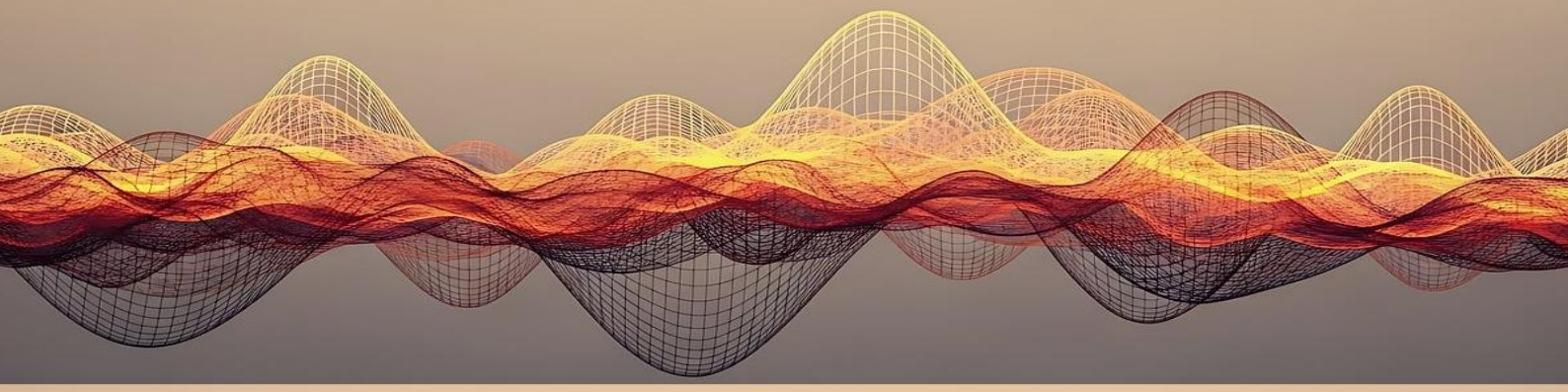


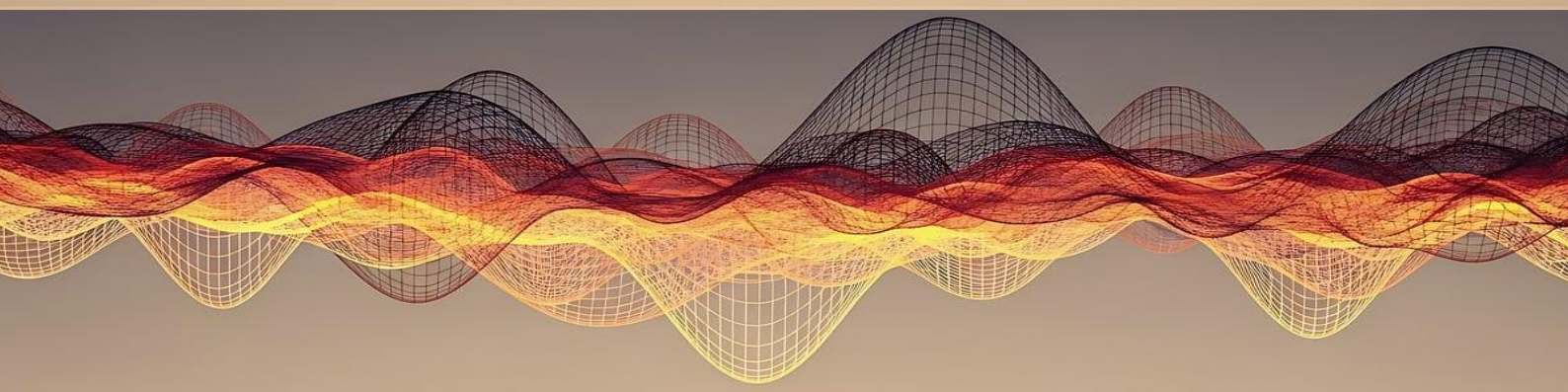
# 5G NR Explained

**Prof. Arturo Azcorra, Dr. Ing., MBA**

**University Carlos III of Madrid**

**and IMDEA Networks**

**Version 1.0, 20 October 2025**



# 1. Introduction

If you are **not** familiar with the overall structure of **5G** (or **LTE**) **radio access network** (i.e. the **overall structure** of **cellular networks** and how the **radio channels** are divided in **carriers**, **bandwidth parts**, **subcarriers**, **symbol intervals**, **resource blocks** and **resource elements**, etc), I strongly recommend that you start reading the **overview** of a **5G network** (click 1.1). This might cause you an **indigestion** of new terms and concepts, but it will provide you a **very useful holistic view**. It is **likely** that you will come back to **re-read** this **overview** once you have read through other parts of the book.

However, if you are already familiar with **5G** (or **LTE**) **networks**, you will likely prefer going directly to chapter 2 on "NR frequency, time and space" or just browse through the **pdf bookmarks** to target the chapter/section that addresses the **matter** that you are looking for. The book uses some color codes that you might want to be aware of (click 1.3).

You may also find useful a description of the **3GPP TS documents** that are **most related** to the **contents** of this **book** (click 1.2).

For the **contents** of the **book**, navigate the **pdf bookmarks**, where you can see the title of **each chapter**, **sections** and **subsections**.

For a brief description of **what is covered**, and **what not**, in this book, click 14.1.

For the **writing principles** followed in this book, click 14.4.

## 1.1 Overview of a 5G network

This **overview** first presents the **3 major subsystems** (click 1.1.1) of a **5G network**. The **overview** then focuses on the description of **one** of these **3 major subsystems** :the **5G Radio Access Network** (**RAN**). First comes a description **5G RAN** (click 1.1.2) is described together with the **cell** and the **new radio** (**NR )node**. Next comes a description of the **channel**, **carrier** and **bandwidth part** (click 1.1.3). This ends with an overview of how a **bandwidth part** is **frequency-time-space multiplexed** (click 1.1.4).

Source web pages:

- https://www.nrexplained.com/
- https://spectrum.ieee.org/3gpp-release-15-overview

### 1.1.1 5G major network subsystems

A **5G network** may be divided in **3 major subsystems**:

- **Radio Access Network** (**RAN**)
- **Transport Network**
- **Core**

**Radio Access Network**

It is composed of the **NR nodes** (click 4), the **cells** (click 3.1.1) plus the **fronthaul** (click 4.6.2) and **midhaul** (click 4.6.2) connectivity. The **RAN** handles **all** the radio-related functionality of the **5G** network, including (among others), connection establishment scheduling, radio-resource management, retransmission protocols, coding, various multi-antenna schemes, routing of user-plane data to the **User Plane Function**

(**UPF**) and control-plane information delivered to the **Access** and **Mobility Management Function** (**AMF**).

A **key element** of the **radio access network** is the **new radio** (**NR**) interface. The **NR interface** is the interface between the **user equipment** (**UE**) and the **NR node**, over the **air**. The **NR interface**, and in particular its **physical layer**, is the **main topic** of this **book**. Some **relevant elements** of the **NR interface** are:

- **NR nodes**, **antenna sites** and **cells** (click 1.1.2).
- **Channels**, **carriers** and **bandwidth parts** (click 1.1.3).
- **Bandwidth part frequency-time-space multiplexing** (click 1.1.4).

**Transport Network**

It is responsible for the **transport** of information **between** the **RAN** and the different **Core** subsystems and functions. This implies connecting **all** the **RAN nodes** deployed throughout the coverage area of the network with the different locations where **Core equipment** is placed.

**Core**

It is responsible for functions **not** related to **radio access** but required to provide a complete network service. This includes, for example, authentication, charging functionality, and setup of end-to-end connections. Handling these functions separately, instead of integrating them into the **RAN**, is beneficial as it allows for several radio-access technologies to be served by the same **Core**.

The **Core** is sometimes called the "**Core Network**". This later term is avoided in this book for considering it **very misleading**, because the **Core** is **not** a **network**.

For a **5G RAN**, **cell** and **NR node overview**, click 1.1.2.

For a **channel**, **carrier** and **bandwidth part overview**, click 1.1.3.

For a **bandwidth part frequency-time-space multiplexing overview**, click 1.1.4.

## 1.1.2 NR overview: NR nodes, antenna sites and cells

This **overview** describes a **very frequent 5G NR configuration**, but there are a **few exceptions** to what is presented here. **These exceptions** are detailed in the sp**ecific sections** (click the corresponding link).

The **radio access networks** (**RAN**) is composed of the **NR nodes** (click 4), the **cells** (click 3.1.1) plus the **fronthaul** (click 4.6.2) and **midhaul** (click 4.6.2) connectivity.

**Each public RAN splits** a **large** geographical **area** (e.g., one country) where the **5G network provides service** into a **large quantity** (e.g., tens of thousands) of **smaller areas** (e.g., of a few square km each) called **cells**.

**Each cell area** (click 3.10.3) is **illuminated** by **1** to **3 standalone antennas** (click 4.5.8.5), or by **1 antenna array** (click 4.5.8.6), that belong to **1** and **only 1 NR node**. This **NR node** provides the **5G services** in the **area** of the **cell** (and typically also in the area of **other cells**). This means that the **NR node**'s **antennas both transmit** to provide service (above the **Downlink performance threshold**, click 3.10.1.1), in the **cell area** and **receive** from the **UEs** (i.e., terminals) to provide service (above the **Uplink performance threshold**) within the **cell area**.

The **antennas** of **each NR node can** be placed at **several locations**, **each** of which is an **antenna site** (click 3.1) that is associated to the **NR node**. The **antennas** at **each antenna site illuminate** a **few** (e.g., 1 to 30) **cells** around it.

**Each cell** provides **service** over **very few** (e.g., 1 to 3) **5G channels** (click 2.2.3).

**Each cell** usually has a **small geographical overlap** with its **neighbor cells** (click 3.1.2) that can **operate** in its **same channel(s)** or in **other channels**. **Cells** can have **significant geographical overlap** (even a **total overlap**) only if they are **illuminated** with **different channels** (to avoid interference).

**Each 5G channel** has a certain **bandwidth** (e.g., 5, 10, 15, 20,…, 100, 200, 400, 800 or 1,600 MHz) and is placed at a **specific spectrum segment** (e.g., one 80 MHz channel from 3,600 MHz to 3,680 MHz).

A **5G User Equipment** (**UE**) is a communications terminal that receives services from the **network**. A **UE** may be a **mobile phone**, a **laptop**, a **fixed internet access point** ("**CPE**"), a **drone**, an **embedded system** in a **car** and many other cases.

**Each cell** hosts a time-varying quantity of **UEs** that receive **services** provided by the **cell**'s **NR node**.

**Each active UE must** be **connected** to **cells** from **one** and **only one NR node**. If **dual connectivity** (click 10.5) is used, **each active UE** may be **simultaneously** connected to **cells** from **2 neighbor NR nodes** (click 4.1).

If **carrier aggregation** (click 10.2) is used, **each active UE** may **aggregate** the capacity from **up to 16 cells** (i.e., a **cell group**, click 3.2.2) of the **same NR node**.

**Each channel** is located within **only one 5G band** (click 2.2.1) and **every channel must comply** with the characteristics that are **defined** for its **band**.

**Each 5G band must** belong to the **3GPP standard bands**. **3GPP** has defined over **80 frequency bands** (click 2.2.13) at different spectrum positions between **600 MHz** and **71 GHz**. **Each such band** has its **specific characteristics** (**frequency location**, **bandwidth**, **duplexing mode**, **channel bandwidths**, …), whose channels have to comply with.

The **cell channel(s)** is/are jointly used to transmit **both** from the **NR node** to the **UE** (i.e., **Downlink**) **and** from the **UE** to the **NR node** (i.e., **Uplink**).

**Each RAN** may be characterized by the c**haracteristics** of its **NR nodes**, and the physical **locations** of its **antenna sites**.

**Each NR node** may be characterized by the **location** of its **antenna sites** and by the characteristics of the **cells** that **it illuminates**.

**Each cell** (click 3.1.1) may be characterized by the **NR node** that **illuminates** it, by its **geographical area**, by the **channels** used to **illuminate** it and by a few other elements.

The **RAN** has to be designed (click 3.10) such that the **combined cells** of **all** the **NR nodes** in the **RAN** provide service, at **each** given **location**, above its corresponding **performance threshold** (click 3.10.1.1).

**5G** and **other mobile communications networks** are sometimes called "**cellular networks**", to specifically indicate that they are composed of **cells**. Although it is true that there are **cell-less mobile communications networks**, the most widespread mobile networks are **cell-based**.

For the **5G network major subsystems**, click 1.1.1.

For a **channel**, **carrier** and **bandwidth part overview**, click 1.1.3.

For a **bandwidth part frequency-time-space multiplexing overview**, click 1.1.4.

**RAN example**

Let us see a **very simple** (but reasonably realistic) configuration **example** of a **5G RAN**, a **NR node**, an **antenna site** and a few **cells**:

- A **5G RAN** covers a **large country** with **thousands** of **NR nodes** and over a **million cells**.
- **One** of its **NR nodes** has **10 antenna sites**.
- **One** of its **antenna sites** is placed at the **top** of given **hill** and illuminates **6 cells**.
- These **6 cells** have the following characteristics:
  - **Three** of the six **cells** have, **each**, an **area** of a **120º circle sector** with a **circle radius** of **.75 km**. The **orientations** of the three **sectors** are **non-overlapping** but with a **joint frontier** (i.e., they cover the whole circle). The **center** of the circle that defines the **3 sectors** is located at the **antenna site**. These three **cells operate** in the **same channel**. This **channel** has a **bandwidth** of **80 MHz**, located from **3,600 MHz** to **3,680 MHz**, within the **TDD** band **n78**.
  - The **other three cells** have, **each**, an **area** of a **120º circle sector** with a **circle radius** of **2 km**. The **orientations** of the three **sectors** are **non-overlapping** but with a **joint frontier** (i.e., they cover the whole circle). The **center** of the circle that defines the **3 sectors** is located at the **antenna site**. The three **cells operate** in the **same FDD channel pair**. **Each channel** has a bandwidth of **10 MHz**. The **Uplink channel** is located from **703 MHz** to **713 MHz**. The **Downlink channel** is located from **758 MHz** to **768 MHz**. This **FDD channel pair** is within the **FDD** band **n28**.

### 1.1.3 NR overview: channels, carriers and bandwidth parts

This **overview** describes a **very frequent channel configuration**, but there are a **few exceptions** to what is presented here. These **exceptions** are detailed in the sp**ecific sections** (click the corresponding link).

**Each radio channel** (click 2.2.3) is a **frequency segment** that is configured in **each cell** (click 3.1.1). A **few** (e.g. 1 to 3) **related radio channels** may be configured in **each cell**. **Each** such **radio channel** is used for the communication between the **cell**'s **NR node** (click 4) and **each** of the active **UEs** (click 4.9) camped in the **cell**. **Each** such **radio channel** is therefore **shared** by **all** the active **UEs** camped in the **cell**.

The **duplexing** (i.e., transmit and receive) can be done in **frequency** or in **time**. The **frequency division duplexing** (**FDD**, click 2.2.12) consists of using **1 radio channel** for **Downlink** (i.e., transmission from the **NR node** to the **UEs**) and a **different radio channel** for **Uplink** (i.e., transmission from the **UEs** to the **NR node**). The **time division duplexing** (**TDD**, click 2.3.4) consists of using a **single radio channel** in **half-duplex** (i.e., it is used **alternating Downlink** and **Uplink**).

**Each radio channel** may be **TDD**, **FDD**, **SUL** (click 10.4) or **SDL** (click 10.5), as it is specified in the **band** (click 2.2.1) to which it belongs. **Each FDD**, **SUL** or **SDL channel** is used in **only one transmission direction** (i.e., a **simplex channel**). **Each FDD channel**

**pair** is composed of **1 only-Downlink channel** and **1 only-Uplink channel**. **Each SUL channel** is **only Uplink**. **Each SDL channel** is **only Downlink**. **Each TDD channel alternates** as **Downlink and Uplink** (i.e., a **half-duplex channel**).

A **carrier** (click 2.2.5) is a **set** of **transmission resources** configured within a **channel**. **Each carrier** can be characterized by its **transmission direction** (either **Uplink** or **Downlink**), by its **bandwidth** (click 2.2.5), by its **DC subcarrier** (click 2.2.10.2), by its **signal type** (see slightly below), by its **subcarrier spacing** (click 2.2.10.3), by its **cyclic prefix** (click 2.3.2), and possibly by other elements. **Every Downlink carrier** corresponds to a **CP-OFDM** (click 11.14) **signal**. **Every Uplink carrier** corresponds to a **CP-DFT-s-OFDM** (click 11.16) **signal** (except in **particular scenarios**, where it can be a **CP-OFDMA** (click 11.15) **signal**).

**Each Downlink**, **Uplink** or **SUL channel** is configured with **Downlink carriers**, **Uplink carriers** or **SUL carriers** (respectively). **Each** of the said **channels** may be configured with a **few** (e.g., 1 to 5) **carriers**. **Each** of the said **channels** may have **only 1 active carrier** at any given time. **Each TDD channel** is configured with a **few** (e.g., 1 to 5) **Downlink carriers and** a **few** (e.g., 1 to 5) **Uplink carriers**, but **each TDD channel** can **only** have **1 Downlink active carrier** or **1 Uplink active carrier** at any given time. As you may see, there may be **Downlink**, **Uplink** and **SUL carriers**.

A **bandwidth part** (click 2.2.7) is one of the **frequency segments** that can be configured **within each carrier**. If **bandwidth adaptation** (**BA**, click 10.1) is used, **each** of the **UE**'s **carriers** may be configured with a **few** (e.g. 1 to 6) **bandwidth parts**. If **BA** is **not** used, **only 1 bandwidth part** can be configured in **each carrier**. The **total quantity** of **bandwidth parts** configured in the **UE**'s **carriers** belonging to **any given type** (**Downlink**, **Uplink** or **SUL**) **cannot** exceed a certain **limit** (e.g., 6).

Within **each time slot** (click 2.3.1), **each UE** may have **only 1 active bandwidth part**, **among** the ones configured **within** any of its **carrier types** (**Downlink**, **Uplink** and **SUL**). Therefore, at any given **slot** the **UE** may have **only 1 Downlink active bandwidth part** and **either only 1 Uplink active bandwidth part** or **only 1 SUL active bandwidth part**.

For the **5G network major subsystems**, click 1.1.1.

For a **5G RAN**, **cell** and **NR node overview**, click 1.1.2.

For a **bandwidth part frequency-time-space multiplexing overview**, click 1.1.4.

### 1.1.4 NR overview: bandwidth part frequency-time-space multiplexing

This **overview** describes a **frequent bandwidth part multiplexing configuration**, but there are a **few exceptions** to what is presented here. **These exceptions** are detailed in **each** of the sp**ecific sections** referenced below (click the corresponding link).

**Each active bandwidth part** (click 1.1.3) is **jointly multiplexed** in **frequency**, in **time** and **possibly** also in **space**:

**Frequency multiplexing** (click 11.6.1) and **frequency multiple access** is done in terms of **subcarriers** (click 2.2.10).

**Time multiplexing** (click 11.6.2) is done in terms of **symbol intervals** (click 2.3.1).

**Space multiplexing** (click 11.6.3), when used, is done in terms of **layers** (click 2.4.3.5), where **each layer** requires at least **1 dedicated antenna port** (click 4.11.5).

**Each active bandwidth part** has therefore a **tridimensional** structure composed of **subcarriers** (in the **frequency** domain), of **symbol intervals** (in the **time** domain) and of **layers** (in the **space** domain).

**Each subcarrier** (click 2.2.10) is an individual **sub-signal** (within the **multicarrier signal** that constitutes the **bandwidth part**) that transports **symbols** (or **sets** of **fractions** of **symbols**, click 11.16) from one (or more) **digital modulation**(s) (click 11.2.1) or from a **complex sequence** (click 11.19). **Each bandwidth part** contains **very many subcarriers multiplexed** in **frequency**, where **all subcarriers usually** (but **not always**) have the **same subcarrier spacing** (click 2.2.10.3). You can think of the **subcarrier spacing** as the **bandwidth** of the **subcarrier**. The **subcarrier spacing must** belong to a **certain set** of **allowed subcarrier spacings**, between **1.5 kHz** and **960 kHz**.

A **symbol interval** is the **time range** during which a **symbol** and its corresponding **cyclic prefix** may be transmitted (except in **PRACH**, click 7.3.1) over **every subcarrier**. Notice that **each transmitted symbol** (or **set** of **symbol fractions**, click 11.16) is **always** preceded by its corresponding **cyclic prefix** (except in **PRACH**). There are **two types** of **cyclic prefix**: **normal** and **extended** (click 2.3.3).

The **average symbol interval duration** is **inversely proportional** to the **subcarrier spacing**. Therefore, **double subcarrier spacing** implies **half duration** of the **symbol interval**. Consequently, the **symbol rate** within a **given bandwidth** is **constant**, **independently** of the **subcarrier spacing** used (assuming that the **given bandwidth** is a **multiple** of the **largest subcarrier spacing**).

For **each Downlink bandwidth part** there is only **one source** (the **NR node**) and there may be **one** (**unicast**) **or several** (**multicast/broadcast**, click 10.8) **destinations** (one or more **UEs**). For **each Uplink** (or **SUL**) **bandwidth part** there are **many sources** (the **UEs**) and only **one destination** (the **NR node**). There is also the **transmission direction** from the **UEs** to other **UEs**, which is called **Sidelink**, which is **not** covered in this book. Because **Uplink** has **multiples transmitters** (i.e., **UEs**) over the **same media**, it requires a **Medium Access Control** (**MAC**) technique. The **MAC** technique used is a **reservation** one (click 11.11), implemented through a **scheduling** (click 4.11.4.1) procedure.

**Do not** confuse the **transmission directions** (**Uplink**, **Downlink**) with the **carrier types** (click 2.2.5), that are **Uplink**, **Downlink** and **SUL**: they are very related, but they are different concepts.

A **Resource Element** (**RE**) is **1 symbol interval** within **1 subcarrier**. **Each** individual **Resource Element** within **each active bandwidth part** may contain **transmitted signal** or **not**. In case the **Resource Element contains transmitted signal**, the **signal** in the **RE** will have **its own specific characteristics**:

- **Subcarrier spacing** (click 2.2.10.3) and **symbol interval duration** (click 2.3.2).
- **Cyclic prefix** type (click 2.3.2).
- **Symbol interval content**: **either 1 symbol** from the **constellation** of a **digital modulation** (click 11.2.1), or **1 complex sequence symbol** (e.g., click 11.20) or **1/n symbol** of **n symbols** (click 11.16).
- **Source system**.
- **Destination system**(**s**).
- **Assigned antenna port**(**s**) (click 4.11.5).

- For **PDSCH** and **PUSCH only**, the **layer(s)** (click 2.4.3.5).
- **Power level**.
- **Assignment** to a **binary channel** or **physical signal** (see below).

A **Resource Element** (**RE**, click 2.2.5.7) in **port** is **1 symbol interval** (click 2.3.1) within **1 subcarrier** (click 2.2.10) and in **1 port** (click 4.11.5). **REs** in **port** that have **multiple ports** for **each RE** are used by **PDSCH** and **PUSCH**.

**Each** of the **very many REs** and **REs** in **each active bandwidth part** is **constantly changing** in terms of containing **signal** or **not**, and if containing signal, in terms of its **signal characteristics**. This results in an **extremely dynamic** content of **each active bandwidth part**.

The **REs** within **each Downlink bandwidth part** are **dynamically scheduled** for each of the following **3 physical binary channels** and **physical signals** (in addition to other **Downlink physical signals**, click 6.20, 6.21 or 6.22):

- The **Synchronization Signal** (**PSS/SSS**, click 6.19) **and Physical Broadcast Channel** (**PBCH**, click 6.2) **Block** (**SSB**, click 6.1).
- The **Physical Downlink Control Channel** (**PDCCH**, click 6.7)
- The **Physical Downlink Shared Channel** (**PDSCH**, click 6.13).

The **Resource Element** within **each Uplink bandwidth part** are **dynamically scheduled** for each of the following **3 physical binary channels** (in addition to **Uplink physical signals**:

- The **Physical Random Access Channel** (**PRACH**, click 7.1).
- The **Physical Uplink Control Channel** (**PUCCH**, click 7.5).
- The **Physical Uplink Shared Channel** (**PUSCH**, click 7.10).

Do **not confuse** these **physical binary channels** with the **analog frequency channels** in which a **band** is divided. These **physical binary channels** are **symbol streams** transported over the **Resource Elements**, **all** of which are **conveyed** over the corresponding **analog frequency channel**.

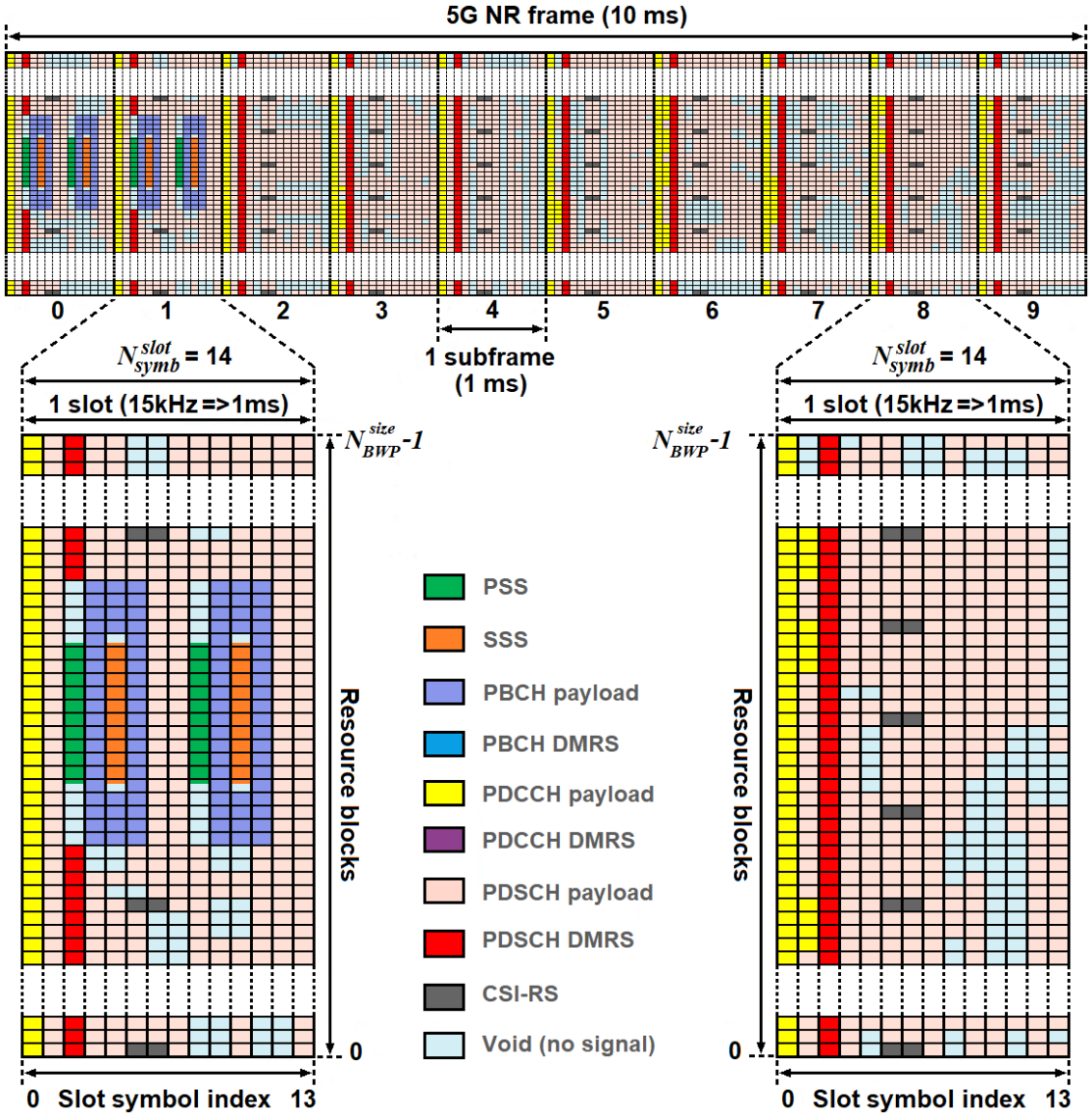
The **REs** corresponding to **each physical channel** or **physical signal** are **not fixed** and may be **constantly changing**. This again results in an **extremely dynamic** configuration of **physical channels** and **physical signals** in each **active bandwidth part**, whose configuration is determined for **each** and **every slot** by the **NR node scheduling** (click 4.11.4.1)

A **Resource Block** (**RB**) is **12 contiguous subcarriers corresponding** to a **Common Resource Block**.

The **12 REs** of **1 Resource Block** are called a **resource element group** (**REG**).

To illustrate the previous explanation, the following figure shows an **approximate** bidimensional (in **frequency** and **time**) representation of a **resource grid** (click 2.2.5.6) of the **Downlink currently active** (click 2.2.7.4) **bandwidth part** (click 2.2.7), for **10 slots** within **1 frame** (click 2.3.1), indicating with colors how **each resource element**

**group** have been **scheduled** (click 4.11.4) to the **different physical binary channels** (click 4.11.3).




Some **relevant remarks** on the **example** in the figure are:

- **Each rectangle** represents **1 resource element group** (**REG**, see above). In the representation of the **5G frame**, **each vertical pixel** represents **3 subcarriers**. In the **zoomed** representation of **2 slots**, **each vertical pixel** represents **1 subcarrier**.
- Notice how the **transmission pattern** is **constantly changing**: **any 2 slots** in the **frame** have **different frequency-time contents**.
- The **SSB** and **PDCCH/PDSCH** have the same **subcarrier spacing**, which is **15 kHz**. For more detail, click 2.2.10.3.
- Since the **PDCCH/PDSCH subcarrier spacing** is **15 kHz**, the **slot duration** is **1 ms**.
- The **frame** contains the **SSB burst** in the **first half-frame**.
- The **SSB burst** corresponds to **SSB pattern Case A**, with **4 SSBs** (the **maximum** possible in **Case A**).

- **Each** of the **4 SSBs** has its **first symbol** located in the **half-frame symbol indexes 2**, **8**, **16** and **22**. For more detail, click 6.1.2.1.3.
- For more detail on the time structure (**frame**, **subframe**, **slot** and **symbol interval**), click 2.3.1.
- This figure is just an example of **one** possible frequency-time pattern out of the **very**, **very many** possible ones.
- If **space multiplexing** were used, **some PDSCH payload allocations** would have **several layers** (click 2.4.3.5), that could be represented in the figure as **planes** in a **third dimension**.
- The figure is drawn to scale: every re

For the system to work properly, all the **individual signals** contained in the **very many Resource Element** and **layers** both **Downlink** and **Uplink**, must be **aligned** in **frequency** (click 2.2.11) and **synchronized** in **time** (click 2.3.4), **both** in a **very precise way**.

For the **5G network major subsystems**, click 1.1.1.

For a **5G RAN**, **cell** and **NR node overview**, click 1.1.2.

For a **channel**, **carrier** and **bandwidth part overview**, click 1.1.3.

Source web pages:

- https://www.rfwireless-world.com/Articles/5G-NR-Physical-Layer.html
- https://www.sharetechnote.com/html/5G/5G_FrameStructure.html
- http://howltestuffworks.blogspot.com/2019/11/5g-nr-resource-blocks.html
- https://www.researchgate.net/figure/5G-NR-Frame-structure-for-0-8-antenna-ports-and-SFI-0_fig1_333719946 (only for the figure above)

## 1.2 Main 3GPP TS documents referred in this book

**3GPP standard documents** (click 10.18) are a moving target and are constantly being updated. To provide some stability and the chance of checking the original **TS document**, this book is based on the **latest TS documents** up to the **end of 2023**.

This implies that **all** the **TS documents** used in this **book**, except the specifications of the **frequency bands** (**TS 38.101**), correspond to **Release 17** (**R17**). **Release 17** is the **last release** of what is considered **5G technology** and **R18** is the **first release** of what is considered **5G Advanced technology**. For the frequency bands (i.e., **TS 38.101** documents), their **initial R18 versions** were already available by the end of **2023**. This allows to combine in this **book** the **most recent 5G technology versions** (**R17**) with the **most recent 5G frequency bands** (**R18**), up to the **end** of the year **2023**.

The **TS documents** that are **most related** to the contents of **this book** are **TS 38.101-1**, **-2** and **-5**, **TS 38.104**, **TS 38.201**, **TS 38.211**, **TS 38.212**, **TS 38.213**, **TS 38.214**, **TS 38.300** and **TS 38.331**.

The specific **version number** and **date**, plus a consent summary, of the different **TS documents** referenced along this book are now presented, **ordered** by **TS number**

- **TS 37.872 V15.1.0** (**2018-12**) "Supplementary **Uplink** (SUL) and LTE-NR co-existence"

  Technical issues and specifications for the co-existence between SUL and LTE-NR.

- **TS 38.101-1 V18.1.0** (**2023-03**) "NR; User Equipment (UE) radio transmission and reception; Part 1: Range 1 Standalone"

- **TS 38.101-2 V18.1.0** (**2023-03**) "NR; User Equipment (UE) radio transmission and reception; Part 2: Range 2 Standalone"

- **TS 38.101-5 V18.3.0** (**2023-09**) "NR; User Equipment (UE) radio transmission and reception; Part 5: Satellite access Radio Frequency (RF) and performance requirements"

  The **previous three TS documents** have the same contents, for different frequency ranges and usages, as in indicated in their title. Frequency, ARFCN, and operating bands. **UE** channel bandwidth. **UE** channel bandwidth for Carrier Aggregation. Channel spacing. Channel raster and synchronization raster.

- **TS 38.104 V17.12.0** (**2023-12**) "NR; Base Station (BS) radio transmission and reception"

  Specification of the radio access for the NR node (base station).

- **TS 38.133 V17.11.0** (**2023-09**) "NR; Requirements for support of radio resource management"

  **RRC_IDLE** state mobility: serving cell.

- **TS 38.201 V17.0.0** (**2021-12**) "NR; Physical layer; General description"

  Overview of the New Radio physical layer. Structure and terminology for the New Radio physical layer. Overall contents of each of the TS 38.2xx documents.

- **TS 38.202 V17.5.0** (**2023-12**) "NR; Services provided by the physical layer"

  Services and functions of the physical layer. Model of physical layer of the **UE**. Parallel transmission of simultaneous physical channels and SRS. Measurements provided by the physical layer.

- **TS 38.211 V17.4.0** (**2022-12**) "NR; Physical channels and modulation"

  For **both** the **NR node** and the **UE**. **Frame structure** and physical resources. **Resource Blocks. Modulation** mapper. **Sequence** generation. **OFDM**, **OFDMA** and **SC-FDMA**. **Downlink physical channels** and **reference signals**, with their **modulation**. **Uplink physical channels** and **reference signals**, with their **modulation**.

- **TS 38.212 17.5.0** (**2023-03**) "NR; Multiplexing and channel coding"

  For **both** the **NR node** and the **UE**. Logical/transport/physical **channel mapping. CRC** Calculation. **CB** segmentation **CRC** attachment. **LDPC**, **polar** and **block coding**.

- **TS 38.213 V17.5.0** (**2023-03**) "NR; Physical layer procedures for control"

  For the **UE** and **not** the **NR node**. **UE cell search**: **SSB** patterns and cell timing. **random access procedure**. **Timing advance** and transmission timing adjustments. **Secondary cell** activation/deactivation. Radio link monitoring. **Uplink** power control. **UE** procedure for determining **PDCCH** assignment.

- **TS 38.214 V17.5.0** (**2023-03**) "NR; Physical layer procedures for data"

  For the **UE** and **not** the **NR node**. Power control. **PDSCH** procedures. **PUSCH** procedures. **Sidelink** procedures. **RTT-based** compensation procedures.

- **TS 38.300 V17.4.0** (**2023-03**) "NR; NR and NG-RAN overall description; Stage-2"

  **RAN** Architecture and functional split. Radio Protocol Architecture. Multi-Radio dual connectivity. Radio Access Network Sharing. Integrated Access and Backhaul. Security. Self-configuration and self-optimization.

- **TS 38.306 V17.7.0** (**2023-12**) "NR; User Equipment (UE) radio access capabilities"

  **UE capabilities**. **UE** capability **parameters**. **UE categories**. **Reduced capability** (**RedCap**) **UEs**.

- **TS 38.321 V17.5.0** (**2023-06**) "NR; Medium Access Control (MAC) protocol specification"

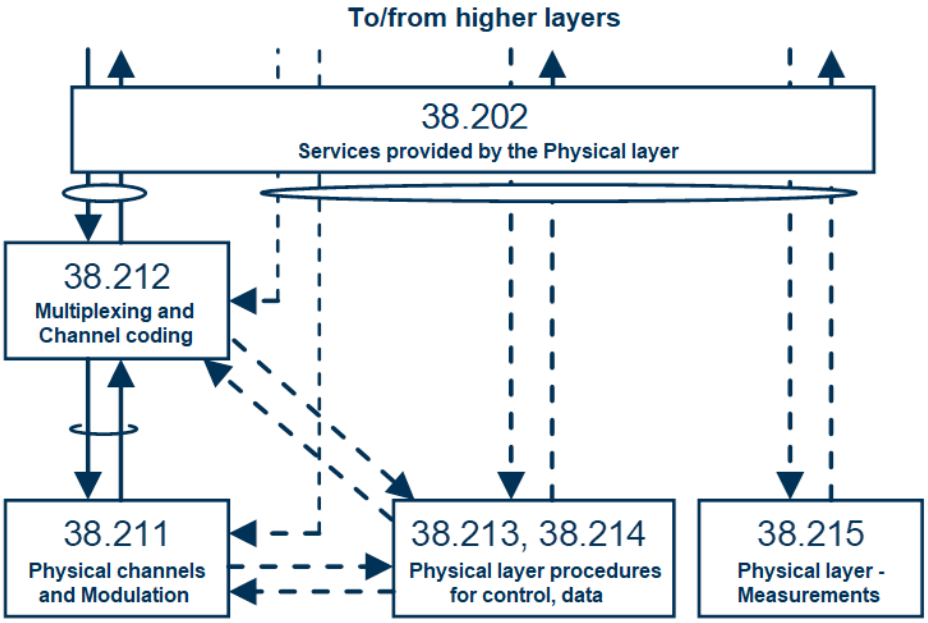
  **Hybrid Automatic Repeat reQuest** (**HARQ**) protocol. **HARQ PDUs** and **procedures**. HARQ codes.

- **TS 38.331 V17.4.0** (**2023-3**) "NR; Radio Resource Control (RRC) protocol specification"

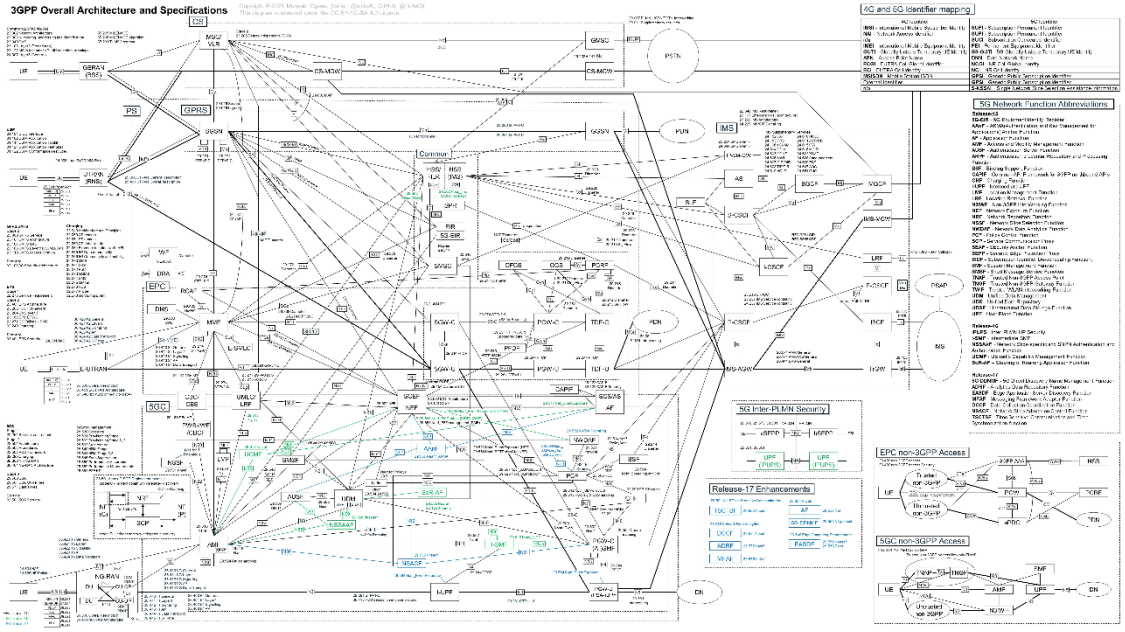
  **ASN.1** specification and description of **all** the **UE configuration parameters**. **RRC** protocol **PDUs** and procedures.

On a few cases, a reference may be done to a different TS document or version other than the ones listed above, but in such cases the specific TS document and version is explicitly indicated.

The following figure shows the relationship between some of these **TS documents**:




The following figure shows a quite complete **map** to **3GPP** systems specification, with the corresponding **Technical Specification** documents:




The following is a list of the TS documents mainly addressed in this **5G** book, with an overview of their contents.

## 1.3 Fonts and colors used for different 5G elements

This book uses the following **font** and **color** conventions to make it easier to **identify** some of the **5G elements** (e.g., **parameters**, **data types**, **data elements**, **subsystems**, etc):

- The **parameters** originated in the **physical layer** itself (e.g., the **PBCH System information** parameters) are written in **brown bold Calibri font** (e.g., **half-frame-flag**).

- The **ASN.1** `parameter names` (i.e., also called information elements (**IEs**)) are written in `medium blue bold Courier font` (e.g., `servingCellConfigCommon`). Recall from 5.7.2.1 that a **parameter** is identified by its **name plus** the **name** of its **enclosing data type**.
- The **ASN.1** `data type names` (i.e., also called information elements (**IEs**) or information element types) are written in `purple bold Courier font` (e.g., `ServingCellConfigCommon`).
- The **ASN.1** `constant names` (e.g., `maxPLMN` or `maxSCSs`), `constant values` (e.g., `1`, `23`, `45`) and `enumerated values` (e.g., `n1`, `n2` or `enabled`) are written in `green bold Courier font`.
- The **ASN.1 keywords** are written in `black bold Courier font` (e.g., **`SEQUENCE`**).
- The **ASN.1** `comments` (e.g., `-- Need R`) are written in `grey bold Courier font`.
- When only **part** of an **ASN.1 specification** is provided, **each fragment** that is **not provided** is replaced by the text:

  "***`other parameters (not shown)`***"

  Notice that an **ellipsis** "**. . .**" is **not used** to indicate this because the **ellipsis** is a **keyword** in **ASN.1** (click 5.9.9).
- The **fields** within **Downlink Control Information** (**DCI**) **messages** and within **Uplink Control Information** (**UCI**) **messages** are written in ***black bold italic Times New Roman font*** (e.g., "***NumCDMGroupsWithoutData***").
- **Terms** related to **subcarriers** are **frequently** written in **purple bold Times New Roman font** (e.g. "**subcarrier**"). **Symbols** representing **subcarriers** are **frequently** written in ***purple bold italic Times New Roman font*** (e.g. "***sc***").
- **Terms** related to **symbol intervals** or to **modulation symbols** are **frequently** written in **medium blue bold Times New Roman font** (e.g. "**symbol interval**"). **Symbols** representing **symbol intervals** or **modulation symbols** are **frequently** written in ***medium blue bold italic Times New Roman font*** (e.g. "***si***").
- **Terms** related to **antenna ports** are **frequently** written in **brownish bold Times New Roman font** (e.g. "**antenna port**"). **Symbols** representing **antenna ports** are **most times** written in ***brownish bold italic Times New Roman font*** (e.g. "***ap***").
- **Terms** related to **MIMO layers** (**not protocol layers**) are **frequently** written in **fuchsia bold Times New Roman font** (e.g. "**layer**"). **Symbols** representing **MIMO layers** (**not protocol layers**) are **frequently** written in ***fuchsia bold italic Times New Roman font*** (e.g. "***lay***").
- **Terms** related to **transport blocks** are **frequently** written in **maroon bold Times New Roman font** (e.g. "**transport block**"). **Symbols** representing **transport blocks** are **frequently** written in ***maroon bold italic Times New Roman font*** (e.g. "***tb***").

- **Terms** related to **codewords** are **frequently** written in **green bold Times New Roman font** (e.g. "**codewords**"). **Symbols** representing **codewords** are **frequently** written in ***maroon bold italic Times New Roman font*** (e.g. "***cwd***").
- **Terms** or **symbols** related to **backhaul** or to the **central unit** are **frequently** written in **dark blue bold Times New Roman font** (e.g. "**backhaul**" or "**CU**").
- **Terms** or **symbols** related to **midhaul** or to the **distributed unit** are **frequently** written in **maroon bold Times New Roman font** (e.g. "**midhaul**" or "**DU**").
- **Terms** or **symbols** related to **fronthaul** or to the **radio unit** are **frequently** written in **purple bold Times New Roman font** (e.g. "**fronthaul**" or "**RU**").

# 2. NR frequency, time and space

The **NR physical layer** performs a **joint usage** of **frequency** (click 2.1.5.5), **time** (click 2.3) and **space** (click 2.4).

**Frequency** is **used** for **Downlink** with **frequency-division multiplexing** (click 11.6.1) and for **Uplink** with **frequency-division multiple-access** (click 11.9). Roughly speaking, the **available system bandwidth** is divided into a **considerable number** of **low bandwidth individual signals**, the **subcarriers** (click 2.2.10).

**Time** is **used** for **Downlink** with **time-division multiplexing** (click 11.6.2) and for **Uplink** with **time-division multiple-access** (click 11.9). Roughly speaking, the **time** of **each subcarrier** is divided into a **sequence** of **time units** called **symbol intervals**.

The **frequency** and **time dimensions** can be **jointly considered** as a **two-dimensional** transmission **resource grid** (click 2.2.5.6). In the **resource grid**, **each** and **every** pair of **1 subcarrier** and **1 symbol interval** conveys **1 symbol**. The **frequency** and **time dimensions** are **coupled** though the concept of **NR numerology** (click 2.1), which is a **very important system parameter**.

**Space** is **used** for **beamforming** (2.4.1) and for **space-division multiplexing** (click 11.6.3) using **MIMO** (click 2.4.3). Roughly speaking, it is possible to transmit as **many non-interfering resource grids**, as **MIMO layers** can be defined in the **cell**. **Each layer** requires, at least, **1 dedicated Tx antenna** and **1 dedicated Rx antenna**.

This chapter describes the characteristics of the **physical layer** of the **Radio Access Network**. The **Air Interface** is a generic term to denote the communication interface between **each NR node** and **each** of the **UEs** within its coverage.

The Air Interface in **5G** is called "**New Radio**" or "**NR**"; in **LTE** it is called "**Evolved Universal Terrestrial Radio Access**" or "**E-UTRA**"; and in **UMTS** it is called "**UMTS Terrestrial Radio Access**" or "**UTRA**".

For more detail on the **physical layer** of **NR**, refer to the **3GPP TS** documents listed in 1.2, where the specific version of each of them that has been used for this book is also indicated.

## 2.1 NR numerology

The **numerology** (denoted "**μ**") is an **integer** number from **0 to 6** that identifies a **pair** of **subcarrier spacing** (**SCS**, click 2.2.10.3) and (average) **slot duration** (click 2.3.1).

Notice that **numerology** is defined in some **3GPP** TS documents as the **combination** of **subcarrier spacing** and **cyclic prefix** (**normal** or **extended**, click 2.3.2). However, throughout the TS documents, the **numerology** "**μ**" **does not have any relation** to the **cyclic prefix** that has been configured. The conclusion is that the **numerology determines** the **subcarrier spacing** and the (average) **slot duration** (and vice versa, click 2.1.1), but the **numerology** does **not** have any direct relationship to the **cyclic prefix**.

Because of the **relationship** between **numerology**, **subcarrier spacing** and (average) **slot duration**, the **symbol rate** for a given **transmission bandwidth** is **approximately constant** (click 2.1.2).

**Numerology** is a **very relevant system parameter**. Having defined **7 numerologies** is **well justified** and has **substantial implications** (click 2.1.3).

**All** the **subcarriers** of **each physical layer element** (e.g., **PBCH**, **CORESET#0**, **PSS**, **PDSCH**) have the **same numerology** at the **same time** instant. However, there are the following **2 cases** of **multiple numerologies**:

- **Several physical layer elements** in the **same radio channel** can have a **different numerology** at **different time instants**.
- **Different physical layer elements** within the **same radio channel** (click 2.2.3) can have a **different numerology** at the **same time** instant.

The **first case** of having **multiple numerologies** leads to **several types** of **numerologies** (click 2.1.4). For more detail on **changing numerologies** along **time**, refer to the **active** (click 2.2.7.4) **bandwidth part** (click 2.2.7) and to the specific descriptions of **each binary channel**.

The **second case** causes that the **transmitted signal simultaneously** contains **multiple subcarrier spacings** and **multiple symbol interval durations**. For more detail on **this case** (i.e., **multiple simultaneous numerologies**), click 2.1.5.

For more detail on the different **parameters** that configure the **subcarrier spacing** (and consequently the **numerology**), click 2.2.10.3.

For more detail on the **numerology configuration** of a **radio channel**, click 2.2.3.2.

For more detail on the **numerology configuration** of a **carrier**, click 2.2.5.

For more detail on **numerology**, refer to clause 4.2 of TS 38.211.

### 2.1.1 Numerology, subcarrier spacing, slot duration and symbol interval duration

The **numerology** of a **subcarrier determines** its **subcarrier spacing** (**SCS**, click 2.2.10.3). The **subcarrier spacing** (except the **PRACH-specific** ones) of a given **subcarrier** is **directly proportional** to $\mathbf{2^{\mu}}$:

$$\textit{Subcarrier_spacing} = \mathbf{15 \cdot 2^{\mu}\ kHz}$$

The **numerology** of a **subcarrier determines** the (average) **slot duration** (click 2.3.2) within the **subcarrier**. The (average) **slot duration** within a given **subcarrier** is **inversely proportional** to $\mathbf{2^{\mu}}$:

$$\textit{Slot_duration} = \mathbf{2^{-\mu}\ ms}$$

The **cyclic prefix** (click 2.3.1) of a **subcarrier determines** (except in **PRACH**) the **quantity** of **symbol intervals** within **each** and **every subcarrier slot**. It is **14 symbol intervals** per **slot** for the **normal cyclic prefix** (**NCP**) and **12** for the **extended cyclic prefix** (**ECP**). Since the **slot duration** is **inversely proportional** to $\mathbf{2^{\mu}}$, the (average) **symbol interval duration** (except the PRACH-specific ones) within a given **subcarrier** is **also inversely proportional** to $\mathbf{2^{\mu}}$, and the **base value** depends on whether NCP or ECP is used:

$$\textit{Symbol_interval_duration}_{\mathrm{NCP}} = \mathbf{1/14 \cdot 2^{-\mu}\ ms \approx 71.4 \cdot 2^{-\mu}\ \mu s}$$

$$\textit{Symbol_interval_duration}_{\mathrm{ECP}} = \mathbf{1/12 \cdot 2^{-\mu}\ ms \approx 83.3 \cdot 2^{-\mu}\ \mu s}$$

Consequently, providing **either** the **numerology** or the **subcarrier spacing** or the **slot duration** or the (average) **symbol interval duration** allows to compute the **other 3 values**, because the **4 of them** are **mathematically interrelated**.

For more detail on the impact of **numerology** on the **subcarrier spacing**, click 2.2.10.3.

For more detail on the impact of the **numerology** on the **transmission time structure**, click 2.3.1.

For more detail on the impact of the **numerology** on the **duration** of a **cyclic prefix**, a **symbol**, a **slot** and a **symbol interval**, click 2.3.2.

### 2.1.2 Symbol rate per MHz of bandwidth

Recall from 2.1.1 that the **subcarrier spacing** and the (average) **symbol interval duration** are **inversely proportional** (click 2.1.1).

For this reason, the **symbol rate** for any given **transmission bandwidth** is **constant**, as it is now shown.

For any **2 subcarriers** "**a**" and "**b**" with **numerologies** "$\mu_a$" and "$\mu_b$", they have **subcarrier spacings** "$2^{\mu a} \bullet x$" and "$2^{\mu b} \bullet x$" **kHz** and they have (average) **symbol interval durations** "$2^{-\mu a} \bullet y$" and "$2^{-\mu b} \bullet y$" **seconds**. Therefore, their **symbol rates** per **MHz** of **transmission bandwidth** "$srw_a$" and "$srw_b$":

$$srw_a = 1{,}000/(2^{\mu a} \bullet x \bullet 2^{-\mu a} \bullet y) = 1{,}000/(x \bullet y) = 1{,}000/(2^{\mu b} \bullet x \bullet 2^{-\mu b} \bullet y) = srw_b$$

As you may see, the **symbol rate** per **MHz** of **transmission bandwidth** is **constant**, **independently** of the **numerology**.

The value of this **constant symbol rate** per **MHz** of **transmission bandwidth** depends on the **cyclic prefix** (click 2.3.1), because it determines the **subcarrier symbol rate** for a given **numerology**. Recall from 2.1.1 that:

$$Subcarrier_spacing = 15 \bullet 2^{\mu} \text{ kHz}$$

$$Symbol_interval_duration_{NCP} = 1/14 \bullet 2^{-\mu} \text{ ms} \approx 71.4 \bullet 2^{-\mu} \text{ µs}$$

$$Symbol_interval_duration_{ECP} = 1/12 \bullet 2^{-\mu} \text{ ms} \approx 83.3 \bullet 2^{-\mu} \text{ µs}$$

Consequently:

$$Symbol_rate_{NCP} \approx 933.3 \text{ kbaud/MHz}$$

$$Symbol_rate_{ECP} = 800.0 \text{ kbaud/MHz}$$

For example, a **20 MHz channel** can be used with **324** (27 RBs) **60 kHz subcarriers**, with **648** (54 RBs) **30 kHz subcarriers** or with **1,296** (108 RBs) **15 kHz subcarriers**. In this example, the **channel transmission bandwidth** is the same: **19.44 MHz**. Regardless of the **numerology** that is used, the **symbol rate** in this **20 MHz channel** is the same:

$$Symbol_rate_{NCP} \approx 933.3 \text{ kbaud/MHz} \bullet 19.44 \text{ MHz} = 18.14 \text{ Mbaud}$$

$$Symbol_rate_{ECP} = 800.0 \text{ kbaud/MHz} \bullet 19.44 \text{ MHz} = 15.55 \text{ Mbaud}$$

Notice that on some **channels** the **transmission bandwidth** is **slightly different** depending on the **subcarrier spacing used**. In these cases, the **symbol rate** is **not completely independent** on the **subcarrier spacing** that is used.

Notice further that **guardband requirements** between **physical layer elements** with **different subcarrier spacings** (click 2.2.5.8) may **slightly reduce** the **symbol rate**.

The **symbol rate** is sometimes called "**modulation rate**" or "**baud rate**".

### 2.1.3 Justification and implications of 7 numerologies

**5G NR** supports **7 numerologies** because it has to work in **bands** within an **extremely wide frequency range**: from **~450 MHz** to **~70 GHz**. Within this **extremely wide range**,

**each band** has its specific characteristics in terms of **propagation**, **doppler**, **inter-symbol interference**, etc. It is therefore **very difficult** to support the complete frequency range with **only 1 numerology** without sacrificing too much efficiency and/or performance.

The **7 numerologies** allow for **subcarrier spacings** from **15 kHz** up to **960 kHz** (plus the **PRACH-specific** ones), and an **inversely proportional** variation in the **average duration** of the corresponding **time elements** (e.g., **symbol interval**, **symbol**, **cyclic prefix** and **slot**).

**Small numerologies** have the advantage of a relatively **long cyclic prefix** in absolute time, at a reasonable overhead. **Large subcarrier spacings** are needed to handle, for example, the **increased phase noise** at higher frequencies and to support **wide channel bandwidths** within the **maximum limit** of **3,300 subcarriers** arising from the **FFT processing load** limit.

**Key points**

- In **OFDM**, the **quantity** of **subcarriers** that can be obtained from a **specific frequency range** is directly related to **spectrum efficiency** (how many bits per second can be transmitted per Hz). The **more subcarriers** that can be sent within a given a **frequency segment** (i.e., the **smaller subcarrier spacing** that is used), the **higher capacity** that can be achieved. In any case, this has to **comply** with the **5G maximum limit** of **3,300 subcarriers**.
- Recall from 2.1.1 that the **symbol interval duration** and the **subcarrier spacing** have an inverse relationship: **narrow subcarrier spacing** means a **longer symbol intervals** With a **longer symbol interval**, it is possible to use a **longer cyclic prefix** (click 2.3.2). With **longer cyclic prefix**, it is possible to get **large cell reach** and better tolerance to fading.
- In **low frequencies** (**FR1** bands) **bands** are quite **narrow**. In order to get **as many subcarriers** as possible in this limited spectrum, it is required to use a **subcarrier spacing as small as possible**. This is why **small subcarrier spacing** (**15**, **30** and **60 kHz**) are used in **low frequency bands**.
- When operating in **very high frequency** (e.g., **mmWave**) as carrier frequency gets higher, the degree of frequency drift by moving transmitter or receiver gets higher. This is, the **Doppler spread** (click 12.1.8) gets **wider** as channel **frequency** gets **higher**. To tolerate this kind of **wide range** of frequency-spread (or **shift**), it is required to use **wider subcarrier spacing**. So **3GPP** has included **very wide subcarrier spacings** (as wide as **960** kHz) in **FR2** bands.
- As **frequency increases**, the degree of **phase noise** also **increases**. Therefore, it is required to implement more sophisticated mechanism for **phase noise estimation** and **correction**. It is **easier** to implement this kind of mechanism with **wider subcarrier spacing**.

Some other implications of using **multiple numerologies** are:

- Because the **average symbol interval duration** is **inversely proportional** to the **subcarrier spacing**, the **raw symbol rate** for a **given bandwidth** is **approximately constant**, **independently** of the **subcarrier spacing** used (click 2.1.2).

- **Larger subcarrier spacings** imply **smaller symbol intervals**. This allows to achieve **lower latency** in the air interface.
- Having designed the **subcarrier spacing** values as **powers** of **2** (of the **numerology**) implies that the corresponding **slot durations** are also **a power of 2**, which allows to **exactly fit** in **1 subframe** an **integer quantity** of **slots** corresponding to **different subcarrier spacings**.

Source web pages:

- https://www.linkedin.com/pulse/motivation-behind-having-multiple-numerology-5g-nr-k-k-upadhayay/
- https://www.techplayon.com/5g-nr-numerology-subcarrier-spcaing-scs/

### 2.1.4 Types of numerologies

Since different **physical layer elements** within the **same radio channel** can have **different numerologies**, when making a reference to a **numerology** it might be required to indicate the **specific NR element** to which it corresponds. There may be the following **different numerologies** within the **same radio channel**:

- **SSB numerology**

  It is the **numerology** used in the **SSB**. It is therefore the **numerology** used for **PSS**, for **SSS** and for **PBCH**. For more detail on its configuration, 6.1.4.

- **CORESET#0 numerology**

  It is the **numerology** used **both** for **CORESET#0** (i.e., for **initial access PDCCH**) and for the delivery of `SIB1` during **initial access** (i.e., for **initial access PDSCH**).

  For more detail on its configuration, click 6.8.8.1.

- **PDSCH numerology**

  It is the **numerology** used for **both PDCCH** and **PDSCH**. It is also the **numerology** for the **Downlink currently active bandwidth part**. It is also the **numerology** for the **transmitted Downlink carrier**. It is also the **numerology** that determines the **Downlink slot duration** (click 2.3.1).

- **PUSCH numerology**

  It is the **numerology** used for **both PUCCH** and **PUSCH**. It is also the **numerology** for the **Uplink currently active bandwidth part**. It is also the **numerology** for the **transmitted Uplink carrier**. It is also the **numerology** that determines the **Uplink slot duration** (click 2.3.1).

The **PDSCH** and **PUCCH numerology(s)** are jointly called the "**control/shared numerology(s)**".

There is **no PRACH numerology**, because in **PRACH**, its **symbol interval duration** is **not directly related** to its **subcarrier spacing**. The reason is that **PRACH** uses **PRACH-specific symbol intervals**.

The **types** of **numerologies** are **very similar**, but **not** the **same**, as the **types** of **subcarrier spacings** (click 2.2.10.4).

### 2.1.5 Multiple simultaneous numerologies

Recall from 2.1 that **different physical layer elements may simultaneously** have **different numerologies**. However, **not all** the **combinations** are **allowed**, as it is explained in the following subsections.

For more detail on the **implications** of **multiple simultaneous subcarrier spacings**, click 2.2.5.8.

For more detail on the **implications** of **multiple simultaneous symbol interval durations** (click 2.1.5.5).

#### 2.1.5.1 Downlink: control/shared and SSB numerology combinations

In a given **Downlink active bandwidth part**, its **non-zero CORESET** and **PDSCH numerology(s)** (i.e., its **Downlink control/shared numerology(s)**, click 2.2.3.2) **may** be **different** to its **SSB numerology** (click 6.1.4), but **not all combinations** are **allowed**.

The following table shows the **forbidden combinations** between **SSB numerology** and **Downlink control/shared numerology**. This table has been produced by consolidating the data from tables 5.3.5-1 (click 2.2.3.2) and 5.4.3.3-1 (click 6.1.5) of TS 38.101-**1**, -**2** and -**5**:

| **Channel** | | | **Downlink control/shared subcarrier spacing [kHz]** | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | **FR1** | | | **FR2-1** | | **FR2-2** | | |
| | | | **15** | **30** | **60** | **60** | **120** | **120** | **480** | **960** |
| **SSB subc. spac. [kHz]** | **FR1** | **15** | 1 | 1/2 | 1/4 | | | | | |
| | | **30** | 2 | 1 | 1/2 | | | | | |
| | **FR2-1** | **120** | | | | 2 | 1 | | | |
| | | **240** | | | | 4 | 2 | | | |
| | **FR2-2** | **120** | | | | | | 1 | 1/4 | 1/8 |
| | | **480** | | | | | | 4 | 1 | 1/2 |
| | | ***960*** | | | | | | *8* | *2* | *1* |

Some **relevant remarks** on this table are:

- The table shows the **numerology** as its corresponding **subcarrier** spacing.
- **Each empty cell** indicates a **combination** between **SSB numerology** and **Downlink control/shared numerology** that is **not allowed** in **any channel**.
- **Each non-empty table cell** indicates a **possible** combination between **SSB numerology** and **Downlink control/shared numerology**, but whether it is **configurable or not** depends on **each specific band** (check the source tables indicated above).
- **Each non-empty table cell** contains the **ratio** of the **SSB subcarrier spacing over** the **Downlink control/shared subcarrier spacing**. You may see that the said **ratio** may take the values **1/8** to **8**, in powers of **2**. It may therefore take the values **1/8** (e.g., 120/960), **1/4** (e.g., 15/60), **1/2** (e.g., 15/30), **1** (e.g., 120/120), **2** (e.g., 120/60), **4** (e.g., 480/120) and **8** (e.g., 960/120). Consequently, the **ratio** of the **SSB symbol interval duration over** a **control/shared symbol interval duration** ranges **from 1/8 to 8**, in powers of **2**.

- The **SSB numerology** "**μ=6**" (i.e., **SCS=960 kHz**) **can only be** used in a **Non Cell-defining SSB** (click 6.1.3), and **not** in a **Cell-defining SSB** (refer to note 2, in table 5.4.3.3-1 of TS 38.101-**2**). This is why its data has been written in italics.
- This table is a **superset** of the table of **combinations** between **SSB numerology** and **CORESET#0 numerology** from the next section (click 2.1.5.2).

### 2.1.5.2 Downlink: SSB and CORESET#0 numerology combinations

In a given **Downlink active bandwidth part**, its **SSB numerology** (click 6.1.4) and its **CORESET#0 numerology** (click 6.8.8.1) may be **different**, but **not all combinations** are **allowed**. Recall that the **PDSCH numerology** for the transmission of **SIB1** is the same as the **PDCCH CORESET#0 numerology**.

The following table shows the **forbidden combinations** between **SSB numerology** and **CORESET#0 numerology**. This table has been produced by consolidating the data from the **headers** of tables 13-1 through 13-10A of TS 38.213:

| **Active bandwidth part** | | | **CORESET#0 numerology [kHz]** | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | **FR1** | | | **FR2-1** | | | **FR2-2** | |
| | | | **15** | **30** | **60** | **60** | **120** | **120** | **480** | **960** |
| **SSB subc. spac. [kHz]** | **FR1** | **15** | 1 | 1/2 | | | | | | |
| | | **30** | 2 | 1 | | | | | | |
| | **FR2-1** | **120** | | | | 2 | 1 | | | |
| | | **240** | | | | 4 | 2 | | | |
| | **FR2-2** | **120** | | | | | | 1 | | |
| | | **480** | | | | | | | 1 | |
| | | ***960*** | | | | | | | | *1* |

Some **relevant remarks** on this table are:

- The table shows the **numerology** as its corresponding **subcarrier** spacing.
- **Each empty cell** indicates a **combination** between **SSB numerology** and **CORESET#0 numerology** that is **not allowed**
- **Each non-empty table cell** is a **possible** combination between **SSB numerology** and **CORESET#0 numerology**, but whether it is **configurable or not** depends on **each specific band** (check the tables indicated above).
- **Each non-empty table cell** contains the **ratio** of **SSB subcarrier spacing over** the **CORESET#0 subcarrier spacing**. You may see that the said **ratio** may take the values **1/2** (e.g., 15/30), **1** (e.g., 120/120), **2** (e.g., 120/60) and **4** (e.g., 240/60). Therefore, the **ratio** ranges **from 1/2 to 4**, in powers of **2**. Consequently, the **ratio** of the **SSB symbol interval duration over** the **CORESET#0 symbol interval duration** ranges **from 1/4 to 2**, in powers of **2**.
- The **SSB numerology** "**μ=6**" (i.e., **SCS=960 kHz**) **can only be** used in a **Non Cell-defining SSB** (click 6.1.3), and **not** in a **Cell-defining SSB** (refer to note 2, in table 5.4.3.3-1 of TS 38.101-**2**). This is why it has been written in italics.
- This table is a **subset** of the table of **combinations** between **SSB numerology** and **Downlink control/shared numerology** from the previous section (click 2.1.5.1).

### 2.1.5.3 Uplink: PRACH and control/shared numerology combinations

In a given **Uplink active bandwidth part** its **PRACH subcarrier spacing** (click 2.2.5.8) and its **control/shared subcarrier spacing** (click 2.2.5.8) may be **different**, but **not all combinations** are **allowed**.

The **allowed combinations** depend on the **PRACH sequence length** "$L_{RA}$" and are listed in the **first three columns** of table 6.3.3.2-1 of TS 38.211 (click 7.2.1).

Notice that **PRACH** has some **PRACH-specific subcarrier spacing** that are **not** bound to **any numerology**, and this is why in **this case** the term **subcarrier spacing** is used **instead** of **numerology**.

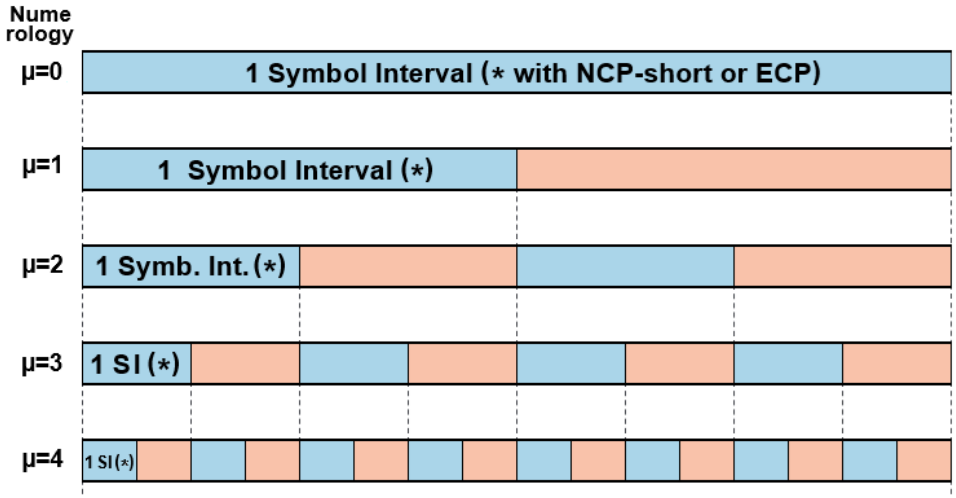
### 2.1.5.4 Downlink, Uplink and SUL: control/shared numerology combinations

In a given **cell**, the **Downlink**, **Uplink** and **SUL control/shared numerology may** be **different**, **without** any **restriction** in respect to the **allowed combinations**.

### 2.1.5.5 Multiple simultaneous symbol interval durations

When **simultaneous physical layer elements** in the **same radio channel** have **different numerologies** (click 2.1.5), it implies that these **elements simultaneously** have **different subcarrier spacings** (click 2.2.5.8) and **different** (average) **symbol interval durations**.

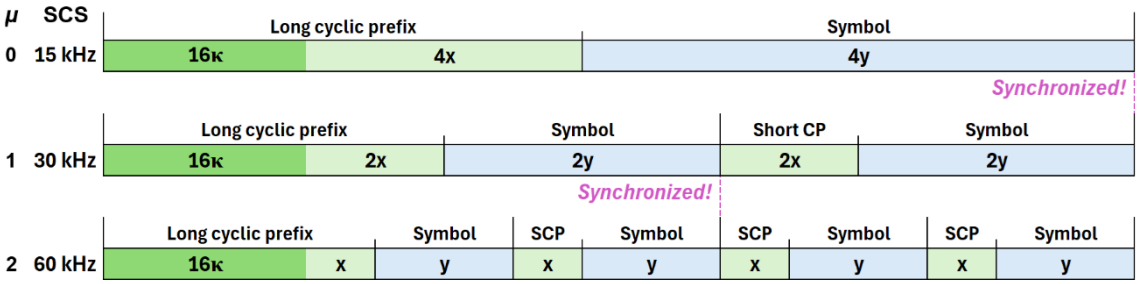
When the **symbol intervals** correspond to the **normal-short CP** (**NCP-short**) or to the **extended CP** (**ECP**), they are **aligned every $2^n$ symbols intervals**. Specifically, for **symbol intervals** corresponding to **numerologies** "$\mu_a$" and "$\mu_b$ (where "$\mu_a < \mu_b$") they are aligned **every $2^{\mu b-\mu a}$ symbol intervals** of **numerology** "$\mu_a$". The following figure shows this:




The **symbol interval durations** for "**μ**=5" and "**μ**=6" are not shown for conciseness' sake, but the **same principle** applies to them.

**After** a **symbol interval** the **normal-long CP** (**NCP-long**), the **alignment rate** for the **normal-short CP** (i.e., **every $2^{\mu b-\mu a}$ symbol intervals** of **numerology** "$\mu_a$") is recovered after the **minimum quantity** of **symbol intervals**, which is **$2^{\mu b-\mu a}$ symbol intervals** of **numerology** "$\mu_a$".

The following figure shows how **symbol interval synchronization** is recovered after a **symbol interval** with the **normal-long cyclic prefix**:




Some **relevant remarks** on this figure are:

- The figure shows how **symbol interval synchronization** between 2 subcarriers with different numerologies "$\mu_a$" and "$\mu_b$" is achieved in the minimum quantity of **symbol intervals**, which is $2^{\mu b-\mu a}$ **symbol intervals** of the smaller numerology "$\mu_a$".
- Recall that the **duration** of **each normal-long CP** is equal to the one of its corresponding **normal-short CP** plus "$16\kappa \cdot T_c$" (whose value is "$16 \cdot 64 \cdot T_c = 1{,}024\ T_c$").
- Recall that the **duration** of **each normal-short CP** and of **each symbol** (e.g., "**x**" and "**y**" for "**μ**") are **exactly half** of the ones of the **next numerology** (i.e., "**x/2**" and "**y/2**"f or "**μ+1**" respectively).
- The duration of a **short cyclic prefix**, of a **symbol** and of "$16\kappa T_c$" are **not** drawn to scale. However, as long as the **relative proportion** indicated in the previous bullet **is kept**, **regardless of their absolute value**, the **synchronization** is achieved **as shown** in the figure.
- Only 3 **numerologies** are shown because the **larger numerologies** would have intervals **too small** to see properly. In any case, the **same principle** applies to the **numerologies** that are not shown in this figure.

## 2.2 NR frequency management

This section presents the spectrum organization in a **top-down** way, under the point of view of the **operator** and the **NR node**. For a **bottom-up UE frequency configuration**, during its **initial access** click 8.

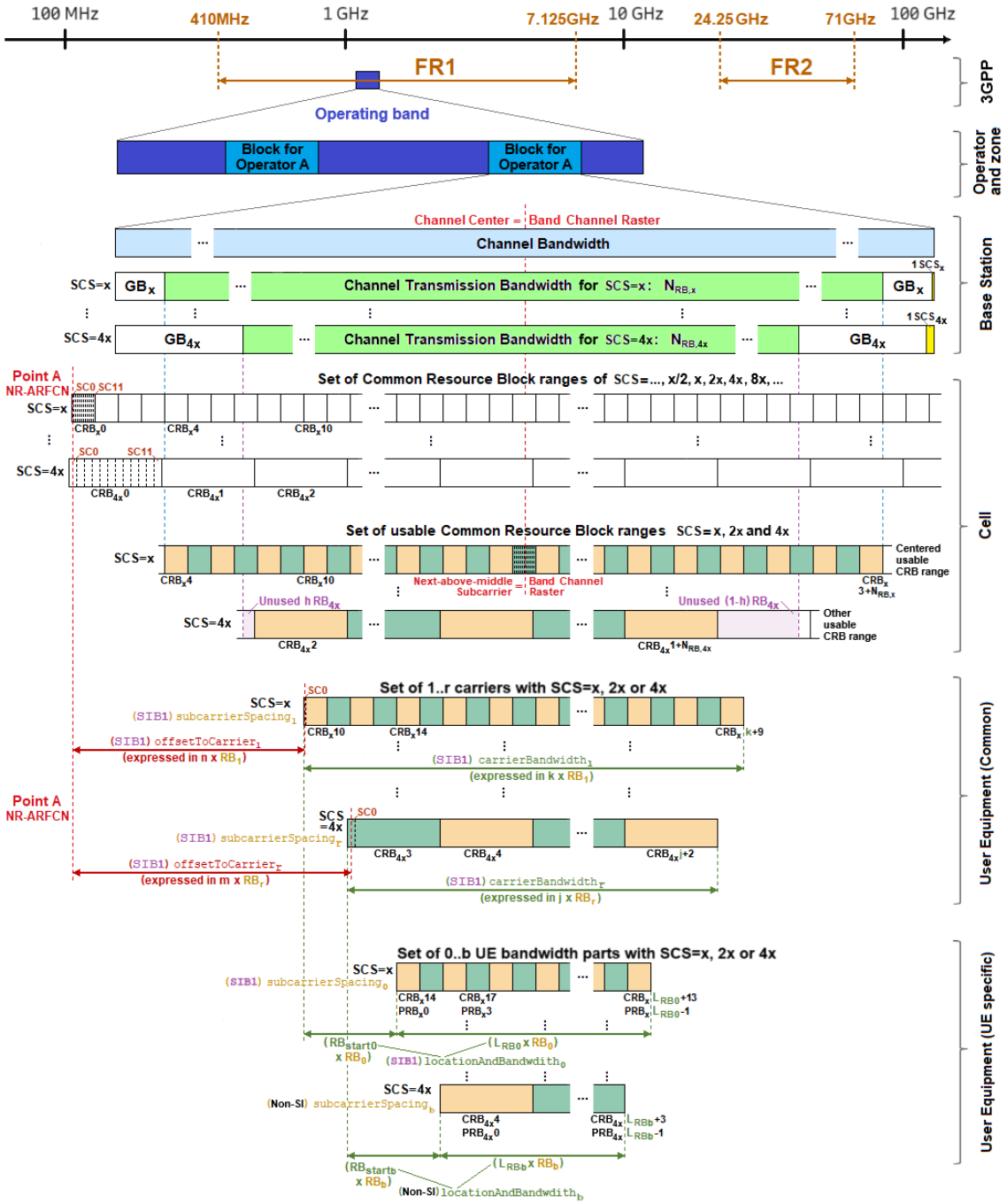
The **5G spectrum** is organized **top-down** in the following way:

- The overall **spectrum** is divided (click 2.2.1) into **2 frequency ranges**, **each** divided into **2 subranges**. **Each subrange** is divided (click 2.2.13) into some (e.g., 1 to 58) **operating bands**, with a definition of their specific characteristics.
- **Each operating band** is divided (click 2.2.2), within **each** given geographical **zone**, into **frequency blocks licensed** to **operators** (or to **other entities**) by **national authorities** (click 2.2.2.1).
- **Each frequency block** is configured (click 2.2.3) with (most frequently) **1 channel** or (less frequent) **several channels**, compliant with the **band**'s **characteristics**. You may therefore have **TDD**, **FDD**, **SUL** or **SDL channels**. **Channels** are assigned to **cells** (click 3.1) by configuring them in **each NR node** (click 4).

- **Each cell**'s **channel** (and for **TDD channels**, in **each** of its **two transmission directions**), is **structured** by configuring (click 2.2.4) **one centered usable CRB range**, **Point A**, **one or more CRB ranges** and **zero or more other usable CRB ranges**.
- **Each cell**'s **channel** (and for **TDD channels**, in **each** of its **two** transmission directions), is configured (click 2.2.5) with **one** or **more carriers**, referred to the corresponding **CRB ranges**. These **carriers** can be **Downlink** (within a **TDD channel** or a **Downlink channel**), **Uplink** (within a **TDD channel** or an **Uplink channel**) or **SUL** (within a **SUL channel**). Notice that **SDL carriers** are **not** currently defined (click 10.5).
- **All** the **cell**'s **Downlink carriers** are jointly organized by configuring (click 2.2.7) them with **1 initial bandwidth part** and possibly with **1 common additional bandwidth part**. The same applies to **all** the **cell**'s **Uplink carriers** and to **all** the **cell**'s **SUL carriers**. **All** the **UE**'s **carriers** belonging to **each cell** are organized by configuring (click 2.2.7) them with a **total** of **1** to **4 dedicated additional bandwidth parts**.
- **Each bandwidth part is organized** in terms of **resource block groups** (**RBGs**, click 2.2.9) for the purpose of **PDSCH** and **PUSCH Type 0 frequency-domain resource allocation**.
- **Each carrier** and **each bandwidth part** has an **integer** quantity of **Resource Blocks**. **Each Resource Block** is **12 contiguous subcarriers** that belong to a **Common Resource Block** (click 2.2.4).
- In addition to the **5G signals**, you may also inject in the channel other signals (click 10.17), placing them in a **frequency range** that does **not overlap** with **any bandwidth part**. One example is **Narrowband IoT signals** (click 10.17),

The **time granularity** between assignments/modifications is **reduced** when moving from **top** to **bottom**: definition of **operating bands** usually takes a few **years**, **channel assignment** to operators usually takes **many months**, **NR node channel configuration** usually remains for **several months**, **Common Resource Block ranges** may stay for **a few months**, **carrier and UE bandwidth part definition** may last **days** to **minutes**, while the **UE active bandwidth part** may change in **seconds**.

The following figure shows an example of this **top-down frequency** organization:




This figure will be explained step by step in the following sections.

The following table shows the **relationship** between the **types** of **bands** (**duplexing mode**), of **channels**, of **CRB ranges**, of **carriers**, of **bandwidth parts** and the **transmission direction**:

| Cell | Element type | | | | | |
|---|---|---|---|---|---|---|
| **Band type (duplex mode)** | SUL | FDD | | TDD | | SDL |
| **Channel type** | | Downlink | Uplink | | | |
| **CRB range (set) type** | | | | Downlink | Uplink | |
| **Carrier type** | | | | | | |
| **Bandwidth part type** | | | | | | |
| **Transmission direction** | Uplink tx | Downlink tx | Uplink tx | Downlink tx | Uplink tx | Downlink tx |

Some **relevant remarks** on this table are:

- Do not confuse the **channel**, **CRB range**, **carrier** or **bandwidth part types** with their **transmission direction**. A **SUL channel**, **SUL CRB range**, **SUL carrier**, or **SUL bandwidth part** are **all** transmitted **Uplink**, but they are **not** the **same** as an **Uplink channel**, an **Uplink CRB range**, an **Uplink carrier**, or an **Uplink bandwidth part** (respectively). The same applies to **SDL** in respect to **Downlink**.
- Notice that a **TDD channel** has **only one TDD CRB range set** (and **one Point A**). **Each TDD CRB range** is therefore used **both** for the **Downlink** and the **Uplink transmission directions**.

The formal **definition** of **bands** and **channels**, plus their **characteristics**, is done in the following **tables** of TS 38.101-**1** (**FR1 terrestrial**), of TS 38.101-**2** (**FR2**) and of TS 38.101-**5** (**FR1 satellite**):

- **Tables 5.1-1**: Definition of **frequency ranges** (click 2.2.1.1).
- **Tables 5.2-1**: **Listing of bands**, with their band **identifier**, **frequency segment(s)** and **duplexing mode** (click 2.2.1.2).
- **Tables 5.4.2.3-1 to -4**: **Channel location** (click 2.2.3.2).
- **Tables 5.3.5-1**: **Subcarrier spacings** and **channel bandwidths** for **each band** (click 2.2.3.2).
- **Tables 5.3.2-1**: **Transmission bandwidth** per "channel bandwidth and subcarrier spacing" (click 2.2.3.6.1).
- **Tables 5.3.3-1**: **Minimum guardband** per "channel bandwidth and subcarrier spacing" (click 2.2.3.6.2).
- **Other related band characteristics** (click 2.2.1.3).

For a **listing** of **5G bands** with a **summary** of their **characteristics**, click 2.2.13.

A good web site with detailed and consolidated band information is:

- https://www.sqimway.com/nr_band.php

### 2.2.1 Band organization of the spectrum, and frequency ranges

**Each 5G operating band** (or just "**5G band** or "**band**") is **one** or **two frequency segment(s)** within the overall **spectrum**.

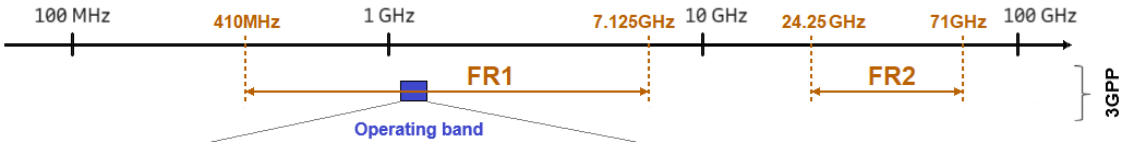
**Each 5G operating band** can be characterized by its **frequency range**, its **frequency segment(s)** and its **duplexing mode** (click 2.2.1.1), by its allowed **channel bandwidths** and its allowed **subcarrier spacings** (click 2.2.3.2), and by a few other attributes (click 2.2.1.4).

The organization of **5G spectrum** into **5G operating frequency bands** is done by **3GPP**, in coordination with **ITU-R**, following agreements in **World Radio Conferences**. The definition of the **characteristics** of **each band** is done by **3GPP**.

The **national authorities** decide which **3GPP** bands (or subsets of) are used, within their country, for **5G communications**.

### 2.2.1.1 Frequency ranges and subranges

The **5G operating frequency bands** are divided into **two groups**, called **Frequency Range 1** (**FR1**) **bands** and **Frequency Range 2** (**FR2**) **bands**, as shown in the following figure:




Some **relevant remarks** on this figure are:

- The figure shows the **two 5G frequency ranges** (**FR1** and **FR2**, click 2.2.1.1), with their **range of frequencies**.
- A **generic FR1 band** (click 2.2.1) is **shown** as a **dark blue rectangle**. Do **not** take into account the **band location**, because the intention is to represent a **generic FR1 band**.
- The **band** could have **TDD**, **FDD**, **SDL** or **SUL duplexing mode** (click 2.2.1.2). If it were an **FDD band**, it would contain **two semi-bands** separated by a **guardband**.

**Each** of the **two frequency ranges** is split into **two subranges**, resulting in **FR1-1**, **FR1-2**, **FR2-1** and **FR2-2**. The **subrange** of **FR1-1** contains (many) **terrestrial** bands and (a few) **satellite** bands. The **definition** of **frequency ranges** and **subranges** is performed in tables **5.1-1** of TS 38.101-**1** (**FR1 terrestrial**), of TS 38.101-**2** (**FR2**) and of TS 38.101-**5** (**FR1 satellite**).

The following table summarizes the information about the **frequency ranges** and **subranges** from the said tables **5.1-1**:

| Freq. Range | Frequency subrange | Frequency | | Band identifiers | | Quantity of bands |
|---|---|---|---|---|---|---|
| | | From | To | From | To | |
| **FR1** | **FR1-1 terrestrial** (click 2.2.13.1) | 410 MHz | 3 GHz | n1 | n105 | 54 |
| | **FR1-2 terrestrial** (click 2.2.13.2) | 3 GHz | 7.125 GHz | | | 9 |
| | **FR1-1 satellite** (click 2.2.13.5) | 410 MHz | 3 GHz | n255 | n256 | 2 |
| **FR2** | **FR2-1** (click 2.2.13.3) | 24.25 GHz | 52.6 GHz | n257 | n262 | 6 |
| | **FR2-2** (click 2.2.13.4) | 52.6 GHz | 71 GHz | n263 | n263 | 1 |

Be aware that the division of **FR1** in **two subranges** is **not** done in table **5.1-1** of TS 38.101 (as is done with the other ones), and it is rather done through references along different clauses from several TS documents.

Notice that **FR2** bands have a **much larger bandwidth** than **FR1** bands, but **FR2** bands have **much shorter range** than **FR1** bands.

The **quantity** of (non-deprecated) operating bands has been **changing** along the different Releases, as follows:

- **Release 15** had **32** operating bands in **FR1 terrestrial** and **4** bands in **FR2**.
- **Release 16** had **45** operating bands in **FR1 terrestrial** and **5** bands in **FR2**.
- **Release 17** had **61** operating bands in **FR1 terrestrial**, **7** bands in **FR2** and **2** bands in **FR1 satellite**.
- **Release 18** (up to Dec-23) had **63** operating bands in **FR1 terrestrial**, **7** bands in **FR2** and **2** bands in **FR1 satellite**.

For a **listing** of **5G bands** with a **summary** of their **characteristics**, click 2.2.13.

### 2.2.1.2 5G bands: band identifier, duplexing mode and frequency segment

**Band identification**

**5G** bands are identified by a **natural number prefixed** by the letter "**n**" (e.g., **n4**, **n34** or **n273**). **LTE** bands are identified by a **natural number** (e.g., **4**, **34** or **273**). **UMTS** bands are identified by a **roman** numeral (e.g., **I**, **II** or **III**). **Each** mobile generation **may** make use of the bands defined for previous generations, doing the so-called "**spectrum refarming**".

When a **5G band** has the **same duplexing** and **same frequency segment** (or overlapping ones) as an **LTE band**, **both** have the **same band integer number**. Such bands are **5G n1** to **n84** that correspond to **LTE** bands **1** to **84** (respectively). Notice however that the **1.4 MHz** and/or **3 MHz channel bandwidths** allowed in some **LTE** bands are **never allowed** in their corresponding **5G bands**, because the said **channel bandwidths do not exist** in **5G**.

**Band duplexing mode**

**Each band**'s **duplexing mode** indicates how the **transmission directions** (**Downlink** and **Uplink**) are managed in the **band**.

According to **duplexing mode**, there are **four mutually exclusive band types**:

- **Time Division Duplexing (TDD) bands**

  **Each channel** within a **TDD band** is used to transmit **Downlink** and **Uplink** (**half-duplex**), alternating in time.

- **Frequency Division Duplexing (FDD) bands**

  **Channels** within **each FDD band** are organized in **channel pairs**. **Each FDD channel pair** has one channel used to transmit **only Downlink** and the other to transmit **only Uplink** (two **simplex channels**).

- **Supplementary Downlink (SDL) bands**

  **Each channel** in an **SDL band** is used to transmit **only Downlink**.

- **Supplementary Uplink (SUL) bands**

  **Each channel** in an **SDL band** is used to transmit **only Downlink**.

For the **relationship** between the **types** of **bands** (**duplexing mode**), of **channels**, of **CRB ranges**, of **carriers**, of **bandwidth parts** and the **transmission direction**, click 1.1.3.

The **FDD**, **SDL** and **SUL bands** are jointly called "**paired spectrum**" **bands**.

For more detail on **Frequency Division Duplexing**, click 2.2.12.

For more detail on **Time Division Duplexing**, click 2.3.4.

**Band frequency segment(s)**

**Each frequency segment** is indicated as the **lowest** and **highest frequency** that the **span** includes. **Each frequency segment** therefore indicates its **frequency location** within the spectrum and its **bandwidth**.

**Each TDD**, **SDL** or **SUL band** is characterized **span** (among other attributes) by **one frequency**.

However, **each FDD band** is divided into **two non-contiguous semi-bands**. One of the **semi-bands only contains Downlink channels**, while the other **semi-band only contains Uplink channels**. **Each FDD band** is therefore characterized (among other attributes) by **one Downlink frequency segment and one Uplink frequency segment**. The **frequency distance** between **both spans** serves as **guardband**.

The **Uplink semi-band** is usually the one with better **SNIR**, which on most cases is the **lower frequency semi-band**. Assigning the **best semi-band** for **Uplink** is done to compensate for the poorer transmission capabilities of the **UE**.

**Most FDD bands** are **symmetric**, this is, **both semi-bands** have the same **bandwidth**. However, **some FDD bands** are **asymmetric**, this is, the **Downlink** and the **Uplink semi-bands** have a **different bandwidth**.

The term i**ntra-band SUL** is sometimes used to refer to a characteristic of **some FDD bands** that contain **more Uplink bandwidth** than **Downlink bandwidth**. The term i**ntra-**

**band SDL** is sometimes used to refer to a characteristic of **some FDD bands** that contain **more Downlink bandwidth** than **Uplink bandwidth**.

*Questions: bands with intra-band SUL/SDL are always handled with paired channels, or you may rather define a different number of channels in Uplink and Downlink?*

Within the **5G** (and the **LTE**) **band tables**, when the **frequency segment** of a given **band** is a **subset** of the **frequency segment** of **another band** (and for **FDD** bands, both **bands** have the **same duplex spacing**), the **narrower band** is considered to be a **subset** of the other **band**.

**Band tables 5.2-1**

The **list of 5G bands**, showing for each band its **identifier**, its **frequency segment** and its **duplexing mode** is indicated in tables **5.2-1** of TS 38.101-**1** (**FR1 terrestrial**), of TS 38.101-**2** (**FR2**) and of TS 38.101-**5** (**FR1 satellite**).

The following is a **fragment** from table "*5.2-1: NR operating bands in FR1*" of TS 38.101-**1**:

| NR operating band | Uplink (UL) operating band BS receive / UE transmit $F_{UL_low}$ – $F_{UL_high}$ | Downlink (DL) operating band BS transmit / UE receive $F_{DL_low}$ – $F_{DL_high}$ | Duplex Mode |
|---|---|---|---|
| **n1** | 1920 MHz – 1980 MHz | 2110 MHz – 2170 MHz | FDD |
| **n2** | 1850 MHz – 1910 MHz | 1930 MHz – 1990 MHz | FDD |
| **…** | … | … | … |
| **n28** | 703 MHz – 748 MHz | 758 MHz – 803 MHz | FDD |
| **n29** | N/A | 717 MHz – 728 MHz | SDL |
| **n30** | 2305 MHz – 2315 MHz | 2350 MHz – 2360 MHz | FDD |
| **n34** | 2010 MHz – 2025 MHz | | TDD |
| **…** | … | … | … |
| **n80** | 1710 MHz – 1785 MHz | N/A | SUL |
| **n81** | 880 MHz – 915 MHz | N/A | SUL |
| **…** | … | … | … |
| **n104** | 6425 MHz – 7125 MHz | | TDD |
| **n105** | 663 MHz – 703 MHz | 612 MHz – 652 MHz | FDD |

The following is **table** "*5.2-1: NR operating bands in FR2*" of TS 38.101-**2**:

| NR operating band | Uplink and Downlink $F_{UL_low}$ – $F_{UL_high}$ | Duplex Mode |
|---|---|---|
| **n257** | 26,500 MHz – 29,500 MHz | TDD |
| **n258** | 24,250 MHz – 27,500 MHz | TDD |
| **n259** | 39,500 MHz – 43,500 MHz | TDD |
| **n260** | 37,000 MHz – 40,000 MHz | TDD |
| **n261** | 27,500 MHz – 28,350 MHz | TDD |
| **n262** | 47,200 MHz – 48,200 MHz | TDD |
| **n263** | 57,000 MHz – 71,000 MHz | TDD[1] |

The following is **table** "*5.2-1: NTN satellite bands in FR1*" of TS 38.101-**5**:

| NTN satellite operating band | Uplink (UL) operating band Satellite Access Node receive / UE transmit $F_{UL,low}$ – $F_{UL,high}$ | Downlink (DL) operating band Satellite Access Node transmit / UE receive $F_{DL,low}$ – $F_{DL,high}$ | Duplex mode |
|---|---|---|---|
| **n256** | 1,980 MHz – 2010 MHz | 2,170 MHz – 2,200 MHz | FDD |
| **n255** | 1,626.5 MHz – 1660.5 MHz | 1,525 MHz – 1,559 MHz | FDD |
| NOTE: NTN satellite bands are numbered in descending order from n256. | | | |

For a **listing** of **5G frequency bands**, with their characteristics, click 2.2.13.

For a **listing** of **LTE frequency bands**, with their characteristics, click 2.2.14.1.

For the **LTE** tables of bands, refer to clause 5 of TS 36.101.

### 2.2.1.3 Example of a 5G band decision

This example is the **first one** in a **series** of **examples** showing the **top-down frequency configuration**, as well as the configuration of a few other elements. For a series of examples showing the **bottom-up UE configuration**, performed through the **initial access** (click 8) procedure, click 8.2.1.

As indicated in 2.2.1.2, the **national authorities** (based on table 5.2-1 of TS 38.101-**2**) **decided** to assign **band n257** for **5G**. The table **entry** for band **n257** is the following:

| Operating Band | Uplink (UL) $F_{UL_low}$ – $F_{UL_high}$ | Downlink (DL) $F_{DL_low}$ – $F_{DL_high}$ | Duplex Mode |
|---|---|---|---|
| **n257** | 26,500 MHz – 29,500 MHz | 26,500 MHz – 29,500 MHz | TDD |

From the information in this table **entry**, **band n257** is in **FR2-1** (click 2.2.1.1), it is **TDD** (click 2.3.4) and it spans from **26,500 MHz** to **29,500 MHz**.

This example continues in 2.2.2.3.

### 2.2.1.4 Other band characteristics

The technical specification documents TS 38.101-**1** (**FR1-terrestrial**), TS 38.101-**2** (**FR2**) and TS 38.101-**5** (**FR1-satellite**) contain clauses to define specific characteristics of **bands**, such as:

- **Carrier aggregation** (**CA**): clause **5.2A** and other "**A-suffix**" clauses.
- **Dual connectivity** (**DC**): clause **5.2B** and other "**B-suffix**" clauses.
- **Supplementary Uplink** (**SUL**): clause **5.2C** and other "**C-suffix**" clauses.
- **Uplink MIMO** (**UL MIMO**): clause **5.2D** and other "**D-suffix**" clauses.
- **Vehicle to anything** (**V2X**): clause **5.2E** and other "**E-suffix**" clauses.

## 2.2.2 Frequency block organization of a band

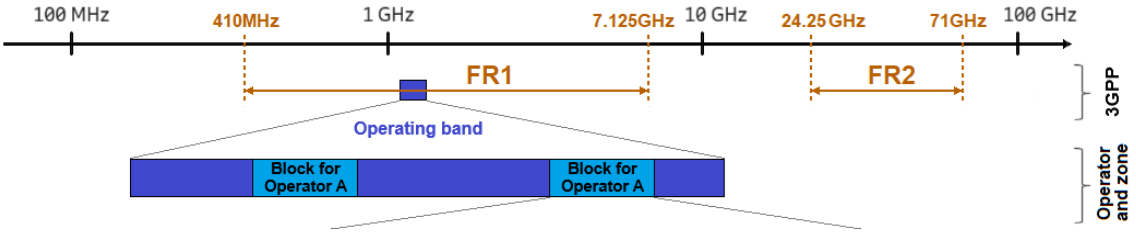
**Each frequency block** (licensed for **5G**) is a **contiguous spectrum segment** within a given **5G band**.

**Each frequency block** can be characterized by its **geographical zone**, its **usage period**, its **cost**, its **usage requirements** (e.g., transmitted power, guard bands, usage for **5G**). and a few other attributes (click 2.2.2.1 or 2.2.2.2).

**Frequency blocks** are licensed by **national authorities** (click 2.2.2.1) to **operators** (or possibly other entities).

The **frequency blocks** in **FDD bands** (click 2.2.1) are licensed in **non-contiguous pairs**. In **each pair**, one of the **frequency blocks** must be within the **FDD Downlink semi-band** and the other within the **FDD Uplink semi-band**. Moreover, the **two frequency blocks** in **each pair** must be **spaced away** the **duplex-spacing** (click 2.2.12) of the **band**.

The following figure shows an example of **two 5G frequency blocks** licensed (within a given 5G **TDD** band) to "**Operator A**":




Some **relevant remarks** on this figure are:

- The figure shows the **two 5G frequency ranges** (**FR1** and **FR2**, click 2.2.1.1), with their **range of frequencies**.
- A **generic FR1 band** (click 2.2.1) is **shown** as a **dark blue rectangle**. Do **not** take into account the **band location**, because the intention is to represent a **generic FR1 band**.
- The **band** could have **TDD**, **FDD**, **SDL** or **SUL duplexing mode**. (click 2.2.1.2). If it were an **FDD band**, it would contain **two semi-bands** separated by a **guardband**.
- **Two frequency blocks** (click 2.2.2) licensed to **operator A** are **shown** as **two blue rectangles**.
- If the **band** had **FDD duplexing mode** (click 2.2.1.2), **each** of the **two frequency blocks** would be placed in **each** of the **band**'s **semi bands**. In this case, **one frequency block** would be for **Downlink** transmission and the **other** for **Uplink** transmission, and they **must** have been licensed for the **same geographical area**.
- If the **band** had **TDD**, **SUL** or **SDL duplexing mode** (click 2.2.1.2), **each** of the **two frequency blocks** could have been licensed **either** for the **same**, **or** for a **different**, **geographical area**.

Notice that the **frequency blocks** licensed by **national authorities** are sometimes called "**channels**". However, the term "**frequency blocks**" is preferred to highlight that a licensed **5G frequency block** is **not** the same as a **5G channel** (click 2.2.2.3). One example of their differences is that **one frequency block** may be configured with **several channels**. Another example is that **one frequency block** may have **different channel configurations** in different zones.

**Channels** within **licensed frequency blocks** are **almost always** operated **without shared spectrum channel access**.

**Channels** within **unlicensed frequency blocks must** be operated **with shared spectrum channel access**.

#### 2.2.2.1 Procedure for spectrum licensing by national authorities

The spectrum is in the **public domain** both **outdoors** and **indoors** in all countries. The **national authorities** decide how to use it, and they usually follow the guidelines from international agreements arising from **ITU** in World Radio Conferences.

National authorities may decide to allocate only a **subset** of a **3GPP frequency band**. For example, in most European countries only part of the **n78** band (from **3,400** to **3,800 MHz**) is being allocated to **5G**.

Each national authority licenses mobile telephony spectrum using different procedures and regulations. The licensing is done for a given period of time, in the order of magnitude of tens of years. There are two extreme licensing models:

- **Beauty contest**

  Consist of defining public service criteria, and the company that commits to offer better service according to the said criteria gets the frequency block assigned. The price to be paid is zero.

- **Auction**

  Consist of calling a public auction and assigning the frequency block to the company with the highest bid.

Most countries follow a **hybrid procedure**, which can be closer to one of the extreme procedures. For example, **auctions** typically include service obligations, spectrum caps, and other regulations usually found on **beauty contest**. Likewise, **beauty contests** sometimes include payment either as fixed fee, or as variable element to be taken into account as an assignment criterion.

To give you an idea of how expensive spectrum is, three **2x10 MHz frequency blocks** in the **700 MHz band**, for 20 year (plus possible extension of 20 additional year) were auctioned in Spain for **1 billion euros each block**. This happened in the 2021 action, for the service zone of the whole Spain. This quantity was **paid flat** on spot by each of the three awardees, and on top of that, each awardee pays a **yearly** spectrum reservation fee in the order of magnitude of **30 M€**.

#### 2.2.2.2 Geographical divisions for frequency allocation

National authorities may decide to dedicate certain parts of the bands for regional allocation. For example, in Spain an allocation region is an autonomous community, while in the United States they usually select the allocation regions based on population.

Large countries (e.g., Brazil, USA) used to hold assignment procedures for each band by zone (e.g., by states), and small countries for the entire country. The trend is to assign each band for the entire national territory in all countries, even the big ones. Some counties (e.g., Germany and Spain) have set up low price assignment procedures for smaller zones, to allow a large industrial plant to have its own frequencies.

An operator usually does not have the same frequency in all its operation zone. Some causes for this may be:

- The operator provides service in several neighboring countries (e.g., in several EU member states), and in each country it got a **different frequency block** licensed by the **national authorities**.
- The national authorities **assigned** part of the spectrum in **smaller geographical zones** (e.g., states), and the allocation process results in the operator having **different blocks** in the **different zones**.
- **Operators trade frequency blocks** among themselves.
- **Operators** can **merge** or **split**, together with **their frequency blocks**.

#### 2.2.2.3 Example of assigned frequency block

This example continues from 2.2.1.3.

**Configuration example up to now**

From 2.2.1.3, this is b**and n257** in **FR2-1**, it is **TDD** and it spans from **26,500 MHz** to **29,500 MHz**.

**Configuration example in this section**

For this example, it is assumed that the **national authorities** have **licensed** (click 2.2.2) to the **operator** one **frequency block within** band **n257**. This **frequency block** has **100 MHz** and is located from **27,650.08 MHz** to **27,750.08 MHz**. Since this is a **licensed frequency block**, it is assumed that its operation is **without shared spectrum channel access.**

This example continues in 2.2.3.4.

### 2.2.3 Channel configuration of a frequency block

**Each 5G channel** is a **contiguous spectrum segment** within a given **5G band**.

For the purpose of **frequency block configuration**, **each channel** can be characterized by the **band** (click 2.2.1) where it is **located**, its **channel type** (click 2.2.3.1), its **bandwidth**, its **subcarrier spacings** and its **exact location**. Also, a **channel** has **other relevant configuration attributes** such as its **CRB ranges** (click 2.2.4) and its **carriers** (click 2.2.5).

**Each frequency block** within a **5G band** is configured, in a certain **geographical zone**, by its **licensee entity** (usually an operator) into **one** or **more channel**(**s**).

The **attributes** of **each channel** within the **frequency block** under configuration may be **determined** in the following **order**:

1. **Channel bandwidth** and **subcarrier spacing**(**s**) (click 2.2.3.2).

   They may be **selected** within the **allowed combinations** that are **stated** in the **band** where the **frequency block** is located.

2. **Channel location** (click 2.2.3.3).

   It may be **selected** among the ones **contained** in the corresponding **band channel raster**. Notice that the **band channel raster** is, on **some bands**, **dependent** on **either** the **channel subcarrier spacings** (**control/shared** and/or **SSB**) **or** on the **channel bandwidth**.

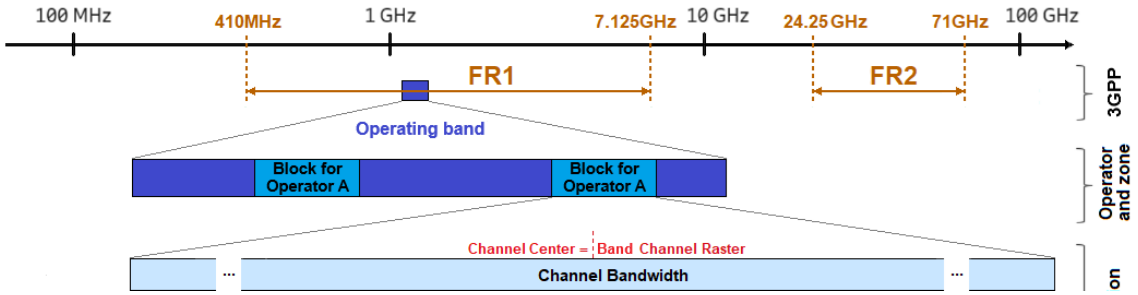
The configured **channel(s)** (with their **channel bandwidth(s)** and **channel raster location(s)**) **must all** be **within** the **frequency block**, taking into account the **adjacent channel spacing** (2.2.3.5) specification.

3. **Channel transmission bandwidth** and **channel minimum guardband pair set** (click 2.2.3.6)

   **Each pair** of **channel transmission bandwidth** and **channel minimum guardband** within this **set** is **determined** by the **channel bandwidth** and by **each** of the **elements** of the **channel control/shared subcarrier spacing set**. Consequently, there is a **bijective correspondence** between the **set** of **pairs** of **channel transmission bandwidth** and **channel minimum guardband** and the **set** of **channel control/shared subcarrier spacings**.

Do not confuse the **frequency block bandwidth** (click 2.2.2), the **channel bandwidth** (click 2.2.3.2) and the **channel transmission bandwidth(s)** (click 2.2.3.5) which are **different concepts**. The **frequency block bandwidth** and the **channel bandwidth** may have a **very different value** (e.g., when **several channels** are configured in **one block**). The **channel bandwidth** and **each** of the **channel transmission bandwidth(s)** **always** have a **different value**, because of the **guardband** at **each channel edge**.

The following figure illustrates an example of **one channel** configured **within** one of the **two frequency blocks** licensed to **Operator A**:




Some **relevant remarks** on this figure are:

- A **generic FR1 band** (click 2.2.1) is **shown** as a **dark blue rectangle**. Do **not** take into account the **band location**, because the intention is to represent a **generic FR1 band**.
- The **band** could have **TDD**, **FDD**, **SDL** or **SUL duplexing mode**. (click 2.2.1.2). If it were an **FDD band**, it would contain **two semi-bands** separated by a **guardband**.
- **Two frequency blocks** (click 2.2.2) licensed to **operator A** are **shown** as **two blue rectangles**.
- If the **band** had **FDD duplexing mode** (click 2.2.1.2), **each** of the **two frequency blocks** would be placed in **each** of the **band**'s **semi-bands**. In this case, **one frequency block** would be for **Downlink channels** and the **other** for **Uplink channels**.
- The **channel bandwidth** (click 2.2.3) is shown as a **light blue rectangle**.
- The figure shows **only one channel** (2.2.3) configured in the **frequency block**, but in the general case it is possible to configure **more than one channel**.

- The **channel type** (**TDD**, **Downlink**, **Uplink**, **SDL** or **SUL**, click 2.2.3.1) **must** correspond to the **band duplexing mode** (**TDD**, **FDD**, **SDL** or **SUL**, click 2.2.1.2).
- The **channel bandwidth must** be one of the **allowed ones** in the **band** (taking into account the **channel**'s **intended control/shared subcarrier spacing**(**s**)), click 2.2.3.2.
- The **channel** is **exactly centered** in **one** of the **reference frequencies** of the **band channel raster** (taking into account the **channel**'s **intended control/shared subcarrier spacing**(**s**)), as it **must** (click 2.2.3.3).

The **channel configuration** within **each frequency block** is **very stable in time**.

Starting with **UMTS** (not with **GSM**) the technology allowed to mitigate co-channel interference between **neighbor cells** operating in the **same channel**. This type of operator's network is called "**single frequency**". This means that a **given frequency block** that an operator has been licensed (click 2.2.2) can be configured to **most** (**even all**) of its **NR nodes** and **cells**. Notice also that the channels used by an operator might not always be the same throughout **all** its coverage zone, because the operator got licenses for different frequency blocks in different regions, within its coverage zone.

**Multiple channel configuration**

As already indicated, it is possible to configure the **frequency block** with **more than one channel**. Fragmenting a **frequency block** in **several channels** is usually **undesirable**, because it implies **inefficiencies** and **over costs**. However, there are **cases** for which **configuring several channels** may be **convenient**.

**One such case** is when there is **mismatch** between the **frequency block bandwidth** and the **band**'s **allowed channel bandwidths** (click 2.2.3.2). This may happen for the following reasons (among others):

- The **operator** got its current **frequency block** by the merger of several previous **frequency blocks** (e.g., from different auctions, from company mergers, from secondary market purchases…).
- The **national authorities did not** dimension the licensed **frequency blocks** according to the **band**'s **channel bandwidths**.
- The **operator** got the **frequency block** as a result of **spectrum refarming** performed by **national authorities**.

To show one example, imagine an **operator** that has a license for an **85 MHz frequency block** in band **n75**. Since **85 MHz** is **not** one of the allowed **channel bandwidths** in **n75**, the **operator cannot** configure a **single channel** in the **frequency block**. The configuration **options** for the **operator** are **two 40 MHz** and **one 5 MHz channels**, **three channels** of **50**, **30** and **5 MHz**, or any other **combination** of the **allowed channel bandwidths** whose **total** is less than or equal **85 MHz**.

**Another such case** is when the **operator** wants to provide service with **two technologies**. For example, an **operator** has been assigned a **50 MHz frequency block** in band **n75** and it configures **one 30 MHz LTE channel** and **one 20 MHz 5G channel** using "**static frequency-domain sharing**" (click 10.6).

Since **spectrum** is **extremely expensive** (click 2.2.2.1), the **operator** will **usually configure each frequency block** with a **channel**, or **channels**, such that the **unused bandwidth** in the **frequency block** is **minimized.**

**Asymmetric channel configuration**

As already indicated, it is possible to configure the **frequency block** with **more than one channel**. Fragmenting a **frequency block** in **several channels** is usually **undesirable**, because it implies **inefficiencies** and **over costs**. However, there are **cases** for which **configuring several channels** may be **convenient**.

**One such case** is when there is **mismatch** between the **frequency block bandwidth** and the **band**'s **allowed channel bandwidths** (click 2.2.3.2). This may happen for the following reasons (among others):

Do not confuse these **analog radio channels** with the **binary channels** (click 4.8) that are conveyed within them.

### 2.2.3.1 Channel types and duplexing mode

**Channel types and duplexing mode**

According to **band duplexing mode** (click 2.2.1.2) and to **transmission direction**, there are the following **four mutually exclusive channel types**:

- **TDD channels**

  **Each TDD channel** transmits **Downlink** and **Uplink** at **alternating times**, and it is configured in a **frequency block** within a **TDD band**. **TDD channels** are **half-duplex channels** used with **time division duplexing** (click 2.3.4).

- **Downlink channels**

  **Each Downlink channel** transmits **Downlink**, and it is configured in a **frequency block** within the **Downlink semi-band** of an **FDD band**. **Downlink channels** are **simplex channels** used with **frequency division duplexing** (click 2.2.12).

- **Uplink channels**

  **Each Uplink channel** transmits **Uplink**, and it is configured in a **frequency block** within the **Uplink semi-band** of an **FDD band**. **Uplink channels** are **simplex channels** used with **frequency division duplexing** (click 2.2.12).

- **SDL channels**

  **Each SDL channel** transmits **Downlink**, and it is configured in a **frequency block** within an **SDL band**. **SDL channels** are **simplex channels** used with **frequency division duplexing** (click 2.2.12).

- **SUL channels**

  **Each SUL channel** transmits **Uplink**, and it is configured in a **frequency block** within a **SUL band**. **SUL channels** are **simplex channels** used with **frequency division duplexing** (click 2.2.12).

Do **not confuse** the **channel type** with the **transmission direction** of a **channel**. A **SUL channel** is **not** and **Uplink channel**. Even though a **SUL channel** is used in the **Uplink transmission direction**, it is **not** an **Uplink channel** (they are two **different terms**). When using the term "**Uplink channels**", this **does not** include **SUL channels**, and it **only** refers to **Uplink channels**, according to the **definition above**. To **jointly** refer to

**Uplink and/or SUL channels** the term that will be used in this book is **either** "**Uplink and/or SUL channels**" or "**channels** in the **Uplink transmission direction**". The same applies between an **SDL channel** and a **Downlink channel**.

Do **not confuse** the **channel type** with the **duplexing mode** of the **band** where the **channel** is located. A **channel** in an **FDD band** is **not** an **FDD channel**. Actually, the "**FDD channel**" **type** has **not been defined** in this book. It can be said that a given **Downlink** or **Uplink channel** is contained in an **FDD band** but it **cannot** be said that it is an **FDD channel**. Notice though that the term "**FDD channel pair**" will be used, to **jointly** refer to a **Downlink channel** and its **paired Uplink channel**, within an **FDD band**.

For the **relationship** between the **types** of **bands** (**duplexing mode**), of **channels**, of **CRB ranges**, of **carriers**, of **bandwidth parts** and the **transmission direction**, click 1.1.3.

**Channel types configured in each band**'s **frequency block**

**Each frequency block** within a **TDD band** is configured with **half-duplex TDD channels**, to be used with **frequency division duplexing** (click 2.3.4).

**Each frequency block** within an **FDD**, **SDL** or **SUL band** is configured with **simplex channels**, to be used with **frequency division duplexing** (click 2.2.12).

**Each frequency block** within an **SDL band** or a **SUL band** is configured with **SDL channels** or **SUL channels** (respectively).

**Each non-contiguous frequency block pair** (click 2.2.1) within an **FDD band** is configured with **non-contiguous channel pairs**. **Each such channel pair** has **one Downlink channel** and **one Uplink channel**. The **Downlink** and **Uplink channel** (both belonging to **each channel pair**) are respectively configured within the **Downlink frequency block** and the **Uplink frequency block** (both belonging to **the frequency block pair** that is being configured). Additionally, the **two channels** in **each channel pair** must be **spaced away** the **duplex-spacing** of the **FDD band** (same as it happens with the **two frequency blocks** in **each frequency block pair**). This is done to **maximize** the **frequency separation** (the **duplex spacing**) between the **two channels** of each **FDD channel pair**, and in this way **minimize self-interference** (click 12.1.14), in particular at the **UEs**. Notice that **each Downlink** or **Uplink frequency block only contains Downlink** or **Uplink channels** (respectively).

### 2.2.3.2 Channel bandwidths and control/shared subcarrier spacings for each band

In **5G-R18** you may **possibly** configure **channels** with a **bandwidth** of **5**, **10**, **15**, **20**, **25**, **30**, **35**, **40**, **45**, **50**, **60**, **70**, **80**, **90** or **100 MHz** (in **FR1** bands) and **50**, **100**, **200**, **400**, **800**, **1,600** or **2,000 MHz** (in **FR2** bands). Notice that **each band only** supports **some** of these **channel bandwidths**. **All** the **possibly** available **channel bandwidths** for **each 5G band** are listed in 2.2.13.

In **5G-R18** you may **possibly** configure **control/shared subcarrier spacings** (click 2.2.10.3) of **15**, **30**, and **60 kHz** (in **FR1** bands) or **60**, **120**, **480**, and **960 kHz** (in **FR2** bands). Notice that **each band only** supports **some** of these **control/shared subcarrier spacings**.

For **each band**, **not all** the **combinations** of **channel bandwidth** and **control/shared subcarrier spacing** from its corresponding **FR** are allowed. The **allowed combinations** for **each band** are indicated in table 5.3.5-1 of TS 38.101-**1**, **-2** and **-5**.

The following is a **fragment** from table "5.*3.5-1: Channel bandwidths for each NR band*" of TS 38.101-**1**:

| NR Band | SCS [kHz] | UE Channel bandwidth [MHz] | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | **5** | **10** | **15** | **20** | **25** | **30** | **35** | **40** | **45** | **50** | **60** | **70** | **80** | **90** | **100** |
| **n1** | **15** | 5 | 10 | 15 | 20 | 25 | 30 | | 40 | 45 | 50 | | | | | |
| | **30** | | 10 | 15 | 20 | 25 | 30 | | 40 | 45 | 50 | | | | | |
| | **60** | | 10 | 15 | 20 | 25 | 30 | | 40 | 45 | 50 | | | | | |
| **…** | **…** | … | | | | | | … | | | | | | | | … |
| **n34** | **15** | 5 | 10 | 15 | | | | | | | | | | | | |
| | **30** | | 10 | 15 | | | | | | | | | | | | |
| | **60** | | 10 | 15 | | | | | | | | | | | | |
| **…** | **…** | … | | | | | | … | | | | | | | | … |
| **n86** | **15** | 5 | 10 | 15 | 20 | | | | 40 | | | | | | | |
| | **30** | | 10 | 15 | 20 | | | | 40 | | | | | | | |
| | **60** | | 10 | 15 | 20 | | | | 40 | | | | | | | |
| **…** | **…** | … | | | | | | … | | | | | | | | … |
| **n105** | **15** | 5 | 10 | 15 | 20 | 25 | 30 | 35 | | | | | | | | |
| | **30** | | 10 | 15 | 20 | 25 | 30 | 35 | | | | | | | | |
| | **60** | | | | | | | | | | | | | | | |

The following is table "5.*3.5-1: Channel bandwidths for each NR band*" of TS 38.101-**2**:

| NR Band | SCS [kHz] | UE Channel bandwidth [MHz] | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | **50** | **100** | **200** | **400** | **800** | **1,600** | **2,000** |
| **n257** | **60** | 50 | 100 | 200 | | | | |
| | **120** | 50 | 100 | 200 | 400[1] | | | |
| **n258** | **60** | 50 | 100 | 200 | | | | |
| | **120** | 50 | 100 | 200 | 400[1] | | | |
| **n259** | **60** | 50 | 100 | 200 | | | | |
| | **120** | 50 | 100 | 200 | 400[1] | | | |
| **n260** | **60** | 50 | 100 | 200 | | | | |
| | **120** | 50 | 100 | 200 | 400[1] | | | |
| **n261** | **60** | 50 | 100 | 200 | | | | |
| | **120** | 50 | 100 | 200 | 400[1] | | | |
| **n262** | **60** | 50 | 100 | 200 | | | | |
| | **120** | 50 | 100 | 200 | 400[1] | | | |
| **n263** | **120** | | | | 400 | | | |
| | **480[2]** | | | | 400 | 800[1] | 1,600[1] | |
| | **960[2]** | | | | 400 | 800[1] | 1,600[1] | 2,000[1] |

Some **relevant remarks** on the two previous tables are:

- **Each empty cell** in the said tables indicates that the **corresponding pair** of **channel bandwidth** and **control/shared subcarrier spacing cannot** be configured in that **band**.
- **Channel bandwidths** with super index "1" are **optional** as of **R18.3**.
- **Subcarrier spacings** with super index "2" are **optional** as of **R18.3**.

Within **each TDD**, SUL or **SDL band**, **different channels** can be configured with **different bandwidth**. Within **each FDD band**, **different channel pairs** can be configured, where the **two channels** of **one channel pair** have a **different bandwidth** in respect to the **two channels** in **another channel pair**.

In **most FDD bands**, the **two simplex channels** of the **same channel pair must** have the **same bandwidth**. However, on a **few FDD bands**, the **said simplex channels** may have **different bandwidth** (click 2.2.3.7).

For the **channel configuration** of the **SSB subcarrier spacing**, click 6.1.4.

For the **channel configuration** of the **CORESET#0 subcarrier spacing**, click 6.8.8.1.

For the **channel configuration** of the **RA preamble subcarrier spacing**, click 9.5.

**Not all** the **combinations** between **SSB subcarrier spacing**, **CORESET#0** (**PDCCH**) **subcarrier spacing** and **control/shared subcarrier spacing** are allowed. For the **allowed combinations**, click 2.1.5.

In **LTE**, depending on each specific band, you may configure channels with a bandwidth of **1.4**, **3**, **5**, **10**, **15** or **20 MHz**. Notice that **LTE** and **5G share** the **5**, **10** and **20 MHz channel bandwidths** in **FR1**.

### 2.2.3.3 Channel location within band channel rasters

The **5G channels cannot** be placed **anywhere** in the **frequency domain**, and rather, **each channel must** be **centered** at an **exact frequency**, that **must** belong to the **correct channel raster** of the **band**.

#### 2.2.3.3.1 Band channel rasters

**Each band channel raster** is a **list** of **exact channel frequencies** (**each identified** by a **channel number**) in one of which **every channel** in the **band must** be **centered**.

**Every band channel raster** is a **subset** of the **5G global frequency raster** (click 2.2.15). It then follows that:

- **Every band**'s **list** of **channel frequencies** (denoted "$CF_{REF,b}$") is a **subset** of the **5G global reference frequencies** "$F_{REF}$" (click 2.2.15).
- **Every band**'s **list** of **channel numbers** (denoted "$CN_{REF,b}$") is a **subset** of the **NR-ARFCNs** (click 2.2.15).
- There is a **one-to-one correspondence** between the elements of **each** "$CF_{REF,b}$" and the ones of its corresponding "$CN_{REF,b}$".
- **Each channel frequency** from **each** "$CF_{REF,b}$" is identified by **one and only one channel number** from its corresponding "$CN_{REF,b}$" (and vice versa).

Some **bands** have **more than one band channel raster**. In these cases, the **band channel raster** to use depends **either** on the **channel subcarrier spacings or** on the **channel bandwidth**.

#### 2.2.3.3.2 Bands with evenly spaced channel frequencies

For these **bands**, their **channel numbers** within **each** "$CN_{REF,b}$" are **evenly spaced**. Consequently, their **channel reference frequencies** within **each** "$CF_{REF,b}$" are also **evenly spaced**.

These bands are **all FR1 bands** (except **n46**, **n96** and **n102**) and **all FR2-1 bands.**

Their **channel numbers** within **each** "$CN_{REF,b}$" are **evenly spaced** in a **band-specific** amount called "$step_size_{,b}$". Consequently, the **spacing** of the **channel frequencies** of **each band** "$b$" is "$step_size_{,b}$" **times** the **global reference frequency spacing** "$\Delta F_{Global,s}$" (click 2.2.15) of the **frequency segment** "$s$" where the **band** "$b$" is located. The

**frequency spacing** between **two consecutive channel frequencies** within **band** "***b***" (located within **segment** "***s***") is denoted by "$\Delta F_{Raster,b}$", and its value is:

$$\Delta F_{Raster,b} = step_size_{,b} \bullet \Delta F_{Global,s}$$

For **each** of **these bands** (and in **paired bands**, for **each** of its **2 semi-bands**), **each** of its (**semi-**)**band channel rasters** is defined with a **range** of **NR-ARFCN** and a "$step_size_{,b}$" value. The **list** "$CN_{REF,b}$" of **all** the **channel numbers** in **each such** (**semi-**)**band channel raster** is obtained by **adding** the **first NR-ARFCN** in the range with a **multiple** (including **zero**) of "$step_size_{,b}$" as long as the resulting value is **lesser or equal than** the **last NR-ARFCN** in the **range**.

These values are listed in table 5.4.2.3-1 of TS 38.101-**1**, **-2** and **-5**.

This is a **fragment** of table 5.4.2.3-1 of TS 38.101-**1**:

| NR operating band | $\Delta F_{Raster,b}$ [kHz] | Uplink range of $CN_{REF,b}$ (First – <$step_size_{,b}$> – Last) | Downlink range of $CN_{REF,b}$ (First – <$step_size_{,b}$> – Last) |
|---|---|---|---|
| **n1** | 100 | 384,000 – <20> – 396,000 | 422,000 – <20> – 434,000 |
| **n2** | 100 | 370,000 – <20> – 382,000 | 386,000 – <20> – 398,000 |
| **…** | … | … | … |
| **n29** | 100 | N/A | 143,400 – <20> – 145,600 |
| **…** | … | … | … |
| **n41** | 15 | 499,200 – <3> – 537,999 | 499,200 – <3> – 537,999 |
| | 30 | 499,200 – <6> – 537,996 | 499,200 – <6> – 537,996 |
| **…** | … | … | … |
| **n47** | 15 | 790,334 – <1> – 795,000 | 790,334 – <1> – 795,000 |
| **…** | … | … | … |
| **n51** | 100 | 285,400 – <20> – 286,400 | 285,400 – <20> – 286,400 |
| **…** | … | … | … |
| **n90** | 15 | 499,200 – <3> – 537,999 | 499,200 – <3> – 537,999 |
| | 30 | 499,200 – <6> – 537,996 | 499,200 – <6> – 537,996 |
| | 100 | 499,200 – <20> – 538,000 | 499,200 – <20> – 538,000 |
| **…** | … | … | … |

This is table 5.4.2.3-1 of TS 38.101-**2**:

| NR operating band | $\Delta F_{Raster,b}$ [kHz] | Uplink and Downlink range of $CN_{REF,b}$ (First – <$step_size_{,b}$> – Last) |
|---|---|---|
| **n257** | 60 | 2,054,166 – <1> – 2,104,165 |
| | 120 | 2,054,167 – <2> – 2,104,165 |
| **n258** | 60 | 2,016,667 – <1> – 2,070,832 |
| | 120 | 2,016,667 – <2> – 2,070,831 |
| **n259** | 60 | 2,270,833 – <1> – 2,337,499 |
| | 120 | 2,270,833 – <2> – 2,337,499 |
| **n260** | 60 | 2,229,166 – <1> – 2,279,165 |
| | 120 | 2,229,167 – <2> – 2,279,165 |
| **n261** | 60 | 2,070,833 – <1> – 2,084,999 |
| | 120 | 2,070,833 – <2> – 2,084,999 |
| **n262** | 60 | 2,399,166 – <1> – 2,415,832 |
| | 120 | 2,399,167 – <2> – 2,415,831 |

This is table 5.4.2.3-1 of TS 38.101-**5**:

| NR operating band | $\Delta F_{Raster,b}$ [kHz] | Uplink range of $CN_{REF,b}$ (First – <*step_size,b*> – Last) | Downlink range of $CN_{REF,b}$ (First – <*step_size,b*> – Last) |
|---|---|---|---|
| **n255** | 100 | 325,300 – <20> – 332,100 | 325,300 – <20> – 332,100 |
| **n256** | 100 | 396,000 – <20> – 402,000 | 396,000 – <20> – 402,000 |

Some **relevant remarks** on the **three** tables above are:

- The **channel numbers** that designate **channel frequencies** so close to a **band edge** that the **channel bandwidth** extends **beyond** the **band edge** shall **not** be used.
- **FDD bands** (e.g., **band n1**) have **pairs** of **band channel rasters**. Within **each pair**, **one band channel raster** is for their **Downlink** channels and the **other** for their **Uplink channels**.
- **SDL bands** (e.g., **band n29**) do **not** have **Uplink** data, because they can only contain **channels** with **Downlink transmission direction**.
- **SUL bands** (e.g., **band n80**) do **not** have **Downlink** data, because they can only contain **channels** with **Uplink transmission direction**.
- **TDD bands** (e.g., **band n51**) have an **identical** channel raster for their **Downlink** and **Uplink channels**. Recall that **all** the **FR2** bands are **TDD**.
- **FR1 bands n41**, **n48**, **n77**, **n78**, **n79** and **n104**, and **all FR2-1 bands**, can be used with **two band channel rasters**. The **band channel raster** derived from the **higher** "$\Delta F_{Raster,b}$" **only** applies to **channels** using the **control/shared subcarrier spacing** that **equals** the **higher** "$\Delta F_{Raster,b}$" **and** using the **SSB subcarrier spacing** that is **equal to or larger than** the **higher** "$\Delta F_{Raster,b}$".
- **FR1 band n90** can be used with **three band channel rasters**.
- **FR1 bands n46**, **n96**, **n102** are included in table 5.4.2.3-1 of TS 38.101-**1**. However, their **channel rasters** are actually defined in tables 5.4.2.3-**2**, **-3** and **-4** of TS 38.101-**1** (respectively).

#### 2.2.3.3.3 Bands with unevenly spaced channel frequencies

For these **bands**, their **channel numbers** within **each** "$CN_{REF,b}$" are **not evenly spaced**. Consequently, their **channel reference frequencies** within **each** "$CF_{REF,b}$" are also **not evenly spaced**. The **spacing** differences are usually small, but on some cases are somehow large.

These bands are **FR1 bands n46**, **n96**, **n102** and **FR2-2 band n263**. **All** of them are **TDD bands**.

**FR1 bands n46**, **n96** and **n102**

**Each** of them has **several channel rasters**, **one** for **each** of its **allowed channel bandwidths**. The **channel raster** for **each band** and **each** of its **allowed channel bandwidth** is provided as **several explicit listings** of **channel numbers** (i.e., **NR-ARFCNs**) **one listing** for **each** of the **channel**'s **allowed bandwidths**. This is detailed in tables 5.4.2.3-**2**, **-3** and **-4** of TS 38.101-**1**.

As an example, this is a **fragment** of table **5.4.2.3-2** of TS 38.101-**1**, that corresponds to **band n46**:

| Channel Bandwidth | Channel numbers ($CN_{REF,b}$) |
|---|---|
| **10 MHz** | 782,000, 788,668 |
| **20 MHz** | 744,000, 745,332, 746,668, 748,000, 749,332, 750,668, 752,000, 753,332, 754,668, 756,000, 765,332, 766,668, 768,000, 769,332, 770,668, 772,000, 773,332, 774,668, 776,000, 777,332, 778,668, 780,000, 781,332, 783,000, 784,332, 785,668, 787,000, 788,332, 789,668, 791,000, 792,332, 793,668 |
| **40 MHz** | … |
| **60 MHz** | … |
| **80 MHz** | … |
| **100 MHz** | 746,668, 753,332, 768,000, 791,000 |

Some **relevant remarks** on the said **three** tables are:

- Notice how the **channel number spacing** is **uneven** (except for the degenerate case of one entry with only two channel numbers).
- Notice how **each band** has several **channel rasters**, **one** for each of the **channel**'s **allowed bandwidths**.
- For **band n46**, the **10 MHz channel bandwidth** shall **only** apply in regions where the **absence** of **non-3GPP technologies** can be guaranteed on a **long-term basis** in this version of specification

**FR2-2 band n263**

It has **several channel rasters**, **one** for **each** of its **allowed channel bandwidths**. The **channel raster** for **each** of its **allowed channel bandwidths** (except for **2,000 MHz**) is provided as a **formula** that results in **evenly spaced channel numbers**. The **channel raster** for its **channel bandwidth** of **2,000 MHz** is provided as a combination of a **formula** and an **explicit list** of **channel numbers**.

This is table **5.4.2.3-2** of TS 38.101-**2**, that corresponds to **band n263**:

| Channel Bandwidth | Channel numbers ($CN_{REF,b}$) |
|---|---|
| **100 MHz** | 2,564,083 + 1,680 • N, N = 0 to 137 |
| **400 MHz** | 2,566,603 + 6,720 • N, N = 0 to 33 |
| **800 MHz** | 2,569,963 + 6,720 • N, N = 0 to 32 |
| **1,600 MHz** | 2,576,683 + 6,720 • N, N = 0 to 30 |
| **2,000 MHz** | 2,580,043 + 6,720 • N, N= 0 to 29<br>and also 2,585,083, 2,655,643, 2,692,603, 2,764,843 |

Some **relevant remarks** on this table are:

- Within **each channel raster**, its **channel numbers** are **evenly space**d, except for the **2,000 MHz channel bandwidth**.

- **Each channel raster** has as many **channel numbers** as possible values of **N**. The exception is the **channel raster** corresponding to **2,000 MHz**, which has **4 additional channel numbers**.
- For the **100 MHz channel bandwidth**, its **channel spacing** is **1,680** units of **NR-ARFCN** which is **100.8 MHz**.
- For the **400**, **800** and **1,600 MHz channel bandwidths**, their **channel spacing** is **6,720** units of **NR-ARFCN** which is **403.2 MHz**.
- For the **2,000 MHz channel bandwidth**, the values of "**N**" that would correspond to the **4 explicit channel numbers** are **0.75**, **11.25**, **16.75** and **27.5** (respectively). In this way you can see how they are located in respect to the other **channel numbers**.

#### 2.2.3.3.4 Summary of band channel raster characteristics

**All FR1 bands** (except **n46**, **n96** and **n102**) and **all FR2-1 bands** have **evenly spaced channel frequencies**.

**All FR1 bands** except **n41**, **n46**, **n48**, **n77**, **n78**, **n79**, **n90**, **n96**, **n102** and **n104** have **only one channel raster**.

**FR1 bands n41**, **n48**, **n77**, **n78**, **n79** and **n104**, and **all FR2-1 bands** have **two channel rasters**. **Band n90** has **three channel rasters**. The **channel raster** to use depends on the **control/shared subcarrier spacing** and on the **SSB subcarrier spacing** that are used in the **channel**.

**FR1 bands n46**, **n96**, **n102** and **FR2-2 band n263** have **as many channel rasters** as **channel bandwidths** are allowed for in the **band**. The **channel raster** to use is the one that corresponds to the configured **channel bandwidth**. **All** of these bands are **TDD bands**.

Restricting the **channel location** to a predefined list of **exact frequencies** (the **band rasters**) allows to simplify the design of the radio frequency elements of the **UE**.

For more detail on the **channel raster**, refer to clause **5.4.2.1** of TS 38.101-**1**, **-2** and **-5**.

Source web pages:

- https://www.nrexplained.com/raster
- https://www.sqimway.com/nr_band.php

#### 2.2.3.3.5 LTE Channel raster

The **LTE channel raster** is a listing of **reference frequencies**, **each** of which is a **valid LTE carrier location**. **Each reference frequency** in the **LTE channel raster** corresponds to the **center frequency** of **next-above-middle subcarrier** of a **valid carrier**. This **next-above-middle subcarrier** is **number 6** of the **middle Common Resource Block** (**CRB**) of the **carrier**. In case the **carrier** has an **even quantity** of **CRBs**, the **next-above-middle subcarrier** is **number 0** from its **CRB** right above the **central carrier frequency**. This location is **half** a **subcarrier above** the **carrier**'s **center frequency**. Recall that **subcarriers** are numbered **0** (lowest frequency) to **11** (highest frequency) within **each Resource Block**.

**Each reference frequency** in the **LTE channel raster** is a function of the **EARFCN**. The **distance** between two **consecutive** reference frequencies in the **LTE channel raster** is **fixed** to **100 kHz**. **Each EARFCN unit** advances the reference frequency in **100 kHz**.

You may calculate (in MHz) the **Downlink** and **Uplink** ref frequencies "$\mathbf{F_{REF_DL}}$" and "$\mathbf{F_{REF_UL}}$" from the **EARFCN** "$\boldsymbol{N_{REF}}$" with the following equations:

$$\mathbf{F_{REF_DL} = 2{,}110 + .1 \cdot N_{REF}}$$

$$\mathbf{F_{REF_UL} = 1{,}920 + .1 \cdot (N_{REF} - 18{,}000)}$$

You may find a calculator of **LTE** channel raster frequencies at:

https://www.sqimway.com/lte_band.php

#### 2.2.3.3.6 GSM Channel raster

The **GSM channel raster** is a listing of **reference frequencies** each of which is a **valid GSM carrier location**. The **distance** between two **consecutive** reference frequencies in the **GSM channel raster** is **fixed** to **200 kHz**, which is the same as the **GSM** channel bandwidth. **Each** reference frequency in the **GSM channel raster** is a function of the **ARFCN**. **Each ARFCN unit** advances the **reference frequency 200 kHz**.

Each **reference frequency** in the **GSM channel raster** corresponds to the **center frequency** of a **valid carrier**.

The following table shows the calculation of the **center frequency** ("$\mathbf{f_{UL}}$" and "$\mathbf{f_{DL}}$") in **MHz** for the **Uplink** and **Downlink** carriers in **each carrier pair** using its associated **ARFCN**:

| Band | Designation | ARFCN | $f_{UL}$ | $f_{DL}$ |
|---|---|---|---|---|
| **GSM 500** | GSM 450 | 259 to 293 | 450.6 + 0.2•(ARFCN - 259) | $f_{UL}$ + 10 |
| | GSM 480 | 306 to 340 | 479.0 + 0.2•(ARFCN - 306) | $f_{UL}$ + 10 |
| **GSM 700** | GSM 750 | 438 to 511 | 747.2 + 0.2•(ARFCN - 438) | $f_{UL}$ + 30 |
| **GSM 850** | GSM 850 | 128 to 251 | 824.2 + 0.2•(ARFCN - 128) | $f_{UL}$ + 45 |
| **GSM 900** | P-GSM | 1 to 124 | 890.0 + 0.2•ARFCN | $f_{UL}$ + 45 |
| | E-GSM | 0 to 124<br>975 to 1,023 | 890.0 + 0.2•ARFCN<br>890.0 + 0.2•(ARFCN - 1,024) | $f_{UL}$ + 45 |
| | GSM-R | 0 to 124<br>955 to 1,023 | 890.0 + 0.2•ARFCN<br>890.0 + 0.2•(ARFCN - 1,024) | $f_{UL}$ + 45 |
| **GSM 1800** | DCS 1800 | 512 to 885 | 1,710.2 + 0.2•(ARFCN - 512) | $f_{UL}$ + 95 |
| **GSM 1900** | PCS 1900 | 512 to 810 | 1,850.2 + 0.2•(ARFCN - 512) | $f_{UL}$ + 80 |

Source web pages:

- https://www.nrexplained.com/raster
- https://www.sqimway.com/nr_band.php
- https://www.techplayon.com/5g-absolute-radio-frequency-channel-number-nr-arfcn/
- https://telcomaglobal.com/p/5g-nr-afrcn-absolute-radio-frequency-channel-number
- https://www.nrexplained.com/rbs
- https://5g-tools.com/4g-lte-earfcn-calculator/

### 2.2.3.4 Example of configuration of a frequency block into channel(s), with its/their bandwidth(s), subcarrier spacing(s) and location(s)

This example continues from 2.2.2.3.

**Configuration example up to now**

From 2.2.1.3, this is b**and n257** in **FR2-1**, it is **TDD** and it spans from **26,500 MHz** to **29,500 MHz**. From 2.2.2.3, we have **one 100 MHz frequency block** from **27,650.08 MHz** to **27,750.08 MHz**.

**Configuration example in this section**

We first configure the **channel bandwidth** and **channel subcarrier spacings**. As indicated in 2.2.3.2, we check table 5.3.5-1 of TS 38.101-**2**, where the table entry for band **n257** is:

| NR operating band | SCS (kHz) | UE channel bandwidth (MHz) | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | 50 | 100 | 200 | 400 | 800 | 1,600 | 2,000 |
| n257 | 60 | 50 | 100 | 200 | | | | |
| | 120 | 50 | 100 | 200 | 400 | | | |

This table entry **allows** for **four** values of **channel bandwidth**. However, the values of **200 MHz** and **400 MHz cannot** be configured in this case, because they are **larger** than the **100 MHz frequency block** assigned to the operator. We may then configure either **one 100 MHz channel** or **two 50 MHz channels**. **Both channel bandwidths** support the same **control/shared subcarrier spacings**. Therefore, in the particular case of this example, there is no interdependency between the **channel bandwidths** and the **channel control/shared subcarrier spacings**.

We can then configure whatever **control/shared subcarrier spacing(s)** and we still can later decide to configure the **frequency block** with either **one 100 MHz channel** or **two 50 MHz channels**.

We choose to configure the **channel** with **only** the **60 kHz control/shared subcarrier spacing**.

We now configure the **channel SSB subcarrier spacing**. As indicated in 6.1.4, we check table 5.4.3.3-1 of TS 38.101-**2**, where the table entry for **band n257** is:

| NR operating band | SSB subcarrier spacing | SSB pattern | Range of GSCN (First – < *step_size*$_{,b}$ > – Last) |
|---|---|---|---|
| n257 | 120 kHz | Case D | 22,388 - <1> - 22,558 |
| | 240 kHz | Case E | 22,390 - <2> - 22,556 |

We have **two options** for the **SSB subcarrier spacing**, and we choose **120 kHz**.

Before configuring the **frequency block** with either **one 100 MHz channel** or **two 50 MHz channels** (and their corresponding **locations**) we first determine what is the

**band channel raster**. From 2.2.3.3, the **band channel raster** is defined in table 5.4.2.3-1, of TS 38.101-**2**. The **table entry** corresponding to **band n257** is:

| NR operating band | $\Delta F_{Raster,b}$ (kHz) | Uplink and Downlink range of $CN_{REF}$ (First – <$step_size_{,b}$> – Last) |
|---|---|---|
| n257 | 60 | 2,054,166 – <1> – 2,104,165 |
| | 120 | 2,054,167 – <2> – 2,104,165 |

We see that **band n257** has **two band channel rasters**. The one to apply depends on the **channel subcarrier spacings**. Since the **control/shared subcarrier spacing** (**60 kHz**) is **not equal** to the **higher** "$\Delta F_{Raster,b}$", (**120 kHz**) then, independently of the **SSB subcarrier spacing** value, we have to use the **band channel raster** with the **lower** "$\Delta F_{Raster,b}$".

We now need to determine what is the **first** (i.e., **lowest**) **frequency** in the **band channel raster**, that is denoted "$CF_{REF\text{-}First,b}$". This is the **reference frequency** that corresponds to the lowest **NR-ARFCN** value (that is denoted "$CN_{REF\text{-}First,b}$") within the **band channel raster**. Checking the corresponding table entry above "$CN_{REF\text{-}First,b}$" is 2,054,166. From 2.2.15, the **channel frequency** corresponding to a **channel number** is obtained with the formula:

$$CF_{REF,b} = F_{REF\text{-}Offs,s} + \Delta F_{Global,s} \bullet (N_{REF,s} - N_{REF\text{-}Offs,s})$$

Also from 2.2.15, the values of parameters "$F_{REF\text{-}Offs,s}$", "$N_{REF\text{-}Offs,s}$" and "$\Delta F_{Global,s}$" are obtained from table 5.4.2.1-1 of TS 38.101-**2**. Since **band n257** is in **frequency segment 3**, the corresponding table entry is:

| $s$ | Frequency segment | Inc. FR | $\Delta F_{Global,s}$ | $F_{REF\text{-}Offs,s}$ | $N_{REF\text{-}Offs,s}$ | Range of $N_{REF,s}$ | Range of $F_{REF,s}$ |
|---|---|---|---|---|---|---|---|
| **3** | **24,250 to 100,000 MHz** | FR2 | 60 kHz | 24,250.08 MHz | 2,016,667 | 2,016,667 to 3,279,165 | 24,250.08 to 99,999.96 MHz |

Therefore, the **first** (**lowest**) **frequency** in the **band channel raster** is:

$$CF_{REF\text{-}First,b} = 24{,}250.08 \text{ MHz} + 60 \bullet 10^{-3} \bullet (2{,}054{,}166 - 2{,}016{,}667)$$

$$CF_{REF\text{-}Frist,b} = 26{,}500.02 \text{ MHz}$$

Consequently, the **band channel raster** frequencies are:

$$CF_{REF,b} = F_{REF\text{-}First,b} + n \bullet \Delta F_{Raster,b}$$

$$CF_{REF,b} = 26{,}500.02 + n \bullet 60 \bullet 10^{-3} \text{ MHz}$$

For a given **channel target frequency** "$CF_{target}$", the **nearest frequency** that belongs to the **channel raster** is:

$$CF_{REF,b} = CF_{REF\text{-}First,b} + (round((CF_{target} - CF_{REF\text{-}First,b})/\Delta F_{Raster,b}) \bullet \Delta F_{Raster,b}$$

The **channel target frequency** "$CF_{target}$" is the one where the **channel** would be **exactly centered** in the **corresponding location** of the **frequency block**.

**Option of configuring one 100 MHz channel**

The **frequency block** is centered is in **27,700.08 MHz**, and this is therefore our **target** frequency "$CF_{target}$".

Using the formula above, the **nearest frequency** to "$\boldsymbol{CF_{target}}$" that belongs to the **channel raster** is:

$$\boldsymbol{CF_{REF,b}} = \boldsymbol{CF_{REF\text{-}First,b}} + (\boldsymbol{round}((\boldsymbol{CF_{target}} - \boldsymbol{CF_{REF\text{-}First,b}})/\boldsymbol{\Delta F_{Raster,b}})\bullet\boldsymbol{\Delta F_{Raster,b}}$$

$$\boldsymbol{CF_{REF,b}} = 26{,}500.02 + (\boldsymbol{round}((27{,}700.08\text{-}26{,}500.02)/(60\bullet10^{-3}))\bullet60\bullet10^{-3} \text{ MHz}$$

$$\boldsymbol{CF_{REF,b}} = 27.700{,}08 \text{ MHz}$$

This is the **center frequency** of the **channel**, which is the **same** as the **channel target frequency**. Therefore, in the particular case of this example, if we configure **one 100 MHz channel** it is **exactly centered** in the **frequency block**.

**Option of configuring two 50 MHz channels**

For the **lower 50 MHz channel**, our **target** frequency "$\boldsymbol{CF_{target}}$" is the **lower quartile** of the **frequency block**, which is **27,675.08 MHz**.

Using the formula above, the **nearest frequency** to "$\boldsymbol{CF_{target}}$" that belongs to the **channel raster** is:

$$\boldsymbol{CF_{REF,b}} = \boldsymbol{CF_{REF\text{-}First,b}} + (\boldsymbol{round}((\boldsymbol{CF_{target}} - \boldsymbol{CF_{REF\text{-}First,b}})/\boldsymbol{\Delta F_{Raster,b}})\bullet\boldsymbol{\Delta F_{Raster,b}}$$

$$\boldsymbol{CF_{REF,b}} = 26{,}500.02 + (\boldsymbol{round}((27{,}675.08\text{-}26{,}500.02)/(60\bullet10^{-3}))\bullet60\bullet10^{-3} \text{ MHz}$$

$$\boldsymbol{CF_{REF,b}} = 27.675{,}06 \text{ MHz}$$

As you may see, the **channel location** has moved **20 kHz downwards** in respect to the **target location**. Therefore, in the particular case of this example, this lower **50 MHz channel overlaps** in **20 kHz** with the **lower adjacent frequency block**. From 2.2.3.5, this **overlap** is **admissible**, because it conforms to the **nominal channel spacing**.

For the **upper 50 MHz channel**, our **target** frequency "$\boldsymbol{CF_{target}}$" is the **upper quartile** of the **frequency block**, which is **27,725.08 MHz**.

Using the formula above, the **nearest frequency** to "$\boldsymbol{CF_{target}}$" that belongs to the **channel raster** is:

$$\boldsymbol{CF_{REF,b}} = \boldsymbol{CF_{REF\text{-}First,b}} + (\boldsymbol{round}((\boldsymbol{CF_{target}} - \boldsymbol{CF_{REF\text{-}First,b}})/\boldsymbol{\Delta F_{Raster,b}})\bullet\boldsymbol{\Delta F_{Raster,b}}$$

$$\boldsymbol{CF_{REF,b}} = 26{,}500.02 + (\boldsymbol{round}((27{,}725.08\text{-}26{,}500.02)/(60\bullet10^{-3}))\bullet60\bullet10^{-3} \text{ MHz}$$

$$\boldsymbol{CF_{REF,b}} = 27{,}725.04 \text{ MHz}$$

As you may see, the **channel location** has moved **40 kHz downwards** in respect to the **target location**. Therefore, in the particular case of this example, this upper **50 MHz** channel leaves a **gap** of **40 kHz** in respect to the **upper adjacent frequency block**. This is admissible because this **channel** does **not overlap** with the **upper adjacent frequency block**.

Notice that another possible configuration is obtained by **moving both channels upwards** in the quantity of "$\boldsymbol{\Delta F_{Raster,b}}$" (i.e., **60 kHz**). If we do this, the **40 kHz gap** would occur in respect to the **lower adjacent frequency block**, while the **20 kHz overlap** would occur in respect to the **upper adjacent frequency block**. From 2.2.3.5, this **overlap** is **admissible**, because it conforms to the **nominal channel spacing**.

For the continuation sections of this example, we choose to configure **one 100 MHz channel exactly centered** in the licensed **frequency block**, as has been shown above.

This example continues in 2.2.3.6.3.

### 2.2.3.5 Adjacent channel spacing

The **spacing** between **adjacent channels** depends on the **bandwidth** of the **band**'s **channel raster granularity** ("$\Delta F_{Raster,b}$", click 2.2.3.3), on the **bandwidths** (click 2.2.3) of the **two adjacent channels** and on the **actual deployment scenario**. The **nominal spacing** between the **central frequencies** of **two adjacent channels** is defined as follows:

**FR1 bands**

- **Bands with "$\Delta F_{Raster,b}$ = 15 kHz"**

  Nominal channel spacing = Cc2Cc + {-5 kHz, 0 kHz, 5 kHz}

- **Bands with "$\Delta F_{Raster,b}$ = 30 kHz"**

  Nominal channel spacing = Cc2Cc + {-10 kHz, 0 kHz, 10 kHz}

- **Bands with "$\Delta F_{Raster,b}$ = 100 kHz"**

  Nominal channel spacing = Cc2Cc

**FR2-1 bands**

- **Bands with "$\Delta F_{Raster,b}$ = 60 kHz"**

  Nominal channel spacing = Cc2Cc + {-20 kHz, 0 kHz, 20 kHz}

- **Bands with "$\Delta F_{Raster,b}$ = 120 kHz"**

  Nominal channel spacing = Cc2Cc + {-40 kHz, 0 kHz, 40 kHz}

**FR2-2 bands** (i.e., band n263)

- Nominal channel spacing = ***ceiling***(Cc2Cc / 50.4) • 50.4 MHz

In the expressions above **Cc2Cc** stands for "**Channel center to Channel center**". It is the frequency **distance** between the **center frequency** of the **two channels** in the **particular case** that the **two channels** were **exactly contiguous**. The value of **Cc2Cc** is computed as:

$$Cc2Cc = (BW_{Channel,x} + BW_{Channel,y}) / 2$$

where $BW_{Channel,a}$ and $BW_{Channel,b}$ are the **channel bandwidths** (**not** the **transmission bandwidths**) of the **two adjacent channels** "**x**" and "**y**".

Some **relevant remarks** on the expressions above are:

- The **nominal channel spacing** is always a **multiple** of the corresponding "$\Delta F_{Raster,b}$" (click 2.2.3.3) **value**. The reason is that **every channel must** be **centered** in a **reference frequency** from the **band channel raster**. Consequently, if the **specific value Cc2Cc** is **not** a **multiple** of "$\Delta F_{Raster,b}$", the **nominal channel spacing cannot** be **Cc2Cc**. In this case, the **two adjacent channels cannot** be **exactly contiguous**. This is clarified in the next bullet.

- For the "$\Delta F_{Raster,b}$" **values** of **15**, **30**, **60** and **120 kHz**, the **nominal channel spacing** has a **slight deviation** over **Cc2Cc** in case that the specific value of **Cc2Cc** is **not** a multiple of "$\Delta F_{Raster,b}$". In the said case, the two **adjacent channels either slightly overlap** (if the **slight deviation** is **negative**) **or** have a **slight gap** between them (if the **slight deviation** is **positive**). The **possible values**

of the **slight deviation** are indicated between curly brackets in the expressions above.

- For the "$\Delta F_{Raster,b}$" **value** of **100 kHz** there is **no slight deviation** and the **nominal channel spacing** is **exactly Cc2Cc**. The reason is obviously that for "$\Delta F_{Raster,b}$ **= 100 kHz**", the value of **Cc2Cc** is **always** a multiple of **100 kHz**, because **all channel bandwidths** are a **multiple** of **100 kHz**.
- In **band n263**, the **nominal channel spacing** is the value of **Cc2Cc rounded up** to a **multiple** of **50.4 MHz**.

For example, **FR1 bands n41**, **n46**, **n47**, **n48**, **n77**, **n78**, **n79**, **n96**, **n102** and **n104** have a **channel raster granularity** "$\Delta F_{Raster,b}$" of **15 kHz** and/or **30 kHz**. Consequently, if we want to configure **two channels**, **both** with a **bandwidth** of either **5**, **20**, **25**, **35**, **40**, **50**, **70**, **80** or **100 MHz**, they **cannot** be **exactly adjacent**. The reason is that the **channels must** be **centered** in a **band raster frequency**, and in this case, this is **incompatible** with them **being exactly adjacent**. These **two channels** should therefore have a **slight gap** or **overlap**, as indicated above.

For a **detailed** configuration example that shows the **adjacent channel spacing**, click 2.2.3.4.

It is important to point out that the **actual channel spacing can be different** to the **nominal channel spacing**, because of **deployment adjustments** performed depending on the **channel raster** to optimize performance in each actual operation scenario. Therefore, it may be **acceptable** that **adjacent channels actually** have a **larger overlap/gap** than the slight **deviation** indicated by the **nominal channel spacing**.

Complying with the **adjacent channel spacing** requirements depends on the **bandwidth** and **location** of the **channel(s)** configured by the **licensee** in its **frequency block**, and on the **adjacent channels configured** by the **licensees** of the **two adjacent frequency blocks**. This may require **coordination** between the **involved licensees**.

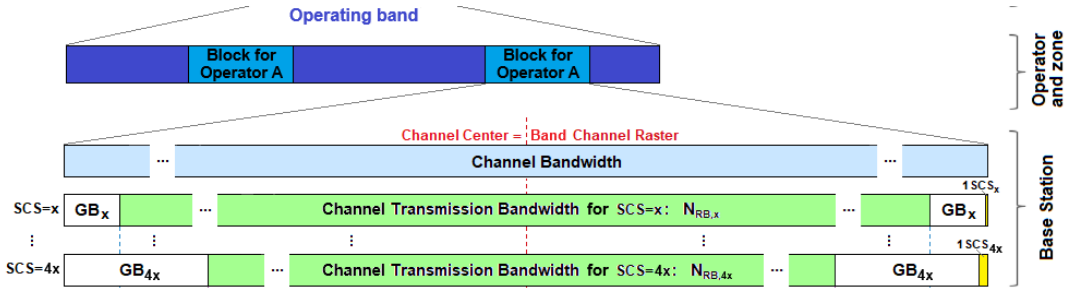
For more detail on **5G adjacent channel spacing**, refer to clause 5.4.1.1 of TS 38.101-**1**, **-2** and **-5**.

### 2.2.3.6 Channel transmission bandwidth and minimum guardband per "channel bandwidth and subcarrier spacing"

**Within each channel**, it is mandatory to leave a **guardband** (click 2.2.3.6.2) at **each** of its **edges** to reduce **inter-channel interference**. Therefore, the **channel transmission bandwidth** (click 2.2.3.6.1) is **always less** than the **channel bandwidth**.

Notice that **both** the **channel transmission bandwidth** and the **channel minimum guardband** depend on the **channel bandwidth** and on the **control/shared subcarrier spacing** in the **channel**.

The following figure shows the **channel transmission bandwidth** and the **channel minimum guardband** (at each **channel edge**) for **two** values ("**x**" and "**y**") of the **control/shared subcarrier spacing**:




Some **relevant remarks** on this figure are:

- A **generic band** (click 2.2.1) is **shown** as a **dark blue rectangle**.
- The **band** could have **TDD**, **FDD**, **SDL** or **SUL duplexing mode**. (click 2.2.1.2). If it were an **FDD band**, it would contain **two semi-bands** separated by a **guardband**.
- **Two frequency blocks** (click 2.2.2) licensed to **operator A** are **shown** as **two blue rectangles**.
- If the **band** had **FDD duplexing mode** (click 2.2.1.2), **each** of the **two frequency blocks** would be placed in **each** of the **band**'s **semi bands**. In this case, **one frequency block** would be for **Downlink** transmission and the **other** for **Uplink** transmission.
- The figure shows **only one channel** (2.2.3) configured in the **frequency block**, but in the general case it is possible to configure **more than one channel**.
- The **channel type** (**TDD**, **Downlink**, **Uplink**, **SDL** or **SUL**, click 2.2.3.1) **must** correspond to the **band duplexing mode** (**TDD**, **FDD**, **SDL** or **SUL**, click 2.2.1.2).
- The **channel bandwidth** (click 2.2.3) is shown as a **light blue rectangle**.
- The **channel bandwidth must** be one of the **allowed ones** in the **band** (taking into account the **channel**'s **intended control/shared subcarrier spacing(s)**) click 2.2.3.2.
- The **channel** is **exactly centered** in **one** of the **reference frequencies** of the **band channel raster** (taking into account the **channel**'s **intended control/shared subcarrier spacing(s)**), as it **must** (click 2.2.3.3).
- **Each** of the **channel**'s (**maximum**) **transmission bandwidths** (2.2.3.6) corresponding to **each** of the **channel**'s **control/shared subcarrier spacing(s)** "***x***" to **4x** is shown as a **light green rectangle**. It is expressed as an **integer quantity** ("$N_{RB,x}$" or "$N_{RB,4x}$") of **Resource Blocks** of the corresponding **subcarrier spacing**.
- **Each** of the **channel**'s (**minimum**) **guardbands** (click 2.2.3.6.2) for the corresponding **control/shared subcarrier spacing** (**x** to **4x**) is shown, at **each channel end**, as a **white rectangle** labeled as "$GB_x$" or "$GB_{4x}$". The **two guardbands** (**one** at **each channel edge**) for a **given control/shared subcarrier**

**spacing** have the **same bandwidth**. The **guardbands** in the figure are **not** drawn to **scale** (see below).

- The **control/shared subcarrier spacing** of **slack** (click 2.2.3.6.2) between the **channel bandwidth** and the **transmission bandwidth** plus the **two minimum guardbands** is shown, at the **upper channel edge**, as a **yellow rectangle** labeled "**1 $SCS_x$**" or "**1 $SCS_y$**".
- The **subindexes** "**x**" and "**4x**" indicate the **corresponding control/shared subcarrier spacing**.

#### 2.2.3.6.1 Channel transmission bandwidth per "channel bandwidth and subcarrier spacing"

The **channel transmission bandwidth** for each **pair** of values of **channel bandwidth** and **channel**'s **control/shared subcarrier spacing** is indicated in tables 5.3.2-1 of TS 38.101-**1** (**FR1 terrestrial**), of TS 38.101-**2** (**FR2**) and of TS 38.101-**5** (**FR1 satellite**).

The following is table 5.3.2-1 "*Maximum transmission bandwidth configuration $N_{RB}$*" of TS 38.101-**1**:

| | **Channel Bandwidth [MHz]** | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **SCS** | **5** | **10** | **15** | **20** | **25** | **30** | **35** | **40** | **45** | **50** | **60** | **70** | **80** | **90** | **100** |
| **[kHz]** | $N_{RB}$ | $N_{RB}$ | $N_{RB}$ | $N_{RB}$ | $N_{RB}$ | $N_{RB}$ | $N_{RB}$ | $N_{RB}$ | $N_{RB}$ | $N_{RB}$ | $N_{RB}$ | $N_{RB}$ | $N_{RB}$ | $N_{RB}$ | $N_{RB}$ |
| **15** | 25 | 52 | 79 | 106 | 133 | 160 | 188 | 216 | 242 | 270 | N/A | N/A | N/A | N/A | N/A |
| **30** | 11 | 24 | 38 | 51 | 65 | 78 | 92 | 106 | 119 | 133 | 162 | 189 | 217 | 245 | 273 |
| **60** | N/A | 11 | 18 | 24 | 31 | 38 | 44 | 51 | 58 | 65 | 79 | 93 | 107 | 121 | 135 |

The following is table 5.3.2-1 "*Maximum transmission bandwidth configuration $N_{RB}$*" of TS 38.101-**2**:

| | **Channel Bandwidth [MHz]** | | | | | | |
|---|---|---|---|---|---|---|---|
| **SCS** | **50** | **100** | **200** | **400** | **800** | **1,600** | **2,000** |
| **[kHz]** | $N_{RB}$ | $N_{RB}$ | $N_{RB}$ | $N_{RB}$ | $N_{RB}$ | $N_{RB}$ | $N_{RB}$ |
| **60** | 1,210 | 2,450 | 4,930 | N/A | N/A | N/A | N/A |
| **120** | 1,900 | 2,420 | 4,900 | 9,860 | N/A | N/A | N/A |
| **480** | N/A | N/A | N/A | 9,680 | 42,640 | 85,520 | N/A |
| **960** | N/A | N/A | N/A | 9,440 | 42,400 | 85,280 | 147,040 |

In these tables, the **channel transmission bandwidth** is expressed as an **integer quantity** of **Resource Blocks** (**RBs**), labeled in the table as "$N_{RB}$". The **RBs** used as **bandwidth units** have the **control/shared subcarrier spacing** indicated in the **first column** of the corresponding table **row**.

#### 2.2.3.6.2 Channel minimum guardband per "channel bandwidth and subcarrier spacing"

The **minimum guardband**, at **each channel edge**, is calculated for each allowed pair of **channel bandwidth** ($Ch_{BW}$) and **control/shared subcarrier spacing** (**SCS**) with the following expression:

$$\textbf{\textit{minimum guardband}} = (Ch_{BW} - N_{RB} \times \text{SCS} \times 12) / 2 - \text{SCS}/2$$

where $N_{RB}$ is the quantity of **Resource Blocks** of the **transmission bandwidth** (click 2.2.3.6.1) for this **same** pair of **channel bandwidth** ($Ch_{BW}$) and **control/shared subcarrier spacing**. This **expression** may be **reordered** as:

$$Ch_{BW} - (N_{RB} \times SCS \times 12 + 2 \times \textit{minimum guardband}) = SCS$$

$$Ch_{BW} - (\textit{transmission bandwidth} + 2 \times \textit{minimum guardband}) = SCS$$

This shows that there is **1 control/shared subcarrier spacing** of **slack** between the **channel bandwidth** and the **channel transmission bandwidth plus** its **two guardbands**. This **slack** is **located** at the **upper channel end** because the **centered usable CRB range** (click 2.2.3.4) is **not exactly centered** in the **channel**: it is **rather half** a **control/shared subcarrier spacing below** the **channel center**. This **slack** may be useful to implement the **required adjacent channel spacing** (click 2.2.3.5).

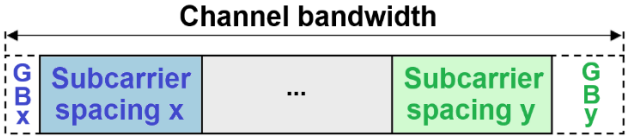
The values resulting from the **first expression** above are presented in tables **5.3.3-1** of TS 38.101-**1** (**FR1 terrestrial**), of TS 38.101-**2** (**FR2**) and of TS 38.101-**5** (**FR1 satellite**).

The following is table "*5.3.3-1: Minimum guardband for each **UE** channel bandwidth and SCS (kHz)*" of TS 38.101-**1**:

| | Channel Bandwidth [MHz] | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **SCS [kHz]** | **5** | **10** | **15** | **20** | **25** | **30** | **35** | **40** | **45** | **50** | **60** | **70** | **80** | **90** | **100** |
| **15** | 242.5 | 312.5 | 382.5 | 452.5 | 522.5 | 592.5 | 572.5 | 552.5 | 712.5 | 692.5 | N/A | N/A | N/A | N/A | N/A |
| **30** | 505 | 665 | 645 | 805 | 785 | 945 | 925 | 905 | 1,065 | 1,045 | 825 | 965 | 925 | 885 | 845 |
| **60** | N/A | 1,010 | 990 | 1,330 | 1,310 | 1,290 | 1,630 | 1,610 | 1,590 | 1,570 | 1,530 | 1,490 | 1,450 | 1,410 | 1,370 |

**Guardbands** for **different simultaneous subcarrier spacings**

When **simultaneous physical layer elements** have **different subcarrier spacings** (click 2.2.5.8) due to **SSB transmission**, the **minimum guardband** on **each channel edge must** be the one stated for the **subcarrier spacing** that is transmitted closer to **that channel edge**. The following figure illustrates this:




where, **GBx** and **GBy** represent the **minimum guardband** required (respectively) for **subcarrier spacing** "**x**" and "**y**" when transmitted across the **complete channel transmission bandwidth** (see tables above).

In **addition** to the previous requirement, the following also hold:

- For **FR1 bands**, in case the **channel bandwidth** is **larger** than **50 MHz**, the **minimum guardband** applied for a **15 kHz subcarrier** (i.e., when it is the one transmitted closer to the **channel edge**) **must** be the one corresponding to a subcarrier spacing of **30 kHz** for the corresponding **channel bandwidth**.
- For **FR2 bands**, in case the **channel bandwidth** is **larger** than **200 MHz**, the **minimum guardband** applied for a **60 kHz subcarrier** (i.e., when it is the one transmitted closer to the **channel edge**) **must** be the one corresponding to a subcarrier spacing of **120 kHz** for the corresponding **channel bandwidth**.

For more detail on **minimum guardband requirements** for **different simultaneous subcarrier spacings**, refer to clause 5.3.3 of TS 38.101-**1**, -**2** and -**5**.

#### 2.2.3.6.3 Example of channel transmission bandwidth and channel guardband configuration

This example continues from 2.2.3.4.

**Configuration example up to now**

From 2.2.1.3, this is b**and n257** is **FR2-1**, it is **TDD** and it spans from **26,500 MHz** to **29,500 MHz**. From 2.2.2.3, we have **one 100 MHz frequency block** from **27,650.08 MHz** to **27,750.08 MHz**. From 2.2.3.4, we have **one 100 MHz channel**, centered in the **frequency block**. The **channel NR-ARFCN** is **2,074,167**. The **channel** has a **control/shared subcarrier spacing** of **60 kHz** and an **SSB subcarrier spacing** of **120 kHz**.

**Configuration example in this section**

As indicated in 2.2.3.6.1, since **n257** is in **FR2**, we check table 5.3.2-1 of TS 38.101-**2** and we see that the entries for a **100 MHz channel** are:

| SCS [kHz] | 100 MHz $N_{RB}$ |
|---|---|
| **60** | 132 |
| **120** | 66 |

Since the previously configured **control/shared subcarrier spacing** is **60 kHz**, the **channel transmission bandwidth** is **132 RBs** (**95.04 MHz**).

Notice that, in the particular case of this example, **both options** of **control/shared subcarrier spacing** result in **exactly** the same **transmission bandwidth** (**95.04 MHz**) but this is **not always** the case.

As indicated in 2.2.3.6.2, the **channel minimum guardband** (at **each channel edge**) is computed with the formula:

$$\textbf{\textit{minimum guardband}} = (Ch_{BW} - N_{RB} \times \text{SCS} \times 12) / 2 - \text{SCS}/2$$

Since the previously configured **control/shared subcarrier spacing** is **60 kHz**, this results in **2.45 MHz**.

Notice that the **slack** between the **channel bandwidth** and the **channel transmission bandwidth** plus the **two channel guardbands** is **one control/shared subcarrier spacing**, this is, **60 kHz**. This **slack** is located at the **upper channel edge**.

This example continues in 2.2.4.2.

### 2.2.3.7 Asymmetric channel bandwidth

**Asymmetric channel bandwidth** consists of configuring a **different channel bandwidth** in the **Downlink transmission direction** in respect to the **Uplink transmission direction**.

The **narrower carrier must** be **confined** within the **frequency span** of the **wider channel**. “Confined within” means that:

- In **FDD bands**, the **duplex spacing must** be **larger or equal than** the one specified for the **band** (click 2.2.12).

- In **TDD bands**, the **narrower carrier must** be **completely contained** within the **frequency span** of the **wider channel**.

**All FR1-2**, **all FR2 bands**, **most symmetric FR1-1 FDD bands** and **all but one FR1-1 TDD bands must** be configured with **symmetric channels**. **All** the **FR1-1 asymmetric FDD bands**, a **few FR1-1 symmetric FDD bands** and **one FR1-1 TDD band** (**n50**) are **usually** configured with **asymmetric channels**. The **FR1-1 bands** that **may** be configured with **asymmetric channels** have been identified with note "**[B9]**" in the table from 2.2.13.1.

For **asymmetric FDD bands**, if the **Downlink semi-band** is **broader** than the **Uplink semi-band**, the **band** is said to have "**intra-band Supplementary Downlink**". In the **opposite case**, the **band** is said to have "**intra-band Supplementary Uplink**".

When the **wider channels** are located in the **higher frequency semi-band**, the **duplex-spacing increases** with the **increase** of the **center frequency** of the **channel pair**, and **vice versa**. Notice that the **duplex-spacing** (and the **self-interference guardband**) is **larger** in **asymmetric channel configuration** than in **symmetric channel configuration**.

The following table combines data for **FDD bands** that support **asymmetric channel bandwidth**, that have been taken from tables 5.2.1, 5.3.6-1, 5.4.4-1 of TS 38.101-**1**:

| FDD band | FDD band type | Uplink [MHz] | | Downlink [MHz] | | Duplex spacing [MHz] | UL channs. [MHz] | DL channs. [MHz] | Asymm. chann. BCS |
|---|---|---|---|---|---|---|---|---|---|
| **n5** | Symmetric | 824 | 849 | 869 | 894 | 45 | 20 | 25 | 0 |
| **n8** | Symmetric | 880 | 915 | 925 | 960 | 45 | 20 | 35 | 0 |
| | | | | | | | 10, 15, 20 | 25, 35 | 1 |
| **n24** | Symmetric | 1,626.5 | 1,660.5 | 1,525 | 1,559 | -101.5, -120.5 | 10 | 5 | |
| **n25** | Symmetric | 1,850 | 1,915 | 1,930 | 1,995 | 80 | 40 | 45 | |
| **n26** | Symmetric | 814 | 849 | 859 | 894 | 45 | 20 | 25, 30 | |
| **n66** | Symmetric | 1,710 | 1,780 | 2,110 | 2,200 | 400 | 5, 10 | 20, 40 | 0 |
| | | | | | | | 20 | 40 | |
| | | | | | | | 5, 10 | 20, 25, 30, 40 | 1 |
| | | | | | | | 20, 25, 30 | 40 | |
| **n70** | Symmetric | 1,695 | 1,710 | 1,995 | 2,020 | 300 | 5, 10 | 15 | 0 |
| | | | | | | | 5, 10, 15 | 20, 25 | |
| **n71** | Symmetric | 663 | 698 | 617 | 652 | -46 | 5 | 10 | 0 |
| | | | | | | | 10 | 15 | |
| | | | | | | | 15 | 20 | |
| | | | | | | | 5 | 10 | 1 |
| | | | | | | | 10 | 15 | |
| | | | | | | | 15 | 20 | |
| | | | | | | | 20 | 35 | |
| | | | | | | | 20 | 25, 30, 35 | 2 |
| **n91**[1] | Intra-b SUL | 832 | 862 | 1,427 | 1,432 | 570 – 595 | 10 | 5 | 0 |
| **n92**[1] | Intra-b SDL | 832 | 862 | 1,432 | 1,517 | μ = 0 => 575 to 680 μ = 1 => 580 to 675 | 5 | 10, 15, 20 | 0 |
| | | | | | | | 10 | 15, 20 | |
| **n93**[1] | Intra-b SUL | 880 | 915 | 1,427 | 1,432 | 527 to 547 | 10 | 5 | 0 |
| **n94**[1] | Intra-b SDL | 880 | 915 | 1,432 | 1,517 | μ = 0 => 522 to 632 μ = 1 => 527 to 627 | 5 | 10, 15, 20 | 0 |
| | | | | | | | 10 | 15, 20 | |
| **n105** | Symmetric | 663 | 703 | 612 | 652 | -51 | 20 | 25, 30, 35 | 0 |

Some **relevant remarks** on this table are:

- The **four bands** with super-index "**1**" require that:
  - **a)** The assignment of the paired **Uplink** and **Downlink** channels are subject to a **Tx-Rx** separation as specified in clause 5.4.4 of TS 38.306;
  - **b)** The range of **Tx-Rx** frequency separation given paired **Uplink** and **Downlink channel bandwidths** "$\boldsymbol{BW_{UL}}$" and "$\boldsymbol{BW_{DL}}$" is given (respectively) by the following **lower limit** and **upper limit**:

    $$\boldsymbol{F_{DL_low}} - \boldsymbol{F_{UL_high}} + 0.5(\boldsymbol{BW_{DL}} + \boldsymbol{BW_{UL}})$$

    $$\boldsymbol{F_{DL_high}} - \boldsymbol{F_{UL_low}} - 0.5(\boldsymbol{BW_{DL}} + \boldsymbol{BW_{UL}});$$

  - **c)** The **Uplink** and **Downlink channel bandwidth combinations** specified in Table 5.3.5-1 and 5.3.6-1 of TS 38.101-**1** depend on the **numerology** "$\boldsymbol{\mu}$".
- In the "**FDD band type**" column "Intra-b" stands for "Intra-band".
- The column headings "**UL channs.**" and "**DL channs.**" stand for **Uplink channel bandwidths** and **Downlink channel bandwidths**.
- The heading of the **last** column stands for **asymmetric channel bandwidth combination set**.
- **Symmetric FDD bands** (with the exception of **n24**) have a **single duplex-spacing** value. **Asymmetric FDD bands** have a **range** of **duplex-spacing** values. **Band n24** is **symmetric** and has **two duplex-spacing** values.

For **FDD bands**, it is **mandatory** for **UEs** to **support asymmetric bandwidth channel combination set 0** (i.e., **BCS0**) if there is an **asymmetric BCS0** defined for the **band**. (refer to TS 38.306).

The following table combines data for the **only TDD band** that supports **asymmetric channel bandwidth**, that have been taken from tables 5.2.1 and 5.3.6-2 of TS 38.101-**1**:

| TDD band | Uplink [MHz] | Downlink [MHz] | UL channs. [MHz] | DL channs. [MHz] |
|---|---|---|---|---|
| **n50** | 1,432 | 1,517 | 60 | 80 |

Some **relevant remarks** on this table are.

- The column headings "**UL channs.**" and "**DL channs.**" stand for **Uplink channel bandwidths** and **Downlink channel bandwidths**.
- **Both** the **center frequency** and the **BWP identifier** shall **match** between **Downlink** and **Uplink carriers** (refer to clause 6.3.2 of TS 38.331 and clause 12 of and TS 38.213).
- In case a **UE** is configured with a **full width bandwidth part** within **both Downlink** and **Uplink channels**, the **center frequency** of the **Downlink** and **Uplink channels** shall be the **same**.
- The **Downlink** and **Uplink CRB ranges** have **only one Point A** location (refer to clause 6.3.2 of TS 38.331).

For more detail on **asymmetric channel bandwidths**, refer to clause 5.3.6 of TS 38.101-**1**.

Source web pages:

- https://www.sqimway.com/nr_band.php#fdd_asymmetric

### 2.2.4 CRB range configuration of a cell's channel

**Each common resource block** (**CRB**) **range** is a quantity of **contiguous resource blocks** (**RBs**) for a given **numerology subcarrier spacing**, at a given **frequency location**. The **RBs** of a **CRB range** are called **common resource blocks** (**CRBs**). The **subcarriers** of a **CRB range** are called **common subcarriers** (**CSCs**). The **CRBs** and the **CSCs** of **each CRB range** are **indexed** starting **from 0** (**lowest frequency**) and **increasing** in **1** for **each CRB** or **CSC** (respectively) **upwards** in **frequency**.

**Each CRB range** can be characterized by its **subcarrier spacing** and the **location** of its **lowest frequency subcarrier**. **Common subcarrier 0** from **each and every** of the channel's **CRB ranges** is **centered** in a frequency location called "**Point A**" (click 2.2.4.3).

**Each usable CRB range** is a quantity of **contiguous CRBs**, that are frequency **aligned** with the **CRB range** of its **same subcarrier spacing**, and **all** of them are **within** the **channel transmission bandwidth**. Its **CRBs** and **CSCs** have the same **indexing** as the ones of its corresponding **CRB range**.

**Each usable CRB range** can be characterized by its **subcarrier spacing**, its **quantity** of **CRBs** and the **index** of its **lowest CRB**. Its **quantity** of **CRBs** is the **same** as the **channel transmission bandwidth** "$N_{RB,\mu}$" (click 2.2.3.6). **Each usable CRB range** is a **subset** of the **CRB range** with its same **subcarrier spacing**.

**The centered usable CRB range** is a **usable CRB range** whose **next-above-middle subcarrier** is **centered** in the **channel center** (i.e., in a **band channel raster reference frequency**).

**The centered usable CRB range** can be characterized by its **subcarrier spacing**, its **quantity** of **CRBs** and the **index** of its **lowest CRB**. The **usable CRB ranges** which are **not** the **centered** one are called the "**other centered CRB ranges**".

**Each cell**'s **channel** is structured by its **licensee entity** (usually an operator) by configuring **one centered usable CRB range**, its **Point A location** (which defines the **location** of **all** its **CRB ranges**) and, **possibly**, **other usable CRB ranges**. For the case of "$\mu$**=0**" or "$\mu$**=1**" it is also configured with **10** or **5 common resource block interlaces** (respectively).

This configuration may be performed in the following **order**:

1. **One centered usable CRB range** (click 2.2.4.1).

   For an example on configuring **one centered CRB range**, click 2.2.4.2.

2. **Point A location** (click 2.2.4.3) and **CRB ranges** (click 2.2.4.4).

3. **Possibly**, **other usable CRB ranges** (click 2.2.4.5).

For an example on **configuring Point A**, some **CRB ranges** and some **other usable CRB ranges**, click 2.2.4.6.

4. **CRB interlaces** (click 2.2.4.7).

   For the case of "$\mu$**=0**" or "$\mu$**=1**", **10** or **5 resource block interlaces** (respectively) are defined within **each CRB range**. These **interlaces** serve to provide **frequency diversity** (click 11.8) to some physical layer elements.

The **frequency location** of a **subcarrier** refers to the **center frequency** of the **subcarrier**.

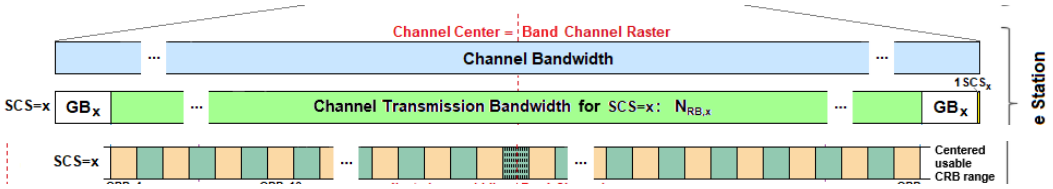
Be aware that the **UEs** do **not know** what the **usable CRB ranges** are. The **UEs** are just aware of the **CRB ranges** (as a **frequency reference**), of the **carriers** (click 2.2.5) and of the **bandwidth parts** (click 2.2.7) that the **NR node** configures in the **UEs**.

### 2.2.4.1 Centered usable CRB range configuration of a cell's channel

**Each cell**'s **channel** is configured by its **frequency block licensee entity** (usually an operator) with **one centered usable CRB range** (click 2.2.4).

In case the quantity "$N_{RB,\mu}$" of **CRBs** in the **centered usable CRB range** is **odd**, its **next-above-middle subcarrier** is the **subcarrier** with **RB index 6** within its **middle CRB**. In case the quantity "$N_{RB,\mu}$" of **CRB**s in the **centered usable CRB range** is **even**, its **next-above-middle subcarrier** is subcarrier with **RB index 0** within its **next-above-middle CRB**. This is expressed in a mathematical way in table 5.4.2.2-1 of TS 38.101-**1**, **-2** and **-5**. Recall that **subcarriers** are indexed **0** (**lowest** frequency) to **11** (**highest** frequency) within **each Resource Block**.

The following figure shows a **cell**'s **channel** and its **centered usable CRB range**:




Some **relevant remarks** on this figure are:

- The **channel type** (**TDD**, **Downlink**, **Uplink**, **SDL** or **SUL**, click 2.2.3.1) **must** correspond to the **band duplexing mode** (**TDD**, **FDD**, **SDL** or **SUL**, click 2.2.1.2).
- The **channel bandwidth** (click 2.2.3) is shown as a **light blue rectangle**.
- The **channel** is **exactly centered** in **one** of the **reference frequencies** of the **band channel raster**, as it **must**.
- The figure **only shows** the **channel maximum transmission bandwidth** of **subcarrier spacing** "**x**" of the **centered usable CRB range**. (click 2.2.4.1).
- **Each rectangle** (alternating green and mauve) represents one **CRB** from the **centered usable CRB range**.
- The **quantity** of **CRBs** of this **centered usable CRB range** is assumed to be **odd**. This is why its **next-above-middle subcarrier** is **subcarrier** with **RB index 6** within its **central CRB**. The said **subcarrier** is **exactly centered** in the **channel center** (red dashed line), as it **must**.
- The **black dashed lines** within a given **CRB** represent the divisions between its **12 subcarriers**.

- **Lowest frequency** is **leftmost**, and **frequency increases left** to **right**.
- The **subcarriers** and **CRBs** are drawn to **scale**: **each horizontal pixel** represents **half** the **subcarrier spacing** "**x**" (i.e., "**x/2**"). **Half** of the **frequency** corresponding to **each 1-pixel diving line** between **two contiguous elements** (e.g., **two subcarriers** or **one subcarrier** and **nothing**) belongs to **each element** (i.e., "**x/4**" **belongs** to **each element**).

### 2.2.4.2 Example of centered usable CRB range configuration

This example continues from 2.2.3.6.3.

**Configuration example up to now**

From 2.2.1.3, this is b**and n257** in **FR2-1**, it is **TDD** and it spans from **26,500 MHz** to **29,500 MHz**. From 2.2.2.3, we have **one 100 MHz frequency block** from **27,650.08 MHz** to **27,750.08 MHz**. From 2.2.3.4, we have **one 100 MHz channel**, **centered** in the **frequency block**. The **channel NR-ARFCN** is **2,074,167**. The **channel** supports **subcarrier spacings** of **60 kHz** and **120 kHz**. From 2.2.3.6.3, its **transmission bandwidth** is **132 RBs** for **60 kHz SCS** and **66 RBs** for **120 kHz SCS**.

**Configuration example in this section**

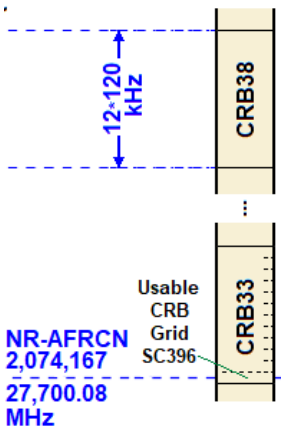
For this example, we **choose** to configure the **centered usable CRB range** with **120 kHz subcarrier spacing**, and **not** the one with **60 kHz**.

As indicated in 2.2.3.4, we have to place the **next-above-middle subcarrier** of the **120 kHz usable CRB range** in the **channel center**.

Based on the channel configuration, the **120 kHz centered usable CRB range** has **66 CRBs**. Since the **centered usable CRB range** has an **even** quantity (**66**) of **CRBs**, its **next-above-middle subcarrier** is **subcarrier 0** from its **next above-middle CRB**, which is its **34th CRB** (counting its **lowest frequency CRB** as the **1st** one). This **subcarrier must** be **centered** in the **channel center**, which is **27,700.08 MHz**.

The **CRB index** of **these CRBs** is **not known** at this stage, because **Point A** has **not** yet been **configured**. For the figure, let us **suppose** that the **lowest frequency CRB** has **CRB index 0**. Consequently, the **next-above-middle CRB** has **CRB index 33** and the **next-above-middle subcarrier** has **common subcarrier index 396**.

The following figure shows the **frequency location** of the **centered usable CRB range** of this example:



You may do the calculation to check that there is **one subcarrier spacing** (**60 kHz**) of **slack** between the **upper edge** of the **upper guardband** and the **channel upper edge**.

This example continues in 2.2.4.6.

### 2.2.4.3 Point A location

**Point A** is a **reference frequency** from the **global frequency raster** (click 2.2.15) that is the **location** of **common subcarrier 0** of **all** the **CRB ranges** (click 2.2.4) in the **set** of **CRB ranges** (click 2.2.4.4), for a given **channel**. Consequently, **every channel** has **1** and **only 1 Point A**. In **FDD bands**, **each channel** in the **channel pair** has its **Point A**. For **TDD bands**, **each TDD channel** has its **only Point A**. For **SUL bands**, **each SUL channel** has its **Point A**.

Recall from 2.2.4 that the **CRBs** from **every usable CRB range** have the same **CRB indexing** as the **CRB range** with its **same subcarrier spacing**. Since the **CRB index cannot** be **negative**, the **highest frequency location** for **Point A** is the **center** of the **lowest frequency subcarrier** of the **centered usable CRB range**. **Point A** may be placed at the **said location**, or **below** it with a separation of an **integer quantity** of **RBs**. of the **subcarrier spacing** of the **centered usable CRB range**. Consequently, the **location** of the **centered usable CRB range** (click 2.2.4.1) determines the **possible locations** for **Point A**.

The **locations** of **Point A** can therefore be:

$$\textbf{Point A} = f_{SC0}^{L} - j \cdot 12 \cdot 2^{\mu} \cdot 15 \text{ kHz}, \text{ for } j \in \{0 \text{ to } max\}$$

Where "$f_{SC0}^{L}$" is the **center** of the **lowest frequency subcarrier** of the **centered usable CRB range**, and "**μ**" is the **numerology** of the **centered usable CRB range**.

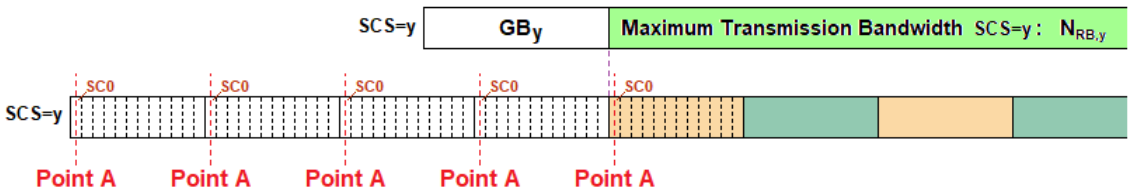
The **next-above middle subcarrier** "$f_{SC0}^{C}$" of the **centered usable CRB range** is spaced away from "$f_{SC0}^{L}$" an **integer number** "$n_{CRBs}$" of **CRBs**. Consequently:

$$f_{SC0}^{L} = f_{SC0}^{C} - n_{CRBs} \cdot 12 \cdot 2^{\mu} \cdot 15 \text{ kHz}$$

$$\textbf{Point A} = f_{SC0}^{C} - (j + n_{CRBs}) \cdot 3 \cdot 2^{\mu} \cdot 60 \text{ kHz}, \text{ for } j \in \{0 \text{ to } max\}$$

Since $f_{SC0}^{C}$ is located in a **band channel raster frequency** (which **all** belong to the **global frequency raster**) and the **reference frequencies** from the **global frequency raster** are spaced away in **either 5**, **15** or **60 kHz**, it follows that **all** the possible **locations** of **Point A** that have been indicated above belong to the **global frequency raster**.

The following figure shows an example of a **centered usable CRB range** as a series of mauve and green rectangles, and **5** of the **possible locations** for **Point A**:




Some **relevant remarks** on this figure are:

- **Lowest frequency** is **leftmost**, and **frequency increases** left to right.
- The **subcarriers** and **CRBs** are drawn to **scale**: **each horizontal pixel** represents **1/8** of **subcarrier spacing**. **Half** of the **frequency** corresponding to **each 1-pixel diving line** between **two contiguous elements** (e.g., **two subcarriers** or **one**

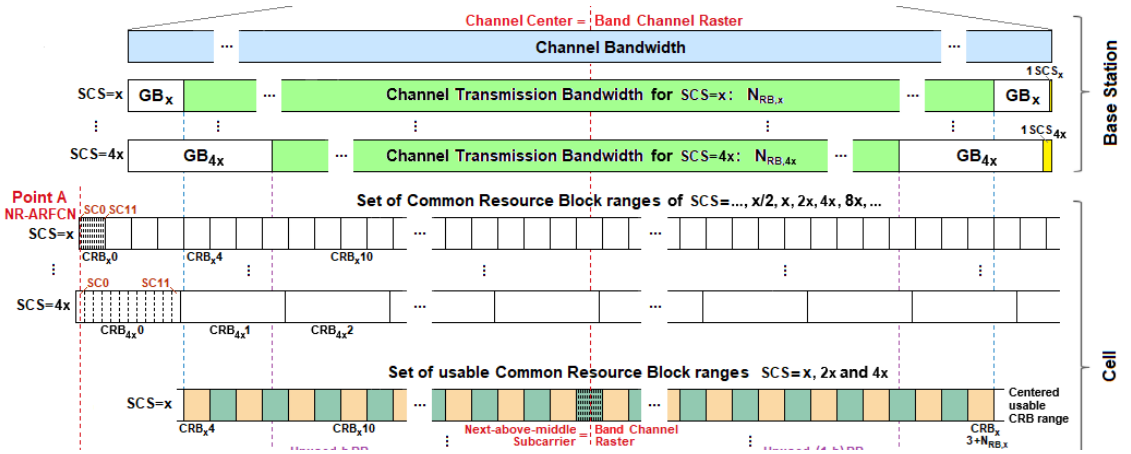
**subcarrier** and **nothing**) belongs to **each element** (i.e., **1/16** of **subcarrier spacing belongs** to **each element**).

Notice that **Point A** may be placed **below** the **lower edge** of the **channel transmission bandwidth** for the corresponding **subcarrier spacing**. Therefore, a **certain range** of the **lowest CRBs** (i.e., **CRB0** to **CRBx**) of a given **subcarrier spacing may** be placed **outside** of the **channel transmission bandwidth** for that same **subcarrier spacing**.

The **chosen location** of **Point A** determines the **frequency location** of **all CRB ranges**, and consequently, of **all other usable CRB ranges**, in respect to the **centered usable CRB range**. Since **all** the **CRB ranges** have **common subcarrier 0 centered** in **Point A**, once **Point A** is configured, the **locations** of **all** the **CRB ranges** and of **all** the other **usable CRB ranges** are determined. The location of **Point A** also determines the **CRB indexing** and the **CSC indexing** of **every CRB range**. The **lowest frequency CRB** (the one **located** in **Point A**) of **each CRB range** has **CRB index 0**, and **each successive CRB upwards** increments the **CRB index** in **1**. The **lowest frequency CSC** (the one **centered** in **Point A**) of **each CRB range** has **CSC index 0**, and **each successive CSC upwards** increments the **CSC index** in **1**.

Notice that **TDD channels only** have **1 Point A**, and **1 set** of **CRB ranges**, which is the reference for **both** its **Downlink** and its **Uplink transmission directions**.

The following figure shows an example of the location of **Point A**, and the corresponding **location** of the **CRB ranges**.




Some **relevant remarks** on this figure are:

- Among the **possible** locations for **Point A**, it has been placed in the **4th CRB below** the **lowest CRB** of the **centered usable CRB range**.
- **Each white rectangle** represents one **CRB** from the **CRB range** of the corresponding **subcarrier spacing**.
- The **location** of **Point A** has determined the **CRB indexing** of **each CRB range**.
- **Each** alternating green and mauve **rectangle** represents one **CRB** from the **centered usable CRB range**. Notice that **CRBs** from the **centered usable CRB range** have the **same indexing** as the ones from the **CRB range** (click 2.2.4.4) that has its **same subcarrier spacing**, as they **must**.
- **All** the **CRBs** from the **centered usable CRB range** fall **within** the **channel transmission bandwidth** for that same **subcarrier spacing**, as they **must**.

- The **quantity** of **CRBs** of this **centered usable CRB range** is assumed to be **odd**. This is why its **next-above-middle subcarrier** is **subcarrier 6** from its **central CRB**. The said **subcarrier** is **exactly centered** in the **channel center** (red dashed line), as it **must**.
- There is **one CRB range** for **each numerology subcarrier spacing** that is **allowed** in the **channel**. In the figure, this is represented as a **set** of **3 CRB ranges** with **subcarrier spacing** from "**x**" to "**4x**".
- The **black dashed lines** within a given **CRB** represent the divisions between its **12 subcarriers**.
- The **subcarriers** and **CRBs** are drawn to **scale**: **each horizontal pixel** represents **1/2** of the **subcarrier spacing**. **Half** of the **frequency** corresponding to **each 1-pixel diving line** between **two contiguous elements** (e.g., **two subcarriers** or **one subcarrier** and **nothing**) belongs to **each element** (i.e., **1/4** of **subcarrier spacing belongs** to **each element**).

For an example of **Point A location**, click 2.2.4.6.

For more detail on **Point A**, refer to clause 4.4.4.2 of TS 38.211.

For more detail on **CRB ranges**, refer to clause 4.4.4.3 of TS 38.211.

Source web page:

- https://www.sharetechnote.com/html/5G/5G_FrameStructure.html
- https://www.sharetechnote.com/html/5G/5G_ResourceBlockIndexing.html

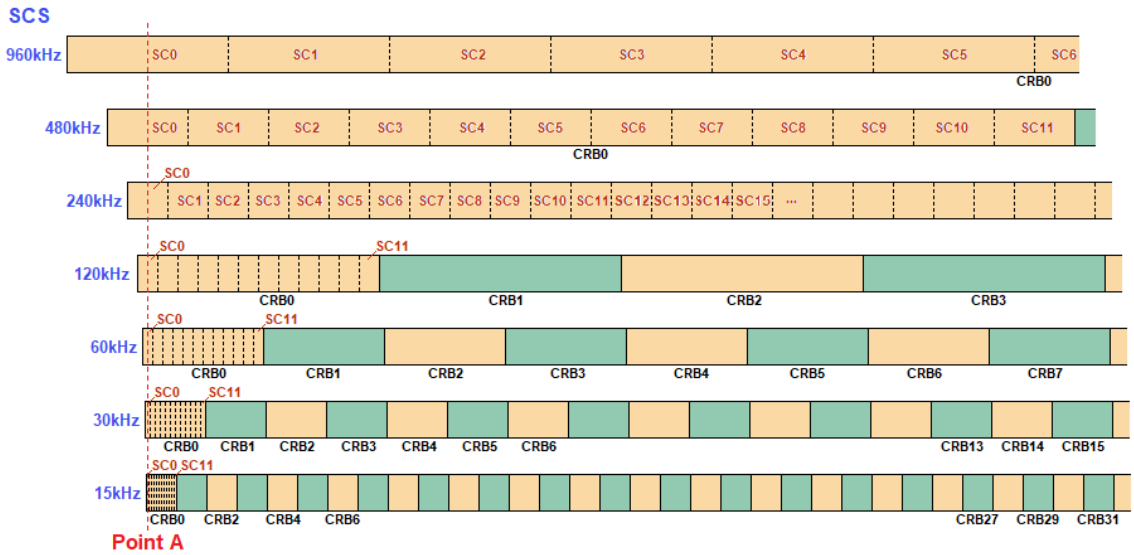
### 2.2.4.4 Set of CRB ranges

When **Point A location** (click 2.2.4.3) is **configured** in a **channel** this also determines the **location** of the **channel**'s **set** of **CRB ranges** (click 2.2.4). The **channel**'s **set** of **CRB ranges** contains **one CRB range** for **each different numerology subcarrier spacing** (click 2.2.10.3) that is **allowed** in the **channel** (click 2.2.3).

There are **no CRB ranges** for the **PRACH-specific subcarrier spacings**. The reason is that the **PRACH subcarriers** with **PRACH-specific subcarrier spacing** are **not aligned** with the **subcarriers** of a **CRB range**. For more detail, click 7.3.1.

In 2.2.4 and 2.2.4.3 it was **stated** that **each** and **every CRB range** has the **center** of its **common subcarrier 0** placed at **Point A**. This implies that the **edges** of the **common subcarriers** and of the **CRBs** belonging to **different CRB ranges** are **never aligned** in frequency (see further below).

The following figure shows one example of **Point A** location, and **all** the **possible CRB ranges** (that may be allowed in **any** channel) with their **common subcarrier 0 centered** in **Point A**.




Some **relevant remarks** on this figure are:

- **Lowest frequency** is **leftmost**, and **frequency increases** left to right.
- The **CRB ranges** for **all** the **numerology subcarrier spacing values** are shown, although in any given **channel only a subset** of them can be used.
- A complete **CRB** with **960 kHz subcarrier spacing** is **not shown**, because the figure is **not wide enough**, and it therefore only shows a few subcarriers within the **960 kHz CRB**.
- The **CRB range** with **240 kHz subcarrier spacing** is shown **without resource blocks**, because the **240 kHz subcarrier spacing** may **only** be used for the **SSB**, and the **SSB RBs** do **not** have to be **aligned** with the **CRBs** of the **240 kHz CRB range**. However, the **SSB subcarriers must** be **aligned** with **Point A**, and therefore, the **SSB subcarriers must** be **aligned** with the **subcarriers** of the corresponding **CRB range**, including the **240 kHz** one.
- The figure does **not** make any reference to the **usable CRB ranges**: it **only shows** the **CRB ranges**.
- The **subcarriers** and **CRBs** are drawn to **scale**: **each horizontal pixel** represents **7.5 kHz**. **Half** of the **frequency** corresponding to **each 1-pixel diving line** between **two contiguous elements** (e.g., **two subcarriers** or **one subcarrier** and **nothing**) belongs to **each element** (i.e., **3.75 kHz belongs** to **each element**).

The **CRB index** and the **common subcarrier index** (for each **CRB range**) are a **reference** used by the **NR node** to **convey** to the **UEs** the **frequency location** of different **physical layer elements** (**carriers**, bandwidth parts, **CORESETs**, **Resource Element** and **Layer**s, …).

There are **five mutually exclusive types** of **CRB ranges** (**exactly** the **same** as the **channel types**): **SUL**, **Downlink**, **Uplink**, **TDD** and **SDL**. The **type** of every **CRB** is the **same** as the **channel type** of its **channel**.

For the **relationship** between the **types** of **bands** (**duplexing mode**), of **channels**, of **CRB ranges**, of **carriers**, of **bandwidth parts** and the **transmission direction**, click 1.1.3.

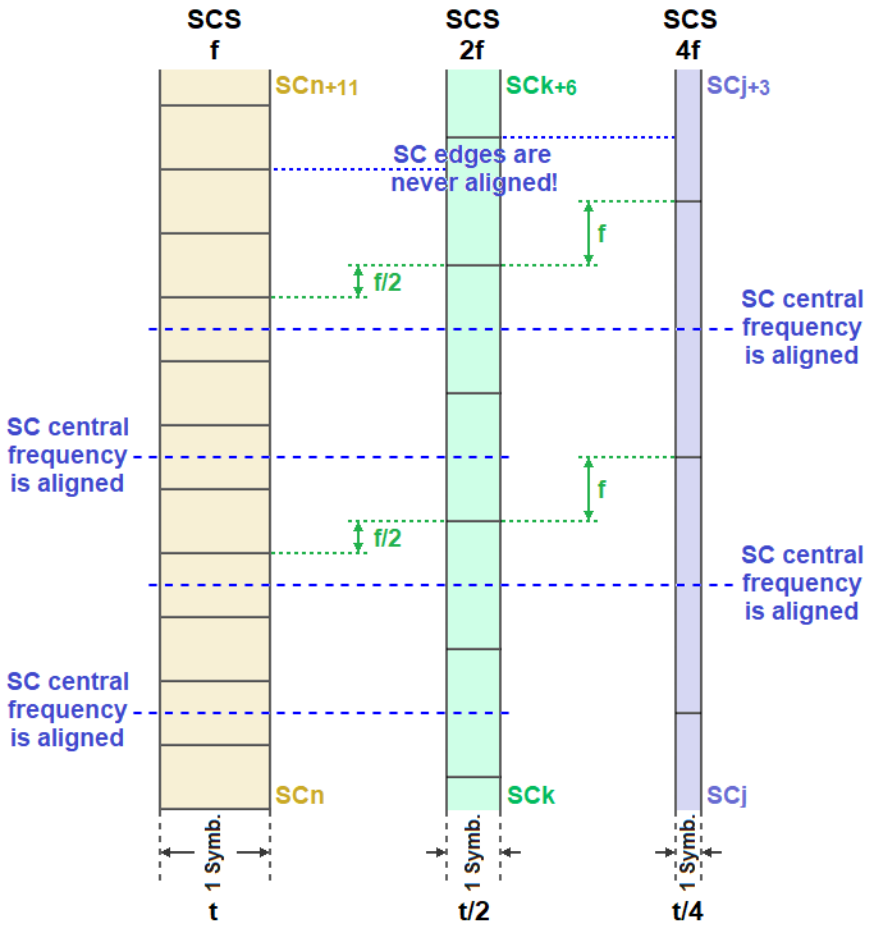
For an example of **CRB ranges location**, click 2.2.4.6.

**(Mis)alignment of CSCs and CRBs with different numerologies**

Since **common subcarrier 0 each and every CRB range** is centered in **Point A**, **every two CRB ranges** with **numerologies** $\mu_a$ and $\mu_b$ (with $\mu_b>\mu_a$) have:

- **Aligned subcarrier centers** every $2^{\mu b-\mu a}$ **subcarriers** of **numerology** $\mu_a$, starting from **common subcarrier 0**.
- **Aligned subcarrier centers** of the **subcarrier 0** of their **CRBs** every $lcm(12, 2^{\mu b-\mu a})$ **CRBs** of **numerology** $\mu_a$, starting from **CRB0**. The function "*lcm*()" is the **least common multiple**.
- **Misaligned subcarrier edges** in $15\cdot 2^{\mu a}\cdot(2^{-1}+2^{0}+2^{1}+\ldots+2^{\mu b-\mu a-2})$ **kHz**, (notice that the **summation** has $\mu_b-\mu_a$ **terms**). The same applies to **CRB edges**.

The following figure illustrates the said statements that refer to **subcarrier centers** and **edges**:




For more detail on **CRB ranges** and **Point A**, refer to clause 4.4.4.3 of TS 38.211.

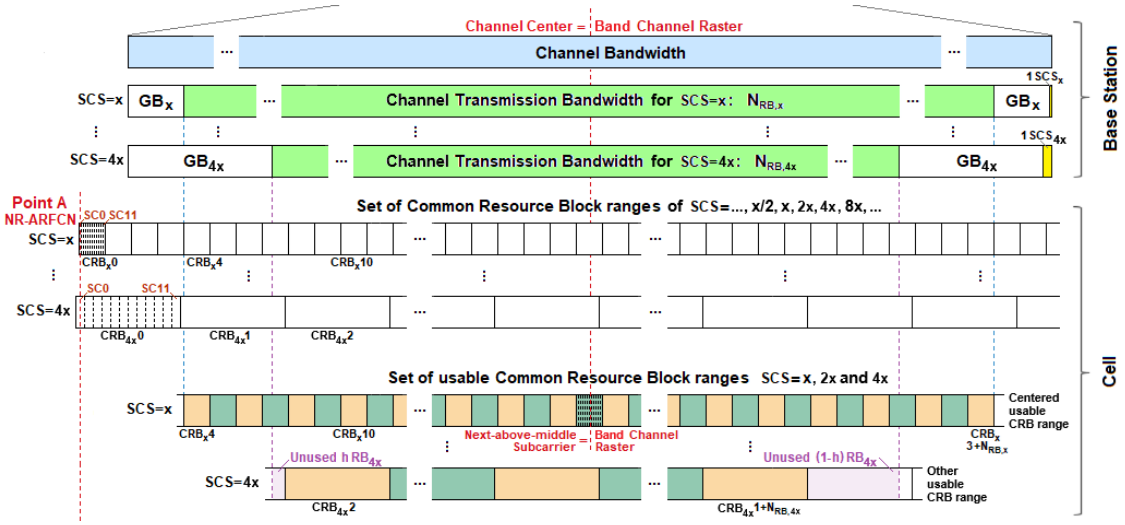
### 2.2.4.5 Other usable CRB ranges

In addition to the **mandatory centered usable CRB range** (click 2.2.4) there is **one other usable CRB range** for **each** of the **other control/shared subcarrier spacings** (click 2.2.3.2) that are configured in the **channel**.

Once **Point A** has been configured (click 2.2.4.3), it determines the **location** and **size** of **all** the **usable CRB ranges other** than the **centered** one (that has been previously configured, click 2.2.4).

Notice that the **other usable CRB ranges may**, **or may not**, have their **next-above-middle subcarrier centered** in the **channel**'s **center frequency**.

The following figure shows an example of the location of **Point A**.

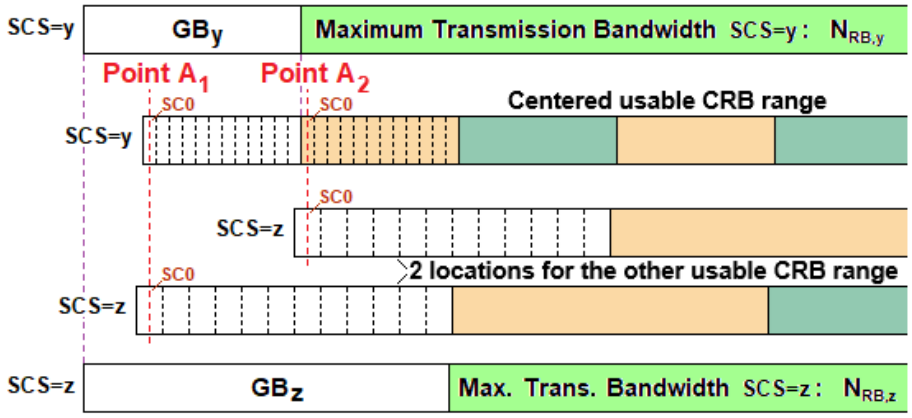



Some **relevant remarks** on this figure are:

- In the example shown in this figure, the **other usable CRB range** with **SCS=4x** does **not** have its **next-above-middle subcarrier centered** in the **channel center**.
- In the example shown in this figure, there are **three usable CRB ranges** (with **SCS** equal "**x**", "**2x**" and "**4x**"), although it is more frequent to have **only** one or **two**.
- In the example shown in this figure, **CRB0** is **below** the **lowest frequency CRB** of the **centered usable CRB range**.
- In the example shown in this figure, the **other usable CRB range** with **SCS=4x loses one CRB** from the **allowed maximum transmission bandwidth** for this **SCS**. This **lost CRB** is always **split** in **two** (as shown in the figure). This **lost** CRB colored in **purple** and labeled as "**Unused**" in the figure.
- Notice how the **location** of **Point A** has determined the **location** of the **other usable CRB ranges** (the one with "**SCS=4x**").

The chosen location for **Point A** may cause **losing one CRB** in some of the **other usable CRB ranges**. **Losing** a **CRB** means that **a usable CRB range** has **one less CRB** than the ones of the **channel transmission bandwidth** for that **subcarrier spacing**. The following

figure shows how the quantity of **CRBs** of the **usable CRB range** with **SCS=z** changes with the location of **Point A**, such that **one CRB** may be lost.




Some **relevant remarks** on this figure are:

- The **centered usable CRB range** is the one with **subcarrier spacing** "**y**".
- The selection of the **centered usable CRB range** determines the **possible locations** of **Point A**. In this example, only **two** possible locations are shown: **Point A1** and **Point A2**.
- The **other centered usable CRB range** is the one with **subcarrier spacing** "**z**". The value of "**z**" is **twice** the one of "**y**".
- **Point A1** and **Point A2** determine **2 possible locations** for the **other usable CRB range**.
- Selecting **Point A1** allows to **fit <u>all</u>** the $N_{RB,z}$ **CRBs** of the **other usable CRB range** (mauve and green rectangles of the **second other usable CRB range**) **within** the **frequency location** of the **maximum transmission bandwidth** for **SCS=z** (light green rectangle at the bottom). Therefore, **no CRB** from the **maximum transmission bandwidth** $N_{RB,z}$ would be **lost.**
- Selecting **Point A2** does **not** allow to **fit <u>all</u>** the $N_{RB,z}$ **CRBs** of the **other usable CRB range** (only the initial mauve rectangle of the **first other usable CRB range** is shown) **within** the **frequency location** of the **maximum transmission bandwidth** for **SCS=z** (light green rectangle at the bottom). Therefore, **one CRB** from the **maximum transmission bandwidth** $N_{RB,z}$ would be **lost.**
- The **subcarriers** and **CRBs** are drawn to **scale**: **each horizontal pixel** represents **1/8** of the **subcarrier spacing** "**y**". **Half** of the **frequency** corresponding to **each 1-pixel diving line** between **two contiguous elements** (e.g., **two subcarriers** or **one subcarrier** and **nothing**) belongs to **each element** (i.e., **1/16** of **subcarrier spacing** "**y**" **belongs** to **each element**).

For an example of **other usable CRB ranges location**, click 2.2.4.6.

Source web pages:

- https://www.nrexplained.com/rbs
- https://5g.smarttelecomedu.com/common-resource-blocks-physical-resource-blocks/
- https://www.sharetechnote.com/html/5G/5G_ResourceGrid.html

### 2.2.4.6 Example of Point A configuration: location of CRB ranges and of other usable CRB ranges

This example continues from 2.2.4.2.

**Configuration example up to now**

From 2.2.1.3, this is b**and n257** in **FR2-1**, it is **TDD** and it spans from **26,500 MHz** to **29,500 MHz**. From 2.2.2.3, we have **one 100 MHz frequency block** from **27,650.08 MHz** to **27,750.08 MHz**. From 2.2.3.4, we have **one 100 MHz channel**, centered in the **frequency block**. The **channel NR-ARFCN** is **2,074,167**. From 2.2.3.4, the **channel** supports **subcarrier spacings** of **60 kHz** and **120 kHz**. From 2.2.3.6.3, the **transmission bandwidth** is **132 RBs** for **60 kHz SCS** and **66 RBs** for **120 kHz SCS**. From 2.2.4.2, **subcarrier 0** from the **next-above-middle CRB** of the **centered usable CRB range** is **centered** at **27,700.08 MHz** and its **subcarrier spacing** is **120 kHz**. This **subcarrier 0** from the **next-above-middle CRB** is the **397$^{th}$ subcarrier** (counting the **lowest frequency subcarrier** as the **1$^{st}$** one).

**Configuration example in this section**

**Subcarrier 0** from the **centered usable CRB range** is **centered** at **27,700.08 MHz** minus **396** times **120 kHz**, which is **27,652.56 MHz**. As indicated in 2.2.4.3, this is the **highest possible location** for **Point A**. In this case, the **subcarrier spacing** of the **centered usable CRB range** is **120 kHz**, and therefore:

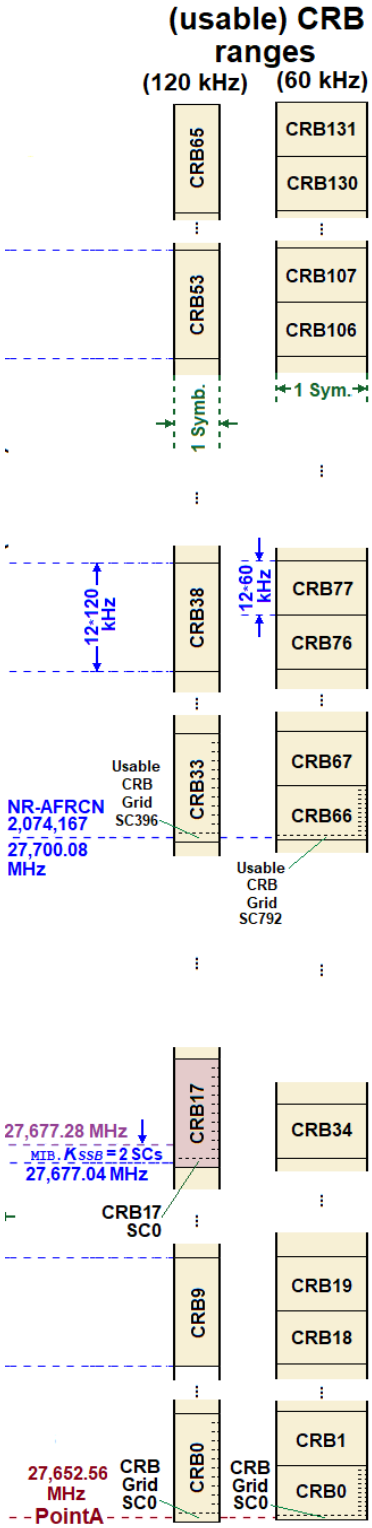
**Point A** = **27,652.56 MHz – j•12•120 kHz**, for **j** $\in$ {**0** to **max**}

**Point A** = **27,652.56 MHz**, **27,651.12 MHz**, **27,649.68 MHz**, **27,648.24 MHz**, etc.

We choose to place **Point A** at **27,652.56 MHz**. Let us check that this location does belong (as is **proven** in 2.2.4.3) to the **global frequency raster**. Using a **NR-ARFCN** web calculator, we see that this location of **Point A** belongs to the **global frequency raster** and its **NR-ARFCN** is **2,073,375**.

The location of **Point A** determines the **location** of **all** the **CRB ranges** and of **all other usable CRB ranges**. The following figure shows the **60 kHz** and the **120 kHz CRB**

**ranges**, and the **location** of the **other usable CRB range**, which is the **60 kHz usable CRB range**. The following figure shows this:




Some **relevant remarks** on this figure are:

- In the case of this example, **both** usable **CRB ranges** have their **next-above-middle subcarrier** centered in the **channel**'s **center frequency**, but this is **not allways** the case.

- In the case of this example, because **both usable CRB ranges** have their **next-above-middle subcarrier** centered in the **channel**'s **center frequency**, both of them have the **maximum transmission bandwidth** (the **quantity** of **RBs** indicated in tables 5.3.2-1, click 2.2.3.4), but this is **not allways** the case.
- In the case of this example, we have **chosen** the location of **Point A** to be the **center frequency** of **subcarrier 0** from the **lowest frequency CRB** of the **centered usable CRB range**. However, in the general case **Point A** may be **below** (and therefore **outside**) of the **centered usable CRB range**.
- In the case of this example, the **lowest frequency CRB** of the **usable CRB ranges** is also the **lowest frequency CRB** (i.e., **CRB0**) from their corresponding CRB ranges. However, in the general case **CRB0** may be **below** the **lowest frequency CRB** of its corresponding **usable CRB range**.

This example continues in 6.1.2.1.3.

### 2.2.4.7 Common resource block interlaces

A **CRB interlace** is a **sequence** of **non-contiguous CRBs**, **evenly spaced** every "$M_\mu$" **CRBs**, where subindex "$\mu$" is the **subcarrier spacing** of the **CRB range**.

Consequently, "$M_\mu$-1" is the **quantity** of **CRBs** between **two consecutive RBs belonging** to the same **CRB interlace**. Another consequence is that there are "$M_\mu$" different **interlaces**.

**CRB interlaces** are defined for **frequency diversity** purposes (e.g., for **PUCCH-PUSCH interlace**, click 2.2.4.8). **CRB interlaces** are only defined for numerologies "$\mu$=**0**" (**15 kHz**) and "$\mu$=**1**" (**30 kHz**).

The value of "$M_\mu$" is **10** for "$\mu$=**0**" (**15 kHz subcarrier spacing**) and it is **5** for "$\mu$=**1**" (**30 kHz subcarrier spacing**).

Each of the "$M_\mu$" different **CRB interlaces** is identified by an **interlace index** "$m$", where "$0 \leq m \leq M_\mu - 1$".

**Interlace** "$m$" is the **sequence** of **CRBs** whose **CRB index** "$n_{CRB}^{\mu}$" complies with:

$$n_{CRB}^{\mu} = m + n_{IRB,m}^{\mu} \cdot M_\mu, \text{ for } n_{IRB,m}^{\mu} = 0, 1, 2, \ldots$$

Therefore, "$m$" is the **CRB index** "$n_{CRB}^{\mu}$" of the **lowest frequency CRB** in **interlace** "$m$" and "$n_{IRB,m}^{\mu}$" is the **interlace RB index** (with **step size 1**) of **CRBs** within **interlace** "$m$". The **interlace RB index** $n_{IRB,m}^{\mu}$ is increased in **1** for **each CRB** in **interlace** "$m$", going **upwards** in frequency. The **interlace RB index** "$n_{IRB,m}^{\mu} = 0$" corresponds to the **lowest frequency CRB** in **interlace** "$m$", which has **CRB index** "$n_{CRB}^{\mu}=m$".

Notice that the **CRBs** (click 2.2.4.3), the **CRBs** from **CRB interlaces**, the **PRBs** (click 2.2.7.1) and the **VRBs** (click 2.2.7.1) that have the **same subcarrier spacing** are **all** exactly **aligned** in the **frequency domain**.

A **CRB interlace** is sometimes called a "**resource block interlace**" or just an "**interlace**".

Do **not confuse** the **CRB interlaces** with the (**non**)**interleaved mapping** of **CORESET**'s **CCEs** (click 6.8.1.1).

For more detail on **CRB interlaces**, refer to clause 4.4.4.6 of TS 38.211.

### 2.2.4.8 PUCCH-PUSCH interlace

**PUCCH-PUSCH interlace** consists of **organizing** the **frequency domain** into **multiple CRB interlaces** (click 2.2.4.7) **each** comprising a **set** of **spaced CRBs** such that the **interlaces** do **not overlap**.

**Each UE** is **assigned 1 unique interlace** for its **PUCCH transmissions** and **1 unique interlace** for its **PUSCH transmissions**. This implies that the **PUCCH transmission** and the **PUSCH transmission** of **each UE** are **interlaced** in the **frequency domain**, and that the **PUCCH/PUSCH transmissions** of **different UEs** are also **interlaced** in the **frequency domain**.

**All** the **UEs** in a given **cell** are **configured** with **PUCCH-PUSCH interlace** when `useInterlacePUCCH-PUSCH-r16` is **present** in the `BWP-UplinkCommon` (click 2.2.8.9) **instance** of **that** given **cell**. In case that `useInterlacePUCCH-PUSCH-r16` from `BWP-UplinkDedicated`, (click 2.2.8.10) is **present** in a **UE** BWP configuration, the **UE** expects that **PUCCH-PUSCH interlace** is **either set or not set** in **all Uplink** (or **SUL**) **bandwidth parts** of the **same serving cell**. Consequently, the **same PUCCH-PUSCH interlace configuration** applies to **all** the **PUCCH transmissions** (i.e., to **all** the **PUCCH resources**, click 7.6.5) and to **all** the **PUSCH transmissions** (click 7.11.3) in **each serving cell**.

### 2.2.4.9 FDD Downlink/Uplink 7.5 kHz subcarrier grid upshift

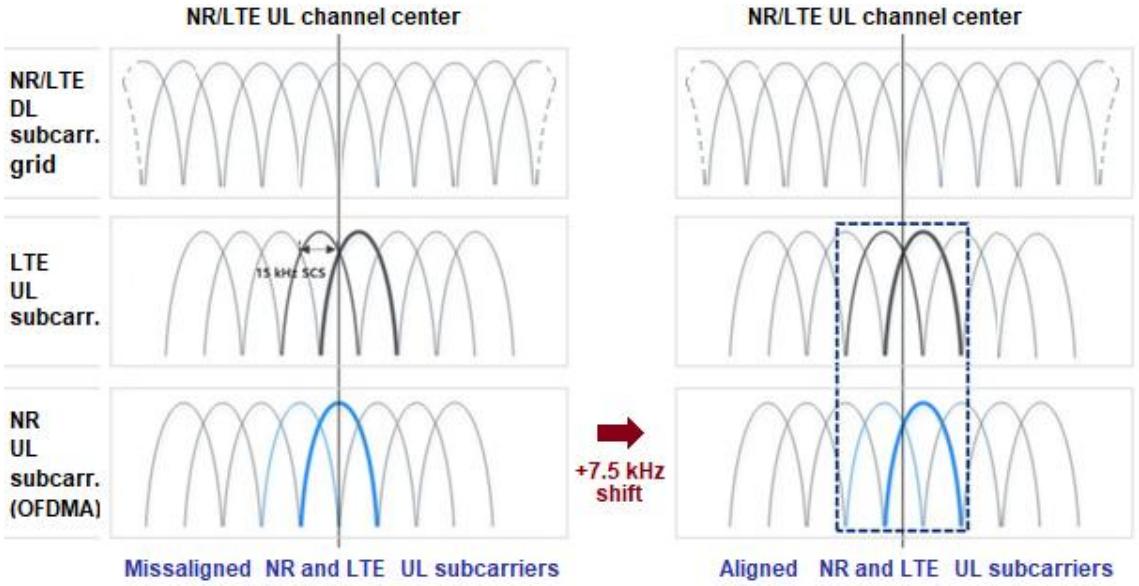
In **5G FDD**, the **subcarriers** of its **Uplink** and **Downlink CRBs** (click 2.2.4) can be **frequency aligned**. However, in **LTE FDD**, its **Uplink subcarrier grid** is **upshifted** by **7.5 kHz** in respect to its **Downlink subcarrier grid**.

The **5G FDD Downlink subcarrier grid** is **aligned** with the **LTE FDD Downlink subcarrier grid**. This implies that the **5G FDD Uplink subcarrier grid** is **not aligned** with the **5G FDD Uplink subcarrier gris**.

**Subcarrier alignment** between **5G** and **LTE** is required to allow for **5G/LTE Dynamic Spectrum Sharing** (click 10.7), which is an important **5G** capability.

In order to **allow** for **FDD carrier alignment** between **5G** and **LTE**, an **optional 7.5 kHz upshift** is supported in the **5G Uplink subcarrier grid**. When this **upshift** is **applied**,

**both** the **Downlink** and the **Uplink FDD subcarrier grids** of **5G** and of **LTE** are **aligned**. This is shown in the following figure:




Some **relevant remarks** on this figure are:

- The **NR** and **LTE FDD Downlink subcarrier grids** are **aligned**.
- The **NR/LTE Downlink subcarrier grid** shown in the figure is the **projection** of the **NR/LTE Downlink subcarrier grid** (from wherever the **FDD Downlink channel** is located in **frequency**) as much as it is required to **overlap** with the **Uplink subcarrier grids** (within the **Uplink FDD channel**).
- **Before** the **7.5 kHz upshift**, the **5G Downlink** and **Uplink subcarrier grids** are **aligned**, but the **LTE** and **5G Uplink subcarrier grids** are **not aligned**. This is shown on the **left** side of the figure.
- **After** the **7.5 kHz upshift**, the **5G Downlink** and **Uplink subcarrier grids** are **not aligned**, but the **LTE** and **5G Uplink subcarrier grids** are **aligned**. This is shown on the **right** side of the figure.

**UEs** that **support DSS must also support** the **7.5 kHz upshift** of the **FDD Uplink subcarrier grid**.

## 2.2.5 Carrier configuration of each cell's channel

### 2.2.5.1 Carriers

**Each carrier** in **5G** is a **set** of **transmission resources** and **characteristics** of a **multicarrier signal** (click 11.3.2). Notice that the **term** "**carrier**" has a **very different semantics** in the **classical single-carrier signals** (click 11.3.1). In **5G NR both** the **Downlink signal** and the **Uplink signal** are **multicarrier**, which implies that the **signal** is composed of **many subcarriers** (click 2.2.10).

**Each carrier** can be characterized by:

- **Identifier** "***i***" (click 2.2.5.3) of the **carrier** within the **channel**.
- **Type** "***x***", **either DL** (for **Downlink**), **UL (for Uplink)** or **SUL**.

- **Numerology** "$\mu_{carrier,i,x}$" (click 2.1), **0** to **7**, which indicates the **subcarrier spacing** (click 2.2.10.3) and the **slot duration** (click 2.3.1).
- **Starting CRB** "$N_{carrier,i,x}^{start,\mu}$" (click 2.2.5.6), which is its **lowest frequency CRB**.
- **Transmission bandwidth** "$N_{carrier,i,x}^{size,\mu}$" (click 2.2.5.6), as a **quantity** of **CRBs**.
- **DC subcarrier** (click 2.2.10.2).
- **Signal type**, **either CP-OFDM** (click 11.14), **CP-DFT-s-OFDMA** (click 11.16) or **CP-OFDMA** (click 11.15).
- For the case of "$\mu_{carrier,i,x}$=2", the **cyclic prefix** (click 2.3.2), **either normal** or **extended**.

When there is **no risk** for **confusion**, the **type subscript** "$x$" may be dropped from the symbols above (e.g., resulting in "$N_{carrier,i}^{size,\mu}$").

It is possible to configure **several carriers** associated to a **channel**, but **only 1 carrier** can be **operative** at any given time.

There is a **1-to-1 relationship** between **carriers** and **resource grids** (click 2.2.5.6). **Each carrier** defines **1 and only 1 resource grid** and **each resource grid** is defined by **1 and only 1 carrier**. The **currently active** (click 2.2.7.4) **bandwidth part** (click 2.2.7) determines which **frequency segment** of the **carrier** and of its corresponding **resource grid** can be **used**.

Notice that the **SUL carriers** (click 10.4) belong to a **different band** than the band of the **Downlink** and **Uplink carriers**, but **all** these **carriers** can belong to the **same cell** (click 3.1).

**Semantics of the term "carrier" in single-carrier signals**

In **single-carrier signals** (click 11.3.1) the **term** "**carrier**" is used for a **completely different concept** than the **term** "**carrier**" in **multicarrier signals** (that has been explained in this section up to now). In **single-carrier signals**, the term **carrier** is a **pure tone** whose characteristic(s) (**phase**, **amplitude**, **frequency**, …) are **modified** (i.e., **modulated**, click 11.2) such that **each modification** along time **conveys** a certain **information**. This **modulated carrier** is the **passband single-carrier signal**, that is conveyed from **transmitter** to **receiver**.

Source web page:

- https://dsp.stackexchange.com/questions/82583/does-non-orthogonal-multicarrier-modulation-use-pulse-shaping

### 2.2.5.2 Configuration of the carriers for each cell's channel

The **different physical binary channels** (click 4.11.3) can have **different numerologies** at the **same time**. The **Downlink carrier** corresponds to the **numerology** of the **PDSCH binary channel**, while the **Uplink** and **SUL carriers correspond** to the **numerology** of the **PUSCH binary channel**.

**Both PDSCH** and **PUSCH** can have **different numerologies** at **different time intervals because** of the usage of **bandwidth parts**. The **PDSCH** and **PUSCH numerology** at a given time correspond (respectively) to the **numerology** of the **Downlink** and **Uplink**

**currently active bandwidth part**, which corresponds to the **carrier** where the **bandwidth part** has been configured.

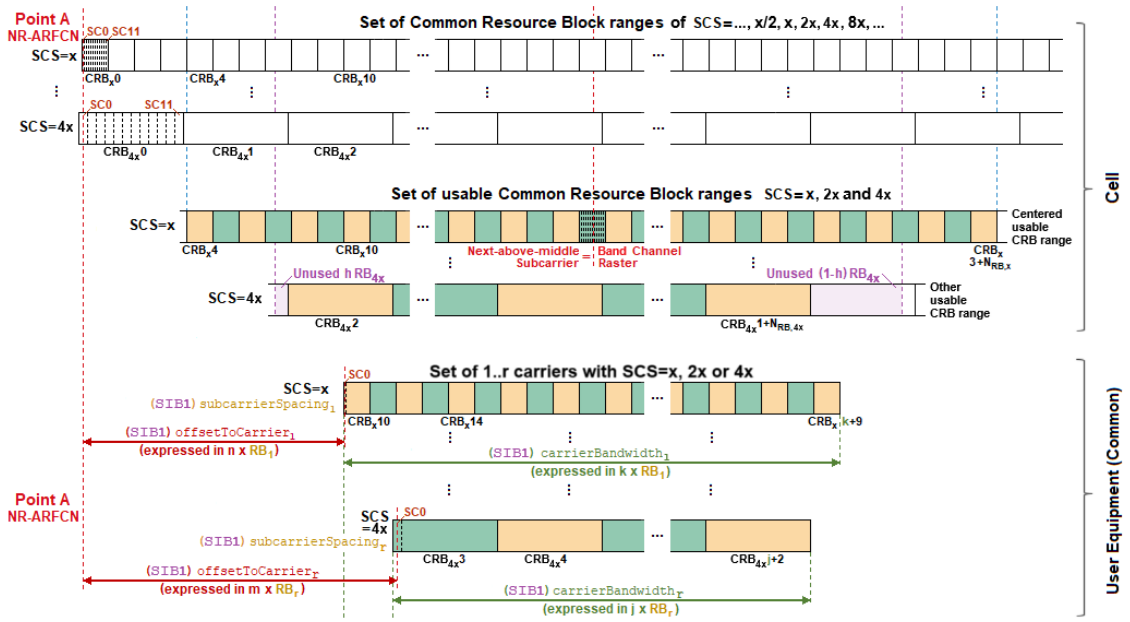
The **Downlink**, **Uplink**, **SUL** (optionally) **carriers** are configured by the **licensee** (usually an operator) of its **enclosing frequency block**. **Each NR node** communicates the corresponding **carrier configuration** to **each** of its **UEs** via **RRC parameters** (click 5).

The **carrier bandwidth** is usually the **transmission bandwidth** (click 2.2.3.6) of the **channel** where it has been defined. It is possible to **configure** the **carrier bandwidth** to a **lower value** than the **channel transmission bandwidth**, but this would require a **strong reason** (e.g., **avoid interference** with a contiguous system) because doing so **wastes** the **very expensive spectrum**.

There is **one set** of **carriers** per **carrier type** (**Downlink**, **Uplink**, **SUL** (if configured) and **SDL** (if configured). The **carrier configuration** corresponds to the **inverse** and **direct Discrete Fourier Transform** (click 4.5.7).

The **CRB indexes** of **each carrier** are the **same** as the ones of the **CRB range** of its **same subcarrier spacing**.

The following figure shows an example of the configuration of **carriers**:




Some **relevant remarks** on this figure are:

- In this example, we are configuring this **cell**'s **channel** and a given **carrier type** (**Downlink**, **Uplink**, **SUL** or **SDL**) with "**r**" **carriers**. The value of "**r**" **must** be **lower or equal** than `maxSCSs` (which is `5`). The **parameters** corresponding to **each carrier** are identified with **subindex 1** or **r**.
- The **subcarrier spacing** of **each carrier must** be the **same** as **one** of the **CRB ranges** (shown as "**x**" to "**4x**"), and it should be **allowed** in the channel.
- **Each carrier**'s **CRBs** are **exactly aligned** with the ones of the **CRB range** of its **same subcarrier spacing**, as they **must**.
- **Each carrier**'s **CRB indexes** are the **same** as the ones of the **CRB range** of its **same subcarrier spacing**, as they **must**.

- A frequent configuration is that **each carrier** has the **same CRB range** as the corresponding **usable CRB range**. However, the example in the figure shows a **different** case, where the **carrier CRB range** is **smaller** than the **usable CRB range**. The reason is that we want to highlight that this case is **possible** (although it is **not** the most **frequent** one).
- This example is for **initial access**, and this is why the **carrier parameters** are marked as being provided within `SIB1` (click 5.3.4) Otherwise, they would be provided within the **Non-SI parameter group** (click 5.3.6).
- The **frequency bars** of the **CRB ranges**, **carriers** and **bandwidth parts** are drawn to **scale**, where **each contiguous horizontal pixel** is **half the subcarrier spacing** "**x**"( i.e., **$SCS_x$/2**).
- The **half $SCS_x$ kHz** corresponding to the **1-pixel** dividing **lines** between **two elements** (e.g., a **1-pixel line** between **two consecutive subcarriers**, or a **1-pixel line between one subcarrier** and **nothing**) belongs at **50%** to **each** of the **two elements** (i.e., **$SCS_x$/4 belongs** to **each element**).

#### 2.2.5.3 Carrier identification

**Each carrier** "***i***" is **identified** with the **index** of the following `SEQUENCE OF`:

```
SEQUENCE (1..maxSCSs) OF SCS-SpecificCarrier
```
(click 2.2.6.12)

The constant `maxSCSs` is `4`, which is the **maximum quantity** of **UE**'s **carriers** within **each** of the **UE**'s **Downlink**, **Uplink** or **SUL carrier types**, in **each** of the **UE**'s **cells**.

This **index uniquely** identifies **each** of the **UE**'s **carriers** within **each** of the **UE**'s **Downlink**, **Uplink** or **SUL carrier types**, within **each** of the **UE**'s **cells**. Notice that the **SUL carriers** belong to a **different band** than the **Downlink** and **Uplink carriers**, but **all** these **carriers** belong to the **same cell**.

**Carrier types**

According to **channel type** and **transmission direction**, there are **three carrier types**:

- **Downlink carriers**

  **Carriers** in the **Downlink transmission direction**, within **TDD** or **FDD bands/channels**.

- **Uplink carriers**

  **Carriers** in the **Uplink transmission direction**, within **TDD** or **FDD bands/channels**.

- **SUL carriers**

  **Carriers** within **SUL bands/channels**. They are **transmitted** in the **Uplink transmission direction**, but they are **not** the same as **Uplink carriers**.

Therefore, there are **three mutually exclusive carrier types** (according to **channel type** and **transmission direction**): **Downlink**, **Uplink**, **SUL carriers** and **SDL carriers**. Notice that the **SUL carriers** belong to a **different band** than the **Downlink** and **Uplink carriers**, but **all** these **carriers** belong to the **same cell**.

Do **not confuse** the **carrier types** with their **transmission direction**. Even though a **SUL carrier** is in the **Uplink transmission direction**, it is **not** an **Uplink carrier** (they are

two **different terms**). When it is said "**Uplink carriers**", this **does not** include **SUL carriers**, and it **only** refers to **Uplink carriers**, as have been **defined above**. To **jointly** refer to **Uplink** and/or **SUL carriers** the term will be **either** "**Uplink** and/or **SUL carriers**" or "**carriers** in the **Uplink transmission direction**".

For the **relationship** between the **types** of **bands** (**duplexing mode**), of **channels**, of **CRB ranges**, of **carriers**, of **bandwidth parts** and the **transmission direction**, click 1.1.3.

#### 2.2.5.4 Configuration of each carrier

The **attributes** (click 2.2.5) of **each carrier** are defined by the following **parameter instances**:

- **Subcarrier spacing** = `subcarrierSpacing`

  The **subcarrier spacing** (click 2.2.10.3) directly determines the **numerology** "$\mu_{carrier,i,x}$" (click 2.1) and indirectly the **slot duration** (click 2.3.1).

- $N_{carrier,x}^{start,\mu}$ = `offsetToCarrier`
- $N_{carrier,x}^{size,\mu}$ = `carrierBandwidth`
- **DC subcarrier location** = `txDirectCurrentLocation`

  For more detail on the **DC subcarrier**, click 2.2.10.2.

These **4 parameters** are **all** the parameters from `SCS-SpecificCarrier` (click 2.2.6.12).

- **Cyclic prefix** = `cyclicPrefix` from several **data types**

  For the configuration of **cyclic prefix** for **bandwidth parts**, **carriers** and **resource grids**, click 2.3.3.

During **initial access** (click 8), the **Downlink carrier configuration** that is required to receive `SIB1` (click 5.3.4) through **PDSCH** is obtained from `MIB` (click 5.3.3): for more detail on this, click 2.2.6.1.

For more detail on **carrier parameters**, click 2.2.5.8.

#### 2.2.5.5 Maximum carrier bandwidth for each subcarrier spacing

There is a limitation of the **maximum quantity** of **subcarriers** in a **carrier** due to the **computational complexity** of the required **Fast Fourier Transforms** at the **Base Band Unit**. In **5G** a **carrier cannot** have more than **275 Resource Blocks**, which is **3,300 subcarriers**, while in **LTE** a **carrier cannot** have more than around **2,200 subcarriers**. This is one of the reasons why the **minimum subcarrier spacing** in **FR2** is **60 MHz**, not to exceed the said limit, in spite of the **very large channel bandwidths** available in **FR2**.

To calculate the **maximum channel bandwidth** for **each subcarrier spacing** in **5G**, let us introduce some background from **LTE**. In **LTE**, the **maximum channel bandwidth** depends on the **DFT size** and on the **subcarrier spacing**. In **LTE**, the **DFT size** is **2,048** and **SCS = 15KHz**, so the resulting **sampling frequency** is **2,048•15 kHz=30.72 MHz**. The **maximum channel bandwidth must** be below the **sampling frequency**. Consequently, the **LTE maximum channel bandwidth** was chosen to be **20 MHz**.

In **5G**, the selection of **subcarrier spacing** and **DFT size** was based on multiple factors, one of them being the support of **channel bandwidths considerably larger** than the **ones**

in **LTE**. In **5G**, the **DFT size** has been **doubled** to **4,096**, which is combined with the introduction of several **larger subcarrier spacings**. The following table presents the **sampling frequencies** and **maximum channel bandwidth** that has been defined for the different **5G subcarrier spacings**.

| SCS [kHz] | Sampling frequency [MHz] | Maximum 5G channel bandwidth [MHz] |
|---|---|---|
| **15** | 61.44 | 50 |
| **30** | 122.88 | 100 |
| **60** | 245.76 | 200 |
| **120** | 491.52 | 400 |
| **240** | 983.04 | Only available for **SSB** |
| **480** | 1,966.08 | 1,600 |
| **960** | 3,932.16 | 2,000 |

For more detail on the **channel bandwidths** that are **available** for **each value** of **control/shared subcarrier spacing**, click 2.2.3.2.

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_FrameStructure.html
- http://howltestuffworks.blogspot.com/2019/11/5g-nr-resource-blocks.html

### 2.2.5.6 Resource grids

**Each resource grid** is **all** the **subcarriers** (click 2.2.10) of a **carrier** (click 2.2.5), over **all** the **symbol interval**s (click 2.3.2) of **one subframe** (click 2.3.1), over **one layer** "***lay***" (click 4.11.4.8.3).

**Each resource grid** can be characterized by the **carrier** to what it is **associated**, and by its **layer** "***lay***". It **inherits** the **characteristic** of its **carrier**: "$\mu_{grid,x}=\mu_{carrier,x}$", "$N_{grid,x}^{start,\mu} = N_{carrier,x}^{start,\mu}$", "$N_{grid,x}^{size,\mu} = N_{carrier,x}^{size,\mu}$" and for "$\mu_{grid,x}=2$" also the **cyclic prefix**. The **subscript** "$x$" may be "$DL$", "$UL$" or "$SUL$" to indicate the **resource grid type**, that corresponds to its **carrier type**. Notice that the **SUL resource grids** belong to a **different band** than the **Downlink** and **Uplink resource grids**, but **all** these **resource grids** belong to the **same cell**. When there is no risk for confusion, the subscript "$x$" may be dropped.

**Each resource grid** has **14** or **12** of **symbol intervals per subframe** "$N_{symb}^{subframe,\mu}$", if its **cyclic prefix** is **normal** or **extended** (respectively). For the configuration of **cyclic prefix** for **bandwidth parts**, **carriers** and **resource grids**, click 2.3.3.

**Each resource grid** is a **relevant reference** to **locate**, in **both frequency** and **time**, the **Resource Element** and **Layers** (2.2.5.7) corresponding to the different **physical channels** and **signals** (click 4.11.3).

There is a **1-to-1 relationship** between **carriers** and **resource grids** (click 2.2.5.6). **Each carrier** defines **1 and only 1 resource grid** and **each resource grid** is defined by **1 and only 1 carrier**. The **currently active** (click 2.2.7.4) **bandwidth part** (click 2.2.7) determines which **frequency segment** of the **carrier** and of its corresponding **resource grid** can be **used**.

Click 1.1.4 for a **clarifying figure** that shows an **approximate** bidimensional (in **frequency** and **time**) representation of a **resource grid** (click 2.2.5.6) of the **Downlink**

**currently active** (click 2.2.7.4) **bandwidth part** (click 2.2.7), for the duration of **1 frame** (click 2.3.1), indicating with colors how the **different RBs** (click 2.2.5.7) have been **scheduled** (click 4.11.4) to the **different physical binary channels** (click 4.11.3).

For more detail on **resource grids**, refer to clause 4.4.2 of TS 38.211.

### 2.2.5.7 Resource elements in port and Resource Blocks

A **Resource Element** is **1 symbol interval**, within **1 subcarrier** (click 2.2.10). A **Resource Element** is denoted as "$(sc, si)_{\mu}$", where "***sc***" is the **subcarrier index**, "***si***" is the **symbol interval index** and "**μ**" is the **numerology** (click 2.1), **all** of them corresponding to a **resource grid**.

A **Resource Element** (**RE**, click 2.2.5.7) in **port** is **1 symbol interval** (click 2.3.1) within **1 subcarrier** (click 2.2.10) and in **1 antenna port** (click 4.11.5). **REs** in **port** that have **multiple ports** for **each RE** are used by **PDSCH** and **PUSCH**. A **Resource Element** in **port** is denoted as "$(sc, si, ap)_{\mu}$", where "***sc***" is the **subcarrier index**, "***si***" is the **symbol interval index**, "***ap***" is the **antenna port number** and "**μ**" is the **numerology** (click 2.1), **all** of them corresponding to a **resource grid**.

An **RE** or a **RE** is the **smallest transmission resource unit** in a **resource grid** (click 2.2.5.5).

**Subcarriers** are **indexed** from "$sc=0$", for the **lowest frequency** one, and **upwards** within **either** the **CRB range**, or the **carrier**, or the **resource grid** or the **bandwidth part**, depending on context.

**Symbols** are **indexed** from "$si=0$", for the **earliest** one, and **upwards** within **either** the **frame**, or the **half-frame**, or the **subframe** or the **slot** (click 2.3.1), depending on context.

The **Resource Element** "$(sc, si)_{\mu}$" is located in the **CRB** with **CRB index**:

$$n^{\mu}_{CRB} = floor(sc / 12)$$

**Each Resource Element** in **port** "$(sc, si, ap)_{\mu}$" contains **1 symbol** (or a **set** of **symbol fractions**, click 11.16) from a **digital modulation** or from a **complex sequence**. The **transmitted symbol** is represented as a **complex value** "$a_{sc,si}^{(ap,\mu)}$" (click 11.2.4), where "***ap***" is the **antenna port number**. When there is no risk for confusion, or no particular **antenna port** or **subcarrier spacing** is specified, the indices "***sc***", "***ap***" and/or "**μ**" may be dropped, resulting in the **Resource Element** "$(sc, si)$" and the complex value "$a_{sc,si}$", respectively.

**Each Resource Block** is **12 contiguous subcarriers** that belong to a **Common Resource Block** (click 2.2.4).

A **Resource Element** may be either **virtual** (a **VRE**) or **physical** (a **PRE**). A **VRE** is a **RE** where its **subcarriers** are **indexed** over a **virtual subcarrier space**. A **PRE** is a **RE** where its **subcarriers** are **indexed** over a **physical subcarrier space**, this is, the **actual transmitted subcarriers**. The **PDSCH Virtual Resource Blocks** (**VRBs**) of the **VREs** are **mapped** to the **PDSCH Physical Resource Blocks** (**PRBs**) of the **PREs either** with a **non-interleaved** or an **interleaved mapping** (click 4.12.15).

**Space-multiplexing** of **data** is performed over the **same resource blocks** (**RBs**) and the **same symbol intervals**. Therefore, **each** and **every space-multiplexing layer** of a given **PDSCH** or **PUSCH transmission** contains the **same set** of (**RB**, **symbol interval**) **pairs**. This also implies that **every PDSCH** or **PUSCH layer** has the **same quantity** of **PDSCH REs** or **PUSCH REs** (respectively).

Click 1.1.4 for a **clarifying figure** that shows an **approximate** bidimensional (in **frequency** and **time**) representation of a **resource grid** (click 2.2.5.6) of the **Downlink currently active** (click 2.2.7.4) **bandwidth part** (click 2.2.7), for the duration of **1 frame** (click 2.3.1), indicating with colors how the **different RBs** (click 2.2.5.7) have been **scheduled** (click 4.11.4) to the **different physical binary channels** (click 4.11.3).

For more detail on **Resource Elements**, refer to clause 4.4.3 of TS 38.211.

#### 2.2.5.8 Multiple simultaneous subcarrier spacings

When **simultaneous physical layer elements** in the **same radio channel** have **different numerologies** (click 2.1.5), it implies that these **elements simultaneously** have **different subcarrier spacings** and **different** (average) **symbol interval durations** (click 2.1.5.5).

One of the implications of **multiple simultaneous subcarrier spacings** is that the **subcarrier edges** of **2 contiguous physical layer elements** with **different subcarrier spacings** are **not aligned** (click 2.2.4.4). This implies that **1 subcarrier** will be **lost** at the **frontier** between the **2** said **physical layer elements**. In case that **both** said **physical layer elements must** be **aligned** to **Resource Blocks**, then **1 complete Resource Block** will be **lost**. In case **only 1** of the said **physical layer elements must** be **aligned** to **Resource Blocks**, then **part** of a **Resource Block** will be **lost**. Moreover, **signal processing constraints** may imply **additional frequency separation** between the **2** said **physical layer elements**. This is why it is usually required to leave some **specific guardband** (click 2.2.3.6.2) between **physical layer elements** with **different subcarrier spacings**.

**Not all** the **combinations** between **SSB subcarrier spacing**, **CORESET#0** (and **PDSCH** for **SIB1**) **subcarrier spacing**, **PRACH subcarrier spacing** and **control/shared subcarrier spacing** are allowed. For the **allowed combinations**, click 2.1.5.

Several **tables** presenting the **subcarrier spacing(s)** (plus many other characteristics) for **each** of the **different physical channels** and **signals** are available at 4.11.3.1.

In **LTE**, there is only **one** subcarrier spacing, and therefore **all** the physical layer elements are **frequency aligned**. There is the exception of the **7.5 kHz SCS**, for **Multimedia Broadcast Multicast Service** (**MBMS**), that is handled in a specific way.

Source web pages:

- https://5g.smarttelecomedu.com/common-resource-blocks-physical-resource-blocks/

### 2.2.6 Carrier parameters

The **UE**'s **carriers** (click 2.2.5) are configured with:

- **Common parameters** that are **broadcast** within **MIB**, as described in 2.2.6.1.
- **Common parameters** that are **broadcast** (or **unicast**) within **SIB1**, as described in 2.2.6.2.
- **Common** and **dedicated parameters** that are **unicast** within the **Non-SI parameter group**, as described in 2.2.6.3.

Possibly existing **carrier parameters** within **other SIBs** and **posSIBs** (click 5.3.5) are **not addressed** in this book.

For an overall explanation on **UE configuration parameters** click 5. For a brief overview of **ASN.1**, click 5.9.

### 2.2.6.1 Carrier parameters within MIB

The **carrier** parameters **within** MIB (click 5.3.3) are subCarrierSpacingCommon and ssb-SubcarrierOffset (click 8.7.1) **both** of them **from** MIB (click 8.12.3). They **jointly** configure the **Downlink carrier** during **initial access** (click 8). **Both** of them are **common**.

The **ancestor tree** (click 5.9.8) of these two **carrier** parameters **up to** the **top** is:

```
                                BCCH-BCH-Message (click 5.8.5.1)
message                           BCCH-BCH-MessageType (click 5.8.5.1)
  mib                               MIB (click 8.12.3)
    subCarrierSpacingCommon       ENUMERATED {scs15or60, scs30or120}
    ssb-SubcarrierOffset          INTEGER (0..15)
```

### 2.2.6.2 Carrier parameters within SIB1

For **carrier** parameters within SIB1 (click 5.3.4), the **most relevant related data types** are:

- DownlinkConfigCommonSIB (click 2.2.6.4): common parameters
- UplinkConfigCommonSIB (click 2.2.6.5): common parameters
- FrequencyInfoDL-SIB (click 2.2.6.6): common parameters
- FrequencyInfoUL-SIB (click 2.2.6.7): common parameters
- SCS-SpecificCarrier (click 2.2.6.12): instantiated as common

The **ancestor tree** (click 5.9.8) of these **most relevant related data types** up to SIB1 is:

```
                                SIB1 (click 5.7.4)
servingCellConfigCommon           ServingCellConfigCommonSIB (0,1) (click 3.9.3)
  downlinkConfigCommon              DownlinkConfigCommonSIB (click 2.2.6.4)
    frequencyInfoDL                   FrequencyInfoDL-SIB (click 2.2.6.6)
      scs-SpecificCarrierList           SCS-SpecificCarrier (1..maxSCSs) (2.2.6.12)
  uplinkConfigCommon                UplinkConfigCommonSIB (click 2.2.6.5)
    frequencyInfoUL                   FrequencyInfoUL-SIB (click 2.2.6.7)
      scs-SpecificCarrierList           SCS-SpecificCarrier (1..maxSCSs) (2.2.6.12)
  supplementaryUplink               UplinkConfigCommonSIB (click 2.2.6.5)
    frequencyInfoUL                   FrequencyInfoUL-SIB (click 2.2.6.7)
      scs-SpecificCarrierList           SCS-SpecificCarrier (1..maxSCSs) (2.2.6.12)
```

Some **relevant remarks** on this **ancestor tree** are:

- For the **ancestor trees** of SIB1 **up to** the **top**, click 5.3.4.
- The constant maxSCSs is 5, which is the **maximum quantity** of **carriers** of a given **type** (**Downlink**, **Uplink** or **SUL**) in **each** of the **UE**'s **cells**.

### 2.2.6.3 Carrier parameters within the Non-SI parameter group

For **carrier** parameters within the **Non-SI parameter group** (click 5.3.6), the **most relevant related data types** are:

- DownlinkConfigCommon (click 2.2.6.8): common parameters
- UplinkConfigCommon (click 2.2.6.9): common parameters

- **FrequencyInfoDL** (click 2.2.6.10): common parameters
- **FrequencyInfoUL** (click 2.2.6.11): common parameters
- ***DownlinkConfig*** **parameter set** (click 2.2.8.3): common **and** dedicated parameters
- **UplinkConfig** (click 2.2.8.4): common **and** dedicated parameters
- **ReconfigurationWithSync** (click 3.9.8): common **and** dedicated parameters
- **SCS-SpecificCarrier** (click 2.2.6.12): common parameters

The **ancestor tree** (click 5.9.8) of **SCS-SpecificCarrier** up to **CellGroupConfig** is:

```
                                        CellGroupConfig (click 3.7.3)
spCellConfig                              SpCellConfig (click 3.9.5)
  reconfigurationWithSync                   ReconfigurationWithSync (0,1) (click 3.9.8)
    spCellConfigCommon                        ServingCellConfigCommon (click 3.9.6)
      downlinkConfigCommon                      DownlinkConfigCommon (click 2.2.6.8)
        frequencyInfoDL                           FrequencyInfoDL (click 2.2.6.10)
          scs-SpecificCarrierList                   SCS-SpecificCarrier (1..maxSCSs)
                                                                                (2.2.6.12)
      uplinkConfigCommon                        UplinkConfigCommon (click 2.2.6.9)
        frequencyInfoUL                           FrequencyInfoUL (click 2.2.6.11)
          scs-SpecificCarrierList                   SCS-SpecificCarrier (1..maxSCSs)
                                                                                (2.2.6.12)
    spCellConfigDedicated                   ServingCellConfig (click 3.9.7)
      begin of DownlinkConfig parameter set (click 2.2.8.3)
      downlinkChannelBW-PerSCS-List         SCS-SpecificCarrier (0..maxSCSs)
      end of DownlinkConfig parameter set
      uplinkConfig                            UplinkConfig (0,1) (click 2.2.8.4)
        uplinkChannelBW-PerSCS-List             SCS-SpecificCarrier (0..maxSCSs)
                                                                              (2.2.6.12)
      supplementaryUplink                     UplinkConfig (0,1) (click 2.2.8.4)
        uplinkChannelBW-PerSCS-List             SCS-SpecificCarrier (0..maxSCSs)
                                                                              (2.2.6.12)
sCellToAddModList                       SCellConfig (0..maxNrofSCells) (click 3.9.4)
  sCellConfigCommon                       ServingCellConfigCommon (0,1) (click 3.9.6)
    downlinkConfigCommon                    DownlinkConfigCommon (click 2.2.6.8)
    uplinkConfigCommon                      UplinkConfigCommon (click 2.2.6.9)
    supplementaryUplinkConfig               UplinkConfigCommon (0,1) (click 2.2.6.9)
  sCellConfigDedicated                  ServingCellConfig (click 3.9.7)
    begin of DownlinkConfig parameter set (click 2.2.8.3)
    downlinkChannelBW-PerSCS-List       SCS-SpecificCarrier (0..maxSCSs)
    end of DownlinkConfig parameter set
    uplinkConfig                          UplinkConfig (0,1) (click 2.2.8.4)
      uplinkChannelBW-PerSCS-List           SCS-SpecificCarrier (0..maxSCSs)
                                                                            (2.2.6.12)
    supplementaryUplink                   UplinkConfig (0,1) (click 2.2.8.4)
      uplinkChannelBW-PerSCS-List           SCS-SpecificCarrier (0..maxSCSs)
                                                                            (2.2.6.12)
```

Some **relevant remarks** on this **ancestor tree** are:

- For the **ancestor trees** of **CellGroupConfig**, **up to** some **RRC NR messages** click 3.7.1 and **up to** some **RRC inter-node messages** click 3.7.2.
- The constant **maxNrofSCells** is **31**, which is the **maximum quantity** of **UE**'s **secondary cells** (i.e., **other** than the **special cells**) in **each cell group**

- The constant `maxSCSs` is `5`, which is the **maximum quantity** of **carriers** of a given **type** (**Downlink**, **Uplink** or **SUL**) in **each** of the **UE**'s **cells**.

Recall that **Sidelink** channels (and consequently **Sidelink** data types and parameters) are **not** covered in this book

#### 2.2.6.4 `DownlinkConfigCommonSIB` data type

It is **only broadcast** (or **unicast**) within `SIB1` (click 5.3.4).

For its **ancestor tree up to** `SIB1`, click 2.2.6.2.

**All** its parameters are **common** (click 5.1.3).

**Each** of its **instances** contains the **Downlink common** configuration for the **carriers**, for the **initial bandwidth part**, for **BCCH**, for **PCCH** and for **other elements**.

##### 2.2.6.4.1 `DownlinkConfigCommonSIB` parameters

**All** its **parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `bcch-Config` `BCCH-Config`: mandatory, common

  It contains the configuration related to the **modification period** (click 5.5.2) of the System Information.

- `frequencyInfoDL` `FrequencyInfoDL-SIB` (click 2.2.6.6): mandatory, common

  It contains the basic parameters of a **Downlink carrier** and transmission thereon.

- `initialDownlinkBWP` `BWP-DownlinkCommon` (click 2.2.8.6): mandatory, common

  It contains the **Downlink initial bandwidth part common** configuration for the **primary cell**. The network configures `locationAndBandwidth` from `BWP` (click 2.2.8.11), **within** `initialDownlinkBWP`, so that the **Downlink initial bandwidth part** contains the entire **CORESET#0** of this serving cell in the **frequency domain**. The **UE** applies `locationAndBandwidth` upon its reception (e.g., to determine the frequency location of signals described in relation to this `locationAndBandwidth`) but it keeps **CORESET#0** until after the reception of either of the **NR RRC messages** `RRCSetup` (click 5.8.5.11), `RRCResume` (click 5.8.5.4) or `RRCReestablishment` (click 5.8.5.4).

- `initialDownlinkBWP-RedCap-r17` `BWP-DownlinkCommon` (click 2.2.8.6): optional, common

  If it is **present**, **reduced capability** (**RedCap**) **UEs** configures its **bandwidth part** with **this** parameter instead of with parameter `initialDownlinkBWP` (see above).

  ***If*** [the `locationAndBandwidth` from `BWP` (click 2.2.8.11), **within** `initialDownlinkBWP-RedCap-r17`, **contains** the **entire CORESET#0**], ***then***, the **UE** applies the `locationAndBandwidth` upon reception of **this** parameter (e.g. to determine the frequency location of signals described in relation to this `locationAndBandwidth`) but it keeps **CORESET#0** until after reception of **NR RRC messages** `RRCSetup`, `RRCResume` or `RRCReestablishment`.

***Else***, (i.e., if the **locationAndBandwidth** **within this** parameter does **not contain** the **entire CORESET#0**), the **UE** uses the information in this parameter for receiving **Downlink messages** during **initial access** (**Msg2**, **MsgB**, **Msg4**, click 9) and **after initial access**.

If it is **absent**, **RedCap UEs** use **initialDownlinkBWP** (see above) provided that it does **not exceed** the **RedCap UE maximum bandwidth** (refer to clause 5.2.2.4.2 of TS 38.331).

- **pcch-Config** **PCCH-Config**: mandatory, common

  It contains the **paging** related configuration.

- **pei-Config-r17** **PEI-Config-r17**: optional, common

  It contains the **paging early indication** (**PEI**) related configuration.

For more detail on these parameters, refer to table "*DownlinkConfigCommonSIB field descriptions*" of TS 38.331.

#### 2.2.6.4.2 DownlinkConfigCommonSIB ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
DownlinkConfigCommonSIB ::= SEQUENCE {
  frequencyInfoDL FrequencyInfoDL-SIB,
  initialDownlinkBWP BWP-DownlinkCommon,
  bcch-Config BCCH-Config,
  pcch-Config PCCH-Config,
  ...,
  [[
  pei-Config-r17 PEI-Config-r17 OPTIONAL,                    -- Need R
  initialDownlinkBWP-RedCap-r17 BWP-DownlinkCommon OPTIONAL  -- Need R
  ]]
}
```

### 2.2.6.5 UplinkConfigCommonSIB data type

It is **only broadcast** (or **unicast**) within **SIB1** (click 5.3.4).

For its **ancestor tree up to** **SIB1**, click 2.2.6.2.

**All** its parameters are **common** (click 5.1.3).

**Each** of its **instances** provides the **common** configuration for the **Uplink carriers** or for the **SUL carriers**.

#### 2.2.6.5.1 UplinkConfigCommonSIB parameters

**All** its **parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- **frequencyInfoUL** **FrequencyInfoUL-SIB** (click 2.2.6.7): mandatory, common

  It contains the absolute **Uplink frequency configuration** and subcarrier specific **virtual carriers**.

- **initialUplinkBWP** **BWP-UplinkCommon** (click 2.2.8.9): mandatory, common

  It contains the **Uplink initial bandwidth part common** configuration for a **primary cell** (refer to clause 12 of TS 38.213).

- `timeAlignmentTimerCommon TimeAlignmentTimer`: mandatory, common

  It does **not** have a **field description** in TS 38.331, but there is some information about it in the **running text**.

  It is then **assumed** that it is the value of the **time alignment timer** as specified in TS 38.321.

  Its data type `TimeAlignmentTimer` is an **`ENUMERATED`** with the following values:

  ```
  {ms500, ms750, ms1280, ms1920, ms2560, ms5120, ms10240, infinity}
  ```

For more detail on these parameters, refer to the table "*UplinkConfigCommonSIB field descriptions*" of TS 38.331.

#### 2.2.6.5.2 `UplinkConfigCommonSIB` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
UplinkConfigCommonSIB ::= SEQUENCE {
  frequencyInfoUL FrequencyInfoUL-SIB,
  initialUplinkBWP BWP-UplinkCommon,
  timeAlignmentTimerCommon TimeAlignmentTimer
}
```

### 2.2.6.6 `FrequencyInfoDL-SIB` data type

It is **only broadcast** (or **unicast**) within `SIB1` (click 5.3.4).

For its **ancestor tree up to** `SIB1`, click 2.2.6.2.

**All** its parameters are **common** (click 5.1.3).

**Each** of its **instances** contains the **common frequency** configuration for **Downlink carriers**.

#### 2.2.6.6.1 `FrequencyInfoDL-SIB` parameters

**Some** of its **relevant parameters** are (in italics, the **literal** text from TS 38.331):

- `frequencyBandList MultiFrequencyBandListNR-SIB`: mandatory, common

  "*List of one or multiple frequency bands to which this carrier(s) belongs.*"

- `offsetToPointA` (click 8.7.3) **`INTEGER (0..2199)`**: mandatory, common

  "*Represents the offset to Point A as defined in TS 38.211 [16], clause 4.4.4.2.*"

  It is the **largest integer quantity** of **RBs** (with the **common subcarrier spacing** "$SCS_c$"), that fits between **Point A** and the **center frequency** of the **lowest subcarrier** of **SSB** (click 6.1) expressed in **units** of **RBs** with **subcarrier spacing** "$SCS_u$".

  In TS 38.331 there is **only 1** parameter `offsetToPointA`, which is this one, that belongs to `SIB1`.

  This parameter is somehow **relevant** and **complex**: this is why it is described in the **specific section** 8.7.3.

- `scs-SpecificCarrierList SEQUENCE (SIZE (1..maxSCSs)) OF SCS-SpecificCarrier` (click 2.2.6.12): mandatory, common

  "*A set of carriers for different subcarrier spacings (numerologies). Defined in relation to Point A (see TS 38.211 [16], clause 5.3). The network configures this for all SCSs that are used in DL BWPs in this serving cell.*"

  The constant `maxSCSs` is `5`, which the **maximum quantity** of **carriers** that can be configured.

For more detail on these parameters, refer to the table "*FrequencyInfoDL-SIB field descriptions*" of TS 38.331.

#### 2.2.6.6.2 `FrequencyInfoDL-SIB` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
FrequencyInfoDL-SIB ::= SEQUENCE {
  frequencyBandList MultiFrequencyBandListNR-SIB,
  offsetToPointA INTEGER (0..2199),
  scs-SpecificCarrierList
         SEQUENCE (SIZE (1..maxSCSs)) OF SCS-SpecificCarrier
}
```

Where `maxSCSs` is `5`, which is the **maximum quantity** of **carriers** in **each** of the **UE**'s **cells**.

### 2.2.6.7 `FrequencyInfoUL-SIB` data type

It is **only broadcast** (or **unicast**) within `SIB1` (click 5.3.4).

For its **ancestor tree up to** `SIB1`, click 2.2.6.2.

**All** its parameters are **common** (click 5.1.3).

**Each** of its **instances** contains the **common frequency** configuration for **Uplink carriers**.

#### 2.2.6.7.1 `FrequencyInfoUL-SIB` parameters

**Some** of its **relevant parameters** are:

- `absoluteFrequencyPointA ARFCN-ValueNR`: optional, common

  "*Absolute frequency of the reference resource block (Common RB 0). Its lowest subcarrier is also known as Point A. Note that the lower edge of the actual carrier is not defined by this field but rather in the scs-SpecificCarrierList (see TS 38.211 [16], clause 4.4.4.2).*"

- `frequencyBandList MultiFrequencyBandListNR-SIB`: optional, common

  "*Provides the frequency band indicator and a list of additionalPmax and additionalSpectrumEmission values as defined in TS 38.101-1 [15], table 6.2.3.1-1, and TS 38.101-2 [39], table 6.2.3.1-2. The **UE** shall apply the first listed band which it supports in the frequencyBandList field.*"

- `frequencyShift7p5kHz ENUMERATED {true}`: optional, common

  If it is **present** (in which case it is "`true`"), the **Uplink subcarriers** and/or **SUL subcarriers** have a **7.5 kHz shift** in respect to the **LTE subcarrier** raster (click 11.15). If it is **absent**, there is no **7.5 kHz shift**.

Notice there is a **same-name parameter** with the **same semantics** from `FrequencyInfoUL` (click 2.2.6.11).

- `scs-SpecificCarrierList` `SEQUENCE (SIZE (1..maxSCSs)) OF SCS-SpecificCarrier` (click 2.2.6.12): mandatory, common

  "*A set of carriers for different subcarrier spacings (numerologies). Defined in relation to Point A (see TS 38.211 [16], clause 5.3). The network configures this for all SCSs that are used in UL BWPs in this serving cell.*"

  The value of `maxSCSs` is `5`, which is the **maximum quantity** of **carriers** in **each** of the **UE**'s **cells**.

For more detail on these parameters, refer to the table "*FrequencyInfoUL-SIB field descriptions*" of TS 38.331.

#### 2.2.6.7.2 `FrequencyInfoUL-SIB` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
FrequencyInfoUL-SIB ::= SEQUENCE {
  frequencyBandList MultiFrequencyBandListNR-SIB OPTIONAL, -- Cond FDD-OrSUL
  absoluteFrequencyPointA ARFCN-ValueNR OPTIONAL,          -- Cond FDD-OrSUL
  scs-SpecificCarrierList
        SEQUENCE (SIZE (1..maxSCSs)) OF SCS-SpecificCarrier
  p-Max P-Max OPTIONAL,                                    -- Need S
  frequencyShift7p5kHz ENUMERATED {true} OPTIONAL,
                                                -- Cond FDD-TDD-OrSUL-Optional
  ...
}
```

Where `maxSCSs` is `5`, which is the **maximum quantity** of **carriers** in **each** of the **UE**'s **cells**.

### 2.2.6.8 `DownlinkConfigCommon` data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 2.2.6.3.

**All** its parameters are **common** (click 5.1.3).

It contains the **Downlink common** configuration for the **carriers**, the **initial bandwidth part** and other elements.

**Each** of its **instances** configures the **UE** with **most** of the **physical layer elements**, of its **initial bandwidth part**, of **Downlink**, of **common configuration**, of its corresponding **cell** (**special** or **secondary**), of its corresponding **cell group** (**master** or **secondary**). The "**corresponding**" elements are determined by the **specific place**, within the overall **ASN.1 specification**, where this **data type** has been **instantiated**.

#### 2.2.6.8.1 `DownlinkConfigCommon` parameters

Its **three parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `frequencyInfoDL` `FrequencyInfoDL` (click 2.2.6.10): optional, common

  It contains the basic configuration of a **Downlink carrier** and transmission thereon.

- `initialDownlinkBWP` `BWP-DownlinkCommon` (click 2.2.8.6): optional, common

  It contains the **Downlink initial bandwidth part** configuration for a **serving cell**. The network configures the `locationAndBandwidth` from `BWP` (click 2.2.8.11), **within** `initialDownlinkBWP`, so that the **Downlink initial bandwidth part** contains the entire **CORESET#0** of this **serving cell** in the **frequency domain**.

- `initialDownlinkBWP-RedCap-r17` `BWP-DownlinkCommon` (click 2.2.8.6): optional, common

  If it is **present**, the **reduced capability** (**RedCap**) **UEs** configure the **Downlink initial bandwidth part** with **this** parameter instead of with `initialDownlinkBWP` (see above).

  If it is **absent**, the **RedCap UEs** uses `initialDownlinkBWP` provided that it does **not exceed** the **RedCap UE maximum bandwidth** (refer to clause 5.2.2.4.2).

For more detail on these parameters, refer to and "*DownlinkConfigCommon field descriptions*" of TS 38.331.

#### 2.2.6.8.2 `DownlinkConfigCommon` ASN.1 specifications

Its **complete ASN.1** (click 5.9) specification is:

```
DownlinkConfigCommon ::= SEQUENCE {
  frequencyInfoDL FrequencyInfoDL
                  OPTIONAL,                -- Cond InterFreqHOAndServCellAdd
  initialDownlinkBWP BWP-DownlinkCommon OPTIONAL,        -- Cond ServCellAdd
  ...,
  [[
  initialDownlinkBWP-RedCap-r17 BWP-DownlinkCommon OPTIONAL        -- Need R
  ]]
}
```

### 2.2.6.9 `UplinkConfigCommon` data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 2.2.6.3.

**All** its parameters are **common** (click 5.1.3).

It contains the **common** configuration for **either Uplink or SUL** (as corresponds) for the **carriers**, the **initial bandwidth part** and other elements.

**Each** of its **instances** configure the **UE** with **most** of the **physical layer elements**, of its **initial bandwidth part**, of **either Uplink or SUL** (as corresponds), of **common configuration**, of its corresponding **cell** (**special** or **secondary**), of its corresponding **cell group** (**master** or **secondary**). The "**corresponding**" elements are determined by the **specific place**, within the overall **ASN.1 specification**, where this **data type** has been **instantiated**.

#### 2.2.6.9.1 `UplinkConfigCommon` parameters

Its **three parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- **`dummy TimeAlignmentTimer`**: mandatory, common

  It does **not** have a **field description** in the table "*UplinkConfigCommon field descriptions*" of TS 38.331, but several **same-name parameters** have a **field description**.

  It is then **assumed** that this field is **not used**: if received it should be **ignored** by the **UE**.

- **`frequencyInfoUL FrequencyInfoUL`** (click 2.2.6.11): optional, common

  It is the absolute **Uplink frequency configuration** and subcarrier **specific virtual** carriers.

- **`initialUplinkBWP BWP-UplinkCommon`** (click 2.2.8.9): optional, common

  It is the **Uplink initial bandwidth part** configuration for a **PCell** (refer to clause 12 of TS 38.213).

For more detail on these parameters, refer to the table "*UplinkConfigCommon field descriptions*" of TS 38.331.

#### 2.2.6.9.2 `UplinkConfigCommon` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
UplinkConfigCommon ::= SEQUENCE {
  frequencyInfoUL FrequencyInfoUL OPTIONAL,
                                                  -- Cond InterFreqHOAndServCellAdd
  initialUplinkBWP BWP-UplinkCommon OPTIONAL,             -- Cond ServCellAdd
  dummy TimeAlignmentTimer
}
```

### 2.2.6.10 `FrequencyInfoDL` data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** **`CellGroupConfig`** (click 3.7.3), click 2.2.6.3.

**All** its parameters are **common** (click 5.1.3).

**Each** of its **instances** contains the **frequency common** configuration for **Downlink carriers**.

#### 2.2.6.10.1 `FrequencyInfoDL` parameters

Its **four parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- **`absoluteFrequencyPointA ARFCN-ValueNR`**: mandatory, common

  It is the **absolute frequency location** of the **reference resource block** (**CRB0**). The **center frequency** of its **lowest subcarrier** is **Point A** (click 2.2.4.3) (refer to clause 4.4.4.2 of TS 38.211). Note that the **lower edge** of the **actual carrier** is **not** defined by this parameter but **rather** in **`scs-SpecificCarrierList`** (see below).

- `absoluteFrequencySSB` `ARFCN-ValueNR`: optional, common

  If it is **present**, it is the **frequency location** of the **SSB** in this **serving cell**. **SSB** related parameters (e.g., **SSB index**) provided for a **serving cell** refer to this **SSB frequency** unless mentioned otherwise. The **cell-defining SSB** (**CD-SSB**) of the **PCell** is always on the **band synchronization raster** (click 6.1.2.1.1).

  If it is **absent**, the **SSB related parameters** should be **absent** (e.g., `ssb-PositionsInBurst`, `ssb-periodicityServingCell` and `subcarrierSpacing`, all from `ServingCellConfigCommon`, click 3.9.6). If it is **absent**, the **UE** obtains the timing reference from the **SpCell** or an **SCell** if applicable as described in clause 4.1 of TS 38.213. This is only supported in case the **SCell** for which the **UE** obtains the timing reference is in the **same frequency band** as the **cell** (i.e., the **SpCell** or the **SCell**, respectively) from which the **UE** obtains the timing reference.

  For **cells** supporting **reduced capability**, this parameter corresponds to the **cell-defining SSB**.

- `frequencyBandList` `MultiFrequencyBandListNR`: mandatory, common

  It is a **list** containing **only one frequency band** to which this **carrier**(**s**) belongs. Multiple values are **not supported**.

- `scs-SpecificCarrierList` **SEQUENCE (SIZE (**`1..maxSCSs`**)) OF** `SCS-SpecificCarrier` (click 2.2.6.12): mandatory, common

  It contains a set of **carriers** for **different subcarrier spacings**, defined in relation to **Point A** (click 2.2.4.3). The network configures one `SCS-SpecificCarrier` **value** at least for **each different subcarrier spacing** that is used (e.g., in a **bandwidth part**, refer to clause 5.3 of TS 38.211).

  The value of `maxSCSs` is `5`, which is the **maximum quantity** of **carriers** in **each** of the **UE**'s **cells**.

**Parameters two to four** are **almost** the **same** as the corresponding ones from `FrequencyInfoUL` (click 2.2.6.11).

For more detail on these parameters, refer to the table "*FrequencyInfoDL-SIB field descriptions*" of TS 38.331.

#### 2.2.6.10.2 `FrequencyInfoDL` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
FrequencyInfoDL ::= SEQUENCE {
  absoluteFrequencySSB ARFCN-ValueNR OPTIONAL,          -- Cond SpCellAdd
  frequencyBandList MultiFrequencyBandListNR,
  absoluteFrequencyPointA ARFCN-ValueNR,
  scs-SpecificCarrierList
         SEQUENCE (SIZE (1..maxSCSs)) OF SCS-SpecificCarrier
  ...
}
```

Where `maxSCSs` is `5`, which is the **maximum quantity** of **carriers** in **each** of the **UE**'s **cells**.

### 2.2.6.11 `FrequencyInfoUL` data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 2.2.6.3.

**All** its parameters are **common** (click 5.1.3).

**Each** of its **instances** contains the **frequency common** configuration for **Uplink carriers**.

#### 2.2.6.11.1 `FrequencyInfoUL` parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `absoluteFrequencyPointA ARFCN-ValueNR`: optional, common

  It is the **absolute frequency location** of the **reference resource block** (**CRB0**). The **center frequency** of its **lowest subcarrier** is **Point A** (click 2.2.4.3) (refer to clause 4.4.4.2 of TS 38.211). Note that the **lower edge** of the **actual carrier** is **not** defined by this parameter but **rather** in `scs-SpecificCarrierList` (see below).

- `frequencyBandList MultiFrequencyBandListNR`: optional, common

  It is a **list** containing **only one frequency band** to which this **carrier(s)** belongs. Multiple values are **not supported**.

- `frequencyShift7p5kHz ENUMERATED {true}`: optional, common

  Its **field description** is **exactly the same** as the one of its **same-name** parameter from `FrequencyInfoUL-SIB` (click 2.2.6.7).

- `scs-SpecificCarrierList SEQUENCE (SIZE (1..maxSCSs)) OF SCS-SpecificCarrier` (click 2.2.6.12): mandatory, common

  It contains a **set** of carriers for different subcarrier spacings (numerologies). Defined in A set of **carriers** for **different subcarrier spacings**, defined in relation to **Point A** (click 2.2.4.3). The network configures one `SCS-SpecificCarrier` **value** at least for **each different subcarrier spacing** that is used (e.g., in a **bandwidth part**, refer to clause 5.3 of TS 38.211).

  The value of `maxSCSs` is `5`, which is the **maximum quantity** of **carriers** in **each** of the **UE**'s **cells**.

These **three first parameters** are **almost** the **same** as the reciprocal ones within `FrequencyInfoDL` (click 2.2.6.10).

For more detail on these parameters, refer to the table "*FrequencyInfoUL field descriptions*" of TS 38.331.

#### 2.2.6.11.2 `FrequencyInfoUL` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
FrequencyInfoUL ::= SEQUENCE {
  frequencyBandList MultiFrequencyBandListNR OPTIONAL,      -- Cond FDD-OrSUL
  absoluteFrequencyPointA ARFCN-ValueNR OPTIONAL,           -- Cond FDD-OrSUL
  scs-SpecificCarrierList
         SEQUENCE (SIZE (1..maxSCSs)) OF SCS-SpecificCarrier,
  additionalSpectrumEmission AdditionalSpectrumEmission OPTIONAL,   -- Need S
  p-Max P-Max OPTIONAL,                                             -- Need S
  frequencyShift7p5kHz ENUMERATED {true} OPTIONAL,
                                              -- Cond FDD-TDD-OrSUL-Optional
  ...
}
```

Where `maxSCSs` is `5`, which is the **maximum quantity** of **carriers** in **each** of the **UE**'s **cells**.

### 2.2.6.12 `SCS-SpecificCarrier` data type

It can be **both broadcast** (or **unicast**) within `SIB1` (click 5.3.4) and **unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `SIB1`, click 2.2.6.2.

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 2.2.6.3.

**Each** of its **instances** contains **either all common parameters or all dedicated parameters** (click 5.1.3), depending on where it is **instantiated**.

The **literal** description of `SCS-SpecificCarrier` is:

"*The IE SCS-SpecificCarrier provides parameters determining the location and width of the actual carrier or the carrier bandwidth. It is defined specifically for a numerology (subcarrier spacing (SCS)) and in relation (frequency offset) to Point A.*"

#### 2.2.6.12.1 `SCS-SpecificCarrier` parameters

Its **four** parameters (in italics, their **literal** description from TS 38.331) are:

- `carrierBandwidth INTEGER (1..maxNrofPhysicalResourceBlocks)`: mandatory, common **or** dedicated
  "*Width of this carrier in number of PRBs (using the* subcarrierSpacing *defined for this carrier) (see TS 38.211 [16], clause 4.4.2).*"

  It therefore provides the **initial** value for "$N_{carrier,x}^{size,\mu}$" (click 2.2.5) and for "$N_{grid,x}^{size,\mu}$" (click 2.2.5.5). The value of "**μ**" in **both** cases is the one of this **carrier**, which is provided by `subcarrierSpacing` (see 2 bullets below).

  The constant `maxNrofPhysicalResourceBlocks-1` is `274`, which is **maximum quantity**, minus **1**, of **CRBs** in **each** of the **UE**'s **carriers**.

  Notice that the TS text refers to "**PRBs**", but it would be clearer to refer to "**CRBs**".

- `offsetToCarrier INTEGER (0..2199)`: mandatory, common **or** dedicated
  "*Offset in frequency domain between Point A (lowest subcarrier of common RB 0) and the lowest usable subcarrier on this carrier in number of PRBs (using the* subcarrierSpacing *defined for this carrier). The maximum value corresponds to 275*8-1. See TS 38.211 [16], clause 4.4.2.*"

It therefore provides the **initial** value for "$N_{carrier,i,x}^{start,\mu}$" (click 2.2.5) and for "$N_{grid,i,x}^{start,\mu}$" (click 2.2.5.5). The value of the **subcarrier spacing** "$\mu$" is provided by `subcarrierSpacing` (see the next bullet). The value of the **carrier index** "$i$" is provided by the **index**, within the **enclosing** "`SEQUENCE OF`", that corresponds to **this instance** of `SCS-SpecificCarrier`.

Notice that the **units** of the related parameter `offsetToPointA` (click 8.7.3) from `FrequencyInfoDL-SIB` (click 2.2.6.6) are **RBs** of **15 kHz** (i.e., 12•15 kHz) in **FR1** and **RBs** of **60 kHz** (i.e., 12•60 kHz) in **FR2**, which may be **different** to the **units** of **this** parameter.

- `subcarrierSpacing SubcarrierSpacing` (click 2.2.10.5.1): mandatory, common **or** dedicated
  "*Subcarrier spacing of this carrier. It is used to convert the* ***offsetToCarrier*** *into an actual frequency.*

  *Only the following values are applicable depending on the used frequency:*
  *FR1: 15 or 30 kHz*
  *FR2-1: 60 or 120 kHz*
  *FR2-2: 120, 480, or 960 kHz*"

  It therefore provides the value for **numerology** "**μ**", **both** for the corresponding **carrier** (click 2.2.5) and for the **resource grid** (click 2.2.5.5) associated to the **carrier**.

  The specification of its data type is an `ENUMERATED` with **all** the **numerology subcarrier spacing values** (click 2.2.10.3). However, in the specification of this parameter the possible **subcarrier spacing values** are **restricted**.

  Notice that there are **several different same-name parameters** `subcarrierSpacing` (click 2.2.10.5).

- `txDirectCurrentLocation INTEGER (0..4095)`: optional, common **or** dedicated
  "*Indicates the downlink Tx Direct Current location for the carrier. A value in the range 0..3299 indicates the subcarrier index within the carrier. The values in the value range 3301..4095 are reserved and ignored by the* ***UE****. If this field is absent for downlink within* `ServingCellConfigCommon` *and* `ServingCellConfigCommonSIB`*, the* ***UE*** *assumes the default value of 3300 (i.e. "Outside the carrier"). (see TS 38.211 [16], clause 4.4.2). Network does not configure this field via* ServingCellConfig *or for uplink carriers.*"

  For the **DC subcarrier** of **each carrier**, click 2.2.10.2.

  Do not confuse this parameter with its **same-name parameter** from `UplinkTxDirectCurrentBWP` (click 2.2.8.12).

For more detail on these parameters, refer to the table "*SCS-SpecificCarrier field descriptions*" of TS 38.331.

#### 2.2.6.12.2 SCS-SpecificCarrier ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
SCS-SpecificCarrier ::= SEQUENCE {
  offsetToCarrier INTEGER (0..2199),
  subcarrierSpacing SubcarrierSpacing,
  carrierBandwidth INTEGER (1..maxNrofPhysicalResourceBlocks),
  ...,
  [[
  txDirectCurrentLocation INTEGER (0..4095) OPTIONAL              -- Need S
  ]]
}
```

Where **maxNrofPhysicalResourceBlocks-1** is **274**, which is **maximum quantity**, minus **1**, of **PRBs** in **each** of the **UE**'s **bandwidth parts**.

## 2.2.7 Bandwidth part configuration of each carrier

**Each bandwidth part** (**BWP**) is a **range** of **contiguous CRBs** (click 2.2.4.3), associated to a given **carrier** (click 2.2.5), **within** a given **cell**'s **channel**, configured to a **UE**.

There is a **1-to-1 relationship** between **carriers** and **resource grids** (click 2.2.5.6). **Each carrier** defines **1 and only 1 resource grid** and **each resource grid** is defined by **1 and only 1 carrier**. The **currently active** (click 2.2.7.4) **bandwidth part** (click 2.2.7) determines which **frequency segment** of the **carrier** and of its corresponding **resource grid** can be **used**.

**Each bandwidth part** can be characterized by its **identifier** "***i***", its **associated carrier** "***j***", its **starting** (lowest frequency) **CRB** "$N_{BWP,i,x}^{start,\mu}$", its **bandwidth** "$N_{BWP,i}^{size,\mu}$" in **CRBs**, its **role**(**s**) and a few other attributes. **Each bandwidth part inherits** the **characteristics** from its **carrier**: its **carrier type** "***x***" (**DL**, **UL**, **SUL** or **SDL**, click 2.2.7.2), that becomes its **bandwidth part type**, its **subcarrier spacing** "$\mu_{carrier,i,x}$", and for the case of "$\mu$=**2**", also its **cyclic prefix**. Notice that the **SUL bandwidth parts** belong to a **different band** than the **Downlink** and the **Uplink bandwidth parts**, but **all** these **bandwidth parts** belong to the **same cell**. When there is no risk for confusion, the subscript "***x***" may be dropped, resulting in "$N_{BWP,i}^{size,\mu}$".

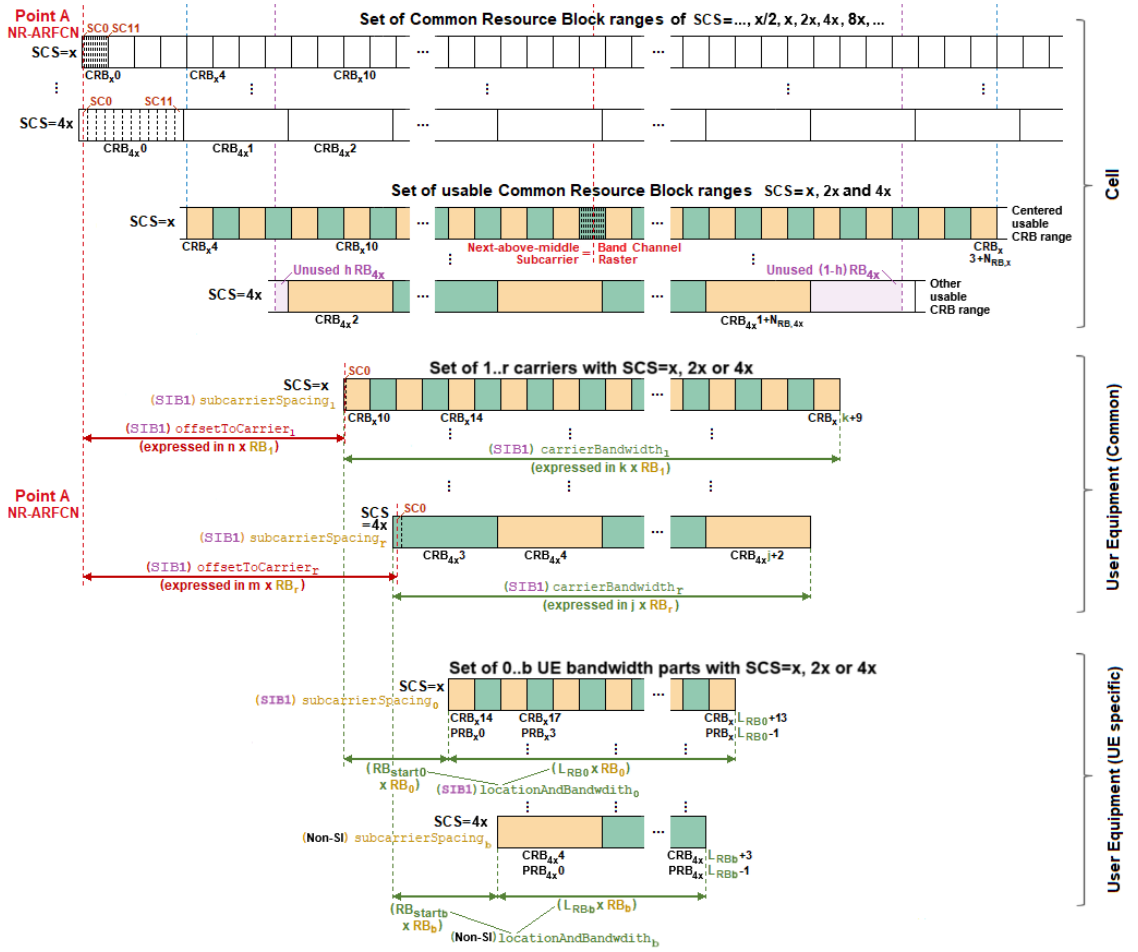
For the **relationship** between the **types** of **bands** (**duplexing mode**), of **channels**, of **CRB ranges**, of **carriers**, of **bandwidth parts** and the **transmission direction**, click 1.1.3.

The **CRBs** of **bandwidth parts** are called "**Physical Resource Blocks** (**PRBs**)". **Each bandwidth parts**' **PRBs** are indexed from **0** (lowest frequency **PRB**) to "$N_{BWP,i}^{size,\mu}$-1" (where "$N_{BWP,i}^{size,\mu}$" is the quantity of **PRBs** in **BWP** "***i***"). Consequently, given a **certain RB**, its **CRB index** and its **PRB index** are **different** (in the general case).

**Each UE**'s **Downlink**, **Uplink**, **SUL** (if configured) and **SDL** (if configured) **bandwidth part** has a configuration determined by the **NR node**. **Each NR node** communicates the corresponding **bandwidth part configuration** to **each** of its **UEs** via **RRC parameters** (click 2.2.8).

There is **one set** of **bandwidth parts** per **bandwidth part type** (**Downlink**, **Uplink**, **SUL** (if configured) and **SDL** (if configured)).

The following figure shows an example of the **CRB ranges**, the **usable CRB ranges**, some **carriers** and some **UE bandwidth parts** configured **within** a given **carrier type**:




Some **relevant remarks** on this figure are:

- In this example, we are configuring this **cell**'s **channel** and a given **carrier type** (**Downlink**, **Uplink**, **SUL** or **SDL**) with the (mandatory) **initial bandwidth part** (**BWP#0**), plus, "**b**" (optional) **additional bandwidth parts**. The value of "**b**" **must** be **lower or equal** than `maxNrofBWPs` (which is `4`). The **parameters** corresponding to **each bandwidth part** are identified with **subindex 0** or **b**.
- For the case of this example (**initial access**), the **initial bandwidth part** (**BWP#0**) configuration is obtained by the **UE** from `SIB1` (click 5.3.4). The configuration of the **UE**'s **additional bandwidth parts** is **always** obtained from the **Non-SI parameter group** (click 5.3.6).
- The **subcarriers spacing** of the different **bandwidth parts** may be different, but it may also be the same. In any case, **each** of their **subcarrier spacings must** be the **one** of one of the **carriers** configured in the **channel** (shown as "**x**" to "**4x**").
- **Each bandwidth part**'s **CRBs/PRBs** are **exactly aligned** with the ones of the **CRB range** of its **same subcarrier spacing**, as they **must**.
- **Each bandwidth part** has its **own specific PRB index**.

- The **frequency bars** of the **CRB ranges**, **carriers** and **bandwidth parts** are drawn to **scale**, where **each contiguous horizontal pixel** is **half the subcarrier spacing** "**x**"(i.e., **$SCS_x/2$**).
- The **half $SCS_x$ kHz** corresponding to the **1-pixel** dividing **lines** between **two elements** (e.g., a **1-pixel line** between **two consecutive subcarriers**, or a **1-pixel line between one subcarrier** and **nothing**) belongs at **50%** to **each** of the **two elements** (i.e, **$SCS_x/4$ belongs** to **each element**).

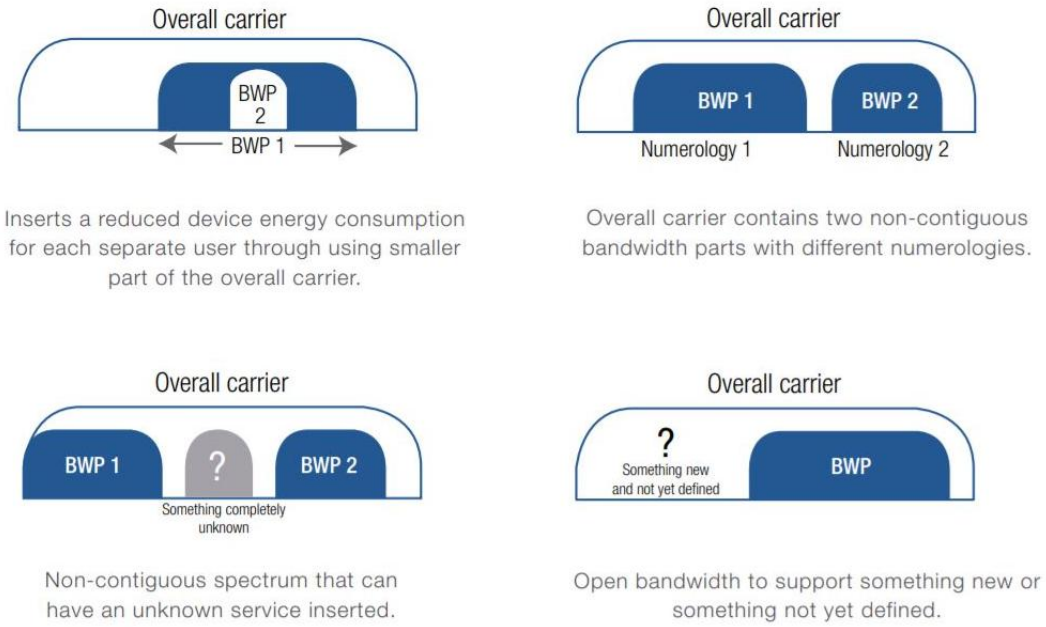
**Bandwidth parts** are used for doing **frequency and time multiplexing** to support **different 5G** (or non-5G) **services** over a **given channel** and **carrier type**. **Each UE** may be configured in **each** of its **cells** and in **each carrier type** with **several bandwidth parts**. **Each** such **bandwidth part** may span a **different frequency range** (**frequency multiplexing**) and **only one** of them is **active** at any given time (**time multiplexing**). Since **only one** of the said **bandwidth parts** is **active** at any given time, their **frequency ranges may overlap**. The **bandwidth part** that is **active** is called the "**active bandwidth part**", and the ones that are **not active** are called the "**inactive bandwidth parts**".

The **frequency** and **time multiplexing** of **bandwidth parts** allows the network to better suite the **requirements/capabilities** of the **different UEs** in the **cell**. For example, using a **bandwidth part** with a **subrange** of the available **bandwidth** allows for **simpler UEs** and **lower UE power consumption**. Since a **cell** may **host UEs** with **substantially different requirements** and/or **capabilities**, the **bandwidth subrange** (i.e., the **bandwidth part**) used can be **dynamically modified** along **time**.

Click 1.1.4 for a **clarifying figure** that shows an **approximate** bidimensional (in **frequency** and **time**) representation of a **resource grid** (click 2.2.5.6) of the **Downlink currently active** (click 2.2.7.4) **bandwidth part** (click 2.2.7), for the duration of **1 fram**e (click 2.3.1), indicating with colors how the **different RBs** (click 2.2.5.7) have been **scheduled** (click 4.11.4) to the **different physical binary channels** (click 4.11.3).

**Each UE** may have **only 1 active bandwidth part** at any given time. However, the **NR node** may have **several simultaneous active bandwidth parts**, where **each** of them serves a (disjoint) **set of UEs**. This allows **dynamic** and **efficient spectrum** usage across **different** types of **services** and **UE needs**. The multiple active bandwidth parts within the same channel allow the **NR node** to provide specific services to different groups of **UEs**, each with its specific capabilities. For example, **one active bandwidth part** may have a **smaller subcarrier spacing** and a **larger bandwidth** to serve a **set** of **eMBB UEs** that demand a large **throughput**, while another **active bandwidth part** with a **larger subcarrier spacing** might serve a **set** of **URLLC UE**s that require **low latency**.

The following figure shows some examples of how a **UE** may be configured with **bandwidth parts** in a given **cell** and **carrier type**:




An example of **BWP** configuration is that one **BWP** may be configured to **UEs** with **reduced energy** requirements, another **BWP** may be configured to **UEs** that require some specific functions or services, and a third **BWP** may provide coexistence with **other systems**.

**Bandwidth parts** are sometimes called "**carrier bandwidth parts**".

For **bandwidth part attributes** and **requirements**, click 2.2.7.1.

For **bandwidth part identification**, **types**, **configuration**, **quantity** and **roles**, click 2.2.7.2.

For **bandwidth interlaces and VRBs**, click 2.2.7.3.

For **bandwidth part switching**, click 2.2.7.4.

**Bandwidth parts** are the **fundamental** element of **bandwidth adaptation** (click 10.1). **Bandwidth adaptation** is a **new 5G** capability, in respect to **LTE**, that does **not** have it.

For more detail on **bandwidth part configuration**, refer to clause 4.4.5 of TS 38.211 and clause 12 of TS 38.213.

For more detail on **bandwidth part operation**, refer to clause 5.15 of TS 38.321.

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_CarrrierBandwidthPart.html
- http://howltestuffworks.blogspot.com/2019/11/5g-nr-bandwidth-part.html
- https://newsletter.mediatek.com/hubfs/mwc/download/bandwidth-part-adaptation.pdf
- https://devopedia.org/5g-nr-bandwidth-part
- Lin et al., 2020: https://arxiv.org/pdf/2004.00761

### 2.2.7.1 Bandwidth part attributes and requirements

#### 2.2.7.1.1 Bandwidth part attributes

A **bandwidth part** is defined by its **identifier** "***i***" (click 2.2.7.2), its **control/shared subcarrier spacing** (2.2.10.3), its **cyclic prefix** (2.3.2), its **start CRB index** "$N_{BWP,i}^{start,\mu}$" and its **bandwidth** as a **quantity** of **PRBs** "$N_{BWP,i}^{size,\mu}$" (click 2.2.7).

The **numerology** of the **carrier** of the **Downlink currently active bandwidth part** is the **PDSCH numerology**. The **numerology** of the **carrier** of the **Uplink** or **SUL currently active bandwidth part** is the **PUSCH numerology**. The **PDSCH** and **PUSCH numerologies** are the ones used (respectively) in the **iDFT** and **DFT functions** of **the NR node** (click 4.5.7). The **same** applies **reciprocally** to the **UE**.

The **control/shared subcarrier spacing** of **each additional BWP must** be the one of a **given carrier** (click 2.2.5), and the **CRBs** of the **BWP must** be a **contiguous subset** of the **CRBs** of that **same carrier**. For the configuration of **cyclic prefix** for **bandwidth parts**, **carriers** and **resource grids**, click 2.3.3.

The relationships between the **CRB index** "$n_{CRB}^{\mu}$" of a **carrier** and the **PRB index** "$n_{PRB,i}^{\mu}$" of its **bandwidth part** "***i***" are the following:

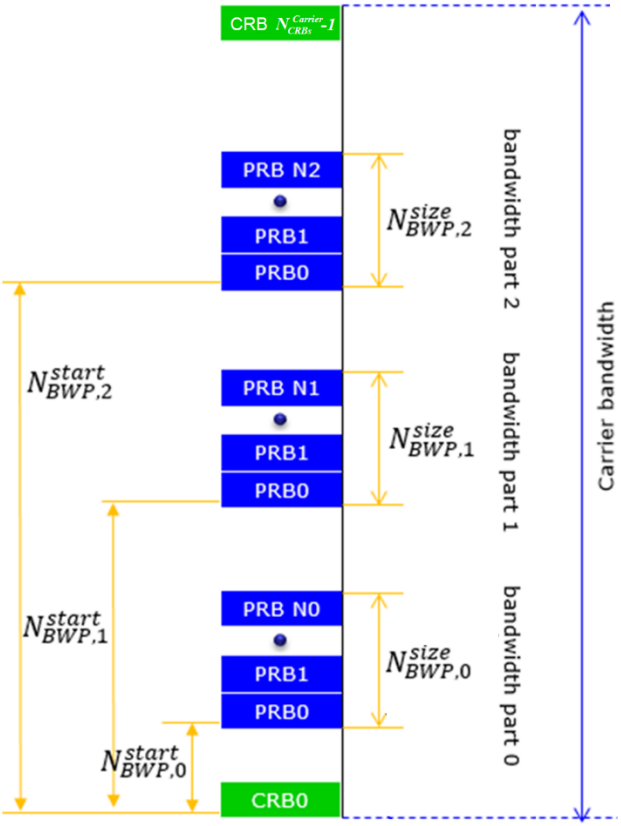
$$n_{CRB}^{\mu} = N_{BWP,i}^{start,\mu} + \mathbf{PRB}_i^{\mu}$$

$$n_{PRB,i}^{\mu} = n_{CRB}^{\mu} - N_{BWP,i}^{start,\mu}$$

Notice that "$n_{PRB,i}^{\mu} = \mathbf{0}$" is always "$n_{CRB}^{\mu} = N_{BWP,i}^{start,\mu}$".

The different **UE**'s **bandwidth parts** configured within to **each pair** of **cell** and **carrier type may overlap** in **frequency**. They do **not interfere** because **only one** of the said **bandwidth parts** can be **active** at **any given time**.

The following figure shows an example of **three bandwidth parts** configured within a **given carrier**. For each **BWP** it is shown its **start CRB**, its **quantity of PRBs** and the **indexes** of its **PRBs**.




Some **relevant remarks** on this figure are:

- For **each BWP**, its **maximum PRB index** is equal to its **PRB size minus 1** (e.g., **N1** = $N_{BWP,I}^{size}$-1).
- The **start** of **each BWP** is the value of a **CRB index**. Therefore, the values of "$N_{BWP,i}^{start}$" are all **CRB indexes**.
- In this example, the **BWPs** do not overlap, but they may overlap.
- In this example, all the **BWPs** have the **same subcarrier spacing**, because all of them correspond to the **same carrier**. However, in the general case they **may have different subcarrier spacings**.
- In this example, the **carrier** starts in **CRB0**, but in a general case it may start in a different **CRB index**.

#### 2.2.7.1.2 Bandwidth part configuration requirements

**The Downlink**, **Uplink** or **SUL bandwidth parts must** be configured to the **UE within** its respective **Downlink**, **Uplink** or **SUL carriers**. This is mathematically expressed as follows:

$$N_{grid,x}^{start,\mu} \leq N_{BWP,i}^{start,\mu} < (N_{grid,x}^{start,\mu} + N_{grid,x}^{size,\mu})$$

$$N_{grid,x}^{start,\mu} < (N_{BWP,i}^{start,\mu} + \text{“}N_{BWP,i}^{size,\mu}\text{”}) \leq (N_{grid,x}^{start,\mu} + N_{grid,x}^{size,\mu})$$

where "$N_{grid,x}^{start,\mu}$" and "$N_{grid,x}^{size,\mu}$" are the **first CRB** and the **CRB size** of the **carrier**, and "$N_{BWP,i}^{start,\mu}$" and "$N_{BWP,i}^{size,\mu}$" are the ones of **bandwidth part** "***i***".

In **every TDD band**, its **Downlink** and **Uplink BWPs** are **paired**. The **two BWPs** in **each** such **DL/UL pair must** have the **same numerology**, the **same cyclic prefix**, the **same center frequency** and **must** be **active/inactive** at the **same time**. **Most frequently** they both have the **same bandwidth**, although it is **possible** that they have a **different bandwidth** (click 2.2.3.7).

**Every Downlink bandwidth part must** be **wide enough** to contain the **SSB**, which requires **20 contiguous RBs** of the **SSB-specific subcarrier spacing**, which **may** be **different** to the **BWP subcarrier spacing**. This is mandatory even if the **SSB** is **not sent** in the **bandwidth part**.

**Every Downlink BWP must** have **at least 1 CORESET** (click 6.8.7) associated to **1 UE-specific search space** (click 6.8.5). **Every initial Downlink BWP must** have at least **1 CORESET** associated to **1 Common search space**.

Within **each** of the **UE**'s **cells** the **subcarrier spacing** and the **cyclic prefix** of the **active Uplink BWP** and of the **active Downlink BWP must** be the **same**. This restriction does **not** apply to the **SUL active BWP**. Refer to page 506 of TS 38.331.

To enable **Bandwidth Adaptation** in the **primary cell**, the **gNB** configures the **UE** with **both Downlink** and **Uplink BWPs**. However, to enable it in **secondary cells**, the **NR node** configures the **UE** with **Downlink BWPs** but **may be not** with **Uplink BWPs**. This is, there **may**, or **may not**, **be** any **Uplink BWP** in a **secondary cell**.

To enable **bandwidth adaptation** (**BA**) on the **primary cell**, the **NR node** configures the **UE** with **Uplink** and **Downlink additional bandwidth part**(**s**). To enable **BA** on **secondary cells** in case of **carrier aggregation** (click 10.2), the **NR node** configures the **UE** with **Downlink additional bandwidth parts**, but there may be **no additional bandwidth parts** in **Uplink** (or **SUL**) **carriers**.

### 2.2.7.2 Bandwidth part identification, types, configuration, quantity and roles

**Bandwidth part identification**

**Each bandwidth part** is **identified** with an **instance** of `BWP-Id` (click 2.2.8.13), which is an `INTEGER (0..maxNrofBWPs)`. The constant `maxNrofBWPs` is `4`, which is the **maximum quantity**, minus **1**, of **UE**'s **bandwidth parts** within **each** of the **UE**'s **Downlink**, **Uplink** or **SUL** in **each** of the **UE**'s **cells**.

This **bandwidth part identifier uniquely** identifies **each** of the **UE**'s **bandwidth parts** within **each** of the **UE**'s **Downlink**, **Uplink** or **SUL**, within **each** of the **UE**'s **cells**.

**Bandwidth part types and subtypes**

Within **each** of the **four mutually exclusive bandwidth part types** which are the **same** as the **carrier type** (**Downlink**, **Uplink**, **SUL** (if configured) or **SDL** (if configured)) to which the **bandwidth part** is associated, there are the **two** following **bandwidth part subtypes**:

- **Initial bandwidth part**

  It **always** has **identifier** `0`. It is **always** configured in the **UE**'s **Downlink**, **Uplink**, **SUL** (if configured) and **SDL** (if configured) **carrier type** of **each** of the **UE**'s **cells**. **All** the **UE**'s in a **cell must** be configured with the **same initial bandwidth part** in **every UE**'s **Downlink carrier type**. The same applies to **every UE**'s **Uplink carrier type**. The same applies to **every UE**'s **SUL carrier type** (if

configured). The same applies to **every UE**'s **SDL carrier type** (if configured). The **initial BWP** is also denoted as **BWP#0**.

- **Additional bandwidth parts**

  Their **identifier** may be `1` to `maxNrofBWPs` (which is `4`). **Each additional bandwidth part** is **specific** to **each UE**'s **Downlink carrier type**, although **several UEs** may **share some/all** of their **Downlink bandwidth parts**. The same applies to **each UE**'s **Uplink carrier type**. The same applies to **each UE**'s **SUL carrier type** (if configured). The same applies to **each UE**'s **SDL carrier type** (if configured).

**Bandwidth part configuration and quantity**

**Each** of the **UE**'s **Downlink**, **Uplink** and **SUL** (if configured) **carrier types** within **each** of the **UE**'s **cells** is configured with the following **bandwidth part types** and **quantities**:

- **1 initial bandwidth part**

  They are configured in **Downlink**, **Uplink** and **SUL** (if configured) during **initial access** (in the **primary cell**) with **1 instance** of `initialDownlinkBWP` and **2 instances** of `initialUplinkBWP` (respectively). These parameters are descendants of `SIB1` (click 5.3.4).

  For more detail on the corresponding configuration parameters, click 2.2.8.1.

  They are configured in **Downlink**, **Uplink** and **SUL** (if configured) **after initial access** (in **each cell**) with **1 instance** of `initialDownlinkBWP` and **2 instances** of `initialUplinkBWP` (respectively). The corresponding **same-named** parameters are descendants **both** of `ServingCellConfigCommon` (click 3.9.6) for common parameters, and of `ServingCellConfig` (click 3.9.4) for dedicated parameters. There is **1 instance** of **both data types** for **each special cell** and for **each secondary cell**.

  For more detail on the corresponding configuration parameters, click 2.2.8.2.

- **Up to 4 additional bandwidth parts**

  They are configured in **Downlink**, **Uplink** and **SUL** (if configured) **after initial access** (in **each cell**) with **1 instance** of `downlinkBWP-ToAddModList` and **2 instances** of `uplinkBWP-ToAddModList`, (respectively). These parameters are descendants of `ServingCellConfig` (click 3.9.4). There is **1 instance** of this **data type** for **each special cell** and for **each secondary cell**.

  For more detail on the corresponding configuration parameters, click 2.2.8.2.

Notice that the **maximum quantity** of **physical layer elements** (**cells**, **bandwidth parts**, **CORESETs**, **search spaces**, etc) that can be **configured** to **each UE** is **not** the same as the **maximum quantity** that a **gNB** can handle.

**Bandwidth part roles**

There are **two roles** that can be assigned through **RRC** to **two different UE BWPs**: the **first active BWP** and the **default Downlink BWP**:

- **First active BWP**

  It is the **BWP** that will **become active** (in **Downlink**, **Uplink** or **SUL** (if configured), as corresponds) right after **RRC establishment** and **configuration**

(click 5.8.3.1) is completed (i.e., when transitioning to **RRC_CONNECTED**, click 5.8.2).

This **role** is assigned to a given **Downlink**, **Uplink** and **SUL** (if configured) **bandwidth part after initial access** (in **each cell**) with `firstActiveDownlinkBWP-Id` and two instances of `firstActiveUplinkBWP-Id` (respectively). These parameters are descendants of `ServingCellConfig` (click 3.9.4). There is **one instance** of this **data type** for **each special cell** and for **each secondary cell**.

For more detail on the corresponding configuration parameters, click 2.2.8.2.

- **Default Downlink BWP**

  It is the **Downlink BWP** that will **become active** right after the **Downlink BWP inactivity timer triggers**. If a specific **default Downlink BWP** is not configured, the **Downlink initial BWP** is the **default Downlink BWP**.

  This **role** is assigned to a given **Downlink bandwidth part after initial access** (in **each cell**) with `defaultDownlinkBWP-Id`. This parameter is a descendant of `ServingCellConfig` (click 3.9.4). There is **one instance** of this **data type** for **each special cell** and for **each secondary cell**.

  For more detail on the corresponding configuration parameters, click 2.2.8.2.

For the **configuration** of **bandwidth part roles**, click 2.2.8.

As indicated in 2.2.7, **only one** of the **UE**'s **banwdith parts** (for **each BWP type**) may be **active** at any given time:

- **Active BWP**

  It is the **UE**'s **BWP** (for **each BWP type**) that is currently in use for **transmission** or **reception**.

- **Inactive BWPs**

  They are **all** the **UE**'s **BWPs** (for **each BWP type**) that are **not** currently in use for **transmission** or **reception**.

For more detail on switching to the **first active BWP**, to the **active BWP** or to the **default Downlink BWP**, click 2.2.7.3.

### 2.2.7.3 Bandwidth part interlaces and virtual resource blocks

**Bandwidth part interlaces**

**Each bandwidth part interlace** is the **mapping** of the corresponding **CRB interlace** (click 2.2.4.7) into **each bandwidth part**.

The **lowest physical resource block index** "$n_{PRB,i}^{\mu}$" (click 2.2.7.1) of **interlace** "$m$" within **bandwidth part** (click 2.2.6.4) "$i$" is:

$$n_{PRB,i}^{\mu} = N_{BWP,i}^{start,\mu} + N_{BWP,i}^{start,\mu} \bmod M_{\mu} + m$$

Therefore, the **physical resource block indexes** "$n_{PRB,i}^{\mu}$" (click 2.2.7.1) of **interlace** "$m$" within **bandwidth part** (click 2.2.6.4) "$i$" are:

$$n_{PRB,i}^{\mu} = N_{BWP,i}^{start,\mu} + N_{BWP,i}^{start,\mu} \bmod M_{\mu} + m + n_{IRB,m}^{\mu} M_{\mu}, \text{ for } n_{IRB,m}^{\mu} = 0, 1, 2, \ldots$$

where "$N_{BWP,i}^{start,\mu}$" is the **CRB index** of the **lowest CRB** of **bandwidth part** "$i$".

The **UE** expects that **each interlace** contains **at least 10 CRBs** within the corresponding **bandwidth part**. This implies that a **bandwidth part** where **interlaces** are defined **must** be **at least 100 CRBs** with **15 kHz subcarrier spacing** and **at least 50 CRBs** with **30 kHz subcarrier spacing**. This implies a **minimum bandwidth part bandwidth** of **18 MHz**.

**Bandwidth part VRBs and CCE mapping**

**Virtual Resource Blocks** (**VRB**s) are defined within a **BWP** in some system contexts (e.g., **PDSCH**, click 4.12.15). The **VRB**s are the **Resource Blocks** in a **virtual** frequency space. In such contexts, the system defines the **mapping** between **each VRB** and **its corresponding PRB** in the **BWP**. **VRBs** and their corresponding mapping to **PRBs** are intended to provide **frequency diversity** (click 11.8).

Notice that the **CRBs** (click 2.2.4.3), the **RBs** from **RB interlaces** (click 2.2.4.6), the **PRBs** and the **VRBs** that have the **same subcarrier spacing** are **all** exactly **aligned** in the **frequency domain**.

For more detail on **VRBs**, refer to clause 4.4.4.5 of TS 38.211.

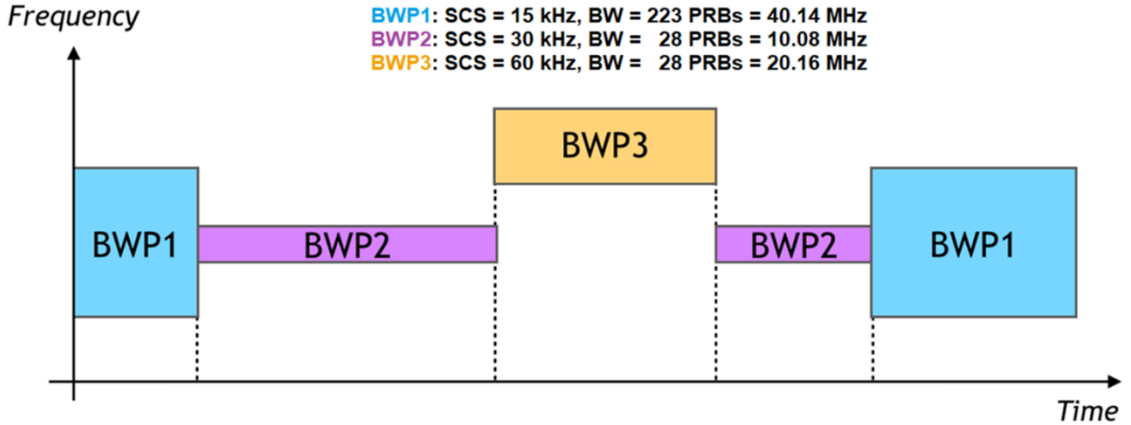
### 2.2.7.4 Bandwidth part switching (active BWP selection)

The procedure to **select** the **active bandwidth part** is called "**bandwidth part switching**".

**Bandwidth part switching overview**

The **active Downlink bandwidth part** is **dynamically** switched **among** the **Downlink bandwidth parts** that have been **configured** (click 2.2.7.2) to the **UE**. The same applies to **Uplink** and to **SUL** (if configured).

The following figure shows an example of **three Uplink BWPs** (with different **start CRB**, **quantity of PRBs** and **subcarrier spacing**) alternatively becoming the **active Uplink BWP** over time.




**Each Uplink** (or **SUL**, if configured) **active UE bandwidth part** (within the **reference** of its corresponding **Uplink** (or **SUL**) **carrier**, click 2.2.5), is an **CP-DFT-s-OFDMA** (click 11.16) **signal** (except in some **specific scenarios**, where it can be a **CP-OFDMA** (click 11.15) **signal**) **produced** by the **joint transmission** of **several UEs**. However, **each Downlink active bandwidth part** is **not a CP-OFDM signal** (click 11.13): rather, it is **the whole Downlink carrier** (click 2.2.5), including **all** the **Downlink active bandwidth parts** that is **one CP-OFDM signal** that is transmitted by the **NR node**.

The **UE** is **not** expected to **receive PDSCH**, **PDCCH**, or **CSI-RS outside** its **active Downlink BWP**. The **UE** shall **not transmit PUSCH** or **PUCCH outside** its **active Uplink** and **SUL bandwidth parts**. For an active cell, the **UE** shall not transmit **Sounding Reference Signals** (**SRS**) **outside** its **active Uplink** and **SUL bandwidth parts**. As an exception, the **UE** may be allowed to send (for Radio Resource Management) **CSI-RS** or **SRS outside** of **its active BWP**, via measurement gap.

**Bandwidth part switching during initial access**

The following table (refer to Mediatek white paper, click 14.6) shows the assignment of **Downlink** and **Uplink bandwidth parts during initial access** (click 8), which is performed in the **UE**'s **primary cell**:

| Sg. | Sp. | DL BWP | UL BWP | Trigger for next step | Comments |
|---|---|---|---|---|---|
| Cell select. | 1 | N/A | N/A | **UE** Rx `MIB` | **UE** is doing **cell search** (click 8.2). |
| | 2 | **CORESET#0** | N/A | **UE** Rx `SIB1` | `MIB` received but contains **no BWPs** => **CORESET#0** is used **as DL BWP** for reception and decoding of `SIB1`. |
| 4-step CBRA | 1 | `SIB1` **initial BWP** | `SIB1` **initial BWP** | **UE** Tx **Msg1** | **Msg1** is **RA preamble** (click 9.3.2). |
| | 2 | `SIB1` **initial BWP** | `SIB1` **initial BWP** | **UE** Rx **Msg2** | **Msg2** is **RA response** (click 9.3.3). |
| | 3 | `SIB1` **initial BWP** | `SIB1` **initial BWP** | **UE** Tx **Msg3** | **Msg3** is `RRCSetupRequest` (click 9.3.4.2). |
| | 4 | `SIB1` **initial BWP** | `SIB1` **initial BWP** | **UE** Rx **Msg4** | **Msg4** is **contention resolution** (click 9.3.5). |
| `RRCSetup` | 1 | **1st Active BWP** | **1st Active BWP** | **N/A** | • **NR node** sends `RRCSetup` (click 9.3.5.2)<br>• Configure **UE-specific BWPs** (default/1st active/other)<br>• If 1st active not configured, use **initial BWPs** |

Some **relevant comments** on this table are:

- In the first column: "**Cell select.**" stands for "the **UE** is doing the **cell selection** procedure" (click 8.1); "**4-step CBRA proc.**" stands for "the **UE** is doing a **4-step contention-based random access procedure**" (click 9.1.3).
- In the first row, "**Sg.**" stands for "**Stage**" and "**Sp.**" stands for "**Step** within the **stage**".
- Before `SIB1` (click 5.3.4) has been acquired, the **UE** does **not know** its **Downlink initial bandwidth part**. However, the **UE requires** to **know** it in order to receive `SIB1` through **PDSCH**. In this very particular case, the **CORESET#0** is used as if it were the **Downlink bandwidth part**.

**Bandwidth part switching after initial access**

**After initial access** the **bandwidth part switching** can be done with one of the following mechanisms:

- **PDCCH DCI messages**
- **RRC signaling**
- **BWP Inactivity Timer**

- **MAC entity**

**Each UE** requires certain **minimum time** to **switch** between **BWPs**. The **minimum switching time** is up to **UE** capability which should be informed to the network via **UE** capability Information.

In **paired bands**, **Downlink** and **Uplink** can switch **BWP** independently. In **unpaired** (i.e., **TDD**) **bands**, **Downlink** and **Uplink** switch **BWP** simultaneously.

For more detail on **BWP switching**, refer to clause 5.15 of TS 38.321.

**BWP switching with DCI messages** (click 6.9)

A specific **BWP** can be **activated** with the **DCI message field** "***Bandwidth part indicator***". The following table shows, for **PDSCH** and for **PUSCH**, the **DCI message Formats** where the said **field** is **present**, what is **assigned** by **each** of these **DCI Formats** and what is the **field**'s **bitlength** in **each** of these **DCI Formats**:

| | **DCI message field "*Bandwidth part indicator*"** | | |
|---|---|---|---|
| | **Assigns** | **DCI Format** | **Bitlength** |
| **Uplink BWP** | **PUSCH Unicast** | Format 0_1 (6.9.3) | 0 to 2 |
| | | Format 0_2 (6.9.4) | 0 to 2 |
| **Downlink BWP** | **PDSCH Unicast** | Format 1_1 (6.9.6) | 0 to 2 |
| | | Format 1_2 (6.9.7) | 0 to 2 |

If a "***Bandwidth part indicator***" field is configured in a **DCI message**, and the **UE supports active BWP change via DCI**, the **UE** switches its **active BWP** to the one indicated by the said field, according to the following table (from Lin et al., 2020):

| **#of RRC configured additional BWPs** | **Bitlength of BWP indicator field** | **BWP indicator** | **bwp-Id** | **Comments** | |
|---|---|---|---|---|---|
| **0** | 0 | Absent | 0 | For Option 1, DCI is not applicable for operation on BWP#0. For Option 2, this is single BWP operation on BWP#0. | |
| **1** | 1 | 0, 1 | 0, 1 | Switch among BWP 0, 1 | For Option 1, once switch to BWP#0 cannot switch out by DCI. |
| **2** | 2 | 00, 01, 10 | 0, 1, 2 | Switch among BWP 0, 1, 2 | |
| **3** | 2 | 00, 01, 10, 11 | 0, 1, 2, 3 | Switch among BWP 0, 1, 2, 3 | |
| **4** | 2 | 00, 01, 10, 11 | 1, 2, 3, 4 | Switch among BWP 1, 2, 3, 4 | |

**BWP switching with RRC signaling** (click 5.8)

When a **UE** transitions from **RRC_IDLE** (click 5.8.2) to **RRC_CONNECTED**, the **RRC** signaling can configure **UE**-specific **BWPs**. **RRC** configuration or reconfiguration message may specify one of these to be **activated**. If so, the **UE** will do **BWP** switching. Due to **RRC** processing delay, this can be in the order of **tens** of **milliseconds**.

**Downlink BWP switching with Downlink BWP inactivity timer**

When there has been **no activity** in the **currently active Downlink bandwidth part** for the duration of the **BWP inactivity timer**, the **UE** automatically switches to the **default Downlink BWP**. The **default Downlink BWP** is configured with **defaultDownlinkBWP-Id** and the **timer value** with **bwp-InactivityTimer**, **both** from the ***DownlinkConfig*** **parameter set** (click 2.2.8.3). The **timer** may be configured to one among some discrete values from **2 ms** to **2,560 ms**. Its maximum value relates to the Discontinuous Reception (**DRX**) inactivity timer.

When the network **releases** the **timer configuration**, the **UE stops** the timer **without switching** to the **default Downlink BWP**.

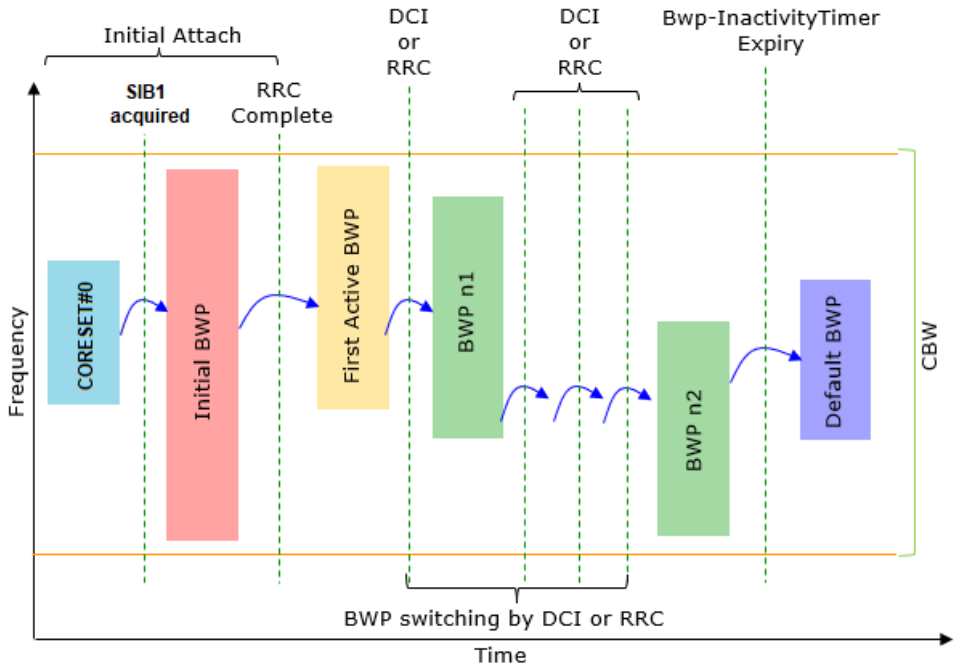
**BWP switching by the MAC entity**

The switching takes place upon initiation of a **random access procedure** (click 9).

**Evolution of the active BWP**

With the different switching mechanisms presented above, a **specific BWP** becomes active depending on various situations.

The following figure shows an example of the evolution of the **active Downlink bandwidth part**.

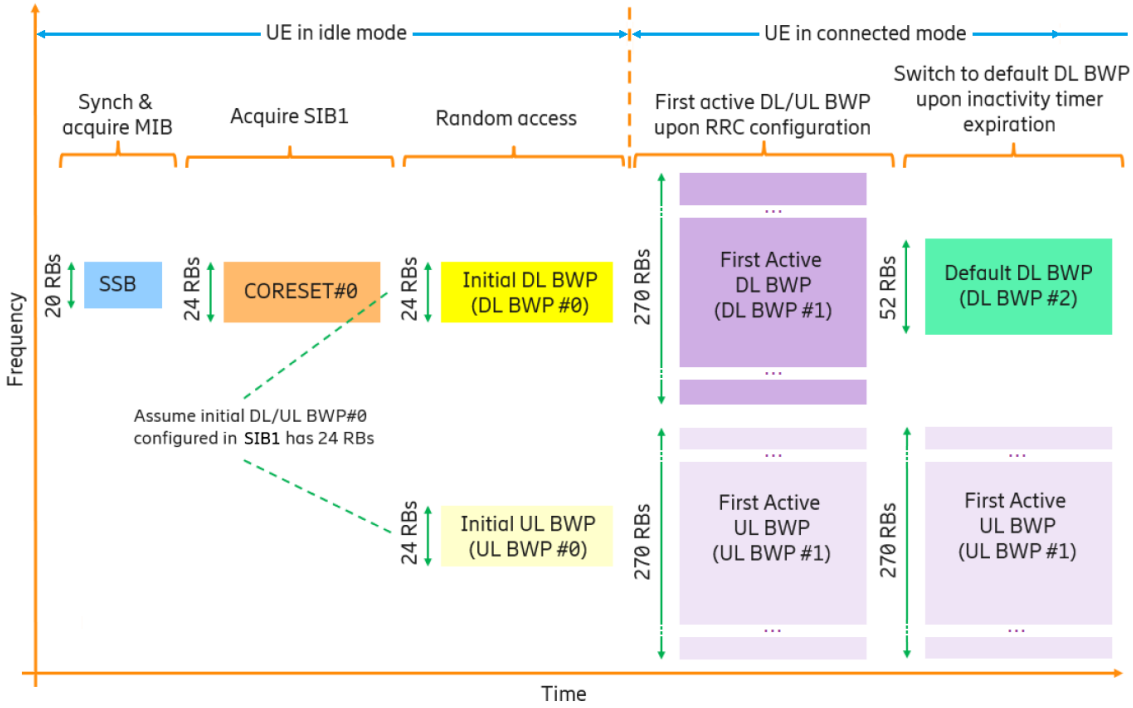



Some **relevant remarks** on this figure are:

- **CORESET#0** is **not** a **bandwidth part**, but it is handled by the **UE** during **initial access** as if it were its **Downlink bandwidth part** in order to get `SIB1` (click 5.3.4) from **PDSCH**.
- The indication "**CBW**" in the figure stands for **carrier bandwidth**.
- **CORESET#0** is (as it **must** be) **within** its **carrier**.
- **Every bandwidth part** is (as it **must** be) **within** its **carrier**.
- Recall that the **First Active BWP must** be different to the **Default BWP**.
- Recall that the **First Active BWP** and the **Default BWP** are **not another BWP**, and **rather**, they are a **role** assigned to one of the **UE**'s **BWPs**. Since in this example the **First Active BWP** and the **Default BWP** are **different** to **BWPs** "**n0**", "**n1**" and "**n2**", it follows that they **must** be **BWPs** "**n3**" and "**n4**" (**not** respectively).
- In this example, **all** the **bandwidth parts** belong to the **same carrier**, (and are located within the carrier bandwidth), but in the general case the **bandwidth parts**

**may** belong to **different carriers**, possibly having **different subcarrier spacing** and/or **cyclic prefix**.

The following figure (from Devopedia) shows an example of the **parallel** evolution of the **active Downlink** and **Uplink bandwidth parts**.




Some **relevant remarks** on this figure are:

- In this example, **CORESET#0** has **24 RBs**, but it in other cases it may have **48** or **96 RBs**.
- Recall that the **RBs** of **SSB**, of **CORESET#0** and of **PDSCH** may be **different** (because they may have a **different subcarrier spacing**).
- The **random access procedure** takes place using the **Downlink** and **Uplink initial BWPs**.
- Once the **RRC connection establishment** is completed, the **UE** can be configured with its **UE-specific additional BWPs**.
- This is an **FDD** example where **Downlink switches** to **DL BWP#2** but **Uplink stays** at **UL BWP#1**. In **TDD**, **BWP switching** happens **together** for **both Downlink** and **Uplink**.

## 2.2.8 Bandwidth part parameters

The **UE**'s **bandwidths parts** (click 2.2.7) are configured with:

- **Common parameters** that are **broadcast** (or **unicast**) within **SIB1**, as described in 2.2.8.1.
- **Common** and **dedicated parameters** that are **unicast** within the **Non-SI parameter group**, as described in 2.2.8.2.

**Bandwidth part** configuration does **not** have **any parameter** within **MIB** (click 5.3.3), because **CSS-Type 0 configuration** is used to **broadcast SIB1** through **PDSCH**.

Possibly existing **bandwidth part parameters** within **other SIBs** and **posSIBs** (click 5.3.5) are **not addressed** in this book.

For an overall explanation on **UE configuration parameters** click 5. For a brief overview of **ASN.1**, click 5.9.

### 2.2.8.1 Bandwidth part parameters within SIB1

The **Downlink initial bandwidth part** and the **Uplink initial bandwidth part** are configured with **common parameters** that are **broadcast** (or **unicast**) within **SIB1** (click 5.3.4), which is sent within **SSB** (click 6.1).

For **Downlink** and **Uplink initial bandwidth parts** parameters within **SIB1** (click 5.3.4), the **most relevant related data types** are:

- **DownlinkConfigCommonSIB** (click 2.2.6.4): common parameters
- **UplinkConfigCommonSIB** (click 2.2.6.5): common parameters
- **BWP-DownlinkCommon** (click 2.2.8.6): common parameters
- **BWP-UplinkCommon** (click 2.2.8.9): common parameters

The following **data type**, **son** of **BWP-DownlinkCommon** (click 2.2.8.6) and of **BWP-UplinkCommon** (click 2.2.8.9), is also relevant:

- **BWP** (click 2.2.8.11): common parameters

For **bandwidth part** parameters **broadcast** (or **unicast**) within **SIB1**, the **ancestor tree** of the **most relevant related data types** (see above) up to **SIB1** is:

```
                                    SIB1 (click 5.7.4)
servingCellConfigCommon               ServingCellConfigCommonSIB (0,1) (click 3.9.3)
  downlinkConfigCommon                  DownlinkConfigCommonSIB (click 2.2.6.4)
    initialDownlinkBWP                    BWP-DownlinkCommon (click 2.2.8.6)
  uplinkConfigCommon                    UplinkConfigCommonSIB (click 2.2.6.5)
    initialUplinkBWP                      BWP-UplinkCommon (click 2.2.8.9)
  supplementaryUplink                   UplinkConfigCommonSIB (click 2.2.6.5)
    initialUplinkBWP                      BWP-UplinkCommon (click 2.2.8.9)
```

For the **ancestor trees** of **SIB1** **up to** the **top** click 5.3.4.

### 2.2.8.2 Bandwidth part parameters within the Non-SI parameter group

**Bandwidth parts** are configured with **common** and **dedicated parameters** that are **unicast** within the **Non-SI parameter group** (click 5.3.6) in the following cases:

- In **the primary cell**, for the **additional bandwidth parts** (i.e., **other** than the **initial bandwidth part**).
- In the **primary-secondary cell** and in **secondary cells**, for **all bandwidth parts**.
- In **the primary cell**, if configuration changes are done **after initial access** (click 8) for the **Downlink initial bandwidth part** and/or for the **Uplink initial bandwidth part**.

For **bandwidth part** parameters within the **Non-SI parameter group** (click 5.3.6), the **most relevant related data types** are:

- **DownlinkConfigCommon** (click 2.2.6.8): common parameters
- **UplinkConfigCommon** (click 2.2.6.9): common parameters

- ***DownlinkConfig*** **parameter set** (click 2.2.8.3): common **and** dedicated parameters
- **UplinkConfig** (click 2.2.8.4): common **and** dedicated parameters
- **BWP-Downlink** (click 2.2.8.5): common **and** dedicated parameters
- **BWP-DownlinkCommon** (click 2.2.8.6): common parameters
- **BWP-DownlinkDedicated** (click 2.2.8.7): dedicated parameters
- **BWP-Uplink** (click 2.2.8.8): common **and** dedicated parameters
- **BWP-UplinkCommon** (click 2.2.8.9): common parameters
- **BWP-UplinkDedicated** (click 2.2.8.10): dedicated parameters
- **BWP-Id** (click 2.2.8.13): dedicated parameters

The following **data type**, **son** of **BWP-DownlinkCommon** (click 2.2.8.6) and of **BWP-UplinkCommon** (click 2.2.8.9) is also relevant:

- **BWP** (click 2.2.8.11): common parameters

The **ancestor tree** (click 5.9.8) of these **most relevant related data types** up to `CellGroupConfig` is:

```
                                CellGroupConfig (click 3.7.3)
spCellConfig                      SpCellConfig (click 3.9.5)
  reconfigurationWithSync       ReconfigurationWithSync (0,1) (click 3.9.8)
    spCellConfigCommon            ServingCellConfigCommon (click 3.9.6)
      downlinkConfigCommon          DownlinkConfigCommon (click 2.2.6.8)
        initialDownlinkBWP            BWP-DownlinkCommon (click 2.2.8.6)
      uplinkConfigCommon            UplinkConfigCommon (click 2.2.6.9)
        initialUplinkBWP              BWP-UplinkCommon (click 2.2.8.9)
      supplementaryUplinkConfig     UplinkConfigCommon (0,1) (click 2.2.6.9)
        initialUplinkBWP              BWP-UplinkCommon (click 2.2.8.9)
    spCellConfigDedicated         ServingCellConfig (click 3.9.7)
      begin of DownlinkConfig parameter set (click 2.2.8.3)
      initialDownlinkBWP            BWP-DownlinkDedicated (click 2.2.8.7)
      downlinkBWP-ToReleaseList     BWP-Id (0..maxNrofBWPs) (click 2.2.8.13)
      downlinkBWP-ToAddModList      BWP-Downlink (0..maxNrofBWPs) (click 2.2.8.5)
        bwp-Id                        BWP-Id (click 2.2.8.13)
        bwp-Common                    BWP-DownlinkCommon (click 2.2.8.6)
        bwp-Dedicated                 BWP-DownlinkDedicated (click 2.2.8.7)
      firstActiveDownlinkBWP-Id     BWP-Id (click 2.2.8.13)
      defaultDownlinkBWP-Id         BWP-Id (click 2.2.8.13)
      end of DownlinkConfig parameter set
      uplinkConfig                  UplinkConfig (0,1) (click 2.2.8.4)
        initialUplinkBWP              BWP-UplinkDedicated (click 2.2.8.10)
        uplinkBWP-ToReleaseList       BWP-Id (0..maxNrofBWPs) (click 2.2.8.13)
        uplinkBWP-ToAddModList        BWP-Uplink (0..maxNrofBWPs) (click 2.2.8.8)
          bwp-Id                        BWP-Id (click 2.2.8.13)
          bwp-Common                    BWP-UplinkCommon (click 2.2.8.9)
          bwp-Dedicated                 BWP-UplinkDedicated (click 2.2.8.10)
        firstActiveUplinkBWP-Id       BWP-Id (click 2.2.8.13)
      supplementaryUplink           UplinkConfig (0,1) (click 2.2.8.4)
        initialUplinkBWP              BWP-UplinkDedicated (click 2.2.8.10)
        uplinkBWP-ToReleaseList       BWP-Id (0..maxNrofBWPs) (click 2.2.8.13)
        uplinkBWP-ToAddModList        BWP-Uplink (0..maxNrofBWPs) (click 2.2.8.8)
          bwp-Id                        BWP-Id (click 2.2.8.13)
          bwp-Common                    BWP-UplinkCommon (click 2.2.8.9)
          bwp-Dedicated                 BWP-UplinkDedicated (click 2.2.8.10)
        firstActiveUplinkBWP-Id       BWP-Id (click 2.2.8.13)
```

```
sCellToAddModList              SCellConfig (0..maxNrofSCells) (click 3.9.4)
  sCellConfigCommon              ServingCellConfigCommon (0,1) (click 3.9.6)
    downlinkConfigCommon           DownlinkConfigCommon (click 2.2.6.8)
      initialDownlinkBWP             BWP-DownlinkCommon (click 2.2.8.6)
    uplinkConfigCommon             UplinkConfigCommon (click 2.2.6.9)
      initialUplinkBWP               BWP-UplinkCommon (click 2.2.8.9)
    supplementaryUplinkConfig      UplinkConfigCommon (0,1) (click 2.2.6.9)
      initialUplinkBWP               BWP-UplinkCommon (click 2.2.8.9)
  sCellConfigDedicated           ServingCellConfig (click 3.9.7)
    begin of DownlinkConfig parameter set (click 2.2.8.3)
    initialDownlinkBWP             BWP-DownlinkDedicated (click 2.2.8.7)
    downlinkBWP-ToReleaseList      BWP-Id (0..maxNrofBWPs) (click 2.2.8.13)
    downlinkBWP-ToAddModList       BWP-Downlink (0..maxNrofBWPs) (click 2.2.8.5)
      bwp-Id                         BWP-Id (click 2.2.8.13)
      bwp-Common                     BWP-DownlinkCommon (click 2.2.8.6)
      bwp-Dedicated                  BWP-DownlinkDedicated (click 2.2.8.7)
    firstActiveDownlinkBWP-Id      BWP-Id (click 2.2.8.13)
    defaultDownlinkBWP-Id          BWP-Id (click 2.2.8.13)
    end of DownlinkConfig parameter set
    uplinkConfig                   UplinkConfig (0,1) (click 2.2.8.4)
      initialUplinkBWP               BWP-UplinkDedicated (click 2.2.8.10)
      uplinkBWP-ToReleaseList        BWP-Id (0..maxNrofBWPs) (click 2.2.8.13)
      uplinkBWP-ToAddModList         BWP-Uplink (0..maxNrofBWPs) (click 2.2.8.8)
        bwp-Id                         BWP-Id (click 2.2.8.13)
        bwp-Common                     BWP-UplinkCommon (click 2.2.8.9)
        bwp-Dedicated                  BWP-UplinkDedicated (click 2.2.8.10)
      firstActiveUplinkBWP-Id        BWP-Id (click 2.2.8.13)
    supplementaryUplink            UplinkConfig (0,1) (click 2.2.8.4)
      initialUplinkBWP               BWP-UplinkDedicated (click 2.2.8.10)
      uplinkBWP-ToReleaseList        BWP-Id (0..maxNrofBWPs) (click 2.2.8.13)
      uplinkBWP-ToAddModList         BWP-Uplink (0..maxNrofBWPs) (click 2.2.8.8)
        bwp-Id                         BWP-Id (click 2.2.8.13)
        bwp-Common                     BWP-UplinkCommon (click 2.2.8.9)
        bwp-Dedicated                  BWP-UplinkDedicated (click 2.2.8.10)
      firstActiveUplinkBWP-Id        BWP-Id (click 2.2.8.13)
```

Some **relevant remarks** on this **ancestor tree** are:

- For the **ancestor trees** of `CellGroupConfig`, **up to** some **RRC NR messages** click 3.7.1 and **up to** some **RRC inter-node messages** click 3.7.2.
- The constant `maxNrofSCells` is `31`, which is the **maximum quantity** of **secondary cells** in **each** of the **UE**'s **cell groups**.
- The constant `maxNrofBWPs` is `4`, which is the **maximum quantity** of **additional bandwidth parts** (i.e., **other** than the **initial bandwidth part**) in **each** of the **UE**'s **bandwidth parts**.
- Notice that the (up to `4`) **instances** of `BWP-Uplink` and `BWP-Downlink` belong to **nameless fields** within **two** different `SEQUENCE OF` (having parameter names `downlinkBWP-ToAddModList` and `uplinkBWP-ToAddModList`).

Recall that **Sidelink** channels (and consequently **Sidelink** data types and parameters) are **not** covered in this book.

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_CarrrierBandwidthPart.html

### 2.2.8.3 *DownlinkConfig* parameter set

Let me first justify why this ***DownlinkConfig*** **parameter set** has been defined in this book.

TS 38.331 specifies the following **data types**:

- `UplinkConfigCommonSIB` (click 2.2.6.5) and
  `DownlinkConfigCommonSIB` (click 2.2.6.4)

  They enclose (respectively) the **Uplink** (or **SUL**) and **Downlink** carrier common configuration. These **2 data types** are instantiated **within** `SIB1` (click 5.3.4). You may click 2.2.6.2 to see their **ancestor tree up to** `SIB1`.

- `UplinkConfigCommon` (click 2.2.6.9) and
  `DownlinkConfigCommon` (click 2.2.6.8)

  They enclose (respectively) **most** of the **Uplink** (or **SUL**) and **Downlink** carrier common configuration. These two data types are instantiated **within** the **Non-SI parameter group** (click 5.3.6). For their **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 2.2.6.3.

- `UplinkConfig` (click 2.2.8.4)

  It encloses **most** of the **Uplink** (or **SUL**) carrier dedicated (and some common) configuration. This data type is instantiated **within** the **Non-SI parameter group** (click 5.3.6). For their **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 2.2.6.3.

Because of what has just been indicated, one would expect to find the data type ~~`DownlinkConfig`~~ which would enclose the carrier dedicated (and some common). configuration for **Downlink**. However, such **data type** does **not exist**. Likewise, the parameter ~~`downlinkConfig`~~, that would be reciprocal to `uplinkConfig` (from `ServingCellConfig`, click 3.9.7) **also does not exist**.

It is convenient that you note this **asymmetry** between **Downlink** and **Uplink** in the structure of the **UE** configuration parameters to avoid confusion.

Since ~~`downlinkConfig`~~ and ~~`DownlinkConfig`~~ do **not exist**, the **common and dedicated parameters** for **Downlink** (equivalent to the ones from `UplinkConfig`) are **directly enclosed** in the data type `ServingCellConfig` (click 3.9.7).

However, for reciprocity with `uplinkConfig` and `supplementaryUplink`, it is considered better to explain the **dedicated** (and some common) **parameters** for **Downlink** grouped in this section, under the term "***DownlinkConfig*** **parameter set**", as if it were the (**non existent**) **data type** ~~`DownlinkConfig`~~.

Presenting the parameters in this way allows a better understanding of their structure, function and location.

The ***DownlinkConfig*** **parameter set** is **unicast** within the **Non-SI parameter group** (click 2.2.8.2).

**Most** of its parameters are **dedicated** and a few are **common** (click 5.1.3).

- **Most** of the **physical layer elements**, of its **Downlink initial bandwidth part**, of **dedicated configuration**.

- Possibly **remove** some **existing Downlink additional bandwidth parts**.
- Possibly **add** some **Downlink additional bandwidth parts**, with **most** of their **physical layer elements**, of **common and dedicated configuration**.
- The **identifier** of the **Downlink first active bandwidth part**.

all of them of its corresponding **cell** (**special** or **secondary**), of its corresponding **cell group** (**master** or **secondary**). The "**corresponding**" elements are determined by the **specific place**, within the overall **ASN.1 specification**, where this **data type** has been **instantiated**.

Notice that there is a proper `UplinkConfig` (click 2.2.8.4) data type that contains the reciprocal configuration for **Uplink** (or **SUL**) **bandwidth parts**.

#### 2.2.8.3.1 *DownlinkConfig* parameter set parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are as follows (grouped by functionality):

Presented in this section on the ***DownlinkConfig*** **parameter set** (but actually belonging to `ServingCellConfig`, click 3.9.7), the parameter for **carrier configuration** (with its data type and semantics) is:

- `downlinkChannelBW-PerSCS-List SEQUENCE (SIZE (1..maxSCSs)) OF SCS-SpecificCarrier` (click 2.2.6.12): optional, dedicated

  It configures a **set** of **UE dedicated channel bandwidth** and **location** configurations for **different subcarrier spacings** (numerologies), defined in relation to **Point A** (click 2.2.4.3).

  The value of `maxSCSs` is `5`, which is the **maximum quantity** of **carriers** in **each** of the **UE**'s **cells**.

  The **UE** uses the configuration provided in this parameter **only** for the purpose of **channel bandwidth** and **location determination**. If this parameter is **absent**, the **UE uses** the configuration in `scs-SpecificCarrierList` from `FrequencyInfoDL-SIB` (click 2.2.6.6), **either** whithin `DownlinkConfigCommonSIB` (click 2.2.6.4) **or** within `DownlinkConfigCommon` (click 2.2.6.8). The network **only** configures **channel bandwidth values** defined in TS 38.101-**1**, **-2** and **-5**.

Presented in this section on the ***DownlinkConfig*** **parameter set** (but actually belonging to `ServingCellConfig`, click 3.9.7), the parameters for **bandwidth part configuration** (indicating for each of them its data type and semantics) are:

- `initialDownlinkBWP BWP-DownlinkDedicated` (click 2.2.8.7): mandatory, dedicated

  It is the **dedicated** (**UE-specific**) configuration for the **Downlink initial bandwidth part** (i.e., **Downlink BWP#0**). The **network always configures** the **UE** with a value for this parameter if **no other bandwidth parts** are configured.

  If any of the optional parameters within `BWP-DownlinkDedicated` are provided, the **UE** considers the **BWP#0** to be an **RRC** configured **BWP** (from **UE** capability viewpoint). Otherwise, the **UE** does not consider the **BWP#0** as an **RRC** configured **BWP** (from **UE** capability viewpoint).

If the **dedicated part** of the **Downlink initial BWP** configuration is absent, the **initial BWP** can be used but with some limitations. For example, changing to another **BWP** requires a `RRCReconfiguration` **message** (click 5.8.5.4) since **DCI** format **1_0** does **not** support **DCI-based switching**.

- `downlinkBWP-ToReleaseList SEQUENCE (SIZE (1..maxNrofBWPs)) OF BWP-Id` (click 2.2.8.13): optional, dedicated

  It contains a **list** of **Downlink additional bandwidth parts** (i.e., **other** than the **Downlink initial bandwidth part**) to be **released** (refer to clause 12 of TS 38.213).

  The value of `maxNrofBWPs` is `4`, which is the **maximum quantity** of **Downlink additional bandwidth parts** in **each** of the **UE**'s **cells**.

- `downlinkBWP-ToAddModList SEQUENCE (SIZE (1..maxNrofBWPs)) OF BWP-Downlink` (click 2.2.8.5): optional, common **and** dedicated

  It contains a **list** of **Downlink additional bandwidth parts** (i.e., **other** than the **Downlink initial bandwidth part**) to be **added** or **modified**, together with their configuration parameters (refer to clause 12 of TS 38.213).

  The value of `maxNrofBWPs` is `4`, which is the **maximum quantity** of **Downlink additional bandwidth parts** in **each** of the **UE**'s **cells**.

  In **TDD bands**, an **Uplink bandwidth part** and a **Downlink bandwidth part** that have the **same identifier** (i.e., the same value of `BandwidthPartId`) are considered as a **bandwidth part pair** and **must** have the **same center frequency**.

- `firstActiveDownlinkBWP-Id BWP-Id` (click 2.2.8.13): mandatory, dedicated

  It is the **BWP** to be active **right after the initial attach** (or **NR** addition) is completed.

  If this parameter is **configured** for an **SpCell**, it contains the **identifier** of the **Downlink BWP** to be **activated** or to be **used** for **RLM**, **BFD** and **measurements** if included in an `RRCReconfiguration` **message** (click 5.8.5.4) contained in an **NR** or **E-UTRA RRC message** indicating that the **SCG** is **deactivated**, upon performing the **RRC** (**re**)**configuration**.

  If this parameter is **configured** for an **SCell**, it contains the **identifier** of the **Downlink BWP** to be used upon **activation** of an **SCell**.

  If this parameter is **absent**, the **RRC** (**re**)**configuration** does **not** impose a **BWP switch**, and:

  - If it is **absent** for the **PSCell** at **SCG deactivation**, the **UE** considers the **previously activated Downlink BWP** as the **BWP** to be used for **RLM**, **BFD** and **measurements**.
  - If it is **absent** for the **PSCell** at **SCG activation**, the **Downlink BWP** to be **activated** is the **Downlink BWP previously** used for **RLM**, **BFD** and **measurements**.

  Upon reconfiguration with `reconfigurationWithSync`, the network sets `firstActiveDownlinkBWP-Id` and `firstActiveUplinkBWP-Id` from `UplinkConfig` (click 2.2.8.4) to the **same value**.

- `defaultDownlinkBWP-Id BWP-Id` (click 2.2.8.13): mandatory, dedicated

  It is the **Downlink BWP** that **UE** and **network** automatically switch to after **no activity** in the **currently active Downlink BWP** for a duration of `bwp-InactivityTimer`. If this parameter is **absent** or it is set to `0`, then the **default Downlink BWP** is the **initial BWP**.

- `bwp-InactivityTimer ENUMERATED`: optional, dedicated

  It is the **inactivity period** to **switch** (click 2.2.7.3) to the **default Downlink bandwidth part** (click 2.2.7.2). If there has been no activity in the currently **active Downlink BWP** (click 2.2.7.2) during this time, the **UE** switches to the **default Downlink BWP**. The value is indicated in **ms**. The value `0.5` ms is only applicable for carriers **above 6 GHz**. When the network releases the timer configuration, the **UE** stops the timer without switching to the **default Downlink BWP**.

  For more detail on **BWP operation**, refer to clause 5.15 from TS 38.321.

Presented in this section on the ***DownlinkConfig*** **parameter set** (but actually belonging to `ServingCellConfig`, click 3.9.7), the parameters for **PDCCH** and **PDSCH configuration** (indicating for each of them its data type and semantics) are:

- `pdcch-ServingCellConfig PDCCH-ServingCellConfig` (click 6.12.10): optional, dedicated

  It does **not** have a **field description** in TS 38.331. Its then **assumed** that it contains **dedicated parameters** that apply to the **PDCCH** contained within **all** the **UE**'s **Downlink bandwidth parts** of the corresponding **UE**'s **serving cell**.

- `pdsch-ServingCellConfig PDSCH-ServingCellConfig` (click 6.18.10): optional, dedicated

  It contains **dedicated parameters** that apply to the **PDSCH** contained within **all** the **UE**'s **Downlink bandwidth parts** of the corresponding **UE**'s **serving cell**.

For more detail on these parameters, refer to the table "*ServingCellConfig field descriptions*" of TS 38.331.

#### 2.2.8.3.2 *DownlinkConfig* parameter set ASN.1 specification

Notice from the explanation in 2.2.8.3 that this is a **subset** of the **ASN.1 specification** of `ServingCellConfig` (click 3.9.7).

```
DownlinkConfig ::=
ServingCellConfig ::= SEQUENCE {
  other ServingCellConfig parameters (not belonging to DownlinkConfig)
  initialDownlinkBWP BWP-DownlinkDedicated OPTIONAL,                -- Need M
  downlinkBWP-ToReleaseList SEQUENCE (SIZE (1..maxNrofBWPs))
                            OF BWP-Id OPTIONAL,                     -- Need N
  downlinkBWP-ToAddModList  SEQUENCE (SIZE (1..maxNrofBWPs))
                            OF BWP-Downlink OPTIONAL                -- Need N
  firstActiveDownlinkBWP-Id BWP-Id OPTIONAL,          -- Cond SyncAndCellAdd
  bwp-InactivityTimer ENUMERATED {ms2, ms3, ms4, ms5, ms6, ms8, ms10,
                                  ms20, ms30, ms40, ms50, ms60, ms80,
                                  ms100, ms200, ms300, ms500, ms750,
                                  ms1280, ms1920, ms2560,
                                  spare10, spare9, spare8, spare7, spare6,
                                  spare5, spare4, spare3, spare2, spare1
                                 } OPTIONAL,                        -- Need R
```

```
    defaultDownlinkBWP-Id BWP-Id OPTIONAL,                          -- Need S
    other ServingCellConfig parameters (not belonging to DownlinkConfig)
    pdcch-ServingCellConfig SetupRelease {PDCCH-ServingCellConfig} OPTIONAL,
                                                                    -- Need M
    pdsch-ServingCellConfig SetupRelease {PDSCH-ServingCellConfig} OPTIONAL,
                                                                    -- Need M
    other ServingCellConfig parameters (not belonging to DownlinkConfig)
  }
```

Where `maxNrofBWPs` is `4`, which is the **maximum quantity** of **additional bandwidth parts** (i.e., **other** than the **initial bandwidth part**) in **each** of the **UE**'s **cells**.

#### 2.2.8.4 `UplinkConfig` data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 2.2.8.22.2.6.3.

It contains **both common parameters** and **dedicated parameters** (click 5.1.3).

**Each** of its **instances** configures the **UE** with:

- **Most** of the **physical layer elements**, of its **either** **Uplink or SUL** (as corresponds) **initial bandwidth part**, of **dedicated configuration**.
- Possibly **remove** some **existing** **either** **Uplink or SUL** (as corresponds) **additional bandwidth parts**.
- Possibly **add** some **either** **Uplink or SUL** (as corresponds) **additional bandwidth parts**, with **most** of their **physical layer elements**, of **common and dedicated configuration**.
- The **identifier** of the **either** **Uplink or SUL first active bandwidth part**.

all of them of its corresponding **cell** (**special** or **secondary**), of its corresponding **cell group** (**master** or **secondary**). The "**corresponding**" elements are determined by the **specific place**, within the overall **ASN.1 specification**, where this **data type** has been **instantiated**.

Notice that the reciprocal data type for **Downlink**, that would be ~~`DownlinkConfig`~~, does **not exist** and the corresponding configuration of **Downlink bandwidth parts** is performed with parameters **directly** **enclosed** in data type `ServingCellConfig` (click 3.9.7). However, for clarity reasons, this book is presenting the reciprocal **Downlink** parameters to the ones from `UplinkConfig` (that are actually enclosed in `ServingCellConfig`) under the ***`DownlinkConfig`*** **parameter set** (click 2.2.8.3), as if it existed.

##### 2.2.8.4.1 `UplinkConfig` parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `firstActiveUplinkBWP-Id` `BWP-Id` (click 2.2.8.13): optional, dedicated

  For the case of a **special cell** (**SpCell**)

  - When **present**, it contains the **ID** of the **Downlink bandwidth part** to be **activated or** to be used for **RLM**, **BFD** and **measurements if included** in an `RRCReconfiguration` **message** (click 5.8.5.4) **contained** in an **NR** or **E-UTRA RRC message** indicating that the **SCG** is deactivated, upon performing the **RRC** (**re**)**configuration**.

- When **absent**, the **RRC (re)configuration** does **not impose** a **BWP switch**. If it is **absent** for the **PSCell** at **secondary cell group deactivation**, the **UE** considers the **previously activated Downlink BWP** as the **BWP** to be used for **RLM**, **BFD** and **measurements**. If it is **absent** for the **PSCell** at **SCG activation**, the **DL BWP** to be **activated** is the **DL BWP previously** to be used for **RLM**, **BFD** and **measurements**.

For the case of a **secondary cell** (**SCell**)

- When **present**, it contains the **ID** of the **Downlink bandwidth part** to be used upon **activation** of the corresponding **SCell**.
- Upon reconfiguration with `reconfigurationWithSync` from `SpCellConfig` (click 3.9.5), the network sets the `firstActiveDownlinkBWP-Id` and `firstActiveUplinkBWP-Id` to the same value.

Its data type `BWP-Id` is just a range of integers from `0` (the **initial** bandwidth part) to `4`.

- `initialUplinkBWP` `BWP-UplinkDedicated` (click 2.2.8.10): optional, dedicated

  It contains the **dedicated** (UE-specific) **parameters** for the **Uplink initial bandwidth part** (i.e., **BWP#0**). If **any** of its **optional parameters** are configured (as part of the parameter `uplinkConfig`), the **UE** considers the **BWP#0** to be an **RRC configured BWP** (from **UE capability viewpoint**). Otherwise, the **UE** does **not consider** the **BWP#0** as an **RRC configured BWP** (from **UE capability viewpoint**). The network **always** configures the **UE** with a value for this parameter if **no other BWPs** are **configured**.

  If the **dedicated part** of the **Uplink BWP** configuration is **absent**, the **initial BWP** can be used **but** with some **limitations**. For example, **changing** to another **BWP** requires `RRCReconfiguration` (click 5.8.5.4) since **DCI Format 1_0** does **not** support **DCI-based switching**.

- `pusch-ServingCellConfig` `PUSCH-ServingCellConfig` (click 7.15.8): optional, dedicated

  It configures **PUSCH related parameters**, that are **not bandwidth part specific**, for the **serving cell**.

- `uplinkBWP-ToAddModList` `SEQUENCE (SIZE (1..maxNrofBWPs)) OF` `BWP-Uplink` (click 2.2.8.8): optional, common **and** dedicated

  It contains a **list** of **Uplink additional bandwidth parts** to be **added** or **modified**, together with their configuration parameters.

  The **Uplink additional bandwidth parts** are the ones **other** than the **Uplink initial bandwidth part**. The value of `maxNrofBWPs` is `4`, which is the **maximum quantity** of **Uplink additional bandwidth parts** in **each** of **the UE**'s **cells**.

  In **TDD bands**, an **Uplink bandwidth part** and a **Downlink bandwidth part** that have the **same identifier** (i.e., the same value of `BandwidthPartId`) are considered as a **bandwidth part pair** and **must** have the **same center frequency**.

- **uplinkBWP-ToReleaseList SEQUENCE (SIZE (1..maxNrofBWPs)) OF BWP-Id** (click 2.2.8.13): optional, dedicated

  It contains a list of **Uplink additional bandwidth parts** to be **released**.

  The **Uplink additional bandwidth parts** are the ones **other** than the **Uplink initial bandwidth part**. The value of **maxNrofBWPs** is **4**, which is the **maximum quantity** of **Uplink additional bandwidth parts** in **each** of **the UE**'s **cells**.

- **uplinkChannelBW-PerSCS-List SEQUENCE (SIZE (1..maxSCSs)) OF SCS-SpecificCarrier** (click 2.2.6.12): optional, dedicated

  It is a set of **channel bandwidth** and **location** configurations for different **subcarrier spacing**s (numerologies). Defined in relation to **Point A**. The **UE** uses this parameter only for the purpose of **channel bandwidth** and **location** determination. If it is **absent**, the **UE** uses **scs-SpecificCarrierList** (either from **UplinkConfigCommon** (click 2.2.6.9) or from **UplinkConfigCommonSIB** (click 2.2.6.5)). The **NR node** only configures **channel bandwidth** that corresponds to the **channel bandwidth values** defined in TS 38.101-**1**, **-2** and **-5**. The constant **maxSCSs** is **5**, which is the **maximum quantity** of **carriers** in **each** of **the UE**'s **cells**.

For more detail on these parameters, refer to the table "*UplinkConfig field descriptions*" of TS 38.331.

#### 2.2.8.4.2 UplinkConfig ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
UplinkConfig ::= SEQUENCE {
  initialUplinkBWP        BWP-UplinkDedicated OPTIONAL,             -- Need M
  uplinkBWP-ToReleaseList SEQUENCE (SIZE (1..maxNrofBWPs)) OF BWP-Id
                          OPTIONAL,                                 -- Need N
  uplinkBWP-ToAddModList  SEQUENCE (SIZE (1..maxNrofBWPs)) OF BWP-Uplink
                          OPTIONAL,                                 -- Need N
  firstActiveUplinkBWP-Id BWP-Id OPTIONAL,             -- Cond SyncAndCellAdd
  pusch-ServingCellConfig SetupRelease {PUSCH-ServingCellConfig} OPTIONAL,
                                                                    -- Need M
  carrierSwitching SetupRelease {SRS-CarrierSwitching} OPTIONAL,    -- Need M
  ...,
  [[
  powerBoostPi2BPSK BOOLEAN OPTIONAL,                               -- Need M
  uplinkChannelBW-PerSCS-List SEQUENCE (SIZE (1..maxSCSs))
                              OF SCS-SpecificCarrier OPTIONAL       -- Need S
  ]],
  [[
  enablePL-RS-UpdateForPUSCH-SRS-r16 ENUMERATED {enabled} OPTIONAL,
                                                                    -- Need R
  enableDefaultBeamPL-ForPUSCH0-0-r16 ENUMERATED {enabled} OPTIONAL,
                                                                    -- Need R
  enableDefaultBeamPL-ForPUCCH-r16 ENUMERATED {enabled} OPTIONAL,   -- Need R
  enableDefaultBeamPL-ForSRS-r16 ENUMERATED {enabled} OPTIONAL,     -- Need R
  uplinkTxSwitching-r16 SetupRelease {UplinkTxSwitching-r16} OPTIONAL,
                                                                    -- Need M
  mpr-PowerBoost-FR2-r16 ENUMERATED {true} OPTIONAL                 -- Need R
  ]]
}
```

Where **maxNrofBWPs** is **4**, which is the **maximum quantity** of **additional bandwidth parts** (i.e., other than the **initial bandwidth part**), and **maxSCSs** is **5**, which is the **maximum quantity** of **carriers**, both of them in **each** of **the UE**'s **cells**.

#### 2.2.8.5 `BWP-Downlink` data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 2.2.8.2.

It contains **both common parameters** and **dedicated parameters** (click 5.1.3).

**Each** of its **instances** contains the configuration parameters for **one Downlink** (either **initial** or **additional**) **bandwidth part**.

##### 2.2.8.5.1 `BWP-Downlink` parameters

Its **three parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `bwp-Common` `BWP-DownlinkCommon` (click 2.2.8.6): optional, common

  It configures the **common parameters** of the corresponding **Downlink bandwidth part**.

- `bwp-Dedicated` `BWP-DownlinkDedicated` (click 2.2.8.7): optional, dedicated

  It configures the **dedicated parameters** of the corresponding **Downlink bandwidth part**.

- `bwp-Id` `BWP-Id` (click 2.2.8.13): mandatory, common **or** dedicated

  It contains the **Downlink bandwidth part identifier** to which the other two parameters apply. If its value is `0`, (i.e., for the **Downlink initial bandwidth part**) it is common. Otherwise, it is dedicated.

For more detail on these parameters, refer to the table "*BWP-Downlink field descriptions*" of TS 38.331.

##### 2.2.8.5.2 `BWP-Downlink` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
BWP-Downlink ::= SEQUENCE {
  bwp-Id          BWP-Id,
  bwp-Common      BWP-DownlinkCommon OPTIONAL,          -- Cond SetupOtherBWP
  bwp-Dedicated BWP-DownlinkDedicated OPTIONAL,         -- Cond SetupOtherBWP
  ...
}
```

#### 2.2.8.6 `BWP-DownlinkCommon` data type

It is **only broadcast** (or **unicast**) within `SIB1` (click 5.3.4).

For its **ancestor tree up to** `SIB1`, click 2.2.8.1.

**All** its parameters are **common** (click 5.1.3).

**Each** of its **instances** contains the **common** configuration parameters for the **Downlink initial bandwidth part** or for **one** of the **up to** `4` **Downlink additional bandwidth parts**.

**Each** of its **instances** configures the **UE** with **most** of the **physical layer elements**, of a **bandwidth part**, of **Downlink**, of **common configuration**, of its corresponding **cell** (**special** or **secondary**), of its corresponding **cell group** (**master** or **secondary**). The "**corresponding**" elements are determined by the **specific place**, within the overall **ASN.1 specification**, where this **data type** has been **instantiated**.

The **common** parameters of the **initial bandwidth part** of the **primary cell** are also provided within `SIB1` (click 5.3.4). For **all other serving cells**, the **network** provides the **common** parameters via **dedicated signaling**.

The network ensures the necessary alignment with corresponding parameters of other **UEs**.

#### 2.2.8.6.1 `BWP-DownlinkCommon` parameters

Its **three parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `genericParameters` `BWP` (click 2.2.8.11): mandatory, common

  It contains the **common configuration** for the corresponding **Downlink bandwidth part**.

- `pdcch-ConfigCommon` `PDCCH-ConfigCommon` (click 6.12.4): optional, common

  It contains the **common configuration** for **PDCCH** in the corresponding **Downlink bandwidth part**.

- `pdsch-ConfigCommon` `PDSCH-ConfigCommon` (click 6.18.5): optional, common

  It contains the **common configuration** for **PDSCH** in the corresponding **Downlink bandwidth part**.

For more detail on these parameters, refer to the table "*BWP-DownlinkCommon field descriptions*" of TS 38.331.

#### 2.2.8.6.2 `BWP-DownlinkCommon` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
BWP-DownlinkCommon ::= SEQUENCE {
  genericParameters BWP,
  pdcch-ConfigCommon SetupRelease {PDCCH-ConfigCommon} OPTIONAL, -- Need M
  pdsch-ConfigCommon SetupRelease {PDSCH-ConfigCommon} OPTIONAL, -- Need M
  ...
}
```

### 2.2.8.7 `BWP-DownlinkDedicated` data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 2.2.8.2.

**All** its parameters are **dedicated** (click 5.1.3).

**Each** of its **instances** configures the **UE** with **most** of the **physical layer elements**, of its corresponding **bandwidth part** (either **initial** or **additional**), of **Downlink**, of **dedicated configuration**, of its corresponding **cell** (**special** or **secondary**), of its corresponding **cell group** (**master** or **secondary**). The "**corresponding**" elements are determined by the **specific place**, within the overall **ASN.1 specification**, where this **data type** has been **instantiated**.

#### 2.2.8.7.1 `BWP-DownlinkDedicated` parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `pdcch-Config` `PDCCH-Config` ( click 6.12.5): optional, dedicated

  It contains the **dedicated configuration** for **PDCCH** in the corresponding **Downlink bandwidth part**.

- `pdsch-Config` `PDSCH-Config` (click 6.18.6): optional, dedicated

  It contains the **dedicated configuration** for **PDSCH** in the corresponding **Downlink bandwidth part**.

- `sps-Config` `SPS-Config`: optional, dedicated

  It contains the **UE-specific** configuration for **PDSCH semi-persistent scheduling** (**SPS**, click 6.14.1) for the **bandwidth part** to which this **instance** of `BWP-DownlinkDedicated` corresponds.

  **Except** for `ReconfigurationWithSync` (click 3.9.8) the **NR node** does **not reconfigure** `sps-Config` when there is an **active PDSCH assignment** (click 4.11.4.3 and refer to TS 38.321). However, the **NR node** can **release** the `sps-Config` at **any time**. The **NR node** can **only** configure **PDSCH semi-persistent scheduling** in a **given Downlink bandwidth part** using **either** this parameter or `sps-ConfigToAddModList-r16` (also from `BWP-DownlinkDedicated`).

  Its **data type** `SPS-Config` is a **`SEQUENCE`** with different parameters.

Notice that it **does not** have the parameter `genericParameters` (with data type `BWP`), as it is the case of `BWP-DownlinkCommon` (click 2.2.8.6).

For more detail on these parameters, refer to the table "*BWP-DownlinkDedicated field descriptions*" of TS 38.331.

#### 2.2.8.7.2 `BWP-DownlinkDedicated` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
BWP-DownlinkDedicated ::= SEQUENCE {
  pdcch-Config SetupRelease {PDCCH-Config} OPTIONAL,                -- Need M
  pdsch-Config SetupRelease {PDSCH-Config} OPTIONAL,                -- Need M
  sps-Config   SetupRelease {SPS-Config} OPTIONAL,                  -- Need M
  radioLinkMonitoringConfig SetupRelease {RadioLinkMonitoringConfig}
                            OPTIONAL,                               -- Need M
  ...,
  [[
  sps-ConfigToAddModList-r16 SPS-ConfigToAddModList-r16 OPTIONAL,   -- Need N
  sps-ConfigToReleaseList-r16 SPS-ConfigToReleaseList-r16 OPTIONAL,
                                                                    -- Need N
  sps-ConfigDeactivationStateList-r16 SPS-ConfigDeactivationStateList-r16
                                      OPTIONAL,                     -- Need R
  beamFailureRecoverySCellConfig-r16 SetupRelease {
                BeamFailureRecoveryRSConfig-r16} OPTIONAL,  -- Cond SCellOnly
  sl-PDCCH-Config-r16     SetupRelease {PDCCH-Config} OPTIONAL,     -- Need M
  sl-V2X-PDCCH-Config-r16 SetupRelease {PDCCH-Config} OPTIONAL
                                                                    -- Need M
  ]],
```

```
    [[
    preConfGapStatus-r17 BIT STRING (SIZE (maxNrofGapId-r17)) OPTIONAL,
                                                        -- Cond PreConfigMG
    beamFailureRecoverySpCellConfig-r17 SetupRelease {
                BeamFailureRecoveryRSConfig-r16} OPTIONAL,  -- Cond SpCellOnly
    harq-FeedbackEnablingforSPSactive-r17 BOOLEAN OPTIONAL,           -- Need R
    cfr-ConfigMulticast-r17 SetupRelease {CFR-ConfigMulticast-r17}
                            OPTIONAL,                                 -- Need M
    dl-PPW-PreConfigToAddModList-r17 DL-PPW-PreConfigToAddModList-r17
                                     OPTIONAL,                        -- Need N
    dl-PPW-PreConfigToReleaseList-r17 DL-PPW-PreConfigToReleaseList-r17
                                      OPTIONAL,                       -- Need N
    nonCellDefiningSSB-r17 NonCellDefiningSSB-r17 OPTIONAL,           -- Need R
    servingCellMO-r17 MeasObjectId OPTIONAL         -- Cond MeasObject-NCD-SSB
    ]]
}
```

### 2.2.8.8 BWP-Uplink data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** **CellGroupConfig** (click 3.7.3), click 2.2.8.2.

It contains **both common parameters** and **dedicated parameters** (click 5.1.3).

**Each** of its **instances** contains the configuration parameters for **one Uplink** or **SUL** (either **initial** or **additional**) **bandwidth part**.

#### 2.2.8.8.1 BWP-Uplink parameters

Its **three parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- **bwp-Common** **BWP-UplinkCommon** (click 2.2.8.9): optional, common

  It configures the **common parameters** of the corresponding **Uplink** or **SUL bandwidth part**.

- **bwp-Dedicated** **BWP-UplinkDedicated** (click 2.2.8.10): optional, dedicated

  It configures the **dedicated parameters** of the corresponding **Uplink** or **SUL bandwidth part**.

- **bwp-Id** **BWP-Id** (click 2.2.8.13): mandatory, common **or** dedicated

  It contains the **Uplink or SUL bandwidth part identifier** to which the other two parameters apply. If its value is **0**, (i.e., for the **Uplink** or **SUL initial bandwidth part**) it is common. Otherwise, it is dedicated.

For more detail on these parameters, refer to the table "*BWP-Uplink field descriptions*" of TS 38.331.

#### 2.2.8.8.2 BWP-Uplink ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
BWP-Uplink ::= SEQUENCE {
  bwp-Id          BWP-Id,
  bwp-Common      BWP-UplinkCommon OPTIONAL,          -- Cond SetupOtherBWP
  bwp-Dedicated BWP-UplinkDedicated OPTIONAL,         -- Cond SetupOtherBWP
  ...
}
```

### 2.2.8.9 BWP-UplinkCommon data type

It is **only broadcast** (or **unicast**) within **SIB1** (click 5.3.4).

For its **ancestor tree up to** `SIB1`, click 2.2.8.1.

**All** its parameters are **common** (click 5.1.3).

**Each** of its **instances** contains the **common** configuration parameters for the **Uplink initial bandwidth part** or for **one** of the **up to** `4` **Uplink additional bandwidth parts**.

Its parameters provide the **UE** with the **configuration** for **most** of the **physical layer elements**, of its corresponding **bandwidth part** (**initial or additional**), of **either Uplink or SUL** (as corresponds), of **common configuration**, of its corresponding **cell** (**special** or **secondary**), of its corresponding **cell group** (**master** or **secondary**). The "**corresponding**" elements are determined by the **specific place**, within the overall **ASN.1 specification**, where this **data type** has been **instantiated**.

#### 2.2.8.9.1 `BWP-UplinkCommon` parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `genericParameters` `BWP` (click 2.2.8.11): mandatory, common

  It contains the **common** configuration for the **bandwidth part** to which the parameter having this **data type** refers to.

- `rach-ConfigCommon` `RACH-ConfigCommon` (click 9.5.3): optional, common

  It contains the **common** configuration for the **PRACH** of this **bandwidth part**.

- `pusch-ConfigCommon` `PUSCH-ConfigCommon` (click 7.15.4): optional, common

  It contains the **common** configuration for the **PUSCH** of this **bandwidth part**.

- `pucch-ConfigCommon` `PUCCH-ConfigCommon` (click 7.9.3): optional, common

  It contains the **common** configuration for the **PUCCH** of this **bandwidth part**.

- `useInterlacePUCCH-PUSCH-r16` `ENUMERATED` `{enabled}`: optional, common

  When **present**, **PUSCH-PUCCH interlace** (click 2.2.4.7) is **set**. This affects **both** the **PUCCH configuration** (click 7.6.5.3) and the **PUSCH configuration** (click 7.11.3).

  Notice that there is a **same-name** (**dedicated**) **parameter**, with **similar** but **slightly different** semantics, from `BWP-UplinkDedicated` (click 2.2.8.10).

  The **UE** expects that `useInterlacePUCCH-PUSCH-r16` is **present either** from **none** or from **both** `BWP-UplinkCommon` (click 2.2.8.9) and `BWP-UplinkDedicated` (click 2.2.8.10).

For more detail on these parameters, refer to the table "*BWP-UplinkDedicated field descriptions*" of TS 38.331.

#### 2.2.8.9.2 BWP-UplinkCommon ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
BWP-UplinkCommon ::= SEQUENCE {
  genericParameters BWP,
  rach-ConfigCommon  SetupRelease {RACH-ConfigCommon} OPTIONAL,     -- Need M
  pusch-ConfigCommon SetupRelease {PUSCH-ConfigCommon} OPTIONAL,    -- Need M
  pucch-ConfigCommon SetupRelease {PUCCH-ConfigCommon} OPTIONAL,    -- Need M
  ...,
  [[
  rach-ConfigCommonIAB-r16 SetupRelease {RACH-ConfigCommon}
                           OPTIONAL,                                -- Need M
  useInterlacePUCCH-PUSCH-r16 ENUMERATED {enabled} OPTIONAL,        -- Need R
  msgA-ConfigCommon-r16 SetupRelease {MsgA-ConfigCommon-r16}
                        OPTIONAL                                    -- Cond SpCellOnly2
  ]],
  [[
  enableRA-PrioritizationForSlicing-r17 BOOLEAN
                                        OPTIONAL,     -- Cond RA-PrioSliceAI
  additionalRACH-ConfigList-r17
              SetupRelease {AdditionalRACH-ConfigList-r17}
              OPTIONAL,                                 -- Cond SpCellOnly2
  rsrp-ThresholdMsg3-r17 RSRP-Range OPTIONAL,                       -- Need R
  numberOfMsg3-RepetitionsList-r17
        SEQUENCE (SIZE (4)) OF NumberOfMsg3-Repetitions-r17
        OPTIONAL,                                               -- Cond Msg3Rep
  mcs-Msg3-Repetitions-r17 SEQUENCE (SIZE (8)) OF INTEGER (0..31)
                           OPTIONAL                             -- Cond Msg3Rep
  ]]
}
```

### 2.2.8.10 BWP-UplinkDedicated data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 2.2.8.2.

**All** its parameters are **dedicated** (click 5.1.3).

Its **2 parent data types** are `UplinkConfig` (click 2.2.8.4) and `BWP-Uplink` (click 2.2.8.8).

**Each** of its **instances** configures the **UE** with **most** of the **physical layer elements**, of its corresponding **bandwidth part** (**initial or additional**), of **either Uplink or SUL** (as corresponds), of **dedicated configuration**, of its corresponding **cell** (**special** or **secondary**), of its corresponding **cell group** (**master** or **secondary**). The "**corresponding**" elements are determined by the **specific place**, within the overall **ASN.1 specification**, where this **data type** has been **instantiated**.

#### 2.2.8.10.1 BWP-UplinkDedicated parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `configuredGrantConfig ConfiguredGrantConfig`: optional, dedicated

  It contains the **UE-specific** configuration for **PUSCH semi-persistent scheduling** (**SPS**, click 6.14.1) for the **bandwidth part** to which this **instance** of `BWP-UplinkDedicated` corresponds.

  It contains a **PUSCH configured grant** of **type1** or **type2**. It may be configured for **Uplink** and/or for **SUL** but in the case of **type1 not** for **both** at a time.

**Except** for `ReconfigurationWithSync` (click 3.9.8), the **NR node** does **not reconfigure** `configuredGrantConfig` when there is an **active configured PUSCH grant type 2** (click and refer to TS 38.321). However, the **NR node** can **release** `configuredGrantConfig` at **any time**. The **NR node** can **only** configure a **configured grant** in a **given bandwidth part** using **either** this parameter or `configuredGrantConfigToAddModList-r16` (also from `BWP-UplinkDedicated`).

Its **data type** `ConfiguredGrantConfig` is a large **`SEQUENCE`** with many different parameters.

- `pucch-Config PUCCH-Config` (click 7.9.4): optional, dedicated

  Each of its **instances** defines a) several parameters for **all PUCCH resources**; b) a certain amount of **individual PUCCH resources** (click 3.2.1); and c) a certain amount of **PUCCH resource sets** (click 4.3.2.4), all of it for the corresponding **UE**'s **Uplink bandwidth part**.

  **Detailed configuration aspects of `pucch-Config`**

  If the **UE** is configured with **SUL**, the **gNB** configures **PUCCH only** on the **BWPs** of **one** of the **Uplinks** (either normal **UL** or **SUL**). The **gNB configures** `pucch-Config` at least on **non-initial BWP(s)** for **SpCell** and on **all BPW(s)** for **PUCCH SCell**. If supported by the **UE**, the **gNB** may configure **at most one additional SCell** of a cell group with `pucch-Config` (i.e. **PUCCH SCell**). If **PUCCH cell switching** is supported by the **UE**, the **gNB** may configure **two TDD serving cells** with `pucch-Config` within **each PUCCH group**. For supporting **PUCCH cell switching** in the **PUCCH group** with the **SpCell**, the **TDD SpCell** and one **TDD SCell** shall have `pucch-Config` on their normal **Uplink**. For supporting **PUCCH cell switching** in the **PUCCH group** with **only SCells**, **two TDD SCells** shall have `pucch-Config` on their normal **Uplink**.

  In **(NG)EN-DC** and **NE-DC**, the **gNB** configures **at most one** serving cell per frequency range with **PUCCH**. In **(NG)EN-DC** and **NE-DC**, if **two PUCCH groups** are configured, the serving cells of the **NR PUCCH group** in **FR2** use the **same numerology**. For **NR-DC**, the **maximum quantity of PUCCH groups** in **each** of the **UE**'s **cell groups** is **1**, and **only the same numerology** is supported for the **cell group** with **carriers only in FR2**.

  The **gNB** may configure **PUCCH** for a **bandwidth part** when setting up the **bandwidth part**. The network may also add/remove the `pucch-Config` in an `RRCReconfiguration` **message** (click 5.8.5.4) with `reconfigurationWithSync` (for **SpCell** or **PUCCH SCell**) or with **SCell release** and add (for **PUCCH SCell**) to move the **PUCCH** between the **Uplink** and the **SUL carrier** of one serving **cell**. In other cases, only modifications of a previously configured `pucch-Config` are allowed.

  If **one** **(S)UL bandwidth part** of a serving **cell** is **configured** with **PUCCH**, **all other** **(S)UL bandwidth parts must** also be **configured** with **PUCCH**.

- `pucch-ConfigMulticast1-r17 PUCCH-Config` (click 7.9.4): optional, dedicated

  It is the **PUCCH configuration** for the **HARQ-ACK codebook** for **MBS multicast** when **multicast feedback** is **not** configured with a **priority value** (refer to clause 9 of TS 38.213). If this parameter is **absent**, then `pucch-Config` (see above) applies.

- `pucch-ConfigMulticast2-r17 PUCCH-Config` (click 7.9.4): optional, dedicated

  It is the **PUCCH configuration** for the **NACK-only feedback** for **MBS multicast** when **multicast feedback** is **not configured** with a **priority value** (refer to clause 9 of TS 38.213). If this parameter is **absent**, then `pucch-Config` (see above) applies.

- `pucch-ConfigurationList-r16 PUCCH-ConfigurationList-r16` (click 7.9.5): optional, dedicated

  It contains the **PUCCH configurations** for **2 simultaneously constructed HARQ-ACK codebooks** (refer to clause 9.1 of TS 38.213). **Different PUCCH Resource IDs** are configured in **different PUCCH-Config** within the **list**.

  This is used for **UCI reporting of different priorities** (click 7.6.8.2).

  Its **data type** `PUCCH-ConfigurationList-r16` is a `SEQUENCE (SIZE (1..2)) OF PUCCH-Config` (click 7.9.4).

- `pucch-ConfigurationListMulticast1-r17 PUCCH-ConfigurationList-r16` (click 7.9.5): optional, dedicated

  It contains the **PUCCH configurations** for **two simultaneously constructed HARQ-ACK codebooks** for **MBS multicast** (refer to clause 9 of TS 38.213).

  This is used for **UCI reporting of different priorities** (click 7.6.8.2).

  Its **data type** `PUCCH-ConfigurationList-r16` is a `SEQUENCE (SIZE (1..2)) OF PUCCH-Config` (click 7.9.4).

- `pucch-ConfigurationListMulticast2-r17 PUCCH-ConfigurationList-r16` (click 7.9.5): optional, dedicated

  It contains the **PUCCH configurations** for **two simultaneously constructed NACK-only feedback** for **MBS multicast** (refer to clause 9 of TS 38.213).

  This is used for **UCI reporting of different priorities** (click 7.6.8.2).

  Its **data type** `PUCCH-ConfigurationList-r16` is a `SEQUENCE (SIZE (1..2)) OF PUCCH-Config` (click 7.9.4).

- `pusch-Config PUSCH-Config` (click 7.15.5): optional, dedicated

  It defines the **PUSCH configuration** for the corresponding **bandwidth part** of the normal **Uplink** or **SUL** carrier of the corresponding **cell**. If the **UE** is configured with **SUL** and if it has a `PUSCH-Config` for both **Uplink** and **SUL**, the ***UL/SUL indicator*** field in **DCI** indicates which of the two to use. Refer to clause 7.3.1 of TS 38.212.

- `sl-PUCCH-Config-r16` `PUCCH-Config` (click 7.9.4): optional, dedicated

  It is the **dedicated PUCCH configuration** used for the **HARQ-ACK** feedback reporting for **NR Sidelink** communication.

- `useInterlacePUCCH-PUSCH-r16` `ENUMERATED {enabled}`: optional, dedicated

  When **present**, **PUSCH-PUCCH interlace** (click 2.2.4.7) is **set**. This affects **both** the **PUCCH configuration** (click 7.6.5.3) and the **PUSCH configuration** (click 7.11.3).

  The **UE** expects that this parameter is configured **either** in **all Uplink bandwidth parts** of a **serving cell** or in **none** of them. Therefore, this parameter is **either** configured for **all** the **PUCCH resources** of a **serving cell** or for **none** of them.

  Notice that there is a **same-name** (**common**) **parameter**, with **similar** but **slightly different** semantics, from `BWP-UplinkCommon` (click 2.2.8.9).

  The **UE** expects that `useInterlacePUCCH-PUSCH-r16` is **present either** from **none** or from **both** `BWP-UplinkCommon` (click 2.2.8.9) and `BWP-UplinkDedicated` (click 2.2.8.10).

Notice that it **does not** contain the parameter `genericParameters` (that has data type `BWP` (click 2.2.8.11)), while `BWP-UplinkCommon` (2.2.8.9) contains it.

For more detail on these parameters, refer to the table "*BWP-UplinkDedicated field descriptions*" of TS 38.331.

### 2.2.8.10.2 `BWP-UplinkDedicated` ASN.1 specification

**Part** of its **ASN.1** (click 5.9) specification is:

```
BWP-UplinkDedicated ::= SEQUENCE {
  pucch-Config           SetupRelease {PUCCH-Config} OPTIONAL,       -- Need M
  pusch-Config           SetupRelease {PUSCH-Config} OPTIONAL,       -- Need M
  configuredGrantConfig  SetupRelease {ConfiguredGrantConfig}
                         OPTIONAL,                                   -- Need M
  srs-Config SetupRelease {SRS-Config} OPTIONAL,                     -- Need M
  sl-PUCCH-Config-r16 SetupRelease {PUCCH-Config} OPTIONAL,          -- Need M
  beamFailureRecoveryConfig SetupRelease {BeamFailureRecoveryConfig}
                            OPTIONAL,                    -- Cond SpCellOnly
  ...,
  [[
  sl-PUCCH-Config-r16 SetupRelease {PUCCH-Config} OPTIONAL,          -- Need M
  cp-ExtensionC2-r16 INTEGER (1..28) OPTIONAL,                       -- Need R
  cp-ExtensionC3-r16 INTEGER (1..28) OPTIONAL,                       -- Need R
  useInterlacePUCCH-PUSCH-r16 ENUMERATED {enabled} OPTIONAL,         -- Need R
  pucch-ConfigurationList-r16
       SetupRelease {PUCCH-ConfigurationList-r16} OPTIONAL,          -- Need M
  lbt-FailureRecoveryConfig-r16 SetupRelease
              {LBT-FailureRecoveryConfig-r16 } OPTIONAL,             -- Need M
  configuredGrantConfigToAddModList-r16
               ConfiguredGrantConfigToAddModList-r16 OPTIONAL,       -- Need N
  configuredGrantConfigToReleaseList-r16
               ConfiguredGrantConfigToReleaseList-r16 OPTIONAL,      -- Need N
  configuredGrantConfigType2DeactivationStateList-r16
      ConfiguredGrantConfigType2DeactivationStateList-r16 OPTIONAL -- Need R
  ]],
  [[
```

```
    other parameters (not shown)
    pucch-ConfigurationListMulticast1-r17
         SetupRelease {PUCCH-ConfigurationList-r16} OPTIONAL,        -- Need M
    pucch-ConfigurationListMulticast2-r17
         SetupRelease {PUCCH-ConfigurationList-r16} OPTIONAL         -- Need M
    ]],
    [[
    pucch-ConfigMulticast1-r17 SetupRelease {PUCCH-Config} OPTIONAL, -- Need M
    pucch-ConfigMulticast2-r17 SetupRelease {PUCCH-Config} OPTIONAL  -- Need M
    other parameters (not shown)
}
```

#### 2.2.8.11 BWP data type

It can be **both broadcast** (or **unicast**) within **SIB1** (click 5.3.4) and **unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to SIB1**, click 2.2.8.1.

For its **ancestor tree up to CellGroupConfig** (click 3.7.3), click 2.2.8.2.

**All** its parameters are **common** (click 5.1.3).

**Each** of its **instances** contains **three common** configuration parameters for the **initial bandwidth part** or for **one** of the **up to 4 additional bandwidth parts**. It applies to **an Uplink bandwidth part** or to a **Downlink bandwidth part**.

##### 2.2.8.11.1 BWP parameters

Its **three** parameters (indicating for each of them its data type and semantics), with definitions **copied** from page 506, are:

- **cyclicPrefix ENUMERATED{extended}**: optional, common

  "*Indicates whether to use the extended cyclic prefix for this bandwidth part. If not set, the **UE** uses the normal cyclic prefix. Normal CP is supported for all subcarrier spacings and slot formats. Extended CP is supported only for 60 kHz subcarrier spacing. (see TS 38.211 [16], clause 4.2). Except for SUL, the network ensures the same cyclic prefix length is used in active DL BWP and active UL BWP within a serving cell.*"

  For the configuration of **cyclic prefix** for **bandwidth parts**, **carriers** and **resource grids**, click 2.3.3.

- **locationAndBandwidth INTEGER (0..37949)**: mandatory, common

  "*Frequency domain location and bandwidth of this bandwidth part. The value of the field shall be interpreted as resource indicator value (RIV) as defined in TS 38.214 [19] with assumptions as described in TS 38.213 [13], clause 12, i.e. setting size $N_{RB}^{BWP}$=275. The first PRB is a PRB determined by **subcarrierSpacing** of this BWP and **offsetToCarrier** (configured in **SCS-SpecificCarrier** contained within **FrequencyInfoDL** / **FrequencyInfoUL** / **FrequencyInfoUL-SIB** / **FrequencyInfoDL-SIB** within **ServingCellConfigCommon** / **ServingCellConfigCommonSIB**) corresponding to this subcarrier spacing. In case of TDD, a BWP-pair (UL BWP and DL BWP with the same **bwp-Id**) must have the same center frequency (see TS 38.213 [13], clause 12).*"

It contains a **resource indicator value** (**RIV**), defined as follows:

*If* $L_{RBs} \leq 138$ *then*

$$RIV = 275\,(L_{RBs} - 1) + RB_{start}$$

*else*

$$RIV = 275\,(276 - L_{RBs}) + (274 - RB_{start})$$

Notice that there is a **same-name parameter** from `LocationAndBandwidthBroadcast-r17`.

- `subcarrierSpacing SubcarrierSpacing` (click 2.2.10.5.1): mandatory, common

  "*Subcarrier spacing to be used in this BWP for all channels and reference signals unless explicitly configured elsewhere. Corresponds to subcarrier spacing according to TS 38.211 [16], table 4.2-1. The value kHz15 corresponds to μ=0, value kHz30 corresponds to μ=1, and so on.*

  *Only the following values are applicable depending on the used frequency:*

  *FR1: 15, 30, or 60 kHz*

  *FR2-1: 60 or 120 kHz*

  *FR2-2: 120, 480, or 960 kHz*

  *For the initial DL BWP and operation in licensed spectrum this field has the same value as the field* ***`subCarrierSpacingCommon`*** *in MIB of the same serving cell. Except for SUL, the network ensures the same subcarrier spacing is used in active DL BWP and active UL BWP within a serving cell. For the initial DL BWP and for shared spectrum, the value of this field corresponds to the subcarrier spacing of the SSB associated to the initial DL BWP.*"

  Be aware that it has **several same-name parameters** (click 2.2.10.5).

For more detail on these parameters, refer to the table "*BWP field descriptions*" of TS 38.331.

#### 2.2.8.11.2 `BWP` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
BWP ::= SEQUENCE {
  locationAndBandwidth INTEGER (0..37949),
  subcarrierSpacing SubcarrierSpacing,
  cyclicPrefix ENUMERATED { extended } OPTIONAL -- Need R
}

SubcarrierSpacing ::= ENUMERATED {kHz15, kHz30, kHz60, kHz120, kHz240,
                                  kHz480-v1700, kHz960-v1700, spare1}
```

### 2.2.8.12 `UplinkTxDirectCurrentBWP` data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

Specifically, it is **unicast either** within the `RRCReconfigurationComplete` **message** (click 5.8.5.4) or within the `RRCResumeComplete` **message** (click 5.8.5.4). Its **ancestor tree up to** the **top** is:

```
                                UL-DCCH-Message (click 5.8.5.9)
message                           UL-DCCH-MessageType
  c1                                CHOICE
    rrcReconfigurationComplete RRCReconfigurationComplete (0..1)
      criticalExtensions             CHOICE
        rrcReconfigurationComplete RRCReconfigurationComplete-IEs (0..1)
          nonCriticalExtension       RRCReconfigurationComplete-v1530-IEs
            uplinkTxDirectCurrentList  UplinkTxDirectCurrentList (0..1)
                                         UplinkTxDirectCurrentCell
                                                    (1..maxNrofServingCells)
              uplinkDirectCurrentBWP     UplinkTxDirectCurrentBWP
                                                      (1.. maxNrofBWPs)
              uplinkDirectCurrentBWP-SUL UplinkTxDirectCurrentBWP
                                                      (1.. maxNrofBWPs)
    rrcResumeComplete              RRCResumeComplete (0..1)
      criticalExtensions             CHOICE
        rrcResumeComplete              RRCResumeComplete-IEs (0..1)
          uplinkTxDirectCurrentList  UplinkTxDirectCurrentList (0..1)
                                       UplinkTxDirectCurrentCell
                                                  (1..maxNrofServingCells)
            uplinkDirectCurrentBWP     UplinkTxDirectCurrentBWP
                                                    (1.. maxNrofBWPs)
            uplinkDirectCurrentBWP-SUL UplinkTxDirectCurrentBWP
                                                    (1.. maxNrofBWPs)
```

Notice that it is **not** a **descendant** of `CellGroupConfig` (click 3.7.3).

**All** its **parameters** are **dedicated** (click 5.1.3).

**Each** of its **instances** indicates the **exact location** of the **DC subcarrier** (click 2.2.10.2) (respectively) of the **enclosing carrier** of **1 Uplink bandwidth part** or **1 SUL bandwidth part**. It also indicates if **there is**, or **not**, a **7.5 kHz shift** (click 11.15) for the **enclosing carrier**.

#### 2.2.8.12.1 `UplinkTxDirectCurrentBWP` parameters

Its **three parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `bwp-Id Bwp-Id`: mandatory, common **or** dedicated

  It is the **Uplink** or **SUL bandwidth part identifier** to which this instance of `UplinkTxDirectCurrentBWP` applies.

- `shift7dot5kHz BOOLEAN`: mandatory, common **or** dedicated

  If "`true`", the location of the **carrier** of the corresponding **Uplink** or **SUL bandwidth part** is **offset** by **7.5 kHz** (click 11.15) relative to the **center** of the **subcarrier** indicated by `txDirectCurrentLocation` (see the next bullet).

- `txDirectCurrentLocation INTEGER (0..3301)`: mandatory, common **or** dedicated

  It indicates the **location**, within the **Uplink** or **SUL carrier**, of the **DC subcarrier** (click 2.2.7.1).

  Values in the range `0` to `3,299` represent the **subcarrier index** (within the **Uplink** or **SUL carrier**) of the **DC subcarrier**. Value `3,300` indicates that the **DC subcarrier** is located **outside** of the **Uplink** or **SUL carrier**. Value `3,301`

indicates that the location of the **Uplink** or **SUL bandwidth part DC subcarrier** is **undetermined**.

For the **DC subcarrier**, click 2.2.10.2.

Do not confuse this parameter with its **same-name parameter** from `SCS-SpecificCarrier` (click 2.2.6.12), that indicates the **DC subcarrier** for the **Downlink carriers**.

#### 2.2.8.12.2 `UplinkTxDirectCurrentBWP` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
UplinkTxDirectCurrentBWP ::= SEQUENCE {
  bwp-Id BWP-Id,
  shift7dot5kHz BOOLEAN,
  txDirectCurrentLocation INTEGER (0..3301)
}
```

### 2.2.8.13 `BWP-Id` data type

It can be **both broadcast** (or **unicast**) within `SIB1` (click 5.3.4) and **unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `SIB1`, click 2.2.8.1.

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 2.2.8.2.

**All** the parameters to which it is assigned are **dedicated**.

**Each** of its **instances** is an **identifier** of a **bandwidth part**. Other parts of the **RRC** configuration use the `BWP-Id` to associate themselves with a **particular bandwidth part**.

Its **complete ASN.1** (click 5.9) specification is:

```
BWP-Id ::= INTEGER (0..maxNrofBWPs)
```

where `maxNrofBWPs` is `4`, which is the **maximum quantity** of **additional bandwidth parts** (i.e., **other** than the **initial bandwidth part**) in **each** of **the UE**'s **cells**.

The `BWP-Id`=`0` is always associated with the **initial bandwidth part** and hence it **cannot** be used for any **additional bandwidth part**.

The **network** may trigger the **UE** to switch the **Uplink** or the **Downlink bandwidth part** using a **DCI message**. The **four code** points in that **DCI message** map to the **RRC-configured** `BWP-Id` depending on **how many additional bandwidth parts** has the **UE** been configured with in the corresponding **UE**'s **cell**:

- If **0 to 3**, the **DCI code point** is the **same** as the `BWP-Id`, (i.e., initial = 0, first additional = 1, second additional = 2 and third additional = 3)
- If **4**, the **DCI code point** is the `BWP-Id` **minus 1** (i.e., first additional = 0, second additional = 1, third additional = 2 and fourth additional = 3). Notice that in this case it is **not** possible to **switch** to the **initial BWP** using the **DCI message field**.

## 2.2.9 Resource block group (RBG) configuration of each bandwidth part

**Each** and **every bandwidth part** is configured in terms of **resource block groups** (**RBGs**). The **RBGs** are used in the **frequency-domain resource allocation Type 0** (i.e.,

**bitmap-based) both** for **PDSCH** (**DFA-Type 0**, click 6.14.3.2) and for **PUSCH** (**UFA-Type 0**, click 7.11.3.2).

An **RBG** is a certain **quantity** of **consecutive Resource Blocks** (**RBs**).

**All** the **RBGs** in **each** and **every bandwidth part** are **contiguous** and **exactly span** the **complete bandwidth part**.

**Size of intermediate RBGs**

The **intermediate RBGs** are **all** of them, **except** the **lowest frequency RBG** and the **highest frequency RBG. Each** and **every** the **intermediate RBG** has the **same size** "***P***" in terms of **RBs**. The **lowest frequency RBG** and/or the **highest frequency RBG might** have a **smaller size** than the **intermediate RBGs**.

The **size** "***P***" of **intermediate RBGs** is determined by the **bandwidth part size** "", parameter `rbg-Size` and a table from TS 38.214. Parameter `rbg-Size` may indicate **either** "`config1`" or "`config2`". For **PDSCH**, `rbg-Size` is from `PDSCH-Config` (click 6.18.6) and the table is 5.1.2.2.1-1. For **PUSCH**, `rbg-Size` is from `PUSCH-Config` (click 6.18.6) and the table is 6.1.2.2.1-1. The table that determines the **size** "***P***" of **intermediate RBGs** is the following:

| **Bandwidth part size [PRBs]** | ***P*** **= RBG size [PRBs]** | |
|---|---|---|
| | `rbg-Size = config1` | `rbg-Size = config2` |
| **1 – 36** | **2** | **4** |
| **37 - 72** | **4** | **8** |
| **73 - 144** | **8** | **16** |
| **145 - 275** | **16** | **16** |

**Location of RBGs**

The **RBGs** are always **aligned** to the **common resource blocks** (**CRBs**, click 2.2.4.3). In case the corresponding **bandwidth part** includes $CRB_0$, its **lowest frequency RBG** spans from $CRB_0$ to $CRB_{P-1}$ (both included). In case the corresponding **bandwidth part** does **not** include $CRB_0$, **extending downwards** in **frequency** its **intermediate RBGs**, its **lowest frequency extended intermediate RBG** spans from $CRB_0$ to $CRB_{P-1}$ (both included).

**Quantity of RBGs**

For the corresponding **bandwidth part**, its **quantity** "$N_{RBG}$" of **RBGs** is given by the expression:

$$N_{RBG} = ceiling((N_{RB}^{BWP} + N_{BWP}^{start} \bmod P) / P)$$

where "$N_{BWP}^{start}$" is the **CRB index** of the **lowest frequency PRB** (i.e., $PRB_0$) and "$N_{RB}^{BWP}$" is the **size** (in **RBs**) of the **bandwidth part**.

Notice that if the **size** of the **bandwidth part** is a **multiple** of "***P***" (i.e., "$N_{RB}^{BWP} \bmod P = 0$") and the **bandwidth part** is **aligned** to the **RBGs** (i.e., "$N_{BWP}^{start} \bmod P = 0$"), its **quantity** "$N_{RBG}$" of **RBGs** is:

$$N_{RBG} = N_{RB}^{BWP} / P$$

In this case, **all** the **RBGs** in the **bandwidth part** have **size** "***P***".

**Size of the lowest and highest RBGs**

For the corresponding **bandwidth part**, the **size** "$RBG_0^{size}$" of its **lowest frequency RBG** is:

$$RBG_0^{size} = P - (N_{BWP}^{start} \bmod P)$$

For the corresponding **bandwidth part**, the **size** "$RBG_0^{size}$" of its **lowest frequency RBG** is obtained as:

***If*** "$(N_{BWP,i}^{size} + N_{BWP}^{start}) \bmod P = 0$"

***then***

$$RBG_{last}^{size} = P$$

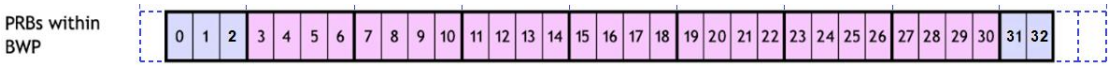
***else***

$$RBG_{last}^{size} = (N_{RB}^{BWP} + N_{BWP}^{start}) \bmod P$$

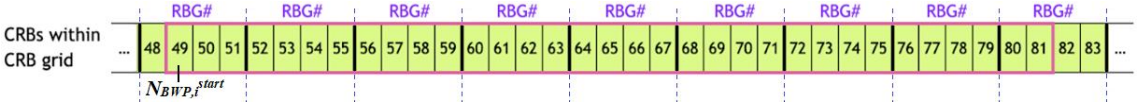
Notice that if the **size** of the **bandwidth part** is a **multiple** of "$P$" (i.e., "$N_{RB}^{BWP} \bmod P = 0$") and the **bandwidth part** is **aligned** to the **RBGs** (i.e., "$N_{BWP}^{start} \bmod P = 0$"), **both** the **size** "$RBG_0^{size}$" of the **lowest frequency RBG** and the **size** "$RBG_{last}^{size}$" of the **highest frequency RBG** is "$P$".

**Example of RBGs**

In this example, the **bandwidth part** has **33 PRBs**, "$N_{BWP,i}^{start}$=49" (i.e., "**PRB**$_0$=**CRB**$_{49}$") and `rbg-size` is **set** to `config2`. The following figure shows this **bandwidth part**:

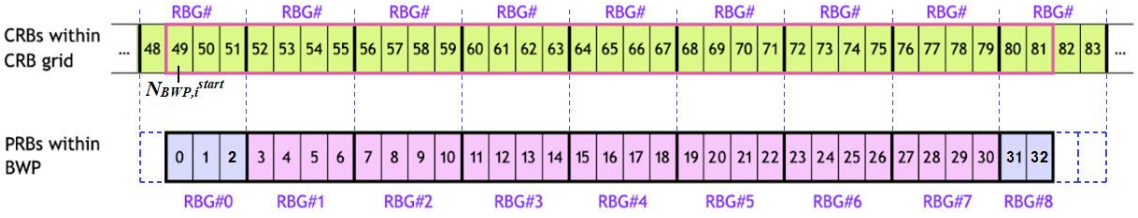



From the table above, the **size** "$P$" of the **intermediate RBGs** is **4 PRBs**. The following figure shows how the **RBGs** are aligned to the **CRB range** (using black vertical lines):




From the formulas above, the **quantity** "$N_{RBG}$" of **RBGs** is **9**.

From the formulas above, the **lowest frequency RBG** has **3 PRBs** and the **highest frequency RBG** has **2 PRBs**. The following figure shows these results graphically, illustrating how they are a consequence of the alignment of the **RBGs** to the **CRB range**:




## 2.2.10 Subcarriers

**Each subcarrier** is one of the **many individual signals** contained in a **multi-carrier signal** (click 11.3.2). **Each subcarrier conveys** an **individual stream** of **modulated data**. The term "**subcarrier**" in **multicarrier signals** is **roughly equivalent** to the term "**carrier**" in **single-carrier signals** (click 11.3.1).

**Each subcarrier** can be characterized by its **subcarrier spacing** (click 2.2.10.3) and at **every symbol interval** (click 2.3.2) by its **type** of **subcarrier** (see below) and by its **amplitude**.

At **every symbol interval** (click 2.3.2) there are the following **types** of **subcarriers**, depending on their **contents** and **characteristics**

- **Data subcarrier**

  It contains **data** that is delivered as **1 symbol** from a given **digital modulation** (click 11.2.1). During **high-traffic hours**, **many subcarriers** are **data subcarriers**.

- **Signaling subcarrier**

  It contains a **physical layer signal** (click 4.11.3) that is delivered **either** as **1 symbol** from a given **digital modulation** (click 11.2.1) or as a **specific complex value** (e.g., from a **Zadoff-Chu sequence**, click 11.20.4).

- **Null subcarrier** (click 2.2.10.1)

  It contains **nothing**. **Its Fourier coefficient is zero**. During **low-traffic hours**, **many subcarriers** are **null subcarriers**.

- **DC subcarrier** (click 2.2.10.2)

  It is **modulated** like a **data subcarrier**, but it **only** contains **low-rate information** because of its expected **distortion**.

#### 2.2.10.1 Null subcarriers

In **multicarrier signals** (click 11.3.2), a **null subcarrier** is the one that has **no amplitude** (i.e., it has a **zero Fourier coefficient**) in a **given symbol interval**.

In **multicarrier signals**, **null subcarriers** may be required to serve as:

- **Inter-channel guardband** at a **channel edge**. In this case, the **subcarrier** is **null** for **every symbol interval** (until channel reconfiguration). For example, **5G** and **LTE** use them to allow for **simple anti-aliasing filters** and **improve compatibility** with neighboring systems.
- **Intra-channel guardband** between **signal components** (e.g., between **border subcarriers** belonging to a **different frequency grid** and/or with **different signal characteristics**). In this case, the **subcarrier** is **null** for **every symbol interval**, until the need for intra-channel guardband ends.
- **Intra-channel guardband** between **bandwidth parts** (click 2.2.7) and **other technologies sharing** the **channel**. In this case, the **subcarrier** is **null** for **every symbol interval**, until the need for intra-channel guardband ends.
- **Empty resource element**. In **multicarrier multiplexing signals** (e.g., **OFDM**, click 11.14) **null subcarriers** can be used at the **required symbol intervals** when there is **less data** to transmit than the **available transmission resources**. In this case, the **subcarrier** can be **null** for as few as **only 1 symbol interval**, until the required data transmission increases.
- **Guardband between interleaves**. In **multicarrier multiple access signals** (e.g., **OFDMA** (click 11.15) or **DFT-s-OFDMA** (click 11.16)) when **interleaved** or **distributed subcarrier mapping** (click 11.17) is used, **every transmitter must**

include **null subcarriers between** the **subcarriers** that have been **assigned** to **it**. This **avoids interfering** between **different transmitters** because their **assigned subcarriers** do **not overlap**.

Source web pages:

- https://www.blacktechnoguys.com/2020/03/dc-carrier-in-nr.html
- https://dsp.stackexchange.com/questions/40335/the-necessity-of-null-subcarrier-in-ofdm

### 2.2.10.2 DC subcarrier

In **multicarrier signals** (click 11.3.2), the **DC subcarrier** of a **complex-valued baseband signal** "$BS(f)$" is the **subcarrier** whose **center** is closer to **frequency 0 Hz**. The **DC subcarrier** contains the **direct current** components of the **signal**. In mathematical terms, there is nothing wrong with using the **DC subcarrier** as any other **subcarrier**. However, in practice, the amplifiers and other components introduce **significant distortion** in the **DC subcarrier**. This is why the **DC subcarrier** is **not** handled as the **other subcarriers**, and it is therefore **required** to know its **location** within the carrier.

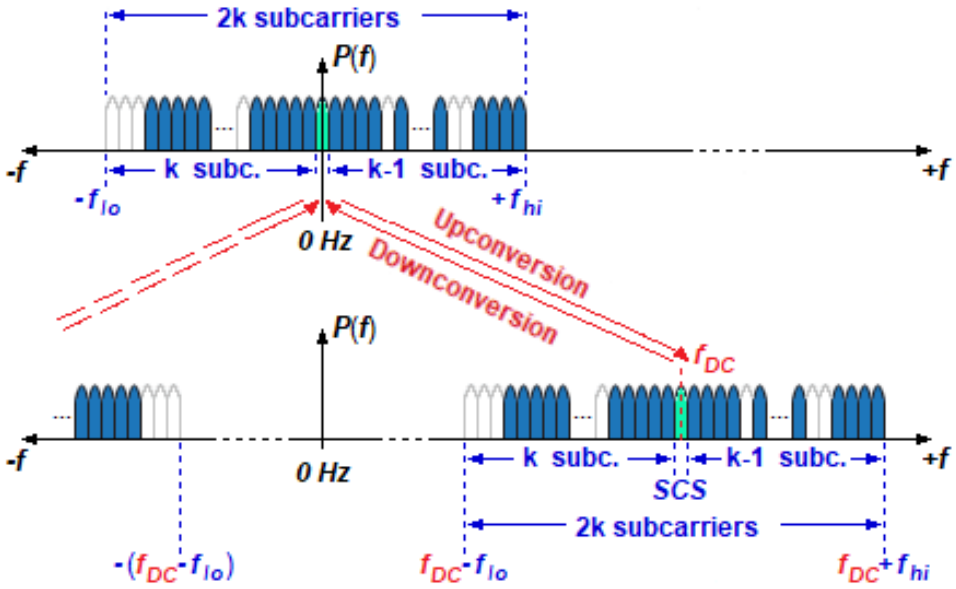
The **location** of **the DC subcarrier** is the **(i)DFT frequency bin** (click 11.19.1) that is **closer** to **0 Hz**. Consequently, the **location** of the **DC subcarrier** within the **carrier** depends on the **relative frequency position** between the **(i)DFT** (click 11.19) and the **carrier** (click 2.2.5.2). Since the **quantity** of **frequency bins** in the **(i)DFT** is typically a **power** of **2**, and the **transmission bandwidth** (click 2.2.3.6) is a **variable quantity** of **RBs**, the **most frequent case** is that the **(i)DFT** has more **frequency bins** than the **quantity** of **subcarriers**. This implies that the **(i)DFT** may be placed in **various locations** in respect to the **carrier**'s **subcarriers**. Hence the need to **indicate** the **location** of the **DC subcarrier**.

Because of **null subcarriers** (click 2.2.10.1), the **DC subcarrier** can be located **outside** of the **power spectral density** (**PSD**) of the **baseband signal** "$BS(f)$".

When a **baseband signal** "$BS(t)$" is **upconverted** (click 4.5.8.12), the **DC subcarrier** of the **resulting passband signal** "$PS(t)$" is the **subcarrier** whose **center** is closer to the **upconversion frequency** "$f_{DC}$" of the **frequency-domain** representation "$PS(f)$".

When a **passband signal** "$PS(t)$" is **down converted** (click 4.5.8.12), the **DC subcarrier** of the **resulting baseband signal** "$BS(t)$" is the **subcarrier** whose **center** was closer to the **down conversion frequency** "$f_{DC}$" of the **frequency-domain** representation "$PS(f)$".

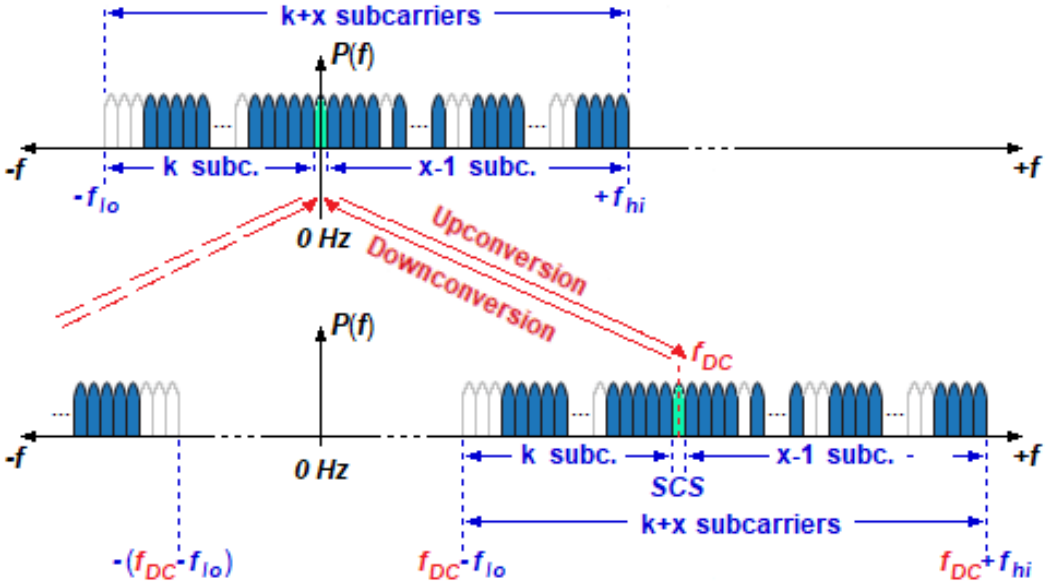
The following figure shows a **complex-valued baseband signal** (where the **DC subcarrier** is the **next-above-middle subcarrier**) that is **up/down converted** into/from the corresponding **real-valued passband signal**:




Some **relevant remarks** on the **example** of this figure are:

- The **DC subcarrier** is colored in **green** with **black borders**. **Non-null subcarriers** (**not** being the **DC one**) are colored in **blue** with **black borders**. **Null subcarriers** (i.e., **zero amplitude**, click 2.2.10.1) are colored in **white** with **grey borders**.
- The **DC subcarrier** is the **next-above-middle subcarrier**.
- After the **upconversion**, the **DC subcarrier** (colored in **green**) is **centered** in the **upconversion frequency** "$f_{DC}$". The **reciprocal** happens in **down-conversion**. This is a **general characteristic**, **not** something specific of this **example**.
- **Both** the **baseband signal** and the **passband signal** are **asymmetric**, in **respect** to the **central frequency** of the **DC subcarrier**, in **1 subcarrier**.
- The figure shows **groups** of **1**, **2** or **3 consecutive null subcarriers** (click 2.2.10.1) but this is **not** very **correct**. The most frequent case is to have **groups** of **12 consecutive null subcarriers** corresponding to **unused REGs** (click 1.1.4) in the **Downlink** or **Uplink** scheduling. However, this has not been shown in the figure because subcarriers would have become too small to view properly.

The following figure shows a **complex-valued baseband signal** (where the **DC subcarrier** is **not** the **next-above-middle subcarrier**) that is **up/down converted** into/from the corresponding **real-valued passband signal**:




Some **relevant remarks** on the **example** of this figure are:

- The **DC subcarrier** is colored in **green** with **black borders**. **Non-null subcarriers** (**not** being the **DC one**) are colored in **blue** with **black borders**. **Null subcarriers** (i.e., **zero amplitude**, click 2.2.10.1) are colored in **white** with **grey borders**.
- The **DC subcarrier** is **not** the **next-above-middle subcarrier**.
- After the **upconversion**, the **DC subcarrier** (colored in **green**) is **centered** in the **upconversion frequency** "$f_{DC}$". The **reciprocal** happens in **down-conversion**. This is a **general characteristic**, **not** something specific of this **example**.
- **Both** the **baseband signal** and the **passband signal** are **asymmetric**, in **respect** to the **central frequency** of the **DC subcarrier**, in **several subcarriers**.
- The figure shows **groups** of **1**, **2** or **3 consecutive null subcarriers** (click 2.2.10.1) but this is **not** very **correct**. The most frequent case is to have **groups** of **12 consecutive null subcarriers** corresponding to **unused REGs** (click 1.1.4) in the **Downlink** or **Uplink** scheduling. However, this has not been shown in the figure because subcarriers would have become too small to view properly.

**Data transmission in the DC subcarrier**

In **5G**, the **DC subcarrier** is treated like any other **subcarrier** in terms of **modulation** and **coding**. However, because of the **expected distortion**, the **DC subcarrier** is **only** used to send **low-bitrate information** that may include control signals, synchronization information, or other signaling components.

In **LTE**, the **DC subcarrier** is a **null subcarrier** (click 2.2.10.1) to allow the use of **simple** (i.e., cheap) direct-conversion (zero-IF) RF receivers. The reason is that **zero-IF** receivers typically encounter a **strong interfering** signal at **DC**. This can be caused by direct leakage of the **local oscillator** signal into the output as well as a **nonzero DC offset**

in the receiver's **analog-to-digital converter** (click 4.5.8.13). This is sidestepped in **LTE** by just **not transmitting any data** in the **DC subcarrier**.

Source web pages:

- https://www.blacktechnoguys.com/2020/03/dc-carrier-in-nr.html
- https://dsp.stackexchange.com/questions/40335/the-necessity-of-null-subcarrier-in-ofdm

### 2.2.10.3 Subcarrier spacing

The **subcarrier spacing** (**SCS**) is the **frequency distance** between the **center frequency** of **2 contiguous subcarriers**. You may think of the **subcarrier spacing** as the **bandwidth** of **each individual subcarrier** (although this is not strictly correct).

The **subcarrier spacing must** belong to a **certain set** of **allowed subcarrier spacings**.

The **7** allowed values **0 to 6** of the **numerology** "**μ**" (click 2.1) determine **7 numerology subcarrier spacings SCS=$2^{\mu}$•15 kHz**, which are (respectively) **15**, **30**, **60**, **120**, **240**, **480** and **960 kHz**. In addition to these **7 numerology subcarrier spacings** there are **2** additional **PRACH-specific subcarrier spacings**: **1.25 kHz** and **5 kHz** that do **not have** an associated **numerology**. These **2 PRACH-specific subcarrier spacing** values can **only** be used for **some cases** of the **RA preamble** (click 7.3.1).

The following table presents **all** the **5G subcarrier spacings** (including **both** the **numerology** ones and the **PRACH-specific** ones), with some of their characteristics:

| Freq. Range | Numerology "μ" | SCS ($15•2^{\mu}$) [kHz] | Cyclic prefix | Used for PDSCH, PDCCH, PUSCH and PUCCH | Used for SSB (PSS, SSS, PBCH) | Used for PRACH |
|---|---|---|---|---|---|---|
| FR1 | N/A | 1.25 | PRACH-specific | No | No | Long Formats |
| | | 5 | | | | |
| | 0 | 15 | Normal | Yes | Yes | Short Formats |
| | 1 | 30 | | Yes | Yes | |
| FR1, FR2-1 | 2 | 60 | Normal or Extended | Yes | No | |
| FR2-1 | 3 | 120 | Normal | Yes | Yes | |
| FR2 | 4 | 240 | | No | Yes | |
| | 5 | 480 | | Yes | Yes | |
| | 6 | 960 | | Yes | Yes | |

As you may see, **numerology 2** (i.e., **60 kHz subcarrier spacing**) is the **only one** that can use the **extended cyclic prefix** (**ECP**, click 2.3.2). Notice that it may also use the **normal cyclic prefix** (**NCP**, click 2.3.2), like the **other numerologies**.

Be aware that **subcarrier spacing** and the (**average**) **slot duration** (click 2.3.2) are **inversely proportional**, because of their relationship through **numerology** (click 2.1).

In **LTE**, **subcarrier spacing** is **fixed** to **15 kHz** and the **carrier** must carry the **maximum** possible **quantity** of **subcarriers** in the **channel**. As an exception, for the **Multimedia Broadcast Multicast Service** (**MBMS**), the **LTE subcarrier spacing** is **7.5 kHz**.

### 2.2.10.4 Types of subcarrier spacing

Since different **physical layer elements** within the **same radio channel** can have **different subcarrier spacings**, when making a reference to a **subcarrier spacing** it

might be required to indicate to which **specific NR element** it corresponds. There may be the following **different subcarrier spacings**, corresponding to **different NR elements**, within the **same radio channel**:

- **SSB subcarrier spacing**

  It is the **subcarrier spacing** used in the **SSB**. It is therefore the **subcarrier spacing** used for **PSS**, for **SSS** and for **PBCH**. For more detail on its configuration, click 6.1.4.

- **CORESET#0 subcarrier spacing**

  It is the **subcarrier spacing** used **both** for **CORESET#0** (i.e., for **initial access PDCCH**) and for the delivery of **SIB1** during **initial access** (i.e., for **initial access PDSCH**).

  For more detail on its configuration, click 6.8.8.1.

- **PDSCH subcarrier spacing**

  It is the **subcarrier spacing** used for **both PDCCH** and **PDSCH**. It is also the **subcarrier spacing** for the **Downlink currently active bandwidth part**. It is also the **subcarrier spacing** for the **transmitted Downlink carrier**. It is also the **subcarrier spacing** that determines the **Downlink slot duration** (click 2.3.1).

- **PUSCH subcarrier spacing**

  It is the **subcarrier spacing** used for **both PUCCH** and **PUSCH**. It is also the **subcarrier spacing** for the **Uplink currently active bandwidth part**. It is also the **subcarrier spacing** for the **transmitted Uplink carrier**. It is also the **subcarrier spacing** that determines the **Uplink slot duration** (click 2.3.1).

- **PRACH subcarrier spacing**

  It is the **subcarrier spacing** used for **PRACH** (i.e., for the **RA preamble**, click 7.3.1).

- **Numerology subcarrier spacings**

  They are the **7 subcarrier spacings** defined by the **7 numerology** (click 2.1) values.

- **PRACH-specific subcarrier spacings**

  They are the **2 subcarrier spacings** that can be used in **PRACH**, and that do **not** have an associated **numerology value**.

The **PDSCH** and **PUCCH subcarrier spacing(s)** are jointly called "**control/shared subcarrier spacing(s)**".

The **types** of **subcarrier spacing** are **very similar**, but **not** the **same**, as the **types** of **numerologies** (click 2.1.4).

### 2.2.10.5 Subcarrier spacing parameters

The **5G** transmission structure may combine **physical layer elements** with **different subcarrier spacings** (click 2.2.5.8), even over **simultaneous symbol intervals**, **changing dynamically** over time.

There are different configuration **parameters** to determine the **subcarrier spacing** for different **physical layer elements**, **situations** and **purposes**, as listed right below. Notice

that there are several cases of **same-name** and **version-different parameters** (click 5.7.2.2). The **data type** of **almost all** of these parameters is `SubcarrierSpacing` (click 2.2.10.5.1), which is an **enumerated** with the **7 numerology subcarrier spacing values** (click 2.2.10.3). Recall that to **uniquely identify** a parameter you need **both** the **name** of the **parameter** and of its **enclosing data type**.

`MIB` **subcarrier spacing parameters**:

- `subCarrierSpacingCommon` from `MIB` (click 8.12.3), that has data type `ENUMERATED {scs15or60, scs30or120}`.

`SIB1` **subcarrier spacing parameters**:

- `subcarrierSpacing` from `BWP` (click 2.2.8.11), that has data type `SubcarrierSpacing`.
- `subcarrierSpacing` from `SCS-SpecificCarrier` (click 2.2.6.12), that has data type `SubcarrierSpacing`.
- `ssbSubcarrierSpacing` from `ServingCellConfigCommon` (click 3.9.6), that has data type `SubcarrierSpacing`.
- `msg1-SubcarrierSpacing` from `RACH-ConfigCommon` (click 9.5.3), that has data type `SubcarrierSpacing`.

**Other system information and non-SI subcarrier spacing parameters**:

- `subcarrierSpacing` from `SCS-SpecificCarrier` (click 2.2.6.12), from `CSI-RS-ResourceConfigMobility`, from `RateMatchPattern` and from `SlotFormatCombinationsPerCell`, all of which have data type `SubcarrierSpacing`.
- `subcarrierSpacing2` from `SlotFormatCombinationsPerCell`, that has data type `SubcarrierSpacing`.
- `ssbSubcarrierSpacing` from `ServingCellConfigCommon` (click 3.9.6), from `InterFreqCarrierFreqInfo`, from `CarrierInfoNR`, from `MeasObjectNR` and from `MeasTiming`, all of which have data type `SubcarrierSpacing`.
- `msg1-SubcarrierSpacing` from `BeamFailureRecoveryConfig`, that has data type `SubcarrierSpacing`.
- `slotListSubcarrierSpacing-r17` from `IAB-ResourceConfig-r17`, that has data type `SubcarrierSpacing`.
- `allowedSCS-List` from `LogicalChannelConfig`, that has data type `SEQUENCE (SIZE 1..maxSCS) OF SubcarrierSpacing`.
- `ssbSubcarrierSpacing-r16` from `MeasIdleCarrierNR-r16` and from `SSB-Configuration-r16`, both of which have data type `SubcarrierSpacing`.
- `srs-SCS-r16` from `SRS-ResourceConfigCLI-r16`, that has data type `SubcarrierSpacing`.

- `rssi-SCS-r16` from `RSSI-ResourceConfigCLI-r16`, that has data type `SubcarrierSpacing`.
- `msgA-SubcarrierSpacing-r16` from `RACH-ConfigCommonTwoStepRA-r16`, that has data type `SubcarrierSpacing`.
- `guardBandSCS-r16` from `IntraCellGuardBandsPerSCS-r16`, that has data type `SubcarrierSpacing`.
- `subcarrierSpacing-r17` from `CO-DurationsPerCell-r17`, that has data type `SubcarrierSpacing`.
- `referenceSubcarrierSpacing` from `TDD-UL-DL-ConfigCommon` (click 2.3.5.3) that has data type `SubcarrierSpacing`.
- `supportedSubcarrierSpacingDL` from `FeatureSetDownlinkPerCC`, that has data type `SubcarrierSpacing`.
- `supportedSubcarrierSpacingUL` from `FeatureSetUplinkPerCC`, that has data type `SubcarrierSpacing`.
- `sl-AllowedSCS-List-r16` from `SL-LogicalChannelConfig-r16`, that has data type `SEQUENCE (SIZE 1..maxSCS) OF SubcarrierSpacing`.
- `subcarrierSpacing-NR-r16` from `FrequencyConfig-NR-r16`, that has data type `SubcarrierSpacing`.
- `csi-RS-SubcarrierSpacing-r16` from `MeasurementTimingConfiguration-v1610-IEs`, that has data type `SubcarrierSpacing`.
- `subcarrierSpacing-r16` from `CO-DurationsPerCell-r16` and from `RA-InformationCommon-r16` (click 9.5.13), that has data type `SubcarrierSpacing`.
- `msg1-SubcarrierSpacing-r16`, `msg1-SubcarrierSpacingCFRA-r16` and `msgA-SubcarrierSpacing-r17`, all of them from `RA-InformationCommon-r16`, all of which have data type `SubcarrierSpacing`.

For an overall explanation on **UE configuration parameters** click 5. For a brief overview of **ASN.1**, click 5.9.

#### 2.2.10.5.1 `SubcarrierSpacing` data type

It can be **both broadcast** (or **unicast**) within `SIB1` (click 5.3.4) and **unicast** within the **Non-SI parameter group** (click 5.3.6).

**Each** of its **instances** contains a **subcarrier spacing value**.

Its **complete ASN.1** (click 5.9) specification is:

```
SubcarrierSpacing ::= ENUMERATED {kHz15, kHz30, kHz60, kHz120, kHz240,
                                  kHz480-v1700, kHz960-v1700, spare1}
```

Notice that on some **parameters** with this data type (e.g., the ones for `SIB1`, for specific frequency ranges, for **control/shared** physical channels, …) the **acceptable values** will be **restricted** in the definition of the corresponding **parameter**.

### 2.2.11 Frequency alignment of different elements

**All** the **physical layer elements must** be **precisely aligned** in **frequency**. A **summary** of **frequency alignment requirements** is:

- **Every radio channel must** be **centered** in a **frequency** from the **band channel raster** (click 2.2.3.3).
- **Every radio channel must leave** a proper **guardband** at **each channel edge** (click 2.2.3.6.2).
- **Every two contiguous radio channels must** be **properly spaced away** (click 2.2.3.5).
- **Every radio channel must** have at least **one centered usable CRB range**,
- The **centered usable CRB range must** have its **next-above-middle subcarrier centered** in the **channel center** (click 2.2.4.1).
- **Every usable CRB range must** have the **same quantity** of **CRBs** as the **radio channel transmission bandwidth** of the **same subcarrier spacing** (click 2.2.4).
- **Every radio channel must** have **1 Point A** and **1 set** of **CRB ranges** (click 2.2.4).
- **Every radio channel**'s **Point A location must** belong to the **global frequency raster** (click 2.2.4.3).
- **Every CRB range must** have its **common subcarrier 0 centered** in **Point A** (click 2.2.4.4).
- The **SSB must** have its **next-above-middle subcarrier centered** in a frequency from the **band synchronization raster** (click 6.1.2.1).
- The **RBs** and the **subcarriers** of **non-zero CORESETs** (**PDCCH**) and of **PDSCH** (except the **RBs** for **SIB1** transmission) **must** be **aligned** to the **CRBs** and **CSCs** of the **Downlink CRB range** of the **same subcarrier spacing**. For more detail, click 2.2.4.4.
- The **RBs** of **SSB** do **not** have to be **aligned** to the **CRBs** of the **Downlink CRB range** of the **same subcarrier spacing**. However, the **subcarriers** of **SSB must** be **aligned** to the **CSCs** of the **Downlink CRB range** of the **same subcarrier spacing**. For more detail, click 6.1.2.1.
- The **RBs** and the **subcarriers** of **SSB**, **CORESET#0** and **PDSCH** for **SIB1 transmission must** be **aligned** among themselves according to the corresponding **multiplexing pattern** (click 6.8.8.4).
- The **RBs** and the **subcarriers** of **PUSCH** and **PUCCH must** be **aligned** to the **CRBs** and the **CSCs** of the **Uplink** (or **SUL**, as corresponds) **CRB range** of the **same subcarrier spacing**.
- **PRACH** does **not** have **RBs**, so **no alignment** can be established in terms of its **RBs**.
- **PRACH subcarriers** do **not** have to be **aligned** with the **common subcarriers** of the **Uplink CRB range** of the **same subcarrier spacing**.
- **Every SSB must** be **completely within** its **Downlink active bandwidth part**.

- **Every PRACH must** be **completely within** its **Uplink active bandwidth part**.
- **Every bandwidth part must** be **completely within** its **carrier**.
- **Every carrier must** be **completely within** its **channel**.

Notice that the **frequency location** of **many** of the **physical layer elements** cited above is configured **independently**. Therefore, **all** these **individual configurations must** be **coherent** so that **all** the above **requirements** are **satisfied**.

For example, the **frequency location** of **each cell**'s **channel** (click 2.2.3.2), of **each channel**'s **carrier** (click 2.2.5), of **each UE**'s **bandwidth part** (click 2.2.7.1.1), of **each cell**'s **SSB** (click 6.1.2.1) and of **each cell**'s **PRACH** (click 7.1), can be configured **independently**. **Every case** of these **individual configurations must satisfy all** the above **requirements**.

**Resource Block indexes**

The **resource blocks** have the following **different indexes**:

- **CRB and CSC indexes**

  They identify (respectively) **each CRB** and **each CSC** within **each CRB range**. **CRB0** and **CSC0** are (respectively) the **lowest frequency ones**. **Each index increases** in **1** (respectively) for **each CRB** or **each CSC upwards** in **frequency**. **CRB0** is the one **located in Point A**. **CSC0** is the one **centered** in **Point A**.

- **Resource grid RB and SC indexes**

  They identify (respectively) **each RB** and **each SC** within **each resource grid**. **RB0** and **SC0** are (respectively) the **lowest frequency ones**. **Each index increases** in **1** (respectively) for **each RB** or **each SC upwards** in **frequency**.

- **PRB and PSC indexes**

  They identify (respectively) **each PRB** and **each PSC** within **each bandwidth part**. **PRB0** and **PSC0** are the **lowest frequency ones** and **each index increases** in **1** (respectively) for **each CRB** or **each CSC upwards** in **frequency**.

- **SSB RB and SC indexes**

  They identify (respectively) **each SSB RB** and **each SSB SC** within **each SSB**. **RB0** and **CS0** are (respectively) the **lowest frequency ones** . **Each index increases** in **1** (respectively) for **each RB** or **each SC upwards** in **frequency**, up to (respectively) **RB19** and **SC239**.

- **CORESET#0 RB and SC indexes**

  They identify (respectively) **each CORESET#0 RB** and **each CORESET#0 SC** within **each CORESET#0**. **RB0** and **SC0** are (respectively) the **lowest frequency ones** and **each index increases** in **1** (respectively)for **each RB** or **each SC upwards** in **frequency** (respectively).

- **PRACH subcarrier index**

  It identifies **each RA preamble subcarrier** within **each RA preamble**. **SC0** is the **lowest frequency one** and the **index increases** in **1** for **each SC upwards** in **frequency**, up to **SC**$_{LRA-1}$.

- **RB subcarrier index**

  It identifies **each** of the **12 SCs** within **each RB**. **SC0** is the **lowest frequency one** and the **index increases** in **1** for **each SC upwards** in **frequency**, up to **SC11**.

**Every value** "$\mathbf{SC_{i,index}}$" from one of the **SC indexes** above may be computed from the **RB index value** "$\mathbf{RB_{j,index}}$" of its **enclosing RB** and the **SC index value** within the **RB** "$\mathbf{SC_{r,RB}}$" as:

$$\mathbf{SC_{i,index}} = 12 \bullet \mathbf{RB_{j,index}} + \mathbf{SC_{r,RB}}$$

### 2.2.12 Frequency division duplexing (FDD) in 5G

The **duplexing mode** (click 2.2.1.2) of **each band** indicates what **duplexing** (click 11.5) technique **must** be used in **its channels** (click 2.2.2.3). **Paired spectrum bands** (**FDD**, **SDL** and **SUL bands**) contain **channels** that use **5G frequency division duplexing**, which is explained in this section. **Time division duplexing** (**TDD**) **bands** contain channels that use **5G time division duplexing** which is explained in 2.3.4. For a comparison between **FDD** and **TDD**, click 11.5.3.

**Every FDD band** uses **channel pairs**, **each** with **one Downlink channel** and **one Uplink channel**, to achieve **duplexing**.

**Every SUL band only** has **channels** in the **Uplink transmission direction**. **Duplexing** is achieved by **combining** a **SUL channel** with a **Downlink channel** from an **FDD band**, in the same **cell**. A **SUL band** therefore **cannot** be used by **itself**, and it **must** be used in **combination** with an **FDD band**. The same applies to **SDL bands**.

**Every FDD band** is **divided** into **one Downlink semi-band** and **one Uplink semi-band**, separated by a **guardband**. The **Downlink semi-band only** has **Downlink channels**, and the **Uplink semi-band only** has **Uplink channels**. If **both semi-bands** have the **same bandwidth**, the **FDD band** is said to be "**symmetric**". Otherwise, it is said to be "**asymmetric**".

The **frequency segment** of **each band**'s **Downlink semi-band** and **Uplink semi-band** is indicated in table 5.2-1 of **TS 38.211-1**, **-2** and **-5**. **Each band**'s **guardband** is the **frequency segment** between **both semi-bands**.

**Each Downlink channel** is **paired** with **one Uplink channel**. In this way, **FDD** allows **full-duplex** transmission, where **both** the **UE** and the **gNB** can transmit **simultaneously**. The **two channels** in **each channel pair** are **located** in the **same relative frequency position** within **each semi-band**. This is, the **1st**, **2nd**, **3rd**, etc **Downlink channel** is **paired** with the **1st**, **2nd**, **3rd**, etc **Uplink channel**, respectively.

If **both channels** in **every channel pair** have the **same bandwidth**, the **FDD band** is said to be configured with "**symmetric channels**". Otherwise, it is said to be configured with "**asymmetric channels**" (click 2.2.3.7).

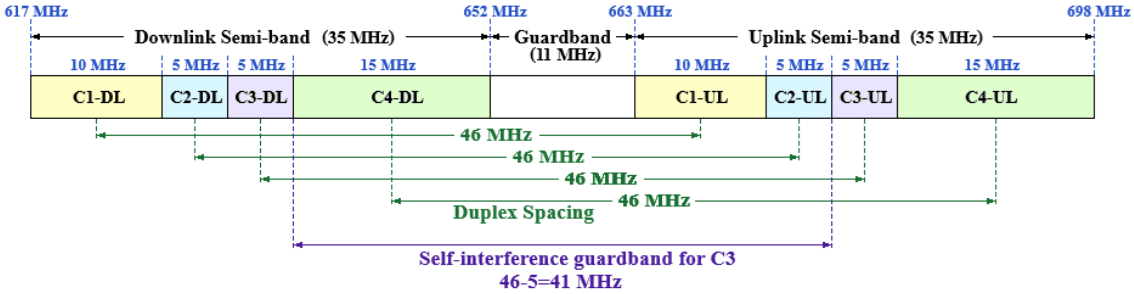
The **frequency distance** between the **channel center** of **two paired channels** is called the "**duplex spacing**". This is sometimes called "Tx-Rx frequency separation" or "carrier center separation". The **default duplex spacing** of **every symmetric FDD band** configured with **symmetric channels** is **constant**. Its value is the **semi-band bandwidth** plus the **guardband bandwidth**. **Each band**'s **default duplex spacing** is specified in table 5.4.4-1 from TS 38.101-**1** and **-5**. The **default duplex spacing** has been consolidated in the **band tables** at 2.2.13. The said tables list the **duplex spacing** computed as the **Downlink channel center minus** the **Uplink channel center** (within

**each channel pair**). This is why in a **few bands** the **duplex spacing** is listed as a **negative number** (which might be a little puzzling).

The **self-interference** (click 12.1.14) **guardband** of a **channel pair** is the **frequency distance** between the **upper edge** of a **channel** in the **lower frequency semi-band** and the **lower edge** of its **paired channel**. Its amount represents the level of **protection** against **self-interference** in this **channel pair**. In **every symmetric FDD band** configured with symmetric channels the **self-interference guardband** of **every channel pair** is the **duplex-spacing minus** the **channel bandwidth**.

In case a **UE** (e.g., because of **carrier aggregation**, click 10.2) or a **NR node** (e.g., because it handles **multiple cells**) uses **more than one channel pair**, the **self-interference guardband** is reduced to the **frequency distance** between the **upper edge** of the **highest frequency channel** (being used) in the **lower frequency semi-band** and the **lower edge** of the **lowest frequency channel** (being used). The **worst case** is when the **UE** or the **NR node** are using **both** the **highest frequency channel** in the **lower semi-band** and the **lowest frequency channel** in the **higher semi-band**. In this **worst case**, the **self-interference guardband** is reduced to the **band guardband**. This type of channel assignment is therefore **quite undesirable** because it reduces the **self-interference guardband**.

The following figure illustrates an example for **FR1-1 FDD band n71**, where the national authorities have licensed **four channel pairs**, **each channel** having **10**, **5**, **5** and **15 MHz** of bandwidth:




Some **relevant remarks** on the example in this figure are:

- The **Uplink semi-band** and the **Downlink semi-band** have the **same bandwidth** (**35 MHz each**). Consequently, the **FDD band n71** is **symmetric**.
- The **two channels** in **every channel pair** have been identified with the **same channel pair number** and the **same color**.
- The **two channels** in **every channel pair** are located in the **same relative positions**, as they **must**.
- The **channel bandwidth** of the **two channels** in **every channel pair** is the **same**, as it **must** (in a **symmetric band**).
- Since it is a **symmetric FDD band**, its **duplex spacing** is **constant**. Its value is the **addition** of the **semi-band bandwidth** (**35 MHz**) and the **guardband bandwidth** (**11 MHz**), resulting in **46 MHz**.
- Since it is a **symmetric FDD band**, the **self-interference** (click 12.1.14) **guardband** for every **channel pair** is the **band duplex-spacing minus** the **channel bandwidth**. The figure illustrates this for the **channel pair C3**, with a resulting value of **41 MHz**.

**Transmission and reception** may be done **simultaneously** at the **NR node** and at **every UE**. However, two **UEs cannot transmit** in the **same RBs** of the **same frame symbol interval index** (within a **frame** with the **same number**), because this would result in **mutual reception interference** at the **NR node** (assuming that the **frame offset** at the **NR node** is **0**, click 2.3.5.7).

The **Uplink semi-band** is almost always the one with **better propagation**, to **mitigate** the **UE**'s **poorer transmission capabilities**. Consequently, the **Uplink semi-band** is **usually** the **lower frequency one**.

The **duplex-spacing** is usually **larger** in **higher frequency bands**, because **self-interference** (click 12.1.14) usually **increases** with **frequency**. Actually, **FDD** is **only** used in some **FR1-1 bands**, while **TDD** is used in **all FR2-1** and **FR2 bands**.

Surprisingly, the **FDD guardband** may be **used** by a **different NR node** (or a different radio system) to the one using the corresponding **FDD band**. The reason is that **guardband** usage by another **NR node** (or system) does **not** cause **self-interference**. It however required to implement the **usual guardbands** between **adjacent bands**. As an example of this, **TDD band n38** (2,600 MHz) is **located** exactly in the **guardband** of **FDD band n7**.

Some **UEs** can **only use half-duplex** (click 2.3.4.5) even in **FDD bands**, where the **FDD channel pair** allows for **full-duplex**. For these **UEs**, the **NR node** signals the **UE** the **semi-duplex slot format** (click 2.3.4.2) that it must use, as it is required in **TDD** (click 2.3.4). Even though these **UEs** use **half-duplex**, the **Downlink** and the **Uplink transmission** is still done in the **corresponding Downlink** and **Uplink channel**.

Source web pages:

- https://www.sqimway.com/nr_band.php#fdd_asymmetric

### 2.2.13 Summary of 5G operating bands

This section contains a **listing** of the **5G operating bands**, summarizing their main characteristics and grouped by **frequency ranges** and subranges: **FR1-1 terrestrial** (click 2.2.13.1), **FR1-2 terrestrial** (click 2.2.13.2), **FR2-1** (click 2.2.13.3), **FR2-2** (click 2.2.13.4) and **FR1-satellite** (click 2.2.13.5).

This section also provides a brief overview on the **future** evolution of **5G bands** (click 2.2.13.7).

A few changes have been made in **R16**, **R17** and **R18** in respect to the **initial band tables** from **R15**. These changes have been mainly to deprecate a few unused bands, to include a few very specific (and unusual) bands requested by a few countries or operators, and to add a few more millimeter wave bands (notably the **FR2-2 band**). A list of **5G frequency bands**, indicating for **each** of them the **release** (and **version**) where it was introduced may be found at:

- https://www.sqimway.com/nr_band.php

For a description of the **overall organization of the spectrum**, click 2.1.5.5.

For the **actual definition of bands** in the **3GPP** documents, refer to the various tables of TS 38.101-**1** (**FR1-terrestrial**), of TS 38.101-**2** (**FR2**) and of TS 38.101-**5** (**FR1-satellite**).

Source web pages:

- https://en.wikipedia.org/wiki/5G_NR_frequency_bands
- https://www.sqimway.com/nr_band.php

### 2.2.13.1 5G Frequency Range 1, subrange 1-1, terrestrial bands

The following table shows the **5G Frequency Range 1**, **subrange 1-1**, **terrestrial bands** (ordered by frequency), with a **summary** of their **characteristics**. For **each band**, the table lists **all** the **possibly available channel bandwidths**. However, the **channel bandwidths actually available** depend on the **control/shared subcarrier spacing** configured in the **channel** (click 2.2.3.2). Notice that the **FDD bands** whose **Downlink** and **Uplink** frequencies are **very different** are **listed twice**, **each** of the two placed in the **frequency position** corresponding to either its **Downlink** or its **Uplink**. **These bands** have been **highlighted** in **soft yellow**.

| Band (Subset of) | Duplex mode | *f* [MHz] | Common name | Uplink [MHz] | | Downlink [MHz] | | Duplex spacing [MHz] | All possible channel bandwidths [MHz] | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| **n105** | FDD | 600 | Digital Dividend (APT) | 663 | 703 | 612 | 652 | -51 | 5, 10, 15, 20, 25[B1], 30[B1], 35[B1] | |
| **n71** | FDD | 600 | Digital Dividend (US) | 663 | 698 | 617 | 652 | -46 | 5, 10, 15, 20, 25[B1], 30[B1], 35[B1] | [B9] |
| **n85** | FDD | 700 | Extended Lower SMH | 698 | 716 | 728 | 746 | 30 | 5, 10, 15 | |
| **n12** | FDD | 700 | Lower SMH | 699 | 716 | 729 | 746 | 30 | 5, 10, 15 | |
| **n28** | FDD | 700 | APT | 703 | 748 | 758 | 803 | 55 | 5, 10, 15, 20, 30 | EU |
| **n83** | SUL | 700 | APT | 703 | 748 | N/A | | N/A | 5, 10, 15, 20, 30 | EU |
| **n13** | FDD | 700 | Upper SMH | 777 | 787 | 746 | 756 | -31 | 5, 10 | |
| **n14** | FDD | 700 | Upper SMH | 788 | 798 | 758 | 768 | -30 | 5, 10 | |
| **n29** | SDL | 700 | Lower SMH | N/A | | 717 | 728 | N/A | 5, 10 | EU |
| **n67** | SDL | 700 | EU 700 | N/A | | 738 | 758 | N/A | 5, 10, 15, 20 | |
| **n20** | FDD | 800 | Digital Dividend (EU) | 832 | 862 | 791 | 821 | -41 | 5, 10, 15, 20 | |
| **n91** | FDD | 800 & 1,500 | DD & L-Band (EU) | 832 | 862 | 1,427 | 1,432 | 570 to 595 [B9] | 5, 10[B12] | [B9], [B11], [B13] |
| **n92** | FDD | 800 & 1,500 | DD & L-Band (EU) | 832 | 862 | 1,432 | 1,517 | 600 to 660 [B9] | 5, 10, 15, 20 | [B9], [B11], [B13] |
| **n82** | SUL | 800 | Digital Dividend (EU) | 832 | 862 | N/A | | N/A | 5, 10, 15, 20 | |
| **n26** | FDD | 850 | Extended CLR | 814 | 849 | 859 | 894 | 45 | 5, 10, 15, 20, 25[B4], 30[B4] | [B9] |
| **n18 (n26)** | FDD | 850 | Lower 800 (Japan) | 815 | 830 | 860 | 875 | 45 | 5, 10, 15 | |
| **n5** | FDD | 850 | CLR | 824 | 849 | 869 | 894 | 45 | 5, 10, 15, 20, 25[B1] | [B9] |
| **n89** | SUL | 850 | CLR | 824 | 849 | N/A | | N/A | 5, 10, 15, 20, 50 | |
| **n100** | FDD | 900 | Rail Mobile Radio (RMR) | 874.4 | 880 | 919.4 | 925 | 45 | 5 | |

| Band (Subset of) | Duplex mode | *f* [MHz] | Common name | Uplink [MHz] | | Downlink [MHz] | | Duplex spacing [MHz] | All possible channel bandwidths [MHz] | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| **n8** | FDD | 900 | Extended GSM | 880 | 915 | 925 | 960 | 45 | 5, 10, 15, 20, 25[B1], 35[B1] | [B9] |
| **n81** | SUL | 900 | Extended GSM | 880 | 915 | N/A | | N/A | 5, 10, 15, 20 | |
| **n93** | FDD | 900 & 1500 | Extended GSM & L-Band (EU) | 880 | 915 | 1,427 | 1,432 | 527 to 547 [B9] | 5, 10[B12] | [B9], [B11], [B13] |
| **n94** | FDD | 900 & 1500 | Extended GSM & L-Band (EU) | 880 | 915 | 1,432 | 1,517 | μ = 0 => 522 to 632 μ = 1 => 527 to 627 [B9] | 5, 10, 15, 20 | [B9], [B11], [B13] |
| **n51** | TDD | 1,500 | Extended L-Band (EU) | 1,427 | | 1,432 | | N/A | 5 | |
| **n74** | FDD | 1,500 | Lower L-Band (US) | 1,427 | 1,470 | 1,475 | 1,518 | 48 | 5, 10, 15, 20 | |
| **n50** | TDD | 1,500 | L-Band (EU) | 1,432 | | 1,517 | | N/A | 5[B6], 10, 15, 20, 30, 40, 50, 60, 80[B1] | [B9] |
| **n76** | SDL | 1,500 | Extended L-Band (EU) | N/A | | 1,427 | 1,432 | N/A | 5 | |
| **n91** | FDD | 800 & 1,500 | DD & L-Band (EU) | 832 | 862 | 1,427 | 1,432 | 570 to 595 [B9] | 5, 10[B12] | [B9], [B11], [B13] |
| **n93** | FDD | 900 & 1500 | Extended GSM & L-Band (EU) | 880 | 915 | 1,427 | 1,432 | 527 to 547 [B9] | 5, 10[B12] | [B9], [B11], [B13] |
| **n75** | SDL | 1,500 | L-Band (EU) | N/A | | 1,432 | 1,517 | N/A | 5, 10, 15, 20, 25, 30, 40, 50 | |
| **n92** | FDD | 800 & 1,500 | DD & L-Band (EU) | 832 | 862 | 1,432 | 1,517 | 600 to 660 [B9] | 5, 10, 15, 20 | [B9], [B11], [B13] |
| **n94** | FDD | 900 & 1500 | Extended GSM & L-Band (EU) | 880 | 915 | 1,432 | 1,517 | μ = 0 => 522 to 632 μ = 1 => 527 to 627 | 5, 10, 15, 20 | [B9], [B11], [B13] |
| **n24** | FDD | 1,600 | Upper L-Band (US) | 1,626.5 | 1,660.5 | 1,525 | 1,559 | -101.5, -120.5 | 5, 10 | [B2], [B3], [B9] |
| **n99** | SUL | 1,600 | Upper L-Band (US) | 1,626.5 | 1,660.5 | N/A | | N/A | 5, 10 | [B2] |
| **n54** | TDD | 1.700 | TD 1700 | 1.670 | | 1.675 | | N/A | 5 | |
| **n86 (n80)** | SUL | 1,700 | Extended AWS (duplex spacing) | 1,710 | 1,780 | N/A | | N/A | 5, 10, 15, 20, 40 | |
| **n66** | FDD | 1,700 & 2,100 | Extended AWS (blocks A-J) | 1,710 | 1,780 | 2,110 | 2,200 | 400 | 5, 10, 15, 20, 25, 30, 35, 40, 45 | [B8], [B9], [B13] |
| **n3** | FDD | 1,800 | DCS | 1,710 | 1,785 | 1,805 | 1,880 | 95 | 5, 10, 15, 20, 25, 30, 35, 40, 45, 50 | |
| **n80** | SUL | 1,800 | DCS | 1,710 | 1,785 | N/A | | N/A | 5, 10, 15, 20, 25, 30, 40 | |
| **n2 (n25)** | FDD | 1,900 | PCS blocks A-F | 1,850 | 1,910 | 1,930 | 1,990 | 80 | 5, 10, 15, 20, 25, 30, 35, 40 | |

| Band (Subset of) | Duplex mode | *f* [MHz] | Common name | Uplink [MHz] | | Downlink [MHz] | | Duplex spacing [MHz] | All possible channel bandwidths [MHz] | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| **n25** | FDD | 1,900 | Extended PCS | 1,850 | 1,915 | 1,930 | 1,995 | 80 | 5, 10, 15, 20, 25, 30, 35, 40, 45[B1] | [B9] |
| **n39** | TDD | 1,900 | DCS–IMT Gap | 1,880 | | 1,920 | | N/A | 5, 10, 15, 20, 25, 30, 40 | |
| **n98** | SUL | 1,900 | DCS–IMT Gap | 1,880 | 1,920 | N/A | | N/A | 5, 10, 15, 20, 25, 30, 40 | |
| **n101** | TDD | 1,900 | Rail Mobile Radio (RMR) | 1,900 | | 1,910 | | N/A | 5, 10 | |
| **n84** | SUL | 1,900 | IMT | 1,920 | 1,980 | N/A | | N/A | 5, 10, 15, 20, 25, 30, 40, 50 | |
| **n70** | FDD | 2,000 | AWS-4 | 1,695 | 1,710 | 1,995 | 2,020 | 300 | 5, 10, 15, 20[B1], 25[B1] | [B9] |
| **n1 (n65)** | FDD | 2,100 | IMT | 1,920 | 1,980 | 2,110 | 2,170 | 190 | 5, 10, 15, 20, 25, 30, 40, 45, 50 | |
| **n65** | FDD | 2,100 | Extended IMT | 1,920 | 2,010 | 2,110 | 2,200 | 190 | 5, 10, 15, 20, 50 | V2X |
| **n66** | FDD | 1,700 & 2,100 | Extended AWS (blocks A-J) | 1,710 | 1,780 | 2,110 | 2,200 | 400 | 5, 10, 15, 20, 25, 30, 35, 40, 45 | [B8], [B9], [B13] |
| **n34** | TDD | 2,100 | IMT | 2,010 | | 2,025 | | N/A | 5, 10, 15 | |
| **n95** | SUL | 2,100 | IMT | 2,010 | 2,025 | N/A | | N/A | 5, 10, 15 | |
| **n40** | TDD | 2,300 | S-Band | 2,300 | | 2,400 | | N/A | 5[B6], 10, 15, 20, 25, 30, 40, 50, 60, 80, 90, 100 | |
| **n97** | SUL | 2,300 | S-Band | 2,300 | 2,400 | N/A | | N/A | 5, 10, 15, 20, 25, 30, 40, 50, 60, 70, 80, 90, 100 | |
| **n30** | FDD | 2,300 | WCS | 2,305 | 2,315 | 2,350 | 2,360 | 45 | 5, 10 | |
| **n53** | TDD | 2,500 | S-Band | 2,484 | | 2,495 | | N/A | 5, 10 | |
| **n41** | TDD | 2,500 | BRS | 2,496 | | 2,690 | | N/A | 10, 15, 20, 30, 40, 50, 60, 70, 80, 90, 100 | |
| **n90 (n41)** | TDD | 2,500 | BRS | 2,496 | | 2,690 | | N/A | 10, 15, 20, 30, 40, 50, 60, 80, 90, 100 | [B10] |
| **n7** | FDD | 2,600 | IMT-E | 2,500 | 2,570 | 2,620 | 2,690 | 120 | 5, 10, 15, 20, 25, 30, 35, 40, 50 | |
| **n38 (n41, n90)** | TDD | 2,600 | IMT-E | 2,570 | | 2,620 | | N/A | 5, 10, 15, 20, 25, 30, 40 | [B5] |

For the notes in this table, click 2.2.13.6.

### 2.2.13.2 5G Frequency Range 1, subrange 1-2, terrestrial bands

The following table shows the **5G Frequency Range 1-2 terrestrial bands** (ordered by frequency), with a **summary** of their **characteristics**. Notice that the **channel bandwidths actually available** depend on the **subcarrier spacing** used (click 2.2.3.2).

| Band (Subset of) | Duplex mode | *f* [MHz] | Common name | Uplink [MHz] | Downlink [MHz] | Duplex spacing [MHz] | All possible channel bandwidths [MHz] | Notes |
|---|---|---|---|---|---|---|---|---|
| **n78 (n77)** | TDD | 3,500 | C-Band | 3,300 | 3,800 | N/A | 10, 15, 20, 25, 30, 40, 50, 60, 70, 80, 90, 100 | EU |
| **n48** | TDD | 3,500 | CBRS (US) | 3,550 | 3,700 | N/A | 5[B6], 10, 15, 20, 30, 40, 50[B4], 60[B4], 70[B4], 80[B4], 90[B4], 100[B4] | |
| **n77** | TDD | 3,700 | C-Band | 3,300 | 4,200 | N/A | 10, 15, 20, 25, 30, 40, 50, 60, 70, 80, 90, 100 | EU |
| **n79** | TDD | 4,700 | C-Band | 4,400 | 5,000 | N/A | 10, 20, 30, 40, 50, 60, 70, 80, 90, 100 | NATO |
| **n46** | TDD | 5,200 | U-NII-1–4 | 5,150 | 5,925 | N/A | 10[B4], 20, 40, 60, 80 | [B7] |
| **n47** | TDD | 5,900 | U-NII-4 | 5,855 | 5,925 | N/A | 10, 20, 30, 40 | V2X |
| **n96** | TDD | 6,000 | U-NII-5–9 | 5,925 | 7,125 | N/A | 20, 40, 60, 80 | [B7] |
| **n102** | TDD | 6,200 | U-NII-5 | 5,925 | 6,425 | N/A | 20, 40, 60, 80 | [B7] |
| **n104 (n96)** | TDD | 6,700 | U-NII-6-8 | 6,425 | 7,125 | N/A | 20, 30, 40, 50, 60, 70, 80, 90, 100 | |

For the notes in this table, click 2.2.13.6.

### 2.2.13.3 5G Frequency Range 2, subrange 2-1, bands

The following table shows the **5G Frequency Range 2**, **subrange 2-1**, **bands** (ordered by frequency), with a **summary** of their **characteristics**. Notice that the **channel bandwidths actually available** depend on the **subcarrier spacing** used (click 2.2.3.2).

| Band (Subset of) | Duplex mode | *f* [GHz] | Common name | Uplink&Downlink [GHz] | | Duplex spacing [MHz] | All possible channel bandwidths [MHz] | Notes |
|---|---|---|---|---|---|---|---|---|
| **n258** | TDD | 24 | K-band | 24.25 | 27.5 | N/A | 50, 100, 200, 400 | EU |
| **n257** | TDD | 26 | LMDS | 26.5 | 29.5 | N/A | 50, 100, 200, 400 | US |
| **n261 (n257)** | TDD | 28 | Ka-band | 27.5 | 28.35 | N/A | 50, 100, 200, 400 | US |
| **n260** | TDD | 39 | Ka-band | 37.0 | 40.0 | N/A | 50, 100, 200, 400 | US |
| **n259** | TDD | 41 | V-band | 39.5 | 43.5 | N/A | 50, 100, 200, 400 | |
| **n262** | TDD | 47 | V-band | 47.2 | 48.2 | N/A | 50, 100, 200, 400 | |

For the notes in this table, click 2.2.13.6.

### 2.2.13.4 5G Frequency Range 2, subrange 2-2, bands

The following table shows the **5G Frequency Range 2**, **subrange 2-2**, **bands** (ordered by frequency), with a **summary** of their **characteristics**. Notice that the **channel bandwidths actually available** depend on the **subcarrier spacing** used (click 2.2.3.2).

| Band (Subset of) | Duplex mode | *f* [GHz] | Common name | Uplink&Downlink [GHz] | | Duplex spacing [MHz] | All possible channel bandwidths [MHz] | Notes |
|---|---|---|---|---|---|---|---|---|
| **n263** | TDD | 60 | V-band | 57.0 | 71.0 | N/A | 100, 200, 400, 800, 1,600, 2,000 | [B7] |

For the notes in this table, click 2.2.13.6.

### 2.2.13.5 5G Frequency Range 1, subrange 1, satellite bands

The following table shows the **5G Frequency Range 1-1 satellite bands** (ordered by frequency), with a **summary** of their **characteristics**. Notice that the **channel bandwidths actually available** depend on the **subcarrier spacing** used (click 2.2.3.2).

| Band (Subset of) | Duplex mode | *f* [MHz] | Common name | Uplink [MHz] | | Downlink [MHz] | | Duplex spacing [MHz] | All possible channel bandwidths [MHz] | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| **n255** | FDD | 1,600 | L-band | 1,626.5 | 1,660.5 | 1,525 | 1,559 | -101.5 | 5, 10, 15, 20 | US |
| **n256** | FDD | 2,100 | S-band | 1,980 | 2,010 | 2,170 | 2,200 | 190 | 5, 10, 15, 20 | IMT |

For the notes in this table, click 2.2.13.6.

### 2.2.13.6 Table notes of 5G operating frequency bands

The notes in the different tables of 5G operating frequency bands are the following:

- B1. **Downlink only**.
- B2. **Uplink restricted** to 1,627.5–1,637.5 MHz and 1,646.5–1,656.5 MHz.
- B3. **Downlink restricted** to 1,526–1,536 MHz.
- B4. **Carrier aggregation** and **Downlink only**.
- B5. Duplex Spacing.
- B6. **Carrier aggregation** or **dual connectivity only**.
- B7. **5G NR** in **unlicensed spectrum**.
- B8. **Downlink** between 2,180–2,200 MHz restricted to **intra-band Supplementary Downlink**.
- B9. Supports **asymmetric channel bandwidth** (click 2.2.3.7).
- B10. Technologically identical to **n41** but requires **Uplink rasterization** to allow for **dynamic spectrum sharing** (**DSS**, click 10.7).
- B11. **Variable duplex operation** is used such that the **Downlink** and **Uplink** frequency ranges are supported **independently** in any valid frequency range for the band.
- B12. **Uplink only**.

- B13. Band appears **twice** in the **table** band ordered by **frequency**, because its **Uplink** and **Downlink semi-bands** have a **very distant center frequency**.

Be aware that **not all** the **channel bandwidths** listed in the tables for a given band are allowed. There are **a few cases** where **some channel bandwidths** are **not allowed** or have **usage restrictions**, depending on the value of the **subcarrier spacing**, **Uplink**/**Downlink** usage, **Carrier Aggregation** and other characteristics. As it has been indicated above, if you want to check the **actual definition** of **5G bands**, refer to the different tables in TS 38.101-**1** (**FR1-terrestrial**), TS 38.101-**1** (**FR2**) and TS 38.101-**5** (**FR1-satellite**).

The tables included in this document **only** contain **active bands** as defined in the **TS document versions** used in this book (click 1.2). Deprecated bands are not shown.

#### 2.2.13.7 Evolution of 5G bands

The process to modify band assignment is quite complex and slow. To modify the use of a given spectrum range it is necessary that in a given **World Radio Conference** (**WRC**) a **proposal** is formally made, which leads to studies being carried out, so that in the **next WRC** the proposal is **debated** and either **approved** or **rejected**. The **2019 WRC** (in October, in Egypt) **approved** the bands that had been **proposed** in the previous **2015 WRC**, which were the ones later standardized by **3GPP**.

In the **2023 WRC** (in Nov-Dec, in Dubai), the following spectrum in mid-frequency was allocated for **international mobile telecommunications** (**IMT**) in various countries and regions:

- **3,300** to **3,400 MHz**
- **3,600** to **3,800 MHz**
- **4,800** to **4,990 MHz**
- **6,425** to **7,125 MHz**

The **2 GHz** and the **2.6 GHz bands** were also identified for **high-altitude platforms** (**HAPS**) as **IMT base stations** (**HIBS**) and established regulations for their operations. This technology offers a new platform to provide mobile broadband with minimal infrastructure using the same frequencies and devices as **IMT networks**. **HIBS** can contribute to bridging the digital divide in remote and rural areas and maintain connectivity during disasters.

### 2.2.14 Summary of LTE and other frequency bands

This section contains a **listing** of the **LTE bands** (click 2.2.14.1), providing for **each band** a **summary** of its **characteristics**. It also contains a summary of other bands related to **5G** (click 2.2.14.2).

For a description of the **overall organization of the spectrum**, click 2.1.5.5.

#### 2.2.14.1 Summary of LTE Frequency Bands

The table below presents the **LTE bands**, summarizing their main characteristics. This table has been taken from Wikipedia (in its version from TS 36.101 R18):

https://en.wikipedia.org/wiki/LTE_frequency_bands

If you want to check the **current definition of bands** in the standard, you may refer to the latest version of TS 36.101.

## LTE: Band table (ordered by frequency)

| Band (Subset of) | Duplex mode | *f* (MHz) | Common name | Uplink (MHz) | | Downlink (MHz) | | Duplex spacing (MHz) | Channel bandwidths (MHz) | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| **87** | FDD | 410 | PMR (APT) | 410 | 415 | 420 | 425 | 10 | 1,4, 3, 5 | |
| **88** | FDD | 410 | PMR (EU) | 412 | 417 | 422 | 427 | 10 | 1,4, 3, 5 | |
| **73** | FDD | 450 | PMR (APT) | 450 | 455 | 460 | 465 | 10 | 1,4, 3, 5 | |
| **72** | FDD | 450 | PMR (EU) | 451 | 456 | 461 | 466 | 10 | 1,4, 3, 5 | |
| **31** | FDD | 450 | NMT | 453 | 458 | 463 | 468 | 10 | 1,4, 3, 5 | |
| **71** | FDD | 600 | Digital Dividend (US) | 663 | 698 | 617 | 652 | -46 | 5, 10, 15, 20 | |
| **85** | FDD | 700 | Extended Lower SMH | 698 | 716 | 728 | 746 | 30 | 5, 10 | |
| **12 (85)** | FDD | 700 | Lower SMH | 699 | 716 | 729 | 746 | 30 | 1,4, 3, 5, 10 | |
| **28** | FDD | 700 | APT | 703 | 748 | 758 | 803 | 55 | 3, 5, 10, 15, 20 | |
| **17 (12, 85)** | FDD | 700 | Lower SMH | 704 | 716 | 734 | 746 | 30 | 5, 10 | |
| **13** | FDD | 700 | Upper SMH | 777 | 787 | 746 | 756 | -31 | 5, 10 | |
| **14** | FDD | 700 | Upper SMH | 788 | 798 | 758 | 768 | -30 | 5, 10 | |
| **29** | SDL | 700 | Lower SMH | N/A | | 717 | 728 | - | 3, 5, 10 | |
| **67** | SDL | 700 | EU 700 | N/A | | 738 | 758 | - | 5, 10, 15, 20 | |
| **103** | FDD | 700 | Upper SMH | 787 | 788 | 757 | 758 | -30 | | [B7] |
| **20** | FDD | 800 | Digital Dividend (EU) | 832 | 862 | 791 | 821 | -41 | 5, 10, 15, 20 | |
| **26** | FDD | 850 | Extended Cellular | 814 | 849 | 859 | 894 | 45 | 1,4, 3, 5, 10, 15 | |
| **18 (26)** | FDD | 850 | Lower 800 (Japan) | 815 | 830 | 860 | 875 | 45 | 5, 10, 15 | |
| **5 (26)** | FDD | 850 | Cellular | 824 | 849 | 869 | 894 | 45 | 1,4, 3, 5, 10 | |
| **19 (26)** | FDD | 850 | Upper 800 (Japan) | 830 | 845 | 875 | 890 | 45 | 5, 10, 15 | |
| **8** | FDD | 900 | Extended GSM | 880 | 915 | 925 | 960 | 45 | 1,4, 3, 5, 10 | |
| **51** | TDD | 1,500 | Extended L-Band (EU) | 1,427 | | 1,432 | | — | 3, 5 | |
| **74** | FDD | 1,500 | Lower L-Band (US) | 1,427 | 1,470 | 1,475 | 1,518 | 48 | 1,4, 3, 5, 10, 15, 20 | |
| **11 (74)** | FDD | 1,500 | Lower PDC | 1,428 | 1,448 | 1,476 | 1,496 | 48 | 5, 10 | |
| **50** | TDD | 1,500 | L-Band (EU) | 1,432 | | 1,517 | | — | 3, 5, 10, 15, 20 | |
| **21 (74)** | FDD | 1,500 | Upper PDC | 1,448 | 1,463 | 1,496 | 1,511 | 48 | 5, 10, 15 | |
| **76** | SDL | 1,500 | Extended L-Band (EU) | N/A | | 1,427 | 1,432 | — | 5 | |
| **75** | SDL | 1,500 | L-Band (EU) | N/A | | 1,432 | 1,517 | — | 5, 10, 15, 20 | |
| **32 (75)** | SDL | 1,500 | L-Band (EU) | N/A | | 1,452 | 1,496 | — | 5, 10, 15, 20 | |
| **24** | FDD | 1,600 | Upper L-Band (US) | 1,627 | 1,661 | 1,525 | 1,559 | -101,5 or -120,5 | 5, 10 | |
| **54** | TDD | 1,600 | L-Band | 1,670 | | 1,675 | | — | 1,4, 3, 5 | |

| Band (Subset of) | Duplex mode | *f* (MHz) | Common name | Uplink (MHz) | | Downlink (MHz) | | Duplex spacing (MHz) | Channel bandwidths (MHz) | Notes |
|---|---|---|---|---|---|---|---|---|---|---|
| **4 (66)** | FDD | 1,700 | AWS-1 | 1,710 | 1,755 | 2,110 | 2,155 | 400 | 1,4, 3, 5, 10, 15, 20 | |
| **66** | FDD | 1,700 | Extended AWS (AWS-1–3) | 1,710 | 1,780 | 2,110 | 2,200 | 400 | 1,4, 3, 5, 10, 15, 20 | |
| **3** | FDD | 1,800 | DCS | 1,710 | 1,785 | 1,805 | 1,880 | 95 | 1,4, 3, 5, 10, 15, 20 | |
| **2 (25)** | FDD | 1,900 | PCS | 1,850 | 1,910 | 1,930 | 1,990 | 80 | 1,4, 3, 5, 10, 15, 20 | |
| **25** | FDD | 1,900 | Extended PCS | 1,850 | 1,915 | 1,930 | 1,995 | 80 | 1,4, 3, 5, 10, 15, 20 | |
| **39** | TDD | 1,900 | DCS–IMT Gap | 1,880 | | 1,920 | | — | 5, 10, 15, 20 | |
| **37** | TDD | 1,900 | PCS | 1,910 | | 1,930 | | — | 5, 10, 15, 20 | [B8] |
| **70** | FDD | 2,000 | AWS-4 | 1,695 | 1,710 | 1,995 | 2,020 | 300 or 295 [B4] | 5, 10, 15, 20[B5] | [B6] |
| **1 (65)** | FDD | 2,100 | IMT | 1,920 | 1,980 | 2,110 | 2,170 | 190 | 5, 10, 15, 20 | |
| **65** | FDD | 2,100 | Extended IMT | 1,920 | 2,010 | 2,110 | 2,200 | 190 | 1,4, 3, 5, 10, 15, 20 | |
| **34** | TDD | 2,100 | IMT | 2,010 | | 2,025 | | — | 5, 10, 15 | |
| **40** | TDD | 2,300 | S-Band | 2,300 | | 2,400 | | — | 5, 10, 15, 20 | |
| **30** | FDD | 2,300 | WCS | 2,305 | 2,315 | 2,350 | 2,360 | 45 | 5, 10 | |
| **53** | TDD | 2,400 | S-Band | 2,484 | | 2,495 | | — | 1,4, 3, 5, 10 | |
| **41** | TDD | 2,500 | BRS | 2,496 | | 2,690 | | — | 5, 10, 15, 20 | |
| **7** | FDD | 2,600 | IMT-E | 2,500 | 2,570 | 2,620 | 2,690 | 120 | 5, 10, 15, 20 | |
| **38 (41)** | TDD | 2,600 | IMT-E | 2,570 | | 2,620 | | — | 5, 10, 15, 20 | [B9] |
| **69** | SDL | 2,600 | IMT-E | N/A | | 2,570 | 2,620 | — | 5, 10, 15, 20 | [B9] |
| **42** | TDD | 3,500 | CBRS (EU, Japan) | 3,400 | | 3,600 | | — | 5, 10, 15, 20 | |
| **48** | TDD | 3,500 | CBRS (US) | 3,550 | | 3,700 | | — | 5, 10, 15, 20 | |
| **43** | TDD | 3,700 | C-Band | 3,600 | | 3,800 | | — | 5, 10, 15, 20 | |
| **46** | TDD | 5,200 | U-NII-1-4 | 5,150 | | 5,925 | | — | 10, 20 | LAA |
| **47** | TDD | 5,900 | U-NII-4 | 5,855 | | 5,925 | | — | 10, 20 | V2X |

The notes indicated in the previous table mean:

- B1. Uplink restricted to 1627.5–1637.5 MHz and 1646.5–1656.5 MHz.
- B2. Downlink restricted to 1526–1536 MHz.
- B3. Downlink between 2180–2200 MHz restricted to intra-band Supplementary Downlink.
- B4. Duplex spacing depends on whether the Uplink is paired with the lower or the upper part of the Downlink, with the remainder of the Downlink available for use as intra-band Supplementary Downlink.
- B5. Carrier aggregation only.

- B6. Downlink is made up of the PCS-H **Downlink** block and the AWS-4 **Uplink** blocks. The **Uplink** is made up of the AWS-3 unpaired blocks.
- B7. Restricted to NB-IoT only.
- B8. PCS Duplex Spacing.
- B9. IMT-E Duplex Spacing.

Be aware that may be not all the channel bandwidths listed in the tables for a given band are allowed. If you want to check the **actual definition** of **LTE bands** in the standard, refer to the different tables in TS 36.101.

The table above only reflect active bands (deprecated bands are not shown).

Source web pages:

- https://www.sqimway.com/lte_band.php
- https://en.wikipedia.org/wiki/LTE_frequency_bands

#### 2.2.14.2 ISM bands

The **ISM** radio bands are portions of the radio spectrum reserved internationally for **industrial**, **scientific**, and **medical** (**ISM**) purposes, excluding applications in telecommunications. Examples of applications for the use of radio frequency (**RF**) energy in these bands include radio-frequency process heating, microwave ovens, and medical diathermy machines. The powerful emissions of these devices can create electromagnetic interference and disrupt radio communication using the same frequency, so these devices are limited to certain bands of frequencies. In general, communications equipment operating in **ISM** bands must tolerate any interference generated by **ISM** applications, and users have no regulatory protection from **ISM** device operation in these bands.

Despite the intent of the original allocations, along the recent years the fastest-growing use of these bands has been for **short-range**, **low-power wireless communications**, since these bands are often approved for such devices, which can be used without a government license, as would otherwise be required for transmitters. **ISM** frequencies are often chosen for this purpose as they already must tolerate interference issues. Cordless phones, Bluetooth devices, near-field communication (**NFC**) devices, garage door openers, baby monitors, and wireless computer networks (**Wi-Fi**) may all use the **ISM** frequencies, although these low-power transmitters are not formally **ISM** devices.

The **ISM** bands are sometimes called “unlicensed bands”, “free bands” or “free spectrum”. The fact that this spectrum can be used without a license does not mean that it is not regulated: actually, its usage is subject to specific regulation in terms of power, type of signals, etc.

Source web pages:

- https://en.wikipedia.org/wiki/ISM_radio_band

#### 2.2.14.3 Summary of other radio bands related to 5G and LTE

**GSM-Railway band**

For the use of **GSM** on **railways**, a special band was defined, which is **GSM-R**.

The **GSM-R** band is at **876-880 MHz** for the upstream channel and **921-925 MHz** for the downstream channel. The Uplink is fully included in **LTE** band **5** and **5G** band n5.

Since they overlap, non-interference is resolved by assignment by country (for example, in the **USA** the **GSM-R** band does not exist, so there is no problem), or by zones (for example, band **5/n5** can be used as long as it is away from railway tracks).

**Wi-Fi Bands**

**Wi-Fi bands** are **unlicensed**, and are regulated as **Industrial**, **Scientific** or **Medical** (**ISM**) purpose **bands**.

The lower **WiFi** (**ISM**) band is from **2.4** to **2.5 GHz**. It is placed between **5G** bands **n40** and **n41**.

The higher **WiFi** (**ISM**) band is from **5.725** to **5.875 GHz**. It is placed **725 MHz** above **FR1** and very much below **FR2**.

The **WiFi** (**ISM**) millimeter band (**WiGig**) is from **57.24** to **70.20 GHz** (having **12.96 GHz** of bandwidth). It is located completely within **FR2 band n263**, and it has a slightly lower bandwidth than **n263**. Therefore, **band n263** has to be used with **unlicensed spectrum operation** mode because it is located within ISM spectrum.

**Radio-altimeter bands**

Radio-altimeters used in commercial airplanes operate in the range from **4.2 GHz** to **4.4 GHz**. This range is placed **exactly** between the **n77** (**EU**) and the **n79** (**NATO**) bands.

### 2.2.15 Global frequency raster and absolute radio frequency channel number (ARFCN)

The **5G global frequency raster** is a **list** of **exact frequencies**, **each** of which is **identified** by an **integer number** called **New Radio Absolute Radio Frequency Channel Number** (**NR-ARFCN**). The **frequencies** within the **5G global frequency raster** are called **global reference frequencies**. Some physical layer elements **must** be **located** at the **exact** value of one of the **global reference frequencies**.

The **list** of **global reference frequencies** in the **global frequency raster** is denoted "$F_{REF}$". The **list** of **NR-ARFCNs** in the **global frequency raster** is denoted "$N_{REF}$". There is a **one-to-one correspondence** between the elements of "$F_{REF}$" and the elements of "$N_{REF}$". Therefore, **each NR-ARFCN** from "$N_{REF}$" **identifies** the **exact value** of **one and only one global reference frequency** from "$F_{REF}$". Conversely, **each global reference frequency** within "$F_{REF}$" can be mapped to **one and only one NR-ARFCN** within "$N_{REF}$".

The **subset** of the **5G global frequency raster** that is used to locate the **5G channels** is the **5G channel raster** (click 2.2.3.3), that contains the **channel reference frequencies** and their corresponding **channel numbers**.

The **NR-ARFCNs** in "$N_{REF}$" are used in **RRC signaling** (click 5.8) and **configuration** (click 8.9) to identify the **exact** frequency **location** of **5G channels** (click 2.2.3), **Point A**, (click 2.2.4.3) and other **5G** physical layer elements.

The **global reference frequencies** within "$F_{REF}$" are distributed in the **range** from **0 Hz** to **99,999.96 MHz**. The **NR-ARFCNs** within "$N_{REF}$" are **all** the **integers** from **0** to **3,279,165**.

Obtaining the **global reference frequency identified** by **each NR-ARFCN** (and vice versa) is done with a simple **equation**. In this **equation**, **each unit** of "$N_{REF,i}$" increases the **global reference frequency** in a **fixed amount** denoted "$\Delta F_{Global,s}$", that is **specific**

of **each** of the **three frequency segments** "***s***" that have been defined (see the table below). The **value** of "$\Delta F_{Global,s}$" is therefore the **exact spacing** between **two consecutive global reference frequencies**, within **each frequency segment** "***s***".

The **equations** to get "$F_{REF}$" from "$N_{REF}$" (and vice versa), for **each frequency segment** "***s***", are (respectively):

$$F_{REF,s} = F_{REF\text{-}Offs,s} + \Delta F_{Global,s} \bullet (N_{REF,s} - N_{REF\text{-}Offs,s})$$

$$N_{REF,s} = N_{REF\text{-}Offs,s} + (F_{REF,s} - F_{REF\text{-}Offs,s}) / \Delta F_{Global,s}$$

For the **second equation**, the resulting value of "$N_{REF,s}$" **must** be an **integer**: otherwise, its **input** frequency "$F_{REF,s}$" **does not** belong to the **5G global frequency raster**.

The values of parameters "$F_{REF\text{-}Offs,s}$", "$N_{Ref\text{-}Offs,s}$" and "$\Delta F_{Global,s}$" in the **two equations** above, for **each frequency segment** "***s***", are listed in the following table (from tables **5.4.2.1-1** of TS 38.101-**1**, -**2** and -**5**):

| ***s*** | **Frequency segment** | **Inc. FR** | $\Delta F_{Global,s}$ | $F_{REF\text{-}Offs,s}$ | $N_{REF\text{-}Offs,s}$ | **Range of** $N_{REF,s}$ | **Range of** $F_{REF,s}$ |
|---|---|---|---|---|---|---|---|
| **1** | **0 to 3,000 MHz** | FR1-1 | 5 kHz | 0 MHz | 0 | 0 to 599,999 | 0 to 2,999.995 MHz |
| **2** | **3,000 to 24,250 MHz** | FR1-2 | 15 kHz | 3,000 MHz | 600,000 | 600,000 to 2,016,666 | 3,000 to 24,249.99 MHz |
| **3** | **24,250 to 100,000 MHz** | FR2 | 60 kHz | 24,250.08 MHz | 2,016,667 | 2,016,667 to 3,279,165 | 24,250.08 to 99,999.96 MHz |

The column "**Inc. FR**" indicates what **frequency (sub)range** is contained within **each** of the **three frequency segments**.

In **higher frequencies** the **bandwidths** are **larger**, and the **spacing** between **global reference frequencies** should also be **larger**. For this reason, the **spacing** $\Delta F_{Global,s}$ is **larger** in **higher frequency segments**.

Although the **NR-ARFCNs** and the **GSCNs** (click 6.1.2.1) cover the **same frequency range** (0 to 100,000 MHz), they are **very different**. To point just one clear difference, the **range** of **NR-ARFCNs** is from **0** to **3,279,165**, while the **range** of **GSCNs** is from **2** to **26,639**.

For more detail on the **5G global frequency raster** and **NR-ARFCNs**, refer to clause **5.4.2.1** of TS 38.101-**1**, -**2** and -**5**.

Source web pages:

- https://www.nrexplained.com/raster
- https://www.sqimway.com/nr_band.php

**ARFCN in other mobile communication technologies**

The **Absolute Radio Frequency Channel Number** (**ARFCN**), as a general concept, is an **integer number** that serves to identify the **exact frequency location** of a **channel** and of a few relevant **channel-related elements**. The way **each ARFCN maps** to a **specific frequency location** depends on **each mobile communications technology** (**5G**, **LTE**, **UMTS** or **GSM**). For this reason, **each technology** has a specific **name** for the **ARFCN**.

In **LTE**, its specific **ARFCN** is called "**Extended** Absolute Radio Frequency Channel Number" or "**EARFCN**". It identifies the **center frequency** of the **next-above-middle subcarrier** of the **carrier**.

In **UMTS**, its specific **ARFCN** is called "**UTRA** Absolute Radio Frequency Channel Number" or "**UARFCN**".

In **GSM**, its specific **ARFCN** is just called "Absolute Radio Frequency Channel Number". It is an **integer number**, that identifies the **center frequency** of **both** the **Downlink and** the **Uplink carriers**, in each **GSM carrier pair**.

## 2.3 NR time management

### 2.3.1 Transmission time structure

The **Downlink**, **Uplink** and **Sidelink transmission time** is structured into **consecutive synchronous frames**, **each** of **10 ms** duration.

**Each frame** is composed of **2 half-frames**, **each** of **5 ms** duration.

**Each half-frame** is composed of **5 subframes**, **each** of **1 ms** duration.

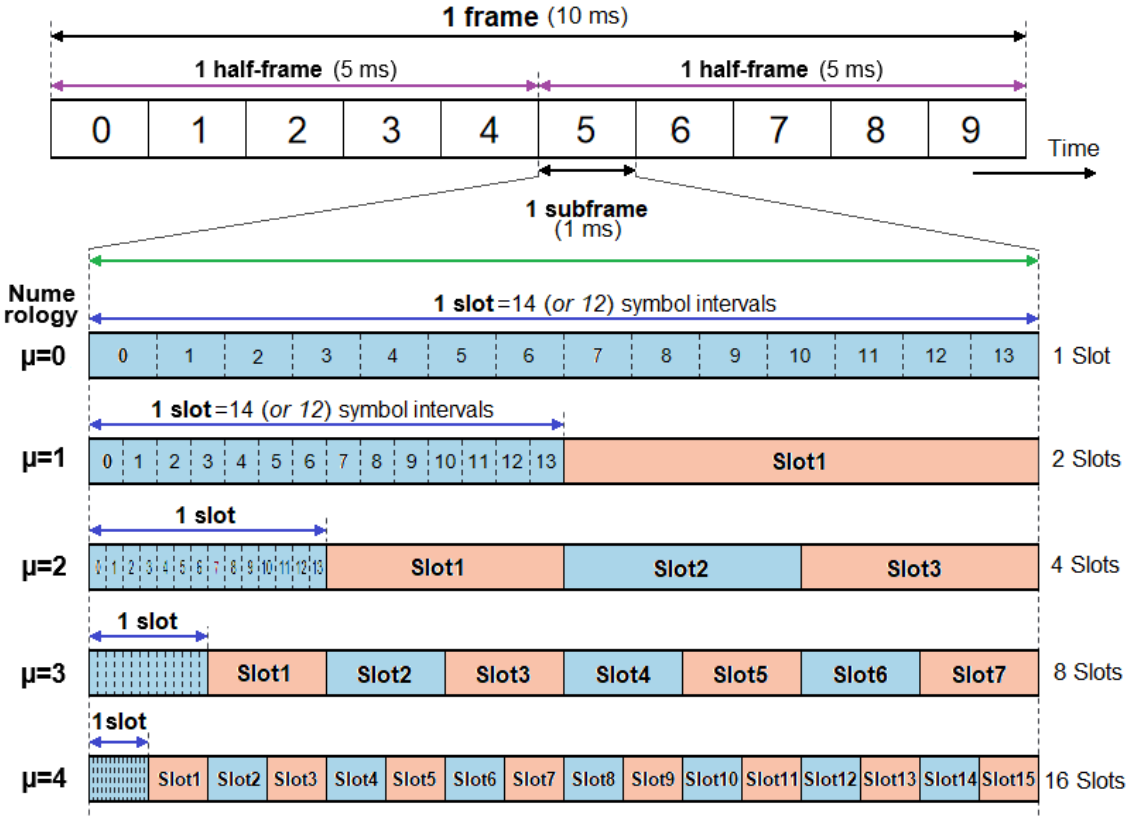
**Each subframe** is composed of **$2^{\mu}$ slots**, **each** of **$2^{-\mu}$ ms average** duration, where "**μ**" is the **data/shared numerology** (click 2.1.4).

**Each slot** is composed of **14 symbol intervals** when **normal cyclic prefix** (**NCP**, click 2.3.2) is used. The duration of **each symbol interval** with **normal CP** has some **slight variations**. The **average** duration of **each symbol interval** with **normal CP** is **$2^{-\mu}$/14 ms**.

There is also the **extended CP** (**ECP**, click 2.3.2), that when used makes a **slot** contain **only 12 symbol intervals**, but its usage is not so frequent. The duration of **each symbol interval** with **extended CP** is **constant**, and it is **$2^{-\mu}$/12 ms**.

Recall from 2.1.1 that the **average slot duration** and the **average symbol interval duration** (for a given **CP**) are **both proportional** to **$2^{-\mu}$**. Therefore, **increasing** the **numerology** "**μ**" in **1 unit halves** the **average slot duration** and the **average symbol interval duration**. Also, the **quantity** of **slots** in **1 subframe doubles** with **each 1 unit increase** in **numerology.**

The following figure shows the **5G time structure** for the **5 smallest** values of the **numerology**, when using **normal cyclic prefix** (which implies **14 symbols intervals** per **slot**):




The 2 missing numerologies "**μ**=5" and "**μ**=6" are not shown in the figure for visibility's sake, but they follow exactly same pattern.

The following table presents the **quantity** of **slots** in each **subframe**, the **quantity** of **symbol intervals** per **slot** and the **symbol rate** depending on the **numerology** and on the type of **cyclic prefix**:

| Freq. Range | Numerology "μ" | SCS ($15 \cdot 2^{\mu}$) [kHz] | Slots per subframe ($2^{\mu}$) | Symbol intervals per slot | Subcarrier symbol rate [kbaud] | Symbol rate for 960 kHz [kbaud] |
|---|---|---|---|---|---|---|
| **FR1** | **0** | **15** | 1 | 14 | 14 | 896 |
| | **1** | **30** | 2 | | 28 | 896 |
| **FR1, FR2-1** | **2 NCP** | **60 NCP** | 4 | | 56 | 896 |
| **FR2-1** | **3** | **120** | 8 | | 112 | 896 |
| **FR2-2** | **4** | **240** | 16 | | 224 | 896 |
| | **5** | **480** | 32 | | 448 | 896 |
| | **6** | **960** | 64 | | 896 | 896 |
| **FR1, FR2-1** | **2 ECP** | **60 ECP** | 4 | ***12*** | ***48*** | ***768*** |

Some **relevant remarks** on this table are:

- **Numerology** "**μ=2**" (i.e., **SCS=60 kHz**) is the **only one** with **two rows**, because it is the only **numerology** that can be used with the **two** options of **cyclic prefix**

**duration**. One **row** is for the **normal CP** (**NCP**) and the other **row** is for the **extended CP** (**ECP**).

- **Numerology** "**μ=4**" (i.e., **SCS=240 kHz**) can **only** be used for the **SSB**. Conversely, **Numerology** "**μ=2**" (i.e., **SCS=60 kHz**) **cannot** be used for **SSB**.
- The **symbol rate** of **1 subcarrier** increases with **subcarrier spacing**. However, the **symbol rate** for a given **bandwidth** is **constant**, (i.e., it does **not depend** on the **subcarrier spacing**).
- The **symbol rate** for a given **bandwidth** is **lower** (in the ratio 12/14) with the **extended cyclic prefix** than with the **normal cyclic prefix**.
- The **PRACH-specific subcarrier spacings** (**1.25 kHz** and **5 kHz**) are **not** included in the table, because most of the concepts shown in the table are non-applicable to them.

For more detail on the **time duration** of **cyclic prefix**, **symbol**, **symbol interval** and **slot**, click 2.3.2.

In **channels** using **frequency division duplexing** (click 2.2.12), corresponding to **FDD**, **SDL** and **SUL bands**, **each carrier type** (**Downlink**, **Uplink**, **SDL** or **SUL**) has **its own**, **independent**, **time domain structure**, as has just been described. The **Downlink** and **Uplink** time domain structures **must** be **precisely synchronized** (click 2.3.6.1).

In **channels** using **time division duplexing** (click 2.3.4), corresponding to **TDD bands**, there are **two partial time domain structures**, one transmitted **Downlink** and the other one transmitted **Uplink**, and **both must match exactly** into **one single time domain structure**, with a very **precise synchronization** (click 2.3.6.2). This is usually **presented** as **one time domain structure**, for **both Downlink** and **Uplink**. In this **single time domain structure**, **some symbol intervals** in **each slot** may be transmitted **Downlink**, while **other symbols intervals** in the **same slot** may be transmitted **Uplink**.

**Frames** are cyclically **numbered** from **0** to **1,023**. The **Downlink 10-bit** frame **number** is **broadcast jointly** between **PBCH System Information** (click 5.3.2) **and** `MIB` (click 5.3.3), both within the **SSB** (which is **not** sent in every frame, click 6.1.2.1.3).

**Subframes** are **indexed** within **each frame**, from **0** to **9**. **Subframes** are **indexed** within **each half-frame**, from **0** to **4**.

**Slots** are **indexed** within **each frame**, from **0** to $\mathbf{10 \cdot 2^{\mu} - 1}$. **Slots** are **indexed** within **each half-frame**, from **0** to $\mathbf{5 \cdot 2^{\mu} - 1}$. **Slots** are **indexed** within **each subframe**, from **0** to $\mathbf{2^{\mu} - 1}$.

**Symbol intervals** and **symbols** are **indexed** within**:**

- **Each frame**, from **0** to $\{\mathbf{14}, \mathbf{12}\} \cdot \mathbf{10 \cdot 2^{\mu} - 1}$.
- **Each half-frame**, from **0** to $\{\mathbf{14}, \mathbf{12}\} \cdot \mathbf{5 \cdot 2^{\mu} - 1}$.
- **Each subframe**, from **0** to $\{\mathbf{14}, \mathbf{12}\} \cdot \mathbf{2^{\mu} - 1}$.
- **Each slot**, from **0** to $\{\mathbf{14}, \mathbf{12}\} - \mathbf{1}$.

where "**14**" corresponds to the **normal cyclic prefix** and "**12**" corresponds to the **extended cyclic prefix**.

The **subframe index**, the **slot index** and the **symbol index** are **not** sent explicitly in a field, and therefore their value is **implicit** in their **relative** temporal location. A given **slot** has **different index values** in respect to its enclosing **frame**, to its enclosing **half-frame**

and to its enclosing **subframe** (except for the first subframe in each frame). The same happens with a given **symbol interval**, that has **different index values** in respect to its enclosing **frame**, to its enclosing **half-frame**, to its enclosing **subframe** and to its enclosing **slot** (except for the first subframe of each frame, the first half-frame of each frame and the first slot of each subframe).

The **concepts** of **multi-slot scheduling** (click 4.11.4.5) and **mini-slot scheduling** (click 4.11.4.6) are more a **scheduling aspect** than a **time structure** aspect.

Notice that for a subcarrier spacing of **15 kHz**, which is the same one as in **LTE**, the **5G slot** duration is **1 ms**, while the **LTE** slot duration is **0.5 ms**, which is **different**.

For more detail on the **transmission time structure**, refer to clause 4.3 of TS 38.211.

**Some timing aspects of LTE**

In **LTE**, same as in **5G**, the **Downlink** and **Uplink** transmission time is divided into **frames** of **10 ms** duration. Each **frame** is divided into **2** equally sized **half-frames**. Each **frame** is itself divided into **10 subframes** of **1 ms**. Each **half-frame** therefore has **5 subframes**.

In **LTE**, **each subframe** is divided into **two 0.5 ms slots** (although the slot in **LTE** has very little technical relevance). In **LTE** the **subcarrier spacing** (**SCS**) is **15 kHz** and **each LTE subframe** contains **14 symbols** (and **each LTE slot 7 symbols**). There is an exception for the **Multimedia Broadcast Multicast Service** (**MBMS**) where **SCS** is **7.5 kHz**. In this case, the **subframe** is still **1 ms**, and therefore it contains **7** symbols instead of **14**.

In **LTE**, in **Downlink**, there are three types of Frame Structure: **Type 1** for **FDD**, **Type 2** for **TDD**, **Type 3** for **LAA**. **Type 3** is the only one that allows for some symbol scheduling, in a limited degree. A few symbols at the beginning of subframe should be reserved for control channel (**PDCCH**, **PHICH**, **PCFICH**). The quantity of reserved symbols can be **1**, **2**, **3** or **4**. (This quantity is informed to **UE** by **PCFICH**).

Source web page:

- https://medium.com/@jessica.chchuang/5g-urllc-mini-slot-in-5gnr-b830a534da8c

### 2.3.2 Duration of cyclic prefix, symbol, symbol interval and slot

The following **5G** concepts are presented in this section:

- **Cyclic prefix** (click 11.13) **duration**

  It is the **time duration** of the **cyclic prefix** of **one OFDM symbol** in a given **subcarrier**. Do not confuse it with the **PRACH-specific cyclic prefix duration** (click 7.3.1).

- **Symbol duration**

  It is the **time duration** of **one OFDM symbol** in a given **subcarrier**. Do not confuse it with the **symbol interval duration**. Do not confuse it with the **PRACH-specific symbol duration** (click 7.3.1).

- **Symbol interval duration**

  It is the **time duration** of **1 symbol plus** the **time duration** of **its cyclic prefix**. Do not confuse it with the **symbol duration**, **nor** with the **cyclic prefix duration** (see above).

- **Slot duration**

  It is the **time duration** of **1 slot**.

In spite of **possibly** having **different simultaneous numerologies**, **each radio channel** has **only 1** (average) **slot duration** at any **given moment**. The **slot duration** of the **radio channel** corresponds to the **numerology** of its **currently active** (click 2.2.7.4) **bandwidth part** (click 2.2.7). This **numerology also** corresponds to the following **elements** in the **radio channel**:

- The **control/shared subcarrier spacing** (click 2.2.5.8).
- The **carrier** (click 2.2.5.1).
- The **resource grid** (click 2.2.5.6).

In the **3GPP** TS documents many **time values** are expressed as a **quantity** of (**basic**) **time units** (clause 4.1 of TS 38.211). The **5G basic time unit** is calculated as:

$$T_c = 1 / (\Delta f_{max} \cdot N_f) = 1 / (480{,}000 \cdot 4{,}096) = \mathbf{508.6\ ps}$$

where "$\Delta f_{max}$" is **480 kHz** and "$N_f$" is the **5G DFT reference size**, which is **4,096**.

**5G** applies a **scaling** of "$2^{-\mu}$" to the **basic time unit** "$T_c$" in order to support its **different subcarrier spacings** (click 2.2.10.3). For **each** value of the **numerology** "**μ**" (click 2.1), the **time duration** of **one subcarrier period** is based on the **time unit** "$T_\mu$", defined as follows:

$$T_\mu = 2^{-\mu} \cdot T_c$$

In **LTE**, the basic time unit is denoted "$T_s$":

$$T_s = 1 / (15{,}000 \cdot 2{,}048) = \mathbf{32.55\ ns}$$ (clause 4 of TS 36.211)

The ratio "$\kappa$" between "$T_s$" and "$T_c$" is also frequently used in the TS documents:

$$\kappa = T_s / T_c = \mathbf{64}$$

In **5G**, there are **two** types of **cyclic prefix** (**CP**): the **normal** cyclic prefix (**NCP**) and the **extended** cyclic prefix (**ECP**), of longer duration. When the **normal CP** is used, each **slot** has **14 symbol intervals**. When the **extended CP** is used, each **slot** has **12 symbol intervals**. The **normal CP** can be used with **all** the **subcarrier spacings** (except the **PRACH-specific** ones). The **extended CP** can **only** be used with **numerology 2** (i.e., **60 kHz subcarrier spacing**).

The **normal CP** has **two durations**: **normal-short CP** (**SCP**) and **normal-long CP** (**LCP**), of longer duration. When the **normal CP** is used, **each subframe** has **2 normal-long CPs**, corresponding to the **first** and the **next-after-middle symbols** of the **subframe**. These **first** and **next-after-middle symbols** have **symbol subframe indexes 0** and $\mathbf{7 \cdot 2^{\mu}}$ (i.e., 7, 14, 28, 56, 112, 224 and 448). **All** the **other subframe symbols** have the **normal-short CP**.

The following table presents the **duration** of a **symbol**, a **normal-short CP**, a **normal-long CP** and an **extended CP**, depending on the **numerology**, and expressed both in **μs** and in multiples of "$T_c$":

| | | | Symbol and CP duration | | | | | | | | Prefix synchr. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Freq. Range | μ | SCS [kHz] | Symbol duration | | Normal CP (NCP) | | | | Extended CP (ECP) | | LCP + $(2^{-\mu}-1)\cdot$SCP [$T_c$] |
| | | | | | Short CP (SCP) | | Long CP (LCP) | | | | |
| | | | $2{,}048\kappa\cdot2^{-\mu}$ [$T_c$] | [μs] | $144\kappa\cdot2^{-\mu}$ [$T_c$] | [μs] | SCP+$16\kappa$ [$T_c$] | [μs] | $512\kappa\cdot2^{-\mu}$ [$T_c$] | [μs] | |
| FR1 | 0 | 15 | 131,072 | 66.67 | 9,216 | 4.69 | 10,240 | 5.21 | N/A | N/A | 10,240 |
| | 1 | 30 | 65,536 | 33.33 | 4,608 | 2.34 | 5,632 | 2.86 | N/A | N/A | 10,240 |
| FR1, FR2-1 | 2 | 60 | 32,768 | 16.67 | 2,304 | 1.17 | 3,328 | 1.69 | 8,192 | 4.2 | 10,240 |
| FR2-1 | 3 | 120 | 16,384 | 8.33 | 1,152 | 0.59 | 2,176 | 1.11 | N/A | N/A | 10,240 |
| FR2-2 | 4 | 240 | 8,192 | 4.17 | 576 | 0.29 | 1,600 | 0.81 | N/A | N/A | 10,240 |
| | 5 | 480 | 4,096 | 2.08 | 288 | 0.15 | 1,312 | 0.67 | N/A | N/A | 10,240 |
| | 6 | 960 | 2,048 | 1.04 | 144 | 0.07 | 1,168 | 0.59 | N/A | N/A | 10,240 |

Some **relevant remarks** on this table are:

- The **symbol duration** is **inversely** proportional to the **subcarrier spacing**.
- The **symbol duration** is **independent** of the **cyclic prefix** used.
- The **normal-short CP duration** and the **extended CP duration** are, **each**, **inversely** proportional to the **subcarrier spacing**.
- The **normal-long CP duration** is the **normal-short CP duration plus** the **constant** quantity "**$16\kappa\cdot T_c$**" (i.e., "**1,024 $T_c$**").
- The **normal-long CP duration** and the **normal-short CP duration** are designed such that the **symbol interval time edges** corresponding to **symbol intervals** of **two subcarriers** with **different subcarrier spacings** will **match** (in powers of 2) after the **symbol interval** with the **longer normal-long CP** (i.e., the **symbol interval** with **normal-long CP** of the **smaller subcarrier spacing**). This is **computed** in the **last column** of the table. For a figure illustrating this, click 2.1.5.5.
- The **symbol interval duration** is the **addition** of a **symbol duration** and of the **corresponding CP duration**. It is **not** shown in the **table above**, and it is **rather** shown in the **table below**.
- **Numerology** "**μ=4**" (**SCS=240 kHz**) can **only** be used for the **SSB**. Conversely, **numerology** "**μ=2**" (**SCS=60 kHz**) **cannot** be used for **SSB**.
- The **PRACH-specific** subcarrier spacings (**1.25 kHz** and **5 kHz**) are **not** included in the table, because most of the concepts shown in the table are non-applicable to them.

The following table presents the **symbol interval duration** with the **short cyclic prefix** and with the **long cyclic prefix**, the **symbol interval duration average** over a **subframe**,

and the **symbol interval duration** with the **extended cyclic prefix**, depending on the **numerology**, and expressed both in **μs** and in multiples of "$T_c$":

| Freq. Range | Numerology "μ" | SCS ($15 \cdot 2^{\mu}$) [kHz] | Symbol interval duration | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | Normal-short CP (SCP) | | Normal-long CP (LCP) | | Normal CP (average) | | Extended CP (ECP) | |
| | | | [$T_c$] | [μs] | [$T_c$] | [μs] | [$T_c$] | [μs] | [$T_c$] | [μs] |
| FR1 | 0 | 15 | 140,288 | 71.35 | 141,312 | 71.87 | 140,434.3 | 71.43 | NA | N/A |
| | 1 | 30 | 70,144 | 35.68 | 71,168 | 36.19 | 70,217.1 | 35.71 | N/A | N/A |
| FR1, FR2-1 | 2 | 60 | 35,072 | 17.84 | 36,096 | 18.35 | 35,108.6 | 17.86 | 40,960 | 20.8 |
| FR2-1 | 3 | 120 | 17,536 | 8.919 | 18,560 | 9.440 | 17,554.3 | 8.929 | N/A | N/A |
| FR2-2 | 4 | 240 | 8,768 | 4.460 | 9,792 | 4.980 | 8,777.1 | 4.464 | N/A | N/A |
| | 5 | 480 | 4,384 | 2.230 | 5,408 | 2.751 | 4,388.6 | 2.232 | N/A | N/A |
| | 6 | 960 | 2,192 | 1.115 | 3,216 | 1.636 | 2,194.3 | 1.116 | N/A | N/A |

Some **relevant remarks** on this table are:

- The **symbol interval duration** with the **normal CP** is **not constant**. Its duration depends on whether the **normal-short CP** or the **normal-long CP** is used.
- The **symbol interval duration** with the **normal-short CP** is "**1,024 $T_c$**" **less** than the one with the **normal-long CP**.
- The **symbol interval duration** with the **normal-short CP** is **inversely** proportional to the **subcarrier spacing**. Notice that **both** the **symbol duration** and the **normal-short CP duration** are **inversely** proportional to the **subcarrier spacing**.
- The **symbol interval duration** with the **normal-long CP** is **not inversely** proportional to the **subcarrier spacing**. Notice that the **symbol duration** is **inversely** proportional to the **subcarrier spacing**, but the **normal-short CP duration** is **not inversely** proportional to the **subcarrier spacing**.
- The **average symbol interval duration** with the **normal CP** is **inversely** proportional to the **subcarrier spacing**. Recall that **normal CP symbol intervals** have **two different durations**. Recall that the **subframe duration** is constant (independent of the **subcarrier spacing**) while the quantity of **slots** per **subframe** is **exactly proportional** to the **subcarrier spacing**.
- The **symbol interval duration** with the **extended CP** is **constant**. However, this is **not** the case for the **normal CP** (see the first bullet above).
- The **PRACH-specific** subcarrier spacings (**1.25 kHz** and **5 kHz**) are **not** included in the table, because most of the concepts shown in the table are non-applicable to them.

The following table presents the **slot duration**, that depends **both** on the **numerology** and on the **cyclic prefix**, expressed in **µs** and in multiples of "$T_c$":

| Freq. Range | Numerology "µ" | SCS ($15 \cdot 2^{\mu}$) [kHz] | Slot duration | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | 2 LCP symbols | | 1 LCP symbol | | 0 LCP symbols | | 12 ECP symbols | |
| | | | [$T_c$] | [µs] | [$T_c$] | [µs] | [$T_c$] | [µs] | [$T_c$] | [µs] |
| **FR1** | **0** | **15** | 1,966,080 | 1,000 | N/A | N/A | N/A | N/A | N/A | N/A |
| **FR1** | **1** | **30** | 984,064 | 500.5 | 983,040 | 500.0 | N/A | N/A | N/A | N/A |
| **FR1, FR2-1** | **2** | **60** | 493,056 | 250.8 | 492,032 | 250.3 | 491,008 | 249.7 | 491,520 | 250.0 |
| **FR2-1** | **3** | **120** | 247,552 | 125.9 | 246,528 | 125.4 | 245,504 | 124.9 | N/A | N/A |
| **FR2-2** | **5** | **480** | 63,424 | 32.26 | 62,400 | 31.74 | 61,376 | 31.22 | N/A | N/A |
| | **6** | **960** | 32,736 | 16.65 | 31,712 | 16.13 | 30,688 | 15.61 | N/A | N/A |

Some **relevant remarks** on this table are:

- With **numerologies 0** or **1** (i.e., **15** or **30 kHz subcarrier spacing**), **all slots** have the **same duration**. The reason is that **all slots** have the same **quantity** of **normal-long CP symbol intervals**. This **quantity** is **2** or **1** for **numerology 0** or **1** (respectively).
- With the **extended CP**, **all slots** have the **same duration**. However, this is **not** the case for the **normal CP** and **numerologies** larger **than 1** (**SCS > 30 kHz**).
- With **normal CP** and a **numerology** larger **than 1** (**SCS > 30 kHz**), the **first** and the **next-after-middle slot** of each **subframe** have a **slightly longer duration** than the **other slots**.
- The **average slot duration** is **inversely** proportional to the **subcarrier spacing**. Notice that the **subframe duration** is constant (independent of the **subcarrier spacing**) while the quantity of **symbol intervals** per **subframe** is **exactly proportional** to the **subcarrier spacing**.
- The **subframe duration** is **always 1 ms** (i.e., "1,966,080 $T_c$").
- There is **no row** for **numerology** "**µ=4**" (**SCS=240 kHz**) because it can **only** be used for the **SSB**. Therefore, it is **not possible** to have a **complete slot**, nor a **complete subframe**, with the **240 kHz subcarrier spacing**.
- The **PRACH-specific** subcarrier spacings (**1.25 kHz** and **5 kHz**) are **not** included in the table, because most of the concepts shown in the table are non-applicable to them.

Be aware that **subcarrier spacing** (click 2.2.10.3) and **average symbol interval duration** are **inversely proportional**, because of their relationship through **numerology** (click 2.1).

**Subcarrier spacing attributes** are summarized in a table at 2.2.10.3. Several **tables** presenting **multiple characteristics** (including **subcarrier spacings** and different **durations**) for **each** of the **different physical channels** and **signals** are available at 4.11.3.1.

For the configuration of **cyclic prefix** for **bandwidth parts**, **carriers** and **resource grids**, click 2.3.3.

In **LTE**, there is only **one subcarrier spacing** (with the exception of MBS services) and **one cyclic prefix** (i.e., **no long/short CPs**). Therefore, **all symbol intervals** and **all symbols** have the same duration. In **LTE**, the **symbol interval duration** is:

1/14 ms = 71.429 μs.

For more detail on **cyclic prefix duration**, refer to clause 5.3.1 of TS 38.211.

Source web pages:

- **LTE**: https://www.sharetechnote.com/html/FrameStructure_DL.html

### 2.3.3 Configuration of cyclic prefix for subcarriers, bandwidth parts and carriers

The **cyclic prefix** (click 2.3.2) is the same for **all** the **subcarriers** within **each bandwidth part** (click 2.2.7.1), and for **all** the **bandwidth parts** associated to the **same carrier** (click 2.2.5). The **UE** learns the **cyclic prefix** of **each bandwidth part** from **`cyclicPrefix`** from **`BWP`** (click 2.2.8.11). **Three instances** of **`cyclicPrefix`** are included within **`SIB1`** (click 5.3.4) to convey the respective **cyclic prefixes** of the **Downlink initial bandwidth part** (click 2.2.7.2), the **Uplink initial bandwidth part** (click 2.2.7.2) and the **SUL initial bandwidth part** (click 2.2.7.2). Other **instances** of **`cyclicPrefix`** are also included in **RRC messages** to configure the **cyclic prefix** of **each** of the **additional bandwidth parts** (click 2.2.8). Parameter **`cyclicPrefix`** is defined **only once** in the whole TS 38.331, within the specification of **data type** **`BWP`** (click 2.2.8.11). Therefore, **`cyclicPrefix`** does **not** have **any same-name parameter**. Notice though that in TS 38.331 there are other parameters related to **cyclic prefix** (e.g., **`ref-SCS-CP`**).

It has just been indicated that the **cyclic prefix** is configured for **each bandwidth part**, and **not** for **each carrier**. You may see this in the specification of **`SCS-specificCarrier`** (click 2.2.6.12). In spite of this, **all** the **bandwidth parts** associated to the **same carrier must** have the **same** value of **`cyclicPrefix`**. This **single value** of **`cyclicPrefix`** is considered to be the **cyclic prefix** of the **carrier** with that **same subcarrier spacing**, and also the **cyclic prefix** of the **resource grid** associated to that **carrier**. The **same** applies to the **single value** of **control/shared subcarrier spacing** for **all** the **bandwidth parts** associated to the **same carrier** (click 2.2.7.1.2).

### 2.3.4 Time division duplexing (TDD) in 5G

The **duplexing mode** (click 2.2.1.2) of **each band** indicates the **duplexing** (click 11.5) technique that **must** be used in **its channels** (click 2.2.2.3). **Time division duplexing bands** contain **channels** that use **time division duplexing**, which is explained in this section. **Paired spectrum bands** (**FDD**, **SDL** and **SUL bands**) contain **channels** using **frequency division duplexing** (**FDD**), which is explained in 2.2.12.

For more detail on **signaling** of **duplexing slot formats**, refer to clause 11.1 of TS 38.213.

For more detail on **time division duplexing**, refer to clause 4.3.2 of TS 38.211.

Source web page:

- https://www.sharetechnote.com/html/5G/5G_tdd_UL_DL_configurationCommon.html

### 2.3.4.1 Overview of TDD

**TDD** is done in terms of **individual symbol intervals**. **Each symbol interval** within **each slot** can be classified as **Downlink** (**D**), **Flexible** (**F**), or **Uplink** (**U**). **Downlink transmission only** occurs in **Downlink** or **Flexible symbol intervals**. **Uplink transmission only** occurs in **Uplink** or **Flexible symbol intervals**. Therefore, a **Flexible symbol interval** can be used for either **Uplink** or **Downlink** transmission as **per requirement**.

The **Flexible symbol intervals** may **also** be **set** to **zero** to implement the **required TDD guard period** for **every Downlink** to **Uplink transition**. The **Downlink** to **Uplink guard period** ranges from **1** to **13 Flexible symbol intervals**. The specific value of the **guard period** depends on the **cell reach**, **subcarrier spacing**, **band**, and other factors.

A **TDD slot format** indicates which **symbol intervals** in the **slot** are **Downlink**, **Uplink** or **Flexible**. This indicates what **symbol intervals** can be used to transmit **Downlink**, **Uplink** or **set** to **zero** for the **guard period**. It is possible to use **many different TDD slot formats** (click 2.3.4.2). **Consecutive slots** may have **different TDD slot formats**. In case the **UE** is **not** configured with **TDD slot formats**, it considers **all symbol intervals** to be **Flexible**.

Allowing to configure a **sequence** of **different slot formats** results in an **extremely flexible Downlink/Uplink scheduling** with the required **guard period** for the **cell**. The **configuration** of **TDD slot formats** has an impact on the relative **capacity split** between **Downlink** and **Uplink**, on the **latency**, and also on the **maximum cell reach**.

The **configuration** of **TDD slot formats** is detailed in 2.3.4.2.

If the **UE** is instructed to **receive SSB** in a **set** of **symbols**, this is obeyed in spite of the **slot format** (e.g., where **one or more symbols** in the set are marked as **Uplink**), and in spite of the **scheduling** using **DCI** (e.g., where the **UE** is instructed to transmit in **one or more** of the **symbols** in the **set**).

If the **UE** is instructed to **transmit PRACH** in a **set** of **symbols**, this is obeyed in spite of the **slot format** (e.g., where **one or more symbols** in the set are marked as **Downlink**), and in spite of the **scheduling** using **DCI** (e.g., where the **UE** is instructed to receive in **one or more** of the **symbols** in the **set**).

As it has already been indicated, in **5G** it is **allowed** (**unlike** in **LTE**) that the **TDD slot format** is **changed** from **one slot** to the **next** one, and it is also allowed that the **TDD slot format** is **different** in **neighbor NR nodes**. This can be done because **beamforming minimizes interference**, and because **5G** has a **more intelligent assignment** of **resource scheduling blocks** to **separate close-by UEs** in the **subcarriers** they use. The **joint usage** of these **two techniques** allows to **largely mitigate** the interference caused by **one UE transmitting** when a **nearby UE** is **receiving**.

Since **5G carriers** have a **very specific time framing** (click 2.3.1), **TDD** requires that the **Uplink carrier** and the **Downlink carrier** be **very precisely synchronized**, such that their signals **exactly interleave** in **time**, **resembling** a **single resulting TDD carrier**.

**All TDD bands** except **FR1-1 n50** have **symmetric channels**. For **asymmetric channel bandwidth** in **n50**, click 2.2.3.7.

Source web pages:

- https://howltestuffworks.blogspot.com/2019/11/5g-nr-time-domain-slots-and-slot-formats.html

- https://www.5gfundamental.com/2020/05/tdd-ul-dl-slot-configuration.html
- https://www.qualcomm.com/content/dam/qcomm-martech/dm-assets/documents/5G-NR-Release-15.pdf
- https://www.sharetechnote.com/html/5G/5G_tdd_UL_DL_configurationCommon.html
- https://www.sharetechnote.com/html/5G/5G_FrameStructure.html

### 2.3.4.2 TDD slot formats configuration

The **TDD slot formats** may be configured in a **static**, **semi-static** or **dynamic** mode.

**Static** and **semi-static slots formats** configuration are done with **RRC common** and/or **dedicated configuration**. **Common configuration** parameters are delivered within `SIB1` (click 5.3.4) or/and within **Non-SI parameters** (2.3.5.2). **Dedicated configuration** parameters are delivered within **Non-SI parameters** (2.3.5.2).

**Dynamic slot formats configuration** is done with **DCI**, **either** with **Slot Format Indicator** (**SFI**) **messages** and/or with **DCI scheduling messages**.

These **four TDD slot formats configuration procedures** are **prioritized**, such that the **symbol intervals** defined as **Downlink** or **Uplink** by a given **procedure cannot** be **modified** by a **lower priority procedure**. **Only Flexible symbol intervals** can be **modified** by a **lower priority procedure**. The **priorities** are (**higher** to **lower**):

1. **Common configuration** (click 2.3.4.2.1)
2. **Dedicated configuration** (click 2.3.4.2.2)
3. **DCI configuration**, either **SFI** (click 2.3.4.2.3) or **scheduling** (click 2.3.4.2.4) messages

If the **UE** has **not** been **configured** with **any TDD slot format**, it assumes that **all** the **symbol intervals** are **Flexible**.

For **small** or **isolated cells**, **dynamic** configuration is **usually preferred** to adapt to **traffic variations**, while for **medium cells**, **semi-static** configuration is **usually preferred** to **mitigate interference**.

As has been shown, **TDD** consists of **alternating periods** of **consecutive Downlink** or **Uplink symbol intervals**, with some interleaved **Flexible symbol intervals**. For **efficiency** reasons (click 11.5.311.5.2), **each** of these alternating **periods** may be **relatively long**, which is **unacceptable** for **low delay URRLC** services. This is why **TDD** includes the scheduling of **mini-slots** (click 4.11.4.6) to provide **low-delay transmissions**.

For more detail on **TDD slots format configuration**, refer to clause 11.1 of TS 38.213.

Source web page:

- https://howltestuffworks.blogspot.com/2019/11/5g-nr-time-domain-slots-and-slot-formats.html

#### 2.3.4.2.1 TDD slot formats common configuration

The **TDD slot formats common configuration** is done with a **TDD pattern** in `pattern1` and **possibly** with as a **second TDD pattern** in (**optional**) `pattern2`, **both** parameters from `TDD-UL-DL-ConfigCommon` (click 2.3.5.3). **Both parameters** have

**data type** `TDD-UL-DL-Pattern` (click 2.3.5.4). These **parameter(s)** are delivered within `SIB1` (click 5.3.4) or/and within **Non-SI parameters** (click 2.3.5.2), the latter contained in a `RRCReconfiguration` **message** (click 5.8.5.4). This configuration is **static** or **semi-static**.

This **TDD slot format definition** consists of the **cyclical repetition either** of `pattern1` **or** of the **concatenation** of `pattern1` and `pattern2` (if defined). **Each TDD pattern** `pattern1` or `pattern2` is a **TDD period** "$P$" or "$P_2$", that **contains** "$S$=$P$·$2^{\mu ref}$" or "$S_2$=$P_2$·$2^{\mu ref}$" **slot formats**, **starts** with "$d_{slots}$" or "$d_{slots,2}$" **all-Downlink slots**, followed by "$d_{syin}$" or "$d_{syin,2}$" **Downlink symbol intervals**, followed by "$f_{syin}$" or "$f_{syin,2}$" **Flexible symbol intervals**, followed by "$u_{syin}$" or "$u_{syin,2}$" **Uplink symbol intervals** and **ends** with "$u_{slots}$" or "$u_{slots,2}$" **all-Uplink slots** (respectively). Notice that "$\mu_{ref}$" is the **same** for **both** `pattern1` and `pattern2`.

The **quantities** "$f_{syin}$" and "$f_{syin,2}$" of **Flexible symbol intervals** are obtained as:

$$f_{syin} = P \cdot 2^{\mu ref} - N_{syin}^{slot} (d_{slots} + u_{slots}) - d_{syin} - u_{syin}$$

$$f_{syin,2} = P_2 \cdot 2^{\mu ref} - N_{syin}^{slot} (d_{slots,2} + u_{slots,2}) - d_{syin,2} - u_{syin,2}$$

where $N_{syin}^{slot}$ is the **quantity** of **symbol intervals** in one **slot format**: **14** if **normal CP** and **12** if **extended CP**.

The **cyclical repetition** of the **one** or **two TDD pattern(s) must** be **synchronized** with **frames**, such that the **first symbol interval** of `pattern1` is also the **first symbol interval** of an **even frame**. If **only** `pattern1` is configured, its **first symbol interval** is the **first symbol interval** of an **even frame every** "**20/$P$**" **TDD periods**, and if `pattern2` is also configured it is **every** "**20/($P$+$P_2$)**" **TDD periods**. For the case of `pattern2` being **configured**, the **UE** expects that **20 ms** (i.e., the **period** of **even frames**) is an **exact multiple** of "$P$+$P_2$".

The **UE expects** that "$\mu_{ref} \leq \mu$", where "$\mu$" is the **numerology** of **any Downlink** or **Uplink bandwidth part** configured to the **UE** in the **cell**.

**Every** $2^{\mu-\mu ref}$ **consecutive symbol intervals** of the **Downlink** or **Uplink active bandwidth part** with **numerology** "$\mu$" have the **same duplexing type** (**Downlink**, **Uplink** or **Flexible**) as its **simultaneous symbol interval** with **numerology** "$\mu_{ref}$". Notice that $2^{\mu-\mu ref}$ **consecutive symbol intervals** with **numerology** "$\mu$" have the **same duration** as **1 symbol interval** with **numerology** "$\mu_{ref}$".

The **TDD period** "$P$" is configured to the **UE** in `dl-UL-TransmissionPeriodicity` from `TDD-UL-DL-Pattern` (click 2.3.5.4). Its value may be `0.5`, `0.625`, `1`, `1.25`, `2`, `2.5`, `3`, `4`, `5` or `10` milliseconds.

The **reference numerology** "$\mu_{ref}$" is configured to the **UE** in `referenceSubcarrierSpacing` from `TDD-UL-DL-ConfigCommon` (click 2.3.5.3). Its value may be `0`, `1`, `2`, `3`, `5`, or `6`.

**Not all** the value **combinations** between "$P$" and "$\mu_{ref}$" are allowed (see the table below).

The values of "$d_{slots}$", "$d_{syin}$", "$u_{slots}$" and "$u_{syin}$" are configured to the **UE** in `nrofDownlinkSlots`, `nrofDownlinkSymbols`, `nrofUplinkSlots` and `nrofUplinkSymbols` (respectively), **all** from `TDD-UL-DL-Pattern` (click 2.3.5.4). Their values may be {`0..320`} for the **slot** parameters and {`0..13`} for the **symbol** parameters.

The instances of `TDD-UL-DL-ConfigCommon` and `TDD-UL-DL-Pattern` are delivered within **either** `SIB1` (click 5.3.4) or **Non-SI parameters** (click 5.3.6).

The following table shows the **allowed combinations** between the **TDD period** values ("***P***" and optionally also "$P_2$"), and the "$\mu_{ref}$" values, with the resulting **quantity** of **slots** within a **TDD period**:

| TDD period [ms] | Slots in a TDD period | | | | | | 20/*P* [periods] |
|---|---|---|---|---|---|---|---|
| | $\mu_{ref}$ | | | | | | |
| | 0 | 1 | 2 | 3 | 5 | 6 | |
| **0.5** | 0.5 | 1 | 2 | 4 | 16 | 32 | 40 |
| **0.625** | | | | 5 | 20 | 40 | 32 |
| **1,25** | | | 5 | 10 | 40 | 80 | 16 |
| **2** | 2 | 4 | 8 | 16 | 64 | 128 | 10 |
| **2.5** | | 5 | 10 | 20 | 80 | 160 | 8 |
| **3** | 3 | 6 | 12 | 24 | 96 | 192 | 6.67 |
| **4** | 4 | 8 | 16 | 32 | 128 | 256 | 5 |
| **5** | 5 | 10 | 20 | 40 | 160 | 320 | 4 |
| **10** | 10 | 20 | 40 | 80 | 320 | | 2 |

Some **relevant remarks** on this table are:

- The **blank cells** correspond to **non-allowed combinations** of "***P***" and "$\mu_{ref}$".
- The **TDD period** values `3` and `4` correspond to `dl-UL-TransmissionPeriodicity-v1530` from `TDD-UL-DL-Pattern` (click 2.3.5.4).
- The **quantity** of **slot formats** in a **TDD period** is **always** an **integer**, **except** for "**TDD period** =`.5`" and "$\mu_{ref}$=`0`".
- For the case of **only** `pattern1`, the **quantity** of **TDD periods** required for the **first symbol interval** of `pattern1` to be also the **first symbol interval** of an **even frame** is **always** an **integer**, **except** for "***P***=`3`".

#### 2.3.4.2.2 TDD slot formats dedicated configuration

The **TDD slot formats configuration** is done with a `SEQUENCE OF` **TDD slot format definitions**. **Each** such **TDD slot format definition** in the said **dedicated configuration** corresponds to **one TDD slot format definition** from the **TDD slot formats common configuration** (click 2.3.4.2.1). **Each TDD slot format definition** from the **dedicated configuration** turns **some** of the **Flexible symbol intervals** of its **corresponding TDD slot format** definition into **either Downlink** or **Uplink symbol intervals**. The **UE** does **not expect** that the **TDD slot formats dedicated configuration attempts** to **modify** a **Downlink** or **Uplink symbol interval** from the **TDD slot format common configuration**.

The **UE dedicated configuration** of the **TDD slot formats** is **activated** with `slotSpecificConfigurationsToAddModList` and **released** with `slotSpecificConfigurationsToReleaseList`, both from `TDD-UL-DL-ConfigDedicated` (click 2.3.5.5), within the **Non-SI parameters** (click 2.3.5.2), and delivered in a `RRCReconfiguration` **message** (click 5.8.5.4). This configuration is

**static** or **semi-static**. The **data type** of the **former** parameter is a **SEQUENCE OF** TDD-UL-DL-SlotConfig (click 2.3.5.6), while the **data type** of the **latter** is a **SEQUENCE OF** TDD-UL-DL-SlotIndex (click 2.3.5.5). The **range** of **both SEQUENCE OF** is (1..maxNrofSlots), where maxNrofSlots is 320, which is the **maximum quantity** of **slots** in a **TDD period** (see above).

**Each** of the **instances** of TDD-UL-DL-SlotConfig (click 2.3.5.6) allows to define **each symbol interval** within the **slot** as **either Downlink** or **Uplink** (or remain **Flexible**).

#### 2.3.4.2.3 TDD slot formats DCI SFI configuration

**TDD slot formats DCI SFI configuration** is done with **DCI Format 2_0 messages** (click 6.9.8) masked with a **group SFI-RNTI** (click 6.9.10). **Each DCI Format 2_0 message** masked with a **specific group SFI-RNTI** contains a **list** of **slot format combination identifiers**, where **each** such **identifier** indicates the **slot format combination** to be used in **one** of the **UE**'s **serving cells**. Recall that a **DCI message** is received by **all** the **UEs** that have been configured with the **specific group SFI-RNTI**. There may be **several UE groups**, where **all** the **UEs** in **each group** are **configured** with a **specific SFI-RNTI** value.

The association between **each TDD slot format combination identifier** in the **payload** and the **UE**'s **serving cell** to which it corresponds (sevingCellId) is configured to **each UE** with positionInDCI. Both parameters are from SlotFormatCombinationsPerCell, click 2.3.5.8.

The **maximum quantity** of **TDD slot format combination identifiers** in **one DCI Format 2_0 message** is in the order of **14**, because the **maximum length** of the **payload** is 128 **bits** (dci-PayloadSize from SlotFormatIndicator, click 2.3.5.7) and **each identifier** may require **9 bits**. The **maximum quantity** of **TDD slot format combinations** in **each TDD slot format combination set** (slotFormatCombinations from SlotFormatCombinationsPerCell, click 2.3.5.8) is 512, hence the need of **up to 9 bits** to identify **each TDD slot format combination**.

**Each TDD slot format combination** (slotFormats from SlotFormatCombination, click 2.3.5.9) is a **sequence** of up to 256 **TDD slot formats**. This **sequence** of **TDD slot formats** is **cyclically repeated**.

**Each TDD slot format** indicates what **symbol intervals**, within a **slot**, are **either Downlink**, **Uplink** or **Flexible**.

The **allowed TDD slot formats** with **normal CP** are defined in table 11.1.1-1 from TS 38.213, which is:

| TDD slot format index | Symbol index within a slot | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
| 0 | D | D | D | D | D | D | D | D | D | D | D | D | D | D |
| 1 | U | U | U | U | U | U | U | U | U | U | U | U | U | U |
| 2 | F | F | F | F | F | F | F | F | F | F | F | F | F | F |
| 3 | D | D | D | D | D | D | D | D | D | D | D | D | D | F |
| 4 | D | D | D | D | D | D | D | D | D | D | D | D | F | F |
| 5 | D | D | D | D | D | D | D | D | D | D | D | F | F | F |
| 6 | D | D | D | D | D | D | D | D | D | D | F | F | F | F |
| 7 | D | D | D | D | D | D | D | D | D | F | F | F | F | F |
| 8 | F | F | F | F | F | F | F | F | F | F | F | F | F | U |
| 9 | F | F | F | F | F | F | F | F | F | F | F | F | U | U |
| 10 | F | U | U | U | U | U | U | U | U | U | U | U | U | U |

| TDD slot format index | Symbol index within a slot | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
| 11 | F | F | U | U | U | U | U | U | U | U | U | U | U | U |
| 12 | F | F | F | U | U | U | U | U | U | U | U | U | U | U |
| 13 | F | F | F | F | U | U | U | U | U | U | U | U | U | U |
| 14 | F | F | F | F | F | U | U | U | U | U | U | U | U | U |
| 15 | F | F | F | F | F | F | U | U | U | U | U | U | U | U |
| 16 | D | F | F | F | F | F | F | F | F | F | F | F | F | F |
| 17 | D | D | F | F | F | F | F | F | F | F | F | F | F | F |
| 18 | D | D | D | F | F | F | F | F | F | F | F | F | F | F |
| 19 | D | F | F | F | F | F | F | F | F | F | F | F | F | U |
| 20 | D | D | F | F | F | F | F | F | F | F | F | F | F | U |
| 21 | D | D | D | F | F | F | F | F | F | F | F | F | F | U |
| 22 | D | F | F | F | F | F | F | F | F | F | F | F | U | U |
| 23 | D | D | F | F | F | F | F | F | F | F | F | F | U | U |
| 24 | D | D | D | F | F | F | F | F | F | F | F | F | U | U |
| 25 | D | F | F | F | F | F | F | F | F | F | F | U | U | U |
| 26 | D | D | F | F | F | F | F | F | F | F | F | U | U | U |
| 27 | D | D | D | F | F | F | F | F | F | F | F | U | U | U |
| 28 | D | D | D | D | D | D | D | D | D | D | D | D | F | U |
| 29 | D | D | D | D | D | D | D | D | D | D | D | F | F | U |
| 30 | D | D | D | D | D | D | D | D | D | D | F | F | F | U |
| 31 | D | D | D | D | D | D | D | D | D | D | D | F | U | U |
| 32 | D | D | D | D | D | D | D | D | D | D | F | F | U | U |
| 33 | D | D | D | D | D | D | D | D | D | F | F | F | U | U |
| 34 | D | F | U | U | U | U | U | U | U | U | U | U | U | U |
| 35 | D | D | F | U | U | U | U | U | U | U | U | U | U | U |
| 36 | D | D | D | F | U | U | U | U | U | U | U | U | U | U |
| 37 | D | F | F | U | U | U | U | U | U | U | U | U | U | U |
| 38 | D | D | F | F | U | U | U | U | U | U | U | U | U | U |
| 39 | D | D | D | F | F | U | U | U | U | U | U | U | U | U |
| 40 | D | F | F | F | U | U | U | U | U | U | U | U | U | U |
| 41 | D | D | F | F | F | U | U | U | U | U | U | U | U | U |
| 42 | D | D | D | F | F | F | U | U | U | U | U | U | U | U |
| 43 | D | D | D | D | D | D | D | D | D | F | F | F | F | U |
| 44 | D | D | D | D | D | D | F | F | F | F | F | F | U | U |
| 45 | D | D | D | D | D | D | F | F | U | U | U | U | U | U |
| 46 | D | D | D | D | D | F | U | D | D | D | D | D | F | U |
| 47 | D | D | F | U | U | U | U | D | D | F | U | U | U | U |
| 48 | D | F | U | U | U | U | U | D | F | U | U | U | U | U |
| 49 | D | D | D | D | F | F | U | D | D | D | D | F | F | U |
| 50 | D | D | F | F | U | U | U | D | D | F | F | U | U | U |
| 51 | D | F | F | U | U | U | U | D | F | F | U | U | U | U |
| 52 | D | F | F | F | F | F | U | D | F | F | F | F | F | U |
| 53 | D | D | F | F | F | F | U | D | D | F | F | F | F | U |
| 54 | F | F | F | F | F | F | F | D | D | D | D | D | D | D |
| 55 | D | D | F | F | F | U | U | U | D | D | D | D | D | D |
| 56 – 254 | Reserved | | | | | | | | | | | | | |
| 255 | The **UE** determines the **TDD slot format** based on **either** `tdd-UL-DL-ConfigurationCommon` from `ServingCellConfigCommon` (click 3.9.6) or `tdd-UL-DL-ConfigurationDedicated` from `ServingCellConfig` (click 3.9.7), and, if any, on detected **DCI Formats** (click 6.9) other than **Format 2_0**. | | | | | | | | | | | | | |

Some **relevant remarks** on this table are:

- **Every TDD slot format** is **identified** with an **index** from **0** to **255**.

- **Every TDD slot format** (except **0**) has **at least one Flexible symbol interval** of **guard period** for **each Downlink** to **Uplink transition**.
- **Every TDD slot format** (except **0**) has **at least one Flexible symbol interval** of **guard period** for **each Downlink** to **Uplink transition**.
- A **slot** with **Format 1 cannot follow** a **slot** with **Format 0**, because in that case there would be no **Downlink** to **Uplink guard period**.

The **allowed TDD slot formats** with **extended CP** are obtained from the ones with **normal CP**. **Each TDD slot format** with **normal CP** produces **one TDD slot format** with **extended CP** with its **same index**. The **UE** determines an **extended CP slot symbol** to be:

- **Downlink** or **Uplink** if **all** the **overlapping normal CP slot symbols** are **Downlink** or **Uplink** (respectively).
- **Flexible** if **one** of the **overlapping normal CP slot symbols** is **Flexible**, or, if **one** of them is **Downlink** and the **other** one is **Uplink**.

This procedure is mainly intended for **UEs** that are **not currently** scheduled; because the **UEs currently scheduled** would receive the **Downlink/Uplink symbol interval** configuration via **DCI scheduling** (click 2.3.5.2.).

The **UE** does **not expect** that the **DCI SFI slot formats configuration attempts** to **modify** a **Downlink** or **Uplink symbol interval** from the **TDD slot format common or dedicated configuration**.

#### 2.3.4.2.4 TDD slot formats DCI scheduling configuration

**TDD slot formats DCI scheduling configuration** consists of using the **conventional scheduling** (click 11.8) mechanism to **allocate Downlink** and **Uplink symbol intervals** to the **UEs**.

If the **UE** is **not configured** with **slot format indicator** (click 2.3.4.2.3) and for the **Flexible symbol intervals** indicated by **common configuration** (click 2.3.4.2.1) and/or by **dedicated configuration** (click 2.3.4.2.2):

- The **UE receives PDSCH** or **CSI-RS** in the **symbol intervals** indicated by a **DCI Format 1_0**, **1_1** or **0_1**.
- The **UE transmits PUSCH**, **PUCCH**, **PRACH**, or **SRS** in the **symbol intervals** indicated by a **DCI Format 0_0**, **0_1**, **1_0**, **1_1** or **DCI Format 2_3**.

For more detail on **DCI Formats**, click 6.9.

### 2.3.4.3 TDD guard period

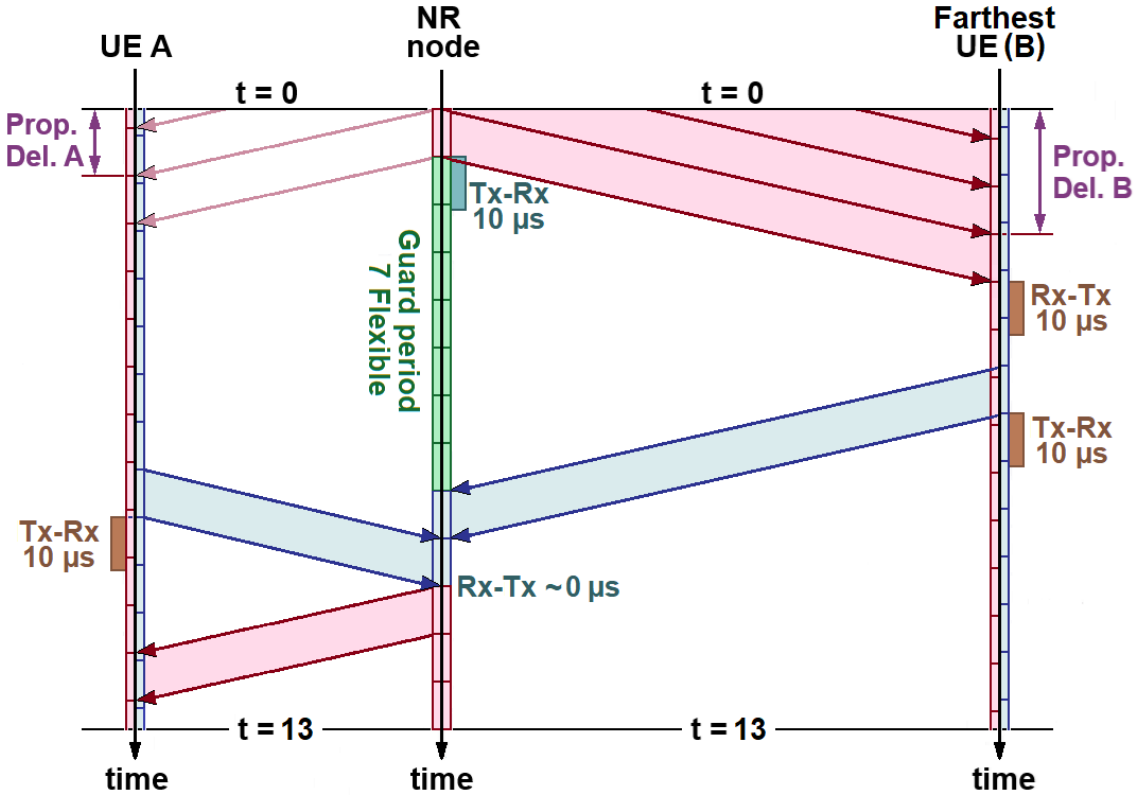
The **TDD guard period** is the **time** that the **NR node must wait** from the **end** of its **last** transmitted **Downlink symbol interval** until it can **receive** the **first Uplink symbol**. The **TDD guard period** is the **time** that the **NR node must wait** in **every Downlink** to **Uplink transition**.

The **TDD guard period must** be **at least twice** the **propagation delay** from the **cell**'s **antenna site** to its **farthest** active **UE**, **plus** the **UE Rx to Tx transceiver switch time**. The **TDD guard period must** be an **integer quantity** of **symbol intervals**. The **TDD guard period** is **required** because the **farthest UE must wait** until having received the **last Downlink symbol before** starting its **Uplink transmission**: otherwise, the **UE self-**

**interference** (click 12.1.14) would **kill** its **Downlink reception** during the **overlapping time**.

In the **Uplink** to **Downlink** transitions there is **no guard period**.

The following figure illustrates the **TDD guard period**:




Some **relevant remarks** on this figure are:

- **Time advances up to down**. The **top** is the **past** and the **bottom** is the **future**.
- **Each vertical rectangle** represents **one symbol interval**: **Downlink** are **colored** in **red**, **Uplink** in **blue** and **Flexible** in **green**.
- The **UE Rx to Tx transceiver switch time** is **smaller** than **one symbol interval**, but the **farthest UE must wait** a **complete symbol interval** to transmit, to comply with the **carrier timing structure**.
- The **TDD guard period** in this example is **two propagation delays** from the **cell**'s **antenna site** to its **farthest** active **UE**, **plus** the **UE Rx to Tx transceiver switch time**, **rounded up** to an **integer quantity** of **symbol intervals**. This results in **7 symbol intervals**, which is the **minimum** possible **TDD guard period**.
- The **TDD guard period** is an **integer quantity** of **symbol intervals**, as it **must**.
- The **NR node transceiver switch times** are colored in **green**, while the ones of the **UEs** are colored in **brown**.
- The **Rx to Tx transceiver switch time** at the **NR node** is **negligible**.
- The **TDD carrier structure** is **referred** to the **NR node**, where it conforms to the **TDD** specification.

- The **carrier structure** at the **UEs** do **not conform** to the **TDD carrier structure** because of the **timing advance offset** (click 2.3.6.1).
- **Solid colored areas** between **diagonal arrows** represent the intended **destination** of the information, while **white areas** represent **signal propagation** to **non-intended** destination.
- **Neither** the **NR node**, **nor any UE**, may **simultaneously transmit and receive**, because this is a **TDD band.**
- In this example, the **propagation delays** of **UE A** and **UE B** are **1.4 symbol intervals** and **2.6 symbol intervals** (respectively). Assuming **numerology 3** (**SCS=120kHz**) and **NCP**, this is roughly a **distance** to the **NR node** of **3.75 km** and **7 km** (respectively). The guard period is **7 symbol intervals**, which is **62.5 µs**.
- The **Downlink** and the **Uplink numerology** is the **same**, as it **must** be in a **TDD band**.
- The figure is drawn to **scale**, where **each vertical pixel** is **0.198 µs**.

The **Uplink to Downlink guard period** is **not zero**, and it is the **NR node Rx to Tx transceiver switch time**, which is equal to "$N_{TA,offset} \bullet T_c$" (click 2.3.6.1).

The **maximum** allowed **over the air transient times** for **base stations type 1-O** and **2-O** are listed in the following table (from tables 9.5.3.2-**1** and **-2** of TS 38.104):

| Transient time [µs] | | |
|---|---|---|
| **Transition** | **Base station type** | |
| | **1-O** | **2-O** |
| **Off to On** | 10 | 3 |
| **On to Off** | 10 | 3 |

The **cell phase synchronization accuracy** measured at **base station** antenna connectors shall be better than **3 µs** (refer to clause 7.4.2 of TS 38.133).

For a general explanation of the **TDD guard period**, click 11.5.2.

Source web pages:

- https://howltestuffworks.blogspot.com/2019/11/5g-nr-time-domain-slots-and-slot-formats.html
- https://www.sharetechnote.com/html/5G/5G_tdd_UL_DL_configurationCommon.html

### 2.3.4.4 Implications of the TDD guard period

The **TDD guard period** causes a **trade-off** between **bitrate** and **latency**.

A **deterministic model** for the **raw bitrate** at the **physical layer** "$BR_{UE}^{Phy}$" may be:

$$BR_{UE}^{Phy} = T_{Tx}^{ULDL} / (T_{Tx}^{ULDL} + GP)$$

where "$T_{Tx}^{ULDL}$"=$T_{Tx}^{UL}$+ $T_{Tx}^{DL}$" is the **duration** of **each** pair of **Uplink** and **Downlink** consecutive **transmissions** and the "$GP$" is the **Downlink** to **Uplink guard period**. This **deterministic model assumes** that the values of "$T_{Tx}^{UL}$" and "$T_{Tx}^{DL}$" are **constant**.

A **deterministic model** for the **one-way Uplink latency** in **TDD** "$L_{TDD}^{UL}$" may be modeled as:

$$L_{TDD}^{UL} = L_{FDD} + GP + T_{Tx}^{DL}/2$$

where "$L_{FDD}$" is the **latency** in **FDD** and "$T_{Tx}^{DL}$" is the **constant duration** of **each Downlink transmission**.

**Downlink traffic** is **most frequently larger** than **Uplink traffic**. Assuming that "$T_{Tx}^{DL}=7T_{Tx}^{UL}$", the result is:

$$BR_{UE}^{Phy} = [(8/7)T_{Tx}^{DL}] / [(8/7)T_{Tx}^{DL} + GP]$$

$$L_{TDD}^{UL} = L_{FDD} + GP + T_{Tx}^{DL}/2$$

As you may see: a) the **raw bitrate** at the **physical layer** "$BR_{UE}^{Phy}$" is **inversely proportional** to the **guard period** "$GP$"; b) the **one-way Uplink latency** "$L_{TDD}^{UL}$" is directly proportional to the **guard period** "$GP$". This shows the **TDD trade-off** between **bitrate** and **latency**.

Combining **both equations**:

$$BR_{UE}^{Phy} = [(16/7)(L_{TDD}^{UL} - L_{FDD} - GP)] / [(16/7)(L_{TDD}^{UL} - L_{FDD} - (9/16)GP)]$$

we get a **mathematical formulation** of the **TDD trade-off** between **bitrate** "$BR_{UE}^{Phy}$" and **latency** "$L_{TDD}^{UL}$".

### 2.3.4.5 Half-duplex UEs

Some **reduced capability UEs** (click 4.9.1) can **only** operate in **half-duplex**, even in **FDD** or **SUL bands**. These **UEs** therefore operate in a **similar mode** to **TDD**.

A **UE not** capable of **full-duplex** communication and **not** supporting **simultaneous transmission** and **reception** as defined by either `simultaneousRxTxInterBandEND` from `MRDC-Parameters`, `simultaneousRxTxInterBandCA` from `CA-ParametersNR` or `simultaneousRxTxSUL` from `CA-ParametersNR` (for these parameters, refer to TS 38.306) among **all cells** within a **cell group** is **not** expected to:

- **Transmit Uplink** in **one cell** within the **cell group earlier** than "$N_{Rx\text{-}Tx}T_c$" **after** the **end** of the **last received Downlink symbol** in the **same** or in a **different cell** within the **cell group**.
- **Receive Downlink** in **one cell** within the **cell group earlier** than "$N_{Tx\text{-}Rx}T_c$" **after** the **end** of the **last transmitted Uplink symbol** in the **same** or **different cell** within the **cell group**.

For **DAPS handover** operation, a **UE not** capable of **full-duplex** communication is **not** expected to:

- **Transmit Uplink** to a **cell earlier** than "$N_{Rx\text{-}Tx}T_c$" **after** the **end** of the **last received Downlink symbol** in the **different cell**.
- **Receive Downlink** from a **cell earlier** than "$N_{Tx\text{-}Rx}T_c$" **after** the **end** of the **last transmitted Uplink symbol** in the **different cell**.

A **UE not** capable of **full-duplex** communication is **not** expected to:

- **Transmit Uplink earlier** than "$N_{Rx\text{-}Tx}T_c$" **after** the **end** of the **last received Downlink symbol** in the **same cell**.

- **Receive Downlink earlier** than "$N_{Tx\text{-}Rx}T_c$" **after** the **end** of the **last transmitted Uplink symbol** in the **same cell**.

The values of the **UE transceiver switch times** "$N_{Tx\text{-}Rx}$" and "$N_{Rx\text{-}Tx}$" are listed in table 4.3.2-3 of TS 38.211:

| | FR1 | | FR2 | |
|---|---|---|---|---|
| **Transition time** | **[$T_c$]** | **[µs]** | **[$T_c$]** | **[µs]** |
| $N_{Tx\text{-}Rx}$ | 25,600 | 13 | 13,792 | 7 |
| $N_{Rx\text{-}Tx}$ | 25,600 | 13 | 13,792 | 7 |

For more detail, refer to clause 4.3.2 of TS 38.211.

Source web page:

- https://www.sharetechnote.com/html/5G/5G_tdd_UL_DL_configurationCommon.html

### 2.3.4.6 TDD in LTE

In **LTE**, **TDD** is done in terms of **subframes** (**1 ms**). The **Downlink** and **Uplink** carrier transmission intervals consist of **one or more subframes**. The **TDD cycle time** consists of either **10 ms** or **5 ms**. The transmission **transitions** are **not always aligned** with **frames**, nor with **slots** (see the table below). **Downlink** and **Uplink subframes** are always **consecutive** within **each frame**, and the **first subframe** is **always Downlink**.

Each **Downlink-to-Uplink transition** requires **one** intermediate **special subframe** (named "**S**" in the table below) that cannot be used for data. This special subframe is used to send **Downlink** synchronization signals (**DwPTS**), a Guard Period (**GP**) and **Uplink** synchronization signals (**UpPTS**). The duration of **each** of these **three** "**S**" **subframe elements** depends on various parameters (cell size, ...). There is a table in the TS documents detailing the different values of the **three elements** depending on the parameters. As an example, with **duplexing frame configuration 0** there are approximately **10 symbol intervals** used for **GP** and this allows a **very large cell** with a theoretical **reach** of up to **100 km**.

The TS documents define **7 duplexing frame configurations** (numbered **0** to **6**). **Configurations 3**, **4** and **5** follow a **10 ms cycle**, while **configurations 0**, **1**, **2** and **6** follow a **5 ms cycle** (**2 cycles per frame**). The following table (table 4.2-2 of TS 36.211 V17.3.0) shows these **7 configurations**, where "**D**", "**U**" and "**S**" stand (respectively) for **Downlink**, **Uplink** and **Special**,.

| **UL-DL Config.** | **DL-to-UL Switch-point periodicity** | **Subframe number** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** |
| **0** | 5 ms | D | S | U | U | U | D | S | U | U | U |
| **1** | 5 ms | D | S | U | U | D | D | S | U | U | D |
| **2** | 5 ms | D | S | U | D | D | D | S | U | D | D |
| **3** | 10 ms | D | S | U | U | U | D | D | D | D | D |
| **4** | 10 ms | D | S | U | U | D | D | D | D | D | D |
| **5** | 10 ms | D | S | U | D | D | D | D | D | D | D |
| **6** | 5 ms | D | S | U | U | U | D | S | U | U | D |

Notice that the **quantity** of **Uplink subframes** in respect to the **quantity** of **Downlink subframes**, determines how the **total available capacity** (channel time) is allocated between **Downlink** and **Uplink** carriers. This allocation can be **strongly asymmetrical** (from **8DL/1UL** to **2DL/6UL**).

Notice also that **subframes 0** and **5** are **always Downlink**.

To avoid interference between neighboring **eNB**s, the **duplexing frame configuration must be the same** in **all neighboring** (same zone) **eNB**s. This prevents the problem of a **UE** that is transmitting to **its eNB** and is interfering with a **close-by UE** that is receiving from **another eNB**. By using the **same duplexing frame configuration** in neighboring **eNB**s, **all UEs** and **eNB**s **transmit and receive** at the same time, preventing the said interference problem. However, this configuration restriction severely limits the flexibility of resource scheduling that can be achieved by **TDD**, and also, implies that the changes in duplexing frame configuration have to be done with a large time granularity (several hours).

The information on the frame configuration to be used is sent through **PDCCH**.

For more detail on **TDD** in **LTE**, refer to clause 4 of TS 36.211.

### 2.3.5 Time division duplexing (TDD) parameters

The **UE**'s **time division duplexing** (**TDD**) (click 2.3.4) is configured with:

- **Common parameters** that are **broadcast** (or **unicast**) within `SIB1`, as described in 2.3.5.1.
- **Common** and **dedicated parameters** that are **unicast** within the **Non-SI parameter group**, as described in 2.3.5.2.

**TDD** configuration does **not** have **any parameter** within `MIB` (click 5.3.3).

Possibly existing **TDD parameters** within **other SIBs** and **posSIBs** (click 5.3.5) are **not addressed** in this book.

For an overall explanation on **UE configuration parameters** click 5. For a brief overview of **ASN.1**, click 5.9.

#### 2.3.5.1 Time division duplexing (TDD) parameters within `SIB1`

For **time division duplexing** (**TDD**) parameters within `SIB1` (click 5.3.4), the **most relevant related data types** are:

- `TDD-UL-DL-ConfigCommon` (click 2.3.5.3): common parameters
- `TDD-UL-DL-Pattern` (click 2.3.5.4): common parameters

The **ancestor tree** (click 5.9.8) of these **most relevant related data types** up to `SIB1` is:

```
                                    SIB1 (click 5.7.4)
servingCellConfigCommon               ServingCellConfigCommonSIB (0,1) (click 3.9.3)
  tdd-UL-DL-ConfigurationCommon       TDD-UL-DL-ConfigCommon (0,1) (click 2.3.5.3)
    pattern1                            TDD-UL-DL-Pattern (click 2.3.5.4)
    pattern2                            TDD-UL-DL-Pattern (0,1) (click 2.3.5.4)
```

Some **relevant remarks** on this **ancestor tree** are:

- For the **ancestor trees** of `SIB1` **up to** the **top**, click 5.3.4.

- **Each cell** using **TDD** has **only one TDD configuration**, and it applies to the **Downlink**, **Uplink** and **SUL** (if exists) **carriers**.

### 2.3.5.2 Time division duplexing (TDD) parameters within the Non-SI parameter group

For **time division duplexing** (**TDD**) parameters within the **Non-SI parameter group** (click 5.3.6), the **most relevant related data types** are:

- `TDD-UL-DL-ConfigCommon` (click 2.3.5.3): common parameters
- `TDD-UL-DL-Pattern` (click 2.3.5.4): common parameters
- `TDD-UL-DL-ConfigDedicated` (click 2.3.5.5): dedicated parameters
- `TDD-UL-DL-SlotConfig` (click 2.3.5.6): dedicated parameters
- `SlotFormatIndicator` (click 2.3.5.7): dedicated parameters

The **ancestor tree** (click 5.9.8) of these **most related data types** up to `CellGroupConfig` is:

```
                                    CellGroupConfig (click 3.7.3)
  spCellConfig                        SpCellConfig (click 3.9.5)
    reconfigurationWithSync             ReconfigurationWithSync (0,1) (click 3.9.8)
      spCellConfigCommon                  ServingCellConfigCommon (click 3.9.6)
        tdd-UL-DL-ConfigurationCommon       TDD-UL-DL-ConfigCommon (0,1)
                                                                    (click 2.3.5.3)
          pattern1                            TDD-UL-DL-Pattern (click 2.3.5.4)
          pattern2                            TDD-UL-DL-Pattern (0,1) (click 2.3.5.4)
      spCellConfigDedicated               ServingCellConfig (click 3.9.7)
        tdd-UL-DL-ConfigurationDedicated  TDD-UL-DL-ConfigDedicated (0,1)
                                                                    (click 2.3.5.5)
          slotSpecificConfigurationsToAddModList
                                              TDD-UL-DL-SlotConfig
                                              (1..maxNrofSlots) (click 2.3.5.5)
          slotSpecificConfigurationsToReleaseList
                                              TDD-UL-DL-SlotIndex
                                              (1..maxNrofSlots) (click 2.3.5.5)
        begin of DownlinkConfig parameter set (click 2.2.8.3)
        pdcch-ServingCellConfig           PDCCH-ServingCellConfig (0,1)
                                                                    (click 6.12.10)
          slotFormatIndicator               SlotFormatIndicator (0,1) (click 2.3.5.7)
        end of DownlinkConfig parameter set
  sCellToAddModList                 SCellConfig (0..maxNrofSCells) (click 3.9.4)
      sCellConfigCommon                 ServingCellConfigCommon (0,1) (click 3.9.6)
        tdd-UL-DL-ConfigurationCommon     TDD-UL-DL-ConfigCommon (0,1)
                                                                    (click 2.3.5.3)
          pattern1                          TDD-UL-DL-Pattern (click 2.3.5.4)
          pattern2                          TDD-UL-DL-Pattern (0,1) (click 2.3.5.4)
      sCellConfigDedicated              ServingCellConfig (click 3.9.7)
        tdd-UL-DL-ConfigurationDedicated  TDD-UL-DL-ConfigDedicated (0,1)
                                                                    (click 2.3.5.5)
          slotSpecificConfigurationsToAddModList
                                              TDD-UL-DL-SlotConfig
                                              (1..maxNrofSlots) (click 2.3.5.5)
          slotSpecificConfigurationsToReleaseList
                                              TDD-UL-DL-SlotIndex
                                              (1..maxNrofSlots) (click 2.3.5.5)
        begin of DownlinkConfig parameter set (click 2.2.8.3)
        pdcch-ServingCellConfig           PDCCH-ServingCellConfig (0,1)
                                                                    (click 6.12.10)
          slotFormatIndicator               SlotFormatIndicator (0,1) (click 2.3.5.7)
        end of DownlinkConfig parameter set
```

Some **relevant remarks** on this **ancestor tree** are:

- For the **ancestor trees** of `CellGroupConfig`, **up to** some **RRC NR messages** click 3.7.1 and **up to** some **RRC inter-node messages** click 3.7.2.
- **Each cell** using **TDD** has **only one TDD configuration**, and it applies to the **Downlink**, **Uplink** and **SUL** (if exists) **carriers**.
- The constant `maxNrofSCells` is `31`, which is the **maximum quantity** of **UE**'s **secondary cells** (i.e., **other** than the **special cells**) in **each cell group**
- The constant `maxNrofSlots` is `320`, which is the **maximum quantity** of **slots** within **any TDD period** (refer to the end of 2.3.5.4.1).
- The constant `maxNrofSCells` is `31`, which is the **maximum quantity** of **secondary cells** in **each** of the **UE**'s **cell groups**.

#### 2.3.5.3 `TDD-UL-DL-ConfigCommon` data type

It can be **both broadcast** (or **unicast**) within `SIB1` (click 5.3.4) and **unicast** within the **Non-SI parameter group** (click 5.3.6)

For its **ancestor tree up to** `SIB1`, click 2.3.5.1.

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 2.3.5.2.

**All** its parameters are **common** (click 5.1.3).

**Each** of its **instances** contains the **common TDD slot format configuration** for **all** the **UEs** in the **cell**.

##### 2.3.5.3.1 `TDD-UL-DL-ConfigCommon` parameters

Its **three parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `pattern1` `TDD-UL-DL-Pattern` (click 2.3.5.4): mandatory, common

  It does **not** have a **field description** in TS 38.331. It is then **assumed** that it is the **TDD pattern** that determines the **TDD slot formats** within **every TDD period** "***P***" (click 2.3.4). The **TDD period** "***P***" is indicated in `pattern1`, within the corresponding `TDD-UL-DL-Pattern` instance.

- `pattern2` `TDD-UL-DL-Pattern` (click 2.3.5.4): optional, common

  It does **not** have a **field description** in TS 38.331. It is then **assumed** that it is the **TDD pattern** that **optionally** determines (**together** with `pattern1`) the **TDD slot formats** during **every TDD period** "$P_2$" (click 2.3.4). The **TDD period** $P_2$ is indicated in `pattern2`, within the corresponding `TDD-UL-DL-Pattern` instance.

- `referenceSubcarrierSpacing` `SubcarrierSpacing` (click 2.2.10.5.1): mandatory, common

  It is the **reference subcarrier spacing** "$\mu_{ref}$" used to determine the **time domain boundaries** of the **TDD pattern** which **must** be **common** across **all subcarrier specific carriers**, i.e., independent of the **actual subcarrier spacing** in use for data transmission.

**Only** the following **values** can be configured, depending on the **frequency range** where the **band** is located:

- **FR1**: **15**, **30**, or **60 kHz**
- **FR2-1**: **60** or **120 kHz**
- **FR2-2**: **120**, **480**, or **960 kHz**

The **network** configures a value **not larger** than any **SCS** of **configured BWPs** for the serving cell. The **network** or `SL-PreconfigGeneral` configures a value a **not larger** than the **SCS** of (**pre**)**configured SL BWP**.

For more detail on these parameters, refer to clause 11.1 of TS 38.213 and to the table "*TDD-UL-DL-ConfigCommon field descriptions*" of TS 38.331.

#### 2.3.5.3.2 `TDD-UL-DL-ConfigCommon` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
TDD-UL-DL-ConfigCommon ::= SEQUENCE {
  referenceSubcarrierSpacing SubcarrierSpacing,
  pattern1 TDD-UL-DL-Pattern,
  pattern2 TDD-UL-DL-Pattern OPTIONAL,                    -- Need R
  ...
}
```

### 2.3.5.4 `TDD-UL-DL-Pattern` data type

It can be **both broadcast** (or **unicast**) within `SIB1`(click 5.3.4) and **unicast** within the **Non-SI parameter group** (click 5.3.6)

For its **ancestor tree up to** `SIB1`, click 2.3.5.1.

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 2.3.5.2.

**All** its parameters are **common** (click 5.1.3).

It contains the configuration of the **TDD** `pattern1` that determines the **TDD slot formats** within **every TDD period** "***P***". **Another instance** may **optionally** contain the configuration of the **TDD** `pattern2` within **every TDD period** "$P_2$".

#### 2.3.5.4.1 `TDD-UL-DL-Pattern` parameters

Its **six parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `dl-UL-TransmissionPeriodicity` `ENUMERATED`: mandatory, common

  It is the **time duration** of **every TDD period** during which the **TDD slot format** is **determined** by the **other parameters** in this **data type**. This parameter is ignored by the **UE** if `dl-UL-TransmissionPeriodicity-v1530` (see slightly below) is **present**.

  Its **data type** is an `ENUMERATED` (click 2.3.5.4.2) that represents `0.5`, `0.625`, `1`, `1.25`, `2`, `2.5`, `5` and `10` milliseconds.

  The value `0.625` is valid **only** for "$\mu_{ref} \in \{3, 5 \text{ or } 6\}$".

  The value `1.25` is valid **only** for "$\mu_{ref} \in \{2, 3, 5 \text{ or } 6\}$".

  The value `2.5` is valid **only** for "$\mu_{ref} \in \{1, 2, 3, 5 \text{ or } 6\}$".

  The value `10` is valid **only** for "$\mu_{ref} \in \{0, 1, 2, 3 \text{ or } 5\}$".

Recall that "$\mu_{ref}$" is conveyed in `referenceSubcarrierSpacing` from `TDD-UL-DL-ConfigCommon` (click 2.3.5.3).

- `dl-UL-TransmissionPeriodicity-v1530 ENUMERATED {ms3, ms4}`: optional, common

  It has the **same semantics** as `dl-UL-TransmissionPeriodicity` (see the bullet right before).

  Its `ENUMERATED` values represent `3` and `4` milliseconds.

- `nrofDownlinkSlots INTEGER (0..maxNrofSlots)`: mandatory, common

  It is the **quantity** of **consecutive all-Downlink slots** at the **beginning** of **each TDD pattern**.

- `nrofDownlinkSymbols INTEGER (0..maxNrofSymbols-1)`: mandatory, common

  It is the **quantity** of **consecutive Downlink symbol intervals right after** the **last all-Downlink slot**.

- `nrofUplinkSlots INTEGER (0..maxNrofSlots)`: mandatory, common

  It is the **quantity** of **consecutive all-Uplink slots** at the **end** of **every TDD pattern**.

- `nrofUplinkSymbols INTEGER (0..maxNrofSymbols-1)`: mandatory, common

  It is the **quantity** of **consecutive Uplink symbol intervals right before** the **first all-Uplink slot**.

Where the constants are:

- `maxNrofSlots` is `320`, which is the **maximum quantity** of **slots** within **any TDD period**. This **maximum quantity** arises from the valid combinations between "***P***" and "$\mu_{ref}$" indicated in the **two first bullets** above. The maximum quantity of slots corresponds to "***P***=`10`" and "$\mu_{ref}$ =`5`" **is allowed**. Consequently, the **maximum quantity** of **slots** in a **TDD period** is $10 \cdot 2^5 = 320$.

- `maxNrofSymbols-1` is `13`, which is the **maximum index** identifying a **symbol** within a **slot** (**14 symbols**, **indexed** from **0** to **13**). This **maximum index** corresponds to the **normal cyclic prefix** and serves **also** for the **extended cyclic prefix**.

For more detail on these parameters, refer to clause 11.1 of TS 38.213 and to the table "*TDD-UL-DL-Pattern field descriptions*" of TS 38.331.

#### 2.3.5.4.2 `TDD-UL-DL-Pattern` ASN.1 specification

```
TDD-UL-DL-Pattern ::=  SEQUENCE {
  dl-UL-TransmissionPeriodicity ENUMERATED {ms0p5, ms0p625, ms1, ms1p25,
                                            ms2, ms2p5, ms5, ms10},
  nrofDownlinkSlots    INTEGER (0..maxNrofSlots),
  nrofDownlinkSymbols  INTEGER (0..maxNrofSymbols-1),
  nrofUplinkSlots      INTEGER (0..maxNrofSlots),
  nrofUplinkSymbols    INTEGER (0..maxNrofSymbols-1),
  ...,
  [[
  dl-UL-TransmissionPeriodicity-v1530
          ENUMERATED {ms3, ms4} OPTIONAL                              -- Need R
  ]]
}
```

### 2.3.5.5 `TDD-UL-DL-ConfigDedicated` data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 2.3.5.2.

**All** its parameters are **dedicated** (click 5.1.3).

**Each** of its **instances** contains the **dedicated TDD slot format configuration** for a **specific UE**.

#### 2.3.5.5.1 `TDD-UL-DL-ConfigDedicated` parameters

Its **two parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `slotSpecificConfigurationsToAddModList` **`SEQUENCE (SIZE (1..maxNrofSlots)) OF`** `TDD-UL-DL-SlotConfig` (click 2.3.5.6): optional, dedicated

  It contains a **`SEQUENCE OF`** **TDD slot formats** to be applied by the corresponding **UE**. **Each TDD slot format** in the **`SEQUENCE OF`** is **configured** with **one instance** of `TDD-UL-DL-SlotConfig`. The **index** of **each slot format** to be added is **not** provided by its position in the **`SEQUENCE OF`**, and its rather **explicitly** indicated by `slotIndex` from `TDD-UL-DL-SlotConfig`.

  If configured, it **overrides** the **Flexible symbol intervals** performed in the **common slot format** configuration performed in `tdd-UL-DL-configurationCommon` from **either** `ServingCellConfigCommonSIB` (click 3.9.3) or `ServingCellConfigCommon` (click 3.9.6).

- `slotSpecificConfigurationsToReleaseList` **`SEQUENCE (SIZE (1..maxNrofSlots)) OF`** `TDD-UL-DL-SlotIndex`: optional, dedicated

  It contains a **`SEQUENCE OF`** **TDD slot formats** to be **removed** from the **`SEQUENCE OF`** **TDD slot formats** previously configured to the **UE**. The **index** of **each slot format** to be removed is **not** provided by its position in the **`SEQUENCE OF`**, and it is rather **explicitly** indicated by `TDD-UL-DL-SlotIndex`.

  The **data type** `TDD-UL-DL-SlotIndex` is an **`INTEGER (0..maxNrofSlots-1)`**.

where `maxNrofSlots` is `320` and `maxNrofSlots-1` is `319`. The constant `maxNrofSlots` is the **maximum quantity** of **slots** within **any TDD period** (refer to the end of 2.3.5.4.1).

For more detail on these parameters, refer to clause 11.1 of TS 38.213 and to the table "*TDD-UL-DL-ConfigDedicated field descriptions*" of TS 38.331.

#### 2.3.5.5.2 `TDD-UL-DL-ConfigDedicated` ASN.1 specification

```
TDD-UL-DL-ConfigDedicated ::= SEQUENCE {
  slotSpecificConfigurationsToAddModList
              SEQUENCE (SIZE (1..maxNrofSlots))
              OF TDD-UL-DL-SlotConfig OPTIONAL,                  -- Need N
  slotSpecificConfigurationsToReleaseList
              SEQUENCE (SIZE (1..maxNrofSlots))
              OF TDD-UL-DL-SlotIndex OPTIONAL,                   -- Need N
  ...
}
```

### 2.3.5.6 `TDD-UL-DL-SlotConfig` data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 2.3.5.2.

**All** its parameters are **dedicated** (click 5.1.3).

**Each** of its **instances** contains the **TDD slot format** configuration for **one slot.**

#### 2.3.5.6.1 `TDD-UL-DL-SlotConfig` parameters

**All** its **parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `allDownlink` **`NULL`**: mandatory (within **`CHOICE`**), dedicated

  **All** the **slot symbols** are **Downlink**.

- `allUplink` **`NULL`**: mandatory (within **`CHOICE`**), dedicated

  **All** the **slot symbols** are **Uplink**.

- `explicit` **`SEQUENCE`**: mandatory (within **`CHOICE`**), dedicated

  The **`SEQUENCE`** contains **two optional** parameters: `nrofDownlinkSymbols` and `nrofUplinkSymbols`. They indicate, respectively, the **quantity** of **Downlink** and **Uplink symbols** at the **beginning** and **end** of the **slot**. If the parameter is absent, it is equivalent to the value `0`.

  **Both** parameters have the **same data type**: **`INTEGER`** `(1..maxNrofSymbols-1)`. The constant `maxNrofSymbols-1` is `13`, which is the **maximum index** identifying a **symbol** within a **slot** (**14 symbols**, **indexed** from **0** to **13**). This **maximum index** corresponds to the **normal cyclic prefix** and serves **also** for the **extended cyclic prefix**.

- `slotIndex` `TDD-UL-DL-SlotIndex`: mandatory, dedicated

  It uniquely **identifies** a **slot** within **every TDD period** "***P***".

  Its **data type** `TDD-UL-DL-SlotIndex` is an **`INTEGER`** `(0..maxNrofSlots-1)`. The constant `maxNrofSlots` is `320`, which is the **maximum quantity** of **slots** within **any TDD period** (refer to the end of 2.3.5.4.1).

  The **TDD period** "***P***" is configured with `dl-UL-TransmissionPeriodicity` from `TDD-UL-DL-Pattern` (click 2.3.5.4), when configured within `pattern1`, from `TDD-UL-DL-ConfigCommon` (click 2.3.5.3). Parameter `pattern1` is

configured within `tdd-UL-DL-configurationCommon` from **either** `ServingCellConfigCommonSIB` (click 3.9.3) **or** `ServingCellConfigCommon` (click 3.9.6).

- `symbols` **`CHOICE`**: mandatory, dedicated

  It indicates which **slot symbols** are **Downlink** and which ones are **Uplink**. Its **`CHOICE`** contains `allDownlink`, `allUplink`, or `explicit` (see above).

For more detail on these parameters, refer to the table "*TDD-UL-DL-SlotConfig field descriptions*" of TS 38.331.

#### 2.3.5.6.2 `TDD-UL-DL-SlotConfig` ASN.1 specification

```
TDD-UL-DL-SlotConfig ::= SEQUENCE {
  slotIndex TDD-UL-DL-SlotIndex,
  symbols CHOICE {
    allDownlink NULL,
    allUplink NULL,
    explicit SEQUENCE {
      nrofDownlinkSymbols INTEGER (1..maxNrofSymbols-1) OPTIONAL,   -- Need S
      nrofUplinkSymbols   INTEGER (1..maxNrofSymbols-1) OPTIONAL   -- Need S
    }
  }
}
```

### 2.3.5.7 `SlotFormatIndicator` data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 6.12.3.

Its **only parent data type** is `PDCCH-ServingCellConfig` (click 6.12.10).

**All** its parameters are **dedicated** (click 5.1.3).

**Each** of its **instances** contains a **slot format combination set** for **each** of the **UE**'s **serving cells**, plus the **group SF-RNTI** value that the **UE** should use to get its **DCI Format 2_0 messages**.

#### 2.3.5.7.1 `SlotFormatIndicator` parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `dci-PayloadSize` **`INTEGER (1..maxSFI-DCI-PayloadSize)`**: mandatory, dedicated

  It is the **total size** (in **bits**) of the **DCI Format 2_0 payload** that contains the **slot format combination identifiers**.

  The constant `maxSFI-DCI-PayloadSize` is `128`, which is the **maximum size** (in **bits**) of the **DCI Format 2_0** (masked with **SFI-RNTI**) **payload**.

- `sfi-RNTI` `RNTI-Value`: mandatory, dedicated

  It configures a **UE** with a **group SFI-RNTI** that it needs to get the **slot format combination** (click 2.3.5.9) of **each** of its **serving cells** from a **DCI message Format 2_0** (click 6.9.8.1) that is **masked** with **this group SFI-RNTI**.

  Its **data type** is an **`INTEGER (0..65535)`** that identifies a **specific group SFI-RNTI**.

- `slotFormatCombToAddModList` **SEQUENCE** `(1..maxNrofAggregatedCellsPerCellGroup)` **OF** `SlotFormatCombinationsPerCell` (click 2.3.5.8): optional, dedicated

  It **adds** a **list** of **slot format combination sets**, where **each slot format combination set** in the **list** corresponds to **one** of the **UE**'s **serving cells**.

  The constant `maxNrofAggregatedCellsPerCellGroup` is `16`, which is the **maximum quantity** of **aggregated cells** (with **carrier aggregation**, click 10.2) per **cell group.**

For more detail on these parameters, refer to clause 11.1.1 of TS 38.213.

For more detail on these parameters, refer to the table "*SlotFormatIndicator field descriptions*" of TS 38.331.

#### 2.3.5.7.2 `SlotFormatIndicator` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
SlotFormatIndicator ::= SEQUENCE {
  sfi-RNTI RNTI-Value,
  dci-PayloadSize INTEGER (1..maxSFI-DCI-PayloadSize),
  slotFormatCombToAddModList SEQUENCE
            (SIZE(1..maxNrofAggregatedCellsPerCellGroup)) OF
            SlotFormatCombinationsPerCell OPTIONAL,                -- Need N
  slotFormatCombToReleaseList SEQUENCE
            (SIZE(1..maxNrofAggregatedCellsPerCellGroup)) OF
            ServCellIndex OPTIONAL,                                -- Need N
  ...,
  [[
  availableRB-SetsToAddModList-r16 SEQUENCE
            (SIZE(1..maxNrofAggregatedCellsPerCellGroup)) OF
            AvailableRB-SetsPerCell-r16 OPTIONAL,                  -- Need N
  availableRB-SetsToReleaseList-r16 SEQUENCE
            (SIZE(1..maxNrofAggregatedCellsPerCellGroup)) OF
            ServCellIndex OPTIONAL,                                -- Need N
  switchTriggerToAddModList-r16 SEQUENCE (SIZE(1..4)) OF
            SearchSpaceSwitchTrigger-r16 OPTIONAL,                 -- Need N
  switchTriggerToReleaseList-r16 SEQUENCE (SIZE(1..4)) OF
            ServCellIndex OPTIONAL,                                -- Need N
  co-DurationsPerCellToAddModList-r16 SEQUENCE
            (SIZE(1..maxNrofAggregatedCellsPerCellGroup)) OF
            CO-DurationsPerCell-r16 OPTIONAL,                      -- Need N
  co-DurationsPerCellToReleaseList-r16 SEQUENCE
            (SIZE(1..maxNrofAggregatedCellsPerCellGroup)) OF
            ServCellIndex OPTIONAL                                 -- Need N
  ]],
  [[
  switchTriggerToAddModListSizeExt-r16 SEQUENCE
            (SIZE(1..maxNrofAggregatedCellsPerCellGroupMinus4-r16)) OF
            SearchSpaceSwitchTrigger-r16 OPTIONAL,                 -- Need N
  switchTriggerToReleaseListSizeExt-r16 SEQUENCE
            (SIZE(1..maxNrofAggregatedCellsPerCellGroupMinus4-r16)) OF
            ServCellIndex OPTIONAL                                 -- Need N
  ]],
  [[
  co-DurationsPerCellToAddModList-r17 SEQUENCE
            (SIZE(1..maxNrofAggregatedCellsPerCellGroup)) OF
            CO-DurationsPerCell-r17 OPTIONAL                       -- Need N
  ]]
}
```

### 2.3.5.8 `SlotFormatCombinationsPerCell` data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 6.12.3.

Its **only parent data type** is `SlotFormatIndicator` (click 2.3.5.7).

**All** its parameters are **dedicated** (click 5.1.3).

**Each** of its **instances** contains a **set** of **slot format combinations**, where **each slot format combination** in the **set** is identified by an **integer** from `0` to `512`.

#### 2.3.5.8.1 `SlotFormatCombinationsPerCell` parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `positionInDCI INTEGER (1..maxSFI-DCI-PayloadSize-1)`: optional, dedicated

  It is the **ordinal** of the **first bit** of the **9-bit** `slotFormatCombinationId` from `SlotFormatCombination` (click 2.3.5.9) **value** for the **UE**'s **cell** identified by `servingCellId` (see above) within the **DCI Format 2_0 message payload**.

  The constant `maxSFI-DCI-PayloadSize-1` is `127`, which is the **maximum quantity** (minus 1) of **payload** of a **DCI masked** with **SFI-RNTI**.

- `servingCellId ServCellIndex`: mandatory, dedicated

  It is the **cell index**, within the **cell group**, of the **cell** to which this **instance** of `SlotFormatCombinationsPerCell` applies.

  Its **data type** is an `INTEGER (0..maxNrofServingCells-1)`, where `maxNrofServingCells-1` is `31`, which is the **maximum quantity**, minus 1, of **cells** per **cell group**.

- `slotFormatCombinations SEQUENCE (1..maxNrofSlotFormatCombinationsPerSet) OF SlotFormatCombination` (click 2.3.5.9): optional, dedicated

  It is a **set** of **slot format combinations**, where **each combination** is a **sequence** of **one or more slot formats**.

  The constant `maxNrofSlotFormatCombinationsPerSet` is `512`, **maximum quantity** of **slot format combinations** in **each slot format set**.

- `subcarrierSpacing SubcarrierSpacing` (click 2.2.10.5.1): mandatory, dedicated

  It is the **reference subcarrier spacing** for this **Slot Format Combination**. The network configures a value **smaller than or equal** to **both**:

  - **Any subcarrier spacing** of **any configured BWP** of the **serving cell** to which this **instance** applies.
  - The **subcarrier spacing** of the **serving cell** where the **UE** monitors for **SFI indications**.

- `subcarrierSpacing2 SubcarrierSpacing` (click 2.2.10.5.1): optional, dedicated

  It is the **reference subcarrier spacing** for a **Slot Format Combination** on an **FDD** or **SUL cell**. For **FDD bands** `subcarrierSpacing` "$SFI_{scs}$" (see the

previous bullet) is the **reference subcarrier spacing** for **Downlink bandwidth parts** and `subcarrierSpacing2` "***SFI***$_{scs2}$" is the **reference subcarrier spacing** for the **Uplink bandwidth parts**. For **SUL bands** `subcarrierSpacing` "***SFI***$_{scs}$" (see the previous bullet) is the **reference subcarrier spacing** for the **non-SUL carrier** and `subcarrierSpacing2` "***SFI***$_{scs2}$" is the **reference subcarrier spacing** for the **SUL carrier**.

Same as for `subcarrierSpacing`, the network configures a value **smaller than or equal** to **both**:

- **Any subcarrier spacing** of **any configured BWP** of the **serving cell** to which this **instance** applies.
- The **subcarrier spacing** of the **serving cell** where the **UE** monitors for **SFI indications**.

Notice that some **low capability UEs** work in **half-duplex** in **FDD bands**, and therefore they require a **slot format configuration**.

For more detail on these parameters, refer to clause 11.1.1 of TS 38.213.

For more detail on these parameters, refer to the table "*SlotFormatCombinationsPerCell field descriptions*" of TS 38.331.

#### 2.3.5.8.2 `SlotFormatCombinationsPerCell` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
SlotFormatCombinationsPerCell ::= SEQUENCE {
  servingCellId ServCellIndex,
  subcarrierSpacing SubcarrierSpacing,
  subcarrierSpacing2 SubcarrierSpacing OPTIONAL,                    -- Need R
  slotFormatCombinations SEQUENCE
            (SIZE (1..maxNrofSlotFormatCombinationsPerSet)) OF
           SlotFormatCombination OPTIONAL,                          -- Need M
  positionInDCI INTEGER(0..maxSFI-DCI-PayloadSize-1) OPTIONAL,      -- Need M
  ...,
  [[
  enableConfiguredUL-r16 ENUMERATED {enabled} OPTIONAL              -- Need R
  ]]
}
```

### 2.3.5.9 `SlotFormatCombination` data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 6.12.3.

Its **only parent data type** is `SlotFormatCombinationsPerCell` (click 2.3.5.8).

It is **only unicast** within the **Non-SI parameter group**

**All** its parameters are **dedicated** (click 5.1.3).

**Each** of its **instances** contains **one slot format combination**.

#### 2.3.5.9.1 `SlotFormatCombination` parameters

Its **two parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `slotFormatCombinationId SlotFormatCombinationId`: mandatory, dedicated

  It is the **identifier** set within the **DCI Format 2_0 message payload** to **dynamically** select this **slot format combination** for the corresponding **UE**'s **cell**.

  Its **data type** `SlotFormatCombinationId` is an `INTEGER (0..maxNrofSlotFormatCombinationsPerSet-1)`, where the value of this constant is `511`, which is the **maximum quantity**, minus 1, of **slot format combinations** in **each slot format set**. Consequently, **each value** of `slotFormatCombinationId` is **represented** with **9 bits**.

- `slotFormats SEQUENCE (SIZE (1..maxNrofSlotFormatsPerCombination)) OF INTEGER (0..255)`: mandatory, dedicated

  It is the **consecutive slot formats** in this **slot format combination**. The **consecutive slot formats** follow the **same order** as **listed** in the `SEQUENCE OF`. For the case of **IAB-MT**, refer to clause 14 of TS 38.213.

  **Each** `INTEGER` **value identifies one slot format** from table 11.1.1-1 of TS 38.213 (click 2.3.4.2.3).

  The constant `maxNrofSlotFormatsPerCombination` is `256`, which is the **maximum quantity** of **slots formats** in **one slot format combination**.

For more detail on these parameters, refer to clauses 11.1.1 of TS 38.213.

For more detail on these parameters, refer to the table "*SlotFormatCombination field descriptions*" of TS 38.331.

#### 2.3.5.9.2 `SlotFormatCombination` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
SlotFormatCombination ::= SEQUENCE {
  slotFormatCombinationId SlotFormatCombinationId,
  slotFormats SEQUENCE (SIZE (1..maxNrofSlotFormatsPerCombination)) OF
              INTEGER (0..255)
}
```

## 2.3.6 Time synchronization

The following elements are **time synchronized** and **time aligned**:

- **Symbol intervals** of **different subcarriers**, in the **same transmission direction**.

  It is **always required**, even between **subcarriers** with **different numerology** (click 2.1.5.5).

- **Transmitter** and **receiver**.

  It is **always required both** in the **Downlink** and in the **Uplink** transmission directions. It is achieved by **clocking** and by **synchronization signals**.

- **Downlink** and **Uplink**

  It is **always required** (click 2.3.6.1).

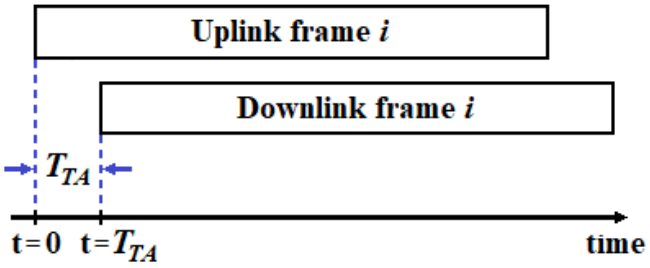
- **TDD band**'s **channels** (click 2.3.6.2)

  It is **almost always required**, in particular in **macro-cells**.

### 2.3.6.1 Synchronization between Downlink and Uplink (timing advance)

The **Downlink** and **Uplink** (and **SUL** if configured) transmissions **must** be **synchronized**. This requires a sufficiently **stable timing source** at **every** UE, plus **periodical synchronization signals**.

The **Downlink** and **Uplink** (and **SUL** if configured) **frames** (with the **same frame number**, click 2.3.1) **must start** at the **same instant** at the **NR node**. To achieve this, the **NR node** indicates **each UE** how much **time** in **advance** over the **beginning** of the **received Downlink frame** it **must** start the **transmission** of the **Uplink frame** that has the **same frame number**. This **time** in **advance** is called the "**timing advance**" $T_{TA}$ and it is required to compensate for the **Uplink propagation delay** from **each UE** to the **NR node**. The following figure shows the **timing advance** at the **UE**:




Since the **UEs** are at **different distances** (i.e., with **different propagation delays**) from the **cell**'s **antenna site**, **each** UE "$x$" has its own **specific timing advance** "$T_{TA,x}$". Since **UEs move** and **channel conditions change**, the **timing advance** "$T_{TA,x}$" of **each UE** is **not constant**, and it rather has to be **periodically updated** by the **NR node**. **Each** UE only receives, and only applies, **one** $T_{TA}$ **value** for **all** its cells that belong to the **same NR node**.

Since **frames** are **time-aligned** at the **NR node**, the **half-frames** and **subframes** are **also time-aligned**. In case the **Downlink active bandwidth part numerology** is the same as the **Uplink bandwidth part** (and **SUL**, if configured) **numerology**, then the **Downlink** and **Uplink** (and **SUL**, if **configured**) **slots** and the **symbol intervals** are **also time-aligned**. In case the **Downlink active bandwidth part numerology** is the **different** to the **Uplink bandwidth part** (and **SUL**, if configured) **numerology**, then the **Downlink** and **Uplink** (and **SUL**, if **configured**) **slots** and the **symbol intervals** are **time-aligned** every $2^{\mu_y-\mu_x}$ **slots** or **symbol intervals** (respectively). The corresponding **numerologies** are "$\mu_y$" and "$\mu_x$", where "$\mu_y > \mu_x$". For the case of **normal CP**, because of the **long cyclic prefix**, some **slots** and **symbol intervals** will **not** be **exactly aligned** (click 2.1.5.5).

The **specific timing advance** $T_{TA,x}$ of a given **UE** "$x$" is obtained with the expression:

$$T_{TA,x} = (N_{TA,x} + N_{TA,offset} + N_{TA,adj}^{common} + N_{TA,adj}^{UE,x}) \cdot T_c$$

The elements in this expression are detailed in clause 4.2 of TS 38.213. A brief overview is:

- "$N_{TA,x}$" and "$N_{TA,offset}$"

  The value of "$N_{TA,offset}$" is the **NR node Tx activation time (Rx to Tx switch delay) plus** the **time** to **avoid NR node** to **NR node interference**.

  For the case of **MsgA** transmission on **PUSCH**, "$N_{TA} = 0$".

  For the case of **random access**, "$N_{TA,offset,x}$" is provided by `n-TimingAdvanceOffset`, **either** from `ServingCellConfigCommonSIB` (click 3.9.3) **or** from `ServingCellConfigCommon` (click 3.9.6). Its possible values are `0`, `25,600` or `39,936`. If it is absent, the **UE** applies the value from table 7.1.2-2 of TS 38.133.

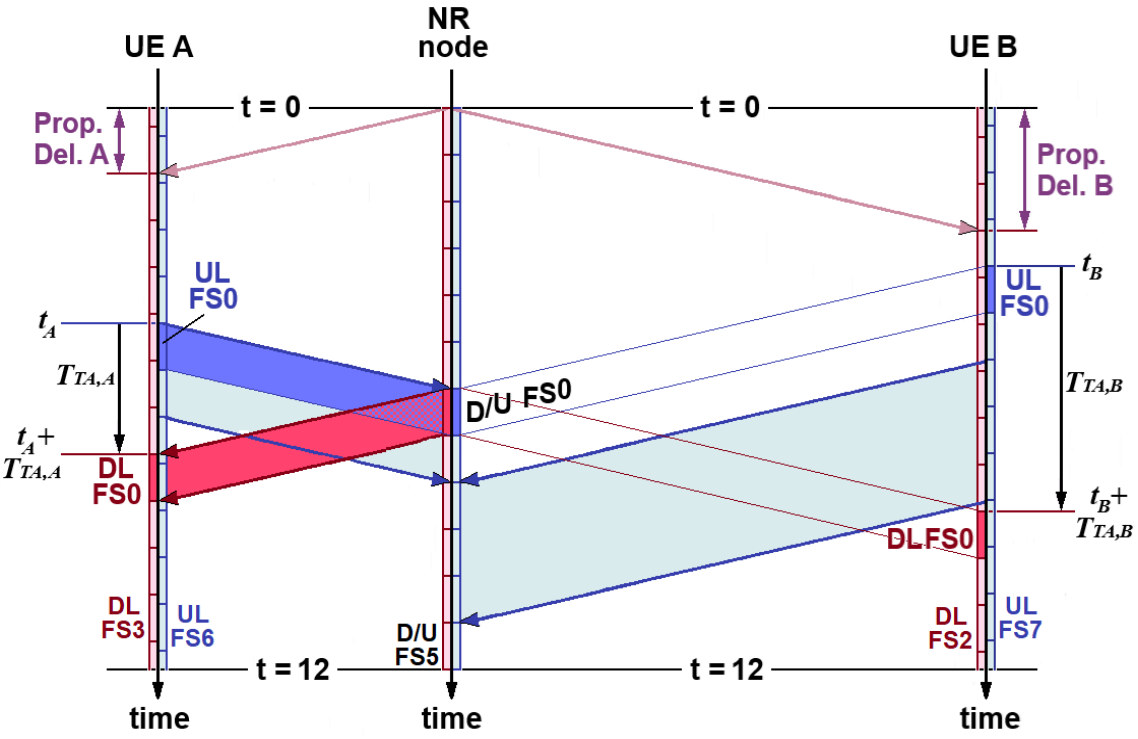
- "$N_{TA,adj}^{common}$"

  It is derived from `ta-Common`, `ta-CommonDrift`, and `ta-CommonDriftVariant-r17`, **all** of them from `TA-Info-r17`. If they are **not** provided, "$N_{TA,adj}^{common} = 0$".

- "$N_{TA,adj}^{UE,x}$"

  It is computed by the **UE** based on the **UE** position and the **serving-satellite-ephemeris-related** parameters. If these parameters are absent, "$N_{TA,adj}^{UE,x} = 0$".

The following figure shows a **time diagram** of an **FDD band** involving the **NR node** and two **UEs** at **different distances**, showing the **alignment** of the **Downlink** and **Uplink frames** at the **NR node** and at **each** of the **two UEs**:




Some **relevant remarks** on this figure are:

- **Time advances up to down**. The **top** is the **past** and the **bottom** is the **future**.

- **Each vertical rectangle** represents **one symbol interval**: **Downlink** are **colored** in **red** and **Uplink** are **colored** in **blue**.
- "**FSn**" stands for "**frame symbol interval *n***", and the **number** "***n***" is the **index** of the **symbol interval** within the **frame**. Consequently, "**FS0**" is the **first symbol** of the **frame**. Notice that **rectangles** corresponding to "**FS0**" are **highlighted** in **dark color**.
- "**DL**", "**UL**" and "**D/U**" stand for "**Downlink**", "**Uplink**" and "**Downlink and Uplink**", respectively.
- The **Downlink** and the **Uplink symbol interval index numbers** shown in the figure correspond to a **Downlink frame** and an **Uplink frame** (respectively) that have the **same frame number**.
- **UE A** is **transmitting** in **Uplink FS0** and **FS1**. Consequently, **UE B cannot transmit** in **Uplink FS0** and **FS1**.
- Since this is an **FDD band**, **transmission and reception** may be done **simultaneously** at the **NR node** and at **every UE**. Actually, in this example the **NR-node** is **simultaneously transmitting and receiving** in **FS0**.
- **Solid colored areas** between **diagonal arrows** represent the intended **destination** of the information, while **white areas** represent **signal propagation** to **non-intended** destination.
- In this example, the **NR node** has **frame offset 0**. Consequently, the **first symbol** of **every Downlink frame** and of **every Uplink frame** (**both frames** with the **same number**) are **simultaneous** at the **NR node**.
- The **first symbol** of **every Downlink frame** and of **every Uplink frame** (**both frames** with the **same number**) are **not simultaneous** at the **UEs**.
- The **first symbol** of **every Uplink frame** is advanced $T_{TA}$ in respect to the **first symbol** of the **Downlink frame** (**both frames** with the **same number**), at **each UE**. This means that the **transmission** of the **first symbol** of **every Uplink frame** is done $T_{TA}$ **before** the **transmission** of the **first symbol** of **every Uplink frame** (**both frames** with the **same number**).
- The **timing advance** $T_{TA}$ is **specific** of **each UE**, and it is approximately equal to its **round-trip time** (**RTT**) to the **NR node**.
- **Each UE**'s **RTT** is the **addition** of its specific **Downlink and Uplink propagation delays**.
- In this example, the **propagation delays** of **UE A** and **UE B** are **1.4 symbol intervals** and **2.6 symbol intervals** (respectively). Assuming **numerology 3** (**SCS=120kHz**) and **NCP**, this is roughly a **distance** to the **NR node** of **3.75 km** and **7 km** (respectively).
- In this example, the **Downlink** and the **Uplink numerology** is the **same**, but this is **not always** the case in an **FDD band**.
- The figure is drawn to **scale**, where **each vertical pixel** is **0.198 µs**.

**FDD channels** do **not** require **synchronization** among them (and in practice they are usually not synchronized).

The **same FDD channel** does **not** require **synchronization** between **neighbor cells** (i.e., **sectors**, click 3.1.4) of the same **antenna site** (click 3.1.3) **nor** between **neighbor cells** (click 3.1.2), and in practice they are usually **not** synchronized.

For more detail on **frame alignment**, refer to clause 4.3.1 of TS 38.211.

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_TimingAdvance.html

#### 2.3.6.2 Synchronization between TDD band's channels

In **every TDD band**, in **macro-cells** (click 3.2.3), **all** the **operator**'s **channels must** have the **same numerology** (i.e., the **same subcarrier spacing** and **slot duration**). Also, the **TDD slot format** (click 2.3.4.2) at every instant **must** also be the **same**. Moreover, **all** its **neighbor NR nodes must start** their **frames** at the **same instant**. This last **requirement** implies that **NR nodes** have **high-precision timing**, based on **GPS**, Precision Time Protocol (**PTP**) or another system. All these requirements **guarantee** that **none** of the **operator**'s **NR nodes** or **served UEs** is simultaneously **receiving** and **transmitting** in the **same band**, which prevents **self-interference** (click 12.1.14). **Channels** in the **same TDD band** of **different operators**, or in **different TDD bands** of the **same operator** are **not** subject to the **above requirements**.

However, in very particular operator's deployments or in private 5G networks, the above **requirements** may be **relaxed**. This will require **individual antenna cables** and **duplexers** for **each channel**, and still may suffer some **self-interference** problems

## 2.4 NR space management

### 2.4.1 Beamforming

**Beamforming** consists of **producing one** (or more) **radiation pattern lobe(s)** with a **high gain** (click 11.21.2) in one (or more) **configurable direction(s)**. **Beamforming** can be done in **Tx** and/or in **Rx**. **Beamforming** usually produces **sidelobes** in addition to the **main lobe(s)**. **Beamforming** consists in the **dynamic configuration** of the **antenna radiation pattern** (click 11.21.5).

**Beamforming** can be characterized by the **endpoint** (**Tx** or **Rx**), by the **lobe beamwidth**, by the **maximum quantity** of configurable **directions** of the **lobe** and by the **time** it takes to **modify** the **lobe direction**.

**Beamforming** only requires **local processing**, and **no processing** is performed at the **remote** communicating **system**. **Tx beamforming** only requires **processing** at the **transmitter**, and it does **not** require any **processing** at the **receiver**. **Rx beamforming** only requires **processing** at the **receiver**, and it does **not** require any **processing** at the **transmitter**. Although the **processing** performed in **Tx beamforming** and in **Rx beamforming** is quite similar, they are **independent techniques**: **Tx beamforming** does **not require** to do **Rx beamforming**, and vice versa. Therefore, **Tx beamforming** is **not** the **reciprocal technique** of **Rx beamforming**, as happens for example with **multiplexing** and **demultiplexing**.

**Beamforming** is a **MIMO** (click 2.4.3) technique and requires an **antenna array** (click 4.5.8.6).

**Beamforming** may be **single-stream** or **multi-stream**. **Single-stream beamforming** radiates a **single signal stream**, while **multi-stream** radiates **several different signal streams**. What is described in this section is **single-stream beamforming**.

For the specific usage of **beamforming** in **5G**, click 10.11.

**Diversity** (click 11.8), **beamforming** (click 11.12), **precoding** (click 2.4.2) and **MIMO** (click 2.4.3) are **different**, but **related**, concepts. For their **relationship**, click 2.4.4.

**Beamforming** may be **analog**, **digital** or **hybrid**, as it is detailed in the following subsections.

**Beamforming** is sometimes called **spatial filtering**.

Source web page:

- https://en.wikipedia.org/wiki/Beamforming

#### 2.4.1.1 Analog beamforming

**Analog beamforming** is processed with **passband signals** in the **time domain**.

In **Tx**, **each analog beamformer** applies **several phase shifts** and **weights** to its **input signal**, and **outputs several time-shifted** and **weighted versions** of its **single input signal**. **Each** of **these output streams** is fed to **one different power amplifier** and **one different Tx array antenna**. Therefore, this requires as **many power amplifiers** and **as many Tx array antennas** as the **quantity** of **streams** that the **Tx beamformer outputs**. The **radiated signals** experience **constructive** and **destructive interference**, resulting in **one Tx beam** that contains the **original signal**, with **high gain** in a **specific direction**. The **Tx beam direction** is configurable with the **phase shift** and **weight** applied to **each** of the **Tx streams**.

In **Rx**, **each analog beamformer** applies a **different phase shift** and **weight** to **each** of its **input signals** and **adds** them to produce a **single signal** with a **significant gain** from a **specific direction**. **Each** of **these input streams** comes from **one different low-noise amplifier** and from **one different Rx array antenna**. Therefore, this requires as **many low-noise amplifiers** and **as many Rx array antennas** as the **quantity** of **streams** that the **Rx beamformer inputs**. The **specific Rx lobe direction** is configurable with the **phase shift** and **weight** applied to **each** of the **Rx streams**. Some **secondary lobes** are produced in **addition** to the **main lobe**. The **optimal lobe direction** is usually a **compromise** between **signal gain** from the desired **transmitter** and **low interference** from **other transmitters** received through the **secondary lobes**.

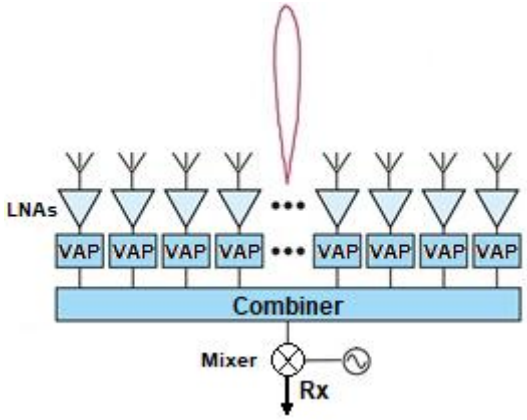
Some characteristics of **each analog beamformer** are:

- It is usually implemented with **passive hardware devices**.
- It manages **passband signals** in the **time domain**.
- It can **only** produce **one main lobe**.

In **Tx analog beamforming**, the **lobe** is oriented towards the **intended receivers**. It is **quite frequent** that the **Tx analog beamformer** is placed **between** the **Tx filter** and the **power amplifiers**. This requires as **many power amplifiers** as **phased** and **weighted signals**, but it has the **advantage** that the **beamformer** works with a **low power signal**. It is **also possible** to find designs where the **beamformer** is placed **after** a **single power amplifier**.

In **Rx analog beamforming**, the **lobe** is oriented to achieve a **compromise** between **good reception** from the **transmitter**, and **poor reception** from **interfering sources**, in order to **maximize SNIR**.

The following figure shows an **analog Rx beamformer**:




Some **relevant remarks** on this figure are:

- **LNA** stands for **low-noise amplifier** (click 4.5.8.11).
- **VAP** stands for **variable amplitude** and **phase**. **Each** such device performs **phase shifting** and **weighting** of **one individual signal**. The **combiner** just **adds** the resulting signals.
- The **mixer** with the **oscillator** performs the signal **down conversion** (click 4.5.8.12) that transforms the **passband signal** into a **baseband signal** (both in the **time domain**).

**Analog beamforming** is sometimes called "**single-lobe beamforming**".

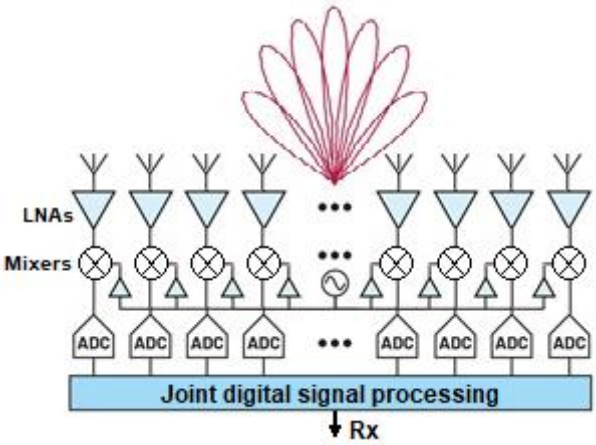
### 2.4.1.2 Digital beamforming

**Digital beamforming** consists in **digitally processing** the **streams** so as to achieve the **desired radiation pattern**.

**Digital beamforming** is much **more flexible** than **analog beamforming**. Among other capabilities, it allows to produce **more than one main lobe**. However, it requires **more RF devices** and the **processing** is considerable more complex.

**Digital beamforming** may be processed **either** in the **frequency domain** or in the **time domain** (in **digital** representation). In 5G it is **most frequently** processed in the **frequency domain**.

The following figure shows a **digital Rx beamformer**:

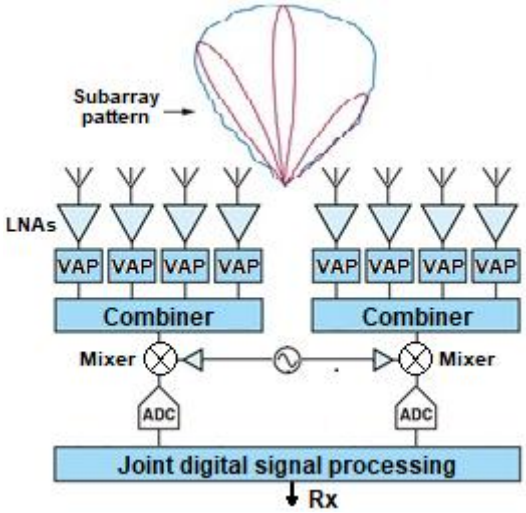



Some **relevant remarks** on this figure are:

- **LNA** stands for **low-noise amplifier** (click 4.5.8.11).
- The **mixer** with the **oscillator** performs the signal **down conversion** (click 4.5.8.12) that transforms the **passband signal** into a **baseband signal** (both in the **time domain**).
- **ADC** stands for **analog to digital converter**.

### 2.4.1.3 Hybrid beamforming

It is combination of **analog** and **digital beamforming**. The following figure shows a **hybrid Rx beamformer** composed of **two analog beamformers**, whose **combiner outputs** are **digitally processed**:




Some **relevant remarks** on this figure are:

- **LNA** stands for **low-noise amplifier** (click 4.5.8.11).
- **VAP** stands for **variable amplitude** and **phase**. **Each** such device performs **phase shifting** and **weighting** of **one individual signal**. The **combiner** just **adds** the resulting signals.

- The **mixer** with the **oscillator** performs the signal **down conversion** (click 4.5.8.12) that transforms the **passband signal** into a **baseband signal** (both in the **time domain**).
- **ADC** stands for **analog to digital converter** (click 4.5.8.13).

Source web page:

- https://www.analog.com/en/resources/technical-articles/power-advantage-of-hybrid-beamforming.html

#### 2.4.1.4 Dynamic and (semi)static beamforming

**Beamforming** can be (**semi**)**static** or **dynamic**. Since **very low switching time** of the **lobe direction** can be achieved, the **highest benefits** are obtained from **dynamic beamforming**. **Dynamic beamforming** is sometimes called "**adaptative beamforming**" and "**phased array**".

As a **simile** of **dynamic analog Tx beamforming** imagine that you have **one flashlight** in a **dark room**: you **change** the **orientation** of the **flashlight** such that at any given moment you **may illuminate** the **room area** in which you are **interested** at that moment.

As a **simile** of **dynamic analog Rx beamforming** imagine that you have **one directional microphone** in a **room** with several **people speaking**: you **change** the **orientation** of the **microphone** such that at any given moment you **may listen** to the **person** in which you are **interested** at that moment.

#### 2.4.1.5 Beam sweeping and beam selection

**Beam sweeping** consists of **transmitting** a **sequence** of **directional radio beams**, in **different directions**, to determine what is the **best beam direction** for a **given receiver**.

The **receiver must** collaborate by performing **beam selection** and **conveying** to the **transmitter** what is the **selected beam**.

**Beam selection** consists of **measuring** the **quality** (e.g., **S/NIR**, **throughput**, …) of **each received beam** in a **sequence beams** (produced by **beam sweeping**) to determine which is the **best beam**, which is the "**selected beam**".

**Beam sweeping** is usually **implemented** using **analog beamforming** (click 2.4.1.1). Same as with **analog beamforming**, **beam sweeping** can be **performed** in **Tx** and/or in **Rx**.

In **5G**, the **NR nodes frequently** uses **Tx beam sweeping** (click 6.1.2.3) in **FR2**. The **5G NR nodes might also** use **Rx beam sweeping**.

### 2.4.2 Precoding and detecting

**Precoding** consists of feeding a **different signal** to **each Tx antenna**, in a way that their **combined transmission maximizes** the **total channel throughput**.

**Precoding** is **only** done in **transmission**. **Precoding** does **not require detecting** (although it is very frequently used). **Precoding requires several Tx antennas** and does **not require** a **Rx antenna array** (although it is very frequently used).

**Detecting** consists of estimating the **vector**(**s**) of **transmitted symbols** from the **received vector** of **signal intervals**, in a way that **maximizes** the **channel throughput**.

**Detecting** is **only** done in **reception**. **Detecting requires precoding**. **Detecting requires both** a **Rx antenna array** and **several Tx antennas**.

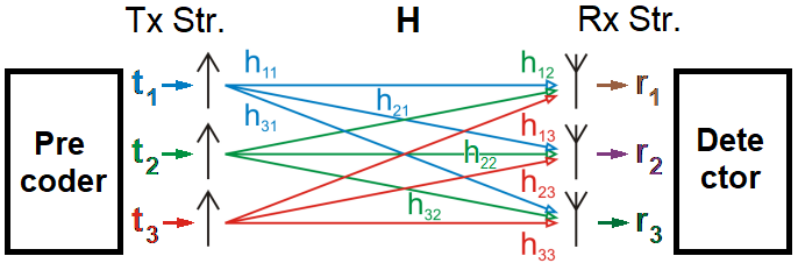
**Both precoding** and **detecting** are **always dynamic**.

**Each signal transmitted** over **each Tx antenna** is called a **Tx stream**. **Each signal received** from **each Rx antenna** is called a **Rx stream**.

There may be **1 transmitter** and **1 receiver** (click 2.4.3.1), **1 transmitter** and **several receivers** (click 2.4.3.2), **several transmitters** and **1 receiver** (click 2.4.3.3) or **several transmitters** and **several receivers** (click 2.4.3.4).

The **total quantity** of **Tx antennas** (from **all transmitters**) and the **tot2.4.3.4al quantity** of **Rx antennas** (from **all receivers**) may be **different** (and usually is).

**Each Rx stream** is **not** one of the **Tx streams**: it is rather the **superposition** of **all** the **Tx streams**, affected by the overall **wireless channel propagation** and by **noise**, as shown in the following figure:




Some **relevant remarks** on this figure are:

- The example in this figure is for **1 transmitter** and **1 receiver**.
- The example in this figure is for the **same total quantity** of **Tx antennas** and **Rx antennas**.
- The **output** of the **precoder** is the **3 streams** "$t_1$" to "$t_3$" that are **fed** to the **3 Tx antennas**.
- The **propagation effects** of the **channel** over the **signals** are **modeled** as the **matrix H**.
- The **input** to the **detector** is the **3 streams** "$r_1$" to "$r_3$" that are **received** from the **3 Tx antennas**.

The mathematical relationship between the **transmitted signal intervals** and the **received signal intervals** may be represented as:

$$\mathbf{r^v = H^{vxm}\, t^m + n^k}$$

where:

- "$t^m$" is the **vertical vector** of "**m**" **transmitted signal intervals**. **Each element** of "$t^m$" is a **signal interval** for **1 Tx antenna** from **1 transmitter**. The **succession** of **signal intervals** in a **given Tx antenna** conforms its **transmitted stream**.
- In case of **several transmitters**, "$t^m$" is the **vertical concatenation** of **every vertical vector** "$t^{mj}{}_j$" sent by **each** of the "**T**" **transmitters**. The value "**m**" is the **addition** of the **dimensions** "$m_j$" of the "**T**" **transmitted vertical matrices** "$t^{mj}{}_j$".
- The **index** "**j**" ranges from **1** to "**T**", which is the **quantity** of **transmitters**.
- **Each** "$r^v$" is the **horizontal vector** of "**v**" **received signal intervals**. **Each element** of "$r^v$" is a **signal interval** from **1 Rx antenna** from **1 receiver**. The

**succession** of **signal intervals** in a **given Rx antenna** conforms its **received stream**.

- In case of **several receivers**, "$\mathbf{r^{v}}$" is the **horizontal concatenation** of **every horizontal vector** "$\mathbf{r^{vi}_{i}}$" received by **each** of the "**v**" **receivers**. The value "**R**" is the **addition** of the **dimensions** "$\mathbf{v_i}$" of the "**R**" **received horizontal matrices** "$\mathbf{r^{vi}_{i}}$".
- The **index** "**i**" ranges from **1** to "**R**", which is the **quantity** of **receivers**.
- "$\mathbf{H^{vxm}}$" is the **channel matrix** that represents the **effects** of the **channel propagation**. The **matrix** $\mathbf{H^{vxm}}$ **transforms** the **vertical matrix** "$\mathbf{t^{m}}$" of "**m**" **transmitted signal intervals** into the **horizontal matrix** "$\mathbf{r^{v}}$" of "**v**" **received signal intervals** (**before** the effect of **noise** "$\mathbf{n^{k}}$"). The **dimension** "**v** × **m**" of the **channel matrix** "$\mathbf{H^{vxm}}$" is the **total quantity** "**v**" of **Rx antennas** (in **all** the **receivers**) by the **total quantity** "**m**" of **Tx antennas** (in **all** the **receivers**).
- "$\mathbf{n^{v}}$" is the **horizontal vector** of "**v**" **noise signals**. **Each element** of "$\mathbf{n^{v}}$" is a **noise interval** at **1 Rx antenna** of **1 receiver**.

For the case of **one receiver**, **maximizing** the **channel**'s **throughput** is tractable as a **single-objective optimization problem**.

For the case of **multiple receivers**, **maximizing** the **channels' throughput** may be defined in **several ways**, such as **maximizing** the **total throughput**, or taking the **Pareto optimal solutions**. In either of these two cases, the result is a **set** of **solutions**. **One way** to **reduce** the **solution set** (possibly to only one solution) is to define a **utility function** as the **weighted sum** of the **receivers' throughput**, with **subjectively** assigned **weights**.

**Precoding** is **most effective** when **Channel State Information** (**CSI**) is available at the **transmitter**.

**Detection** is **most effective** when **CSI** is available at the **receiver**.

The **precoding/detection algorithms** that actually **maximize** the **throughput** are **nonlinear** and consume considerable **computing resources**. However, there are **linear precoding algorithms** that can achieve **high throughput** with **much lower computational complexity**. **Nonlinear precoding algorithms** are based on the concept of **dirty paper coding** (**DPC**). **Linear precoding algorithms** include **maximum ratio transmission** (**MRT**), **zero-forcing** (**ZF**) **precoding**, and **transmit Wiener precoding**. There are also **precoding algorithms** tailored for **low-rate feedback** of **CSI**, such as **random beamforming**.

**Precoding** and **detection** are a key element of **Multiple Input Multiple Output** (**MIMO**, click 2.4.3)

**Diversity** (click 11.8), **beamforming** (click 11.12), **precoding** (click 2.4.2) and **MIMO** (click 2.4.3) are **different**, but **related**, concepts. For their **relationship**, click 2.4.4.

### 2.4.3 Multiple-Input Multiple-Output (MIMO)

**MIMO** consists of using, in a **coordinated** way, **several Tx antennas** and/or **several Rx antennas** from **one** (or more) **transmitter**(**s**) to **one** (or more) **receiver**(**s**). The **quantity** of **Tx antennas** may be **different** to the **quantity** of **Rx antennas**. The **combined configuration** of the **transmitter**(**s**) and the **receiver**(**s**) determines the way **MIMO perform**s. **MIMO** can be used for **beamforming** (click 11.12), for **spatial diversity** (click 11.8), for **spatial multiplexing** (click 11.6), or for a **combination** of them. **MIMO**

can also be used for **CSI estimation**, by **transmitting** and **receiving specific channel estimation signals** over **exclusive Tx antennas and/or Rx antennas** (click 4.11.4.8.3).

The maximum performance of **MIMO** is obtained when used for **space-division multiplexing** or for **space-division multiple-access** based on **Channel State Information** (click 2.4.5) at the **transmitter** and/or at the **receiver**. **MIMO** for **space-division multiplexing** or for **space-division multiple-access allows** to **transmit** "***n***" **parallel flows** of information, **spatially reusing** "***n***" times the **same spectrum segment**. **Each** of the **parallel flows** of **information** is called a "**layer**" (that should **not** be **confused** with the **layers** of **protocol architectures**). This is why **space-division MIMO** is also called **multi-layer MIMO**. The **quantity** of **layers** is **at most** the **smallest** between the **quantity** of **Tx antennas** and of **Rx antennas**.

On most recent texts (including this book), when using the **term MIMO** it implicitly refers to **MIMO** for **space-division**, possibly in combination with **beamforming** (click 11.12) and/or **diversity** (click 11.8). Notice that when the **quantity** of **Tx antennas exceeds** the **quantity** of **Tx layers** (click 2.4.3.5), the **excess Tx antennas** may be used for **beamforming** and/or **diversity** for **some** (or all) the **transmitted layers**.

A **MIMO device** (e.g., an **NR node**, a **WiFi access point** or a **UE**) can be characterized by its **quantity** "**m**" of **Tx antennas** (indicated as "**mT**" or "**mTx**") and its **quantity** "**v**" of **Rx antennas** (indicated as "**vR**" or "**vRx**"). If the **quantity** "**m**" of **Tx antennas** is the **same** as the **quantity** "**v**" of **Rx antennas**, this may be indicated as "**mTR**".

For example, a **MIMO device** "**2T4R**" means that it has **2 Tx antennas** and **4 Rx antennas**. A **MIMO device** "**64TR**" means that it has **64 Tx antennas** and **64 Rx antennas**.

**MIMO devices** frequently have a **different quantity** of **Tx** and **Rx antennas**. In **5G** the **NR node almost always** has **more antennas** than **each** of the **UEs**. The **NR node** may be for example **64T64R**, while its camped **UEs** may be **2T4R**, **1T2R** and **1T1R**.

**Depending** on the **communication mode**, there are **four** different **MIMO types**:

- **Point-to-point MIMO** (click 2.4.3.1)

  It consists of **only one transmitter** and **only one receiver**.

- **Multipoint-to-point MIMO** (click 2.4.3.2)

  It consists of **several transmitters** and **only one receiver**.

- **Point-to-multipoint MIMO** (click 2.4.3.2)

  It consists of **one transmitter** and **several receivers**.

- **Multipoint-to-multipoint MIMO** (click 2.4.3.2)

  It consists of **several transmitters** and **several receivers**.

**Depending** on the **quantity** of **users**, there are **two** different **MIMO types**:

- **Single-User MIMO** (**SU-MIMO**)

  It is the same as **point-to-point MIMO**.

- **Multi-User MIMO** (**MU-MIMO**)

  It includes **multipoint-to-point MIMO**, **point-to-multipoint MIMO** and **multipoint-to-multipoint MIMO**.

**Two particular cases** of **MIMO** systems are:

- **Multiple-input and single-output** (**MISO**)

  A particular **MIMO** case where the **receiver** has **only one antenna**.

- **Single-input and multiple-output** (**SIMO**)

  A particular **MIMO** case where the **transmitter** has **only one antenna**.

**Single-input single-output** (**SISO**) systems are **not MIMO**. A **SISO** system is a conventional radio system where the **transmitter** has **only one Tx antenna** and the **receiver** has **only one Rx antenna**.

In **5G**, the **NR node**'s **array antennas** may be **considerably spaced away** (i.e., **multiple wavelengths**). However, the **UE**'s **array antenna separation** is **heavily space-constrained** due to their **physical size**. The **limitations** arising for **small antenna separation** at the **UE** may be **mitigated** through **advanced algorithms** and **antenna design**.

**Diversity** (click 11.8), **beamforming** (click 11.12), **precoding** (click 2.4.2) and **MIMO** (click 2.4.3) are **different**, but **related**, concepts. For their **relationship**, click 2.4.4.

For the allowed **configurations** (bands, channels, power, …) of **5G Uplink MIMO**, refer to clauses 5.2D, 6.2D, 6.3D, 6.4D, 6.5D, 7.3D, 7.4D, 7.5D, 7.6D, 7.7D and 7.8D of TS 38.101-**1**, **-2** and **-5**.

### 2.4.3.1 Point-to-point MIMO

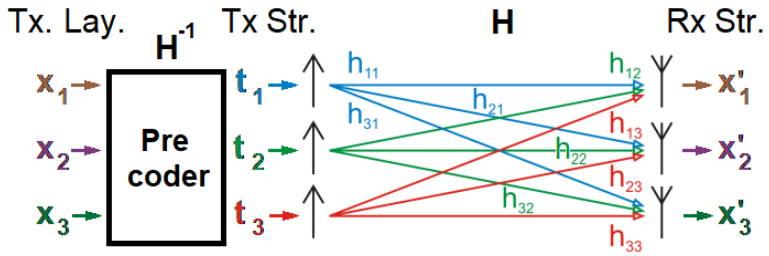
It consists of **MIMO** with **only one transmitter** and **only one receiver**.

The **quantity** of **Tx antennas** may be **different** to the **quantity** of **Rx antennas**.

When using **point-to-point MIMO** for **space-division multiplexing** (click 11.6), it is done with **precoding** (click 2.4.2), that is mathematically modeled as:

$$\mathbf{r}^{v} = \mathbf{H}^{vxm}\,\mathbf{t}^{m} + \mathbf{n}^{v}$$

For the case of **more** (or equal) **Tx antennas** than **Rx antennas** and a **perfectly known channel matrix** at the **transmitter**, it is theoretically possible to **precode** the **modulated layers** (click 2.4.3.5) such that the **transmitted streams** result in that the **Tx layers** are **directly received** (with the corresponding noise) as the **Rx streams**. Notice that because of the condition on the **quantity** of **antennas**, the **quantity** of **layers** may be the **same** as the **quantity** of **Rx antennas**. Notice further that this theoretical approach does **not** require **channel state information** (**CSI**) at the **receiver**. The following figure shows an example for **3 Tx antennas** and **3 Rx antennas**:



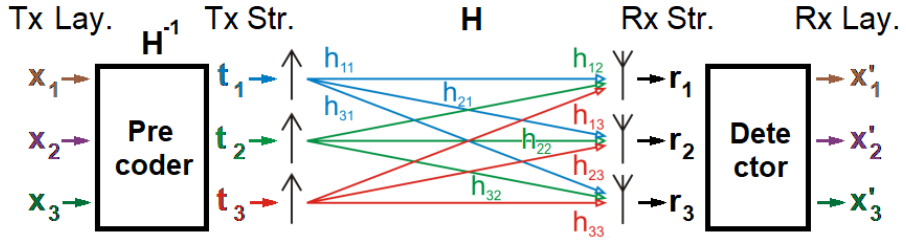

Some **relevant remarks** on this figure are:

- "**H**" is the **channel state information matrix.**

- The **precoder** consists of the **inverse matrix** "**$H^{-1}$**" of the **channel matrix** "**H**". It converts the **3 Tx layers** "**$x_1$**" to "**$x_3$**" into the **3 Tx streams** "**$t_1$**" to "**$t_3$**".
- The **3 received streams** are the **same** as the **3 transmitted layers** "**$x'_1$**" to "**$r'_3$**" (with the effect of noise and distortion, not shown in the figure).
- This example is for the **same quantity** of **Tx antennas** as of **Rx antennas**.

Since the **CSI cannot** be **perfectly known** and the **channel** has **non-linear** effects, this theoretical approach is **not feasible**, and using **only CSI** at the **transmitter** does **not maximize** the **throughput**. To **maximize** the **throughput** it is **required** to **complement precoding** at the **transmitter** with a **specific detector** at the **receiver**, and therefore, **also** the **receiver** requires **CSI knowledge**.

The following figure illustrates this case:




Some **relevant remarks** on this figure are:

- "**H**" is the **channel state information matrix.**
- The **precoder** consists of the **inverse matrix** "**$H^{-1}$**" of the **channel matrix** "**H**". It transforms the **3 Tx layers** "**$x_1$**" to "**$x_3$**" into the **3 Tx streams** "**$t_1$**" to "**$t_3$**".
- The **3 Rx streams** are "**$r_1$**" to "**$r_3$**", which are **similar**, but **different** to the **3 Tx layers** "**$x'_1$**" to "**$r'_3$**" (with the effect of noise and distortion, not shown in the figure).
- The **detector** transforms the **3 Rx streams** "**$r_1$**" to "**$r_3$**" into the **3 transmitted layers** "**$x'_1$**" to "**$r'_3$**" (with the effect of noise and distortion, not shown in the figure).
- This example is for the **same quantity** of **Tx antennas** as of **Rx antennas**.

**Point-to-point MIMO** technology is used in **WiFi** (e.g., in some **802.11n** products).

Depending on the **channel** and **system characteristics**, there are **several point-to-point MIMO techniques**, such as:

- **BLAST** (Bell Laboratories Layered Space-Time): Gerard. J. Foschini (1996).
- **PARC** (Per Antenna Rate Control): Varanasi, Guess (1998), Chung, Huang, Lozano (2001).
- **SPARC** (Selective Per Antenna Rate Control): Ericsson (2004).

### 2.4.3.2 Point-to-multipoint MIMO

It consists of **MIMO** with **one transmitter** and **several receivers**.

**Different receivers** may have a **different quantity** of **Rx antennas**. The **total quantity** of **Rx antennas** may be **different** to the **quantity** of **Tx antennas**.

In case the **total quantity** of **Rx antennas** is **lower** than the **quantity** of **Rx antennas**, the **transmitter** has to decide the **allocation** of **layers** among the different **receivers**.

Recall from 2.4.2 that whenever the **receiver only** has **1 Rx antenna**, it is **not possible** to perform **detection**.

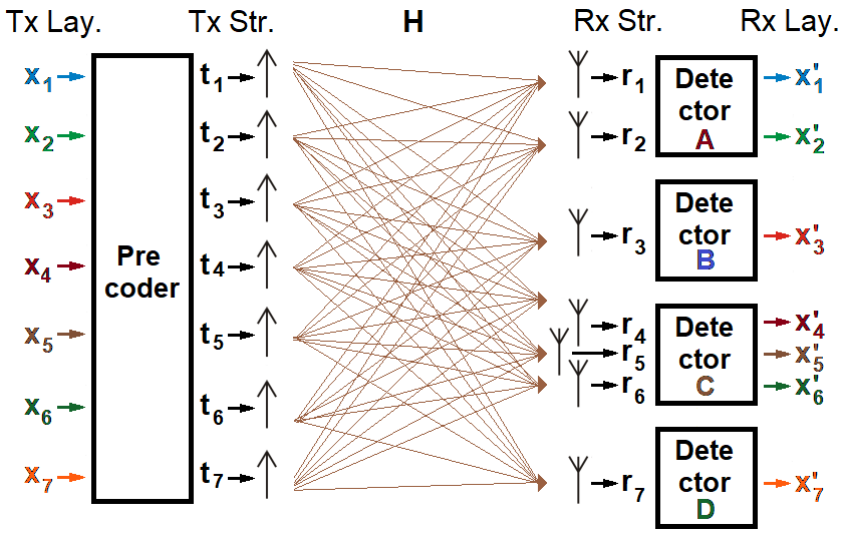
When using **point-to-multipoint MIMO** for **space-division multiplexing** (click 11.6), it is done with **precoding** and **detecting** (click 2.4.2), that is mathematically modeled as:

$$\mathbf{r}^{v} = \mathbf{H}^{vxm}\,\mathbf{x}^{m} + \mathbf{n}^{v}$$

Because the **single precoder** controls **all** the **transmitted streams**, and has the **CSI**, it can **largely mitigate** the **cross-interference** at **every receiver**. The **detector** at **each receiver**, also having the **CSI**, contributes to **mitigate** the **cross-interference**.

The **layers** may be **disjoint among receivers**, or they may **rather** be **shared** by **several receivers**. When there are **shared layers**, the **quantity** of **Rx antennas must** be **larger** than the **quantity** of **layers**. When there are **shared layers**, they may be used for **multicast** communication and/or for **unicast** communication by using some **multiplexing** technique within **each shared layer**. For example, in **5G each Downlink layer** is **multiplexed** in **frequency** and **time** with **OFDM** (click 11.14) and it may also be used for **multicast**.

The following figure illustrates **point-to-multipoint MIMO** with **disjoint layers among receivers**:

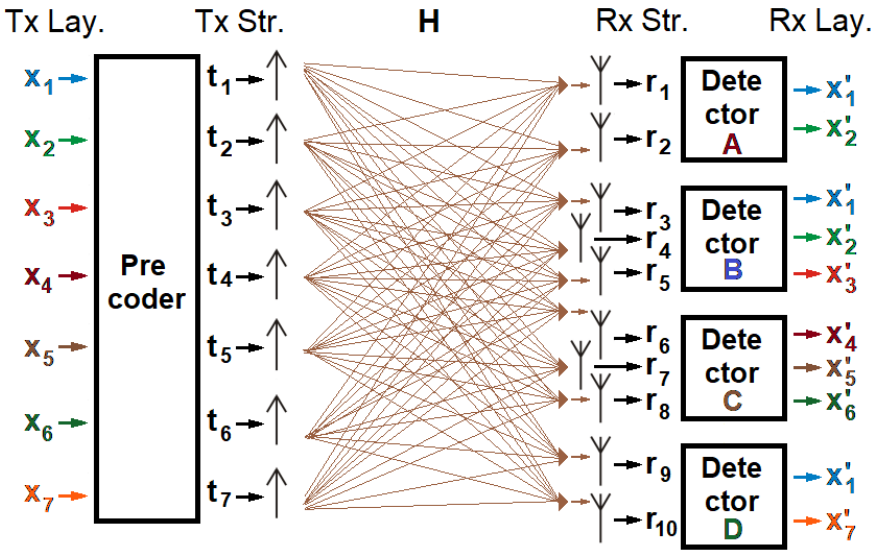



Some **relevant remarks** on this figure are:

- "**H**" is the **channel state information matrix** (click 2.4.5).
- This example has **4 receivers**, labeled **A** to **D**, **each** with its corresponding **detector**.
- **Different receivers** have the **same** or **different quantity** of **Rx antennas**. In this example, **receiver A** has **2 Rx antennas**, **receivers B** and **C** have **1 Rx antenna**, and **receiver C** has **3 Rx antennas**.
- The **Tx layers** are "$\mathbf{x}_1$" to "$\mathbf{x}_7$" and the **Rx layers** are "$\mathbf{x'}_1$" to "$\mathbf{x'}_7$".
- The **Tx streams** are "$\mathbf{t}_1$" to "$\mathbf{t}_7$" and the **Rx streams** are "$\mathbf{r}_1$" to "$\mathbf{r}_7$".

- This example has **1 Tx stream**, **1 Tx antenna**, **1 Rx antenna** and **1 Rx stream** for **each layer**, but there may be **many other cases**.
- The **channel addition** of **all** the **Tx streams** "$t_i$" is **received** by **each Rx antenna** "**j**" as the **Rx stream** "$r_j$".
- **Noise exists**, but it is **not shown** in the figure.

The following figure illustrates the previous example of **point-to-multipoint MIMO**, but in this case with **some shared layers among receivers**:




Some **relevant remarks** on this figure are:

- **All** of the **relevant remarks** on the **previous figure apply** to **this figure**, **except** the one on **1 Tx stream**, **1 Tx antenna**, **1 Rx antenna** and **1 Rx stream** for **each layer** (the sixth remark).
- This example has **2 shared layers**: a) **layer** "$x_2$" is **shared** by **transmitters A**, **B** and **D**; b) **layer** "$x_2$" is **shared** by receivers **A** and **B**.
- The **quantity** of **Rx antennas** (**10**) is **larger than** the **quantity** of **layers** (**7**).

Depending on the **precoding** being **linear** or **non-linear**, and on the **CSI available** to the **transmitter**, there may be the following **three cases** of **point-to-multipoint MIMO**:

- **Non-linear precoding** with **complete CSI** (click 2.4.3.2.1)
- **Linear precoding** with **complete CSI** (click 2.4.3.2.2)
- **Linear precoding** with **partial CSI** (click 2.4.3.2.3)

**Massive MIMO**

**Massive MIMO** is a particular case of **point-to-multipoint MIMO** when the **quantity** of **Tx antennas** is **considerably larger** than the **quantity** of **receivers**. The value of "**considerably larger**" is **not formally defined** and is frequently understood to be **around** (or **above**) **2** or **3 times more**.

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_MassiveMIMO_MU_MIMO.html

#### 2.4.3.2.1 P-to-mp MIMO: non-linear precoding with complete CSI

**Precoding algorithms** based on **dirty paper coding** (**DCP**) do **pre-cancel known interference without power penalty**. The **transmitter only** requires to know **interference**, and full channel state information is required everywhere to achieve the weighted **sum capacity**. Some specific algorithms are **Costa precoding**, **Tomlinson-Harashima precoding** and the **vector perturbation technique**.

#### 2.4.3.2.2 P-to-mp MIMO: linear precoding with complete CSI

Recall from (click 2.4.2) that **linear precoding algorithms** are **suboptimal** because they **cannot maximize** the **channel throughput**. However, they can **maximize** the **weighted sum performance** (or some **other metric** of **achievable** rates under **linear precoding**). The **optimal linear precoding** does **not** have any **closed-form** expression, but it takes the form of a **weighted Minimum Mean-Square Error** (**MMSE**) precoding for **single-antenna receivers**. The **precoding weights** for a **given receiver** are selected to **maximize** a **ratio** between the **signal gain** at **this receiver** and the **interference** generated at **other receivers** (with **some weights**) plus **noise**. Thus, **linear precoding** can be interpreted as the **optimal balance** between **signal gain** and **inter-user interference**.

The **mathematical optimal weighted MMSE precoding** is **non tractable**, leading to different **approximate heuristic approaches**. Usual approaches concentrate on **either** the **numerator** or the **denominator** of the **mentioned ratio**: **either maximum ratio transmission** (**MRT**) or **zero-forcing** (**ZF**). **MRT precoding only maximizes** the **signal gain** at the **intended receiver**. **MRT** is **close-to-optimal** in **noise-limited systems**, where the **inter-user interference** is **negligible** compared to the **noise**. **ZF precoding** aims at **nulling** the **inter-user interference**, at the **expense** of **losing some signal gain**. **ZF precoding** can achieve a **performance close** to the **sum capacity** when the **number** of **users** is **large** or the system is **interference-limited** (i.e., the **noise** is **weak** compared to the **interference**). A **balance** between **MRT** and **ZF** is obtained by **regularized zero-forcing** (also known as **signal-to-leakage-and-interference ratio** (**SLNR**) or as **transmit Wiener filtering**). **All** of **these** heuristic algorithms can **also** be applied to **multiple antenna receivers**.

Another approach has been used to **reformulate** the **weighted sum rate optimization problem** to a **weighted sum men-square error** (**MSE**) **problem** with **additional optimization MSE weights** for **each symbol**.

**Optimal linear precoding** can be **computed** using **monotonic optimization algorithms**, but the **computational complexity** grows **exponentially** with the **quantity** of **receivers**. Consequently, **optimal linear algorithms** are only useful for a **low quantity** of **receivers** or for **benchmarking**.

#### 2.4.3.2.3 P-to-mp MIMO: linear precoding with limited CSI

In practice, **CSI** at the **transmitter** cannot be **complete** due to **estimation errors** and **quantization**. **Inaccurate channel knowledge** may result in **significant loss** of **throughput**, because the **interference** between the **multiplexed streams cannot** be sufficiently **controlled**. In **closed-loop systems**, the **feedback capabilities** decide which **precoding strategies** are **feasible**. **Each receiver** can **either feedback** a **quantized version** of its **complete channel knowledge** or **focus** on **certain critical performance indicators** (e.g., the **channel gain**).

- When the **complete channel knowledge** is **fed back** with **good accuracy**, it is convenient to use **algorithms** designed for having **full channel knowledge** with

**minor performance degradation**. For example, **zero-forcing** may achieve the **full multiplexing gain**, but **only** provided that the **accuracy** of the **channel feedback increases linearly** with **SNR** (in **decibels**). **Quantization** and feedback of **CSI** is based on **vector quantization**, and **codebooks** based on **Grassmannian line packing** have shown **good performance**.

- When the **feedback rates** are **very low**, it is convenient to use **suboptimal algorithms** that work well with **very low channel knowledge**. For example, **random beamforming** or **opportunistic beamforming** can achieve a **reasonably good performance** that **scales** like the **sum capacity** for a **large quantity** of **receivers**. In **random beamforming**, a **set** of **beamforming directions** are **selected randomly** and **each receiver feedbacks** a **few bits** to **inform** the **transmitter which beam** provides it the **best performance** and what **rate** can be **supported**. For a **large quantity** of **users**, there is a **high probability** that **each random beamforming weight** will **provide good performance** for at least **one user**.

#### 2.4.3.3 Multipoint-to-point MIMO

It consists of **MIMO** with **several transmitters** and **only one receiver**.

**Different transmitters** may have a **different quantity** of **Tx antennas**. The **total quantity** of **Tx antennas** may be **different** to the **quantity** of **Rx antennas**.

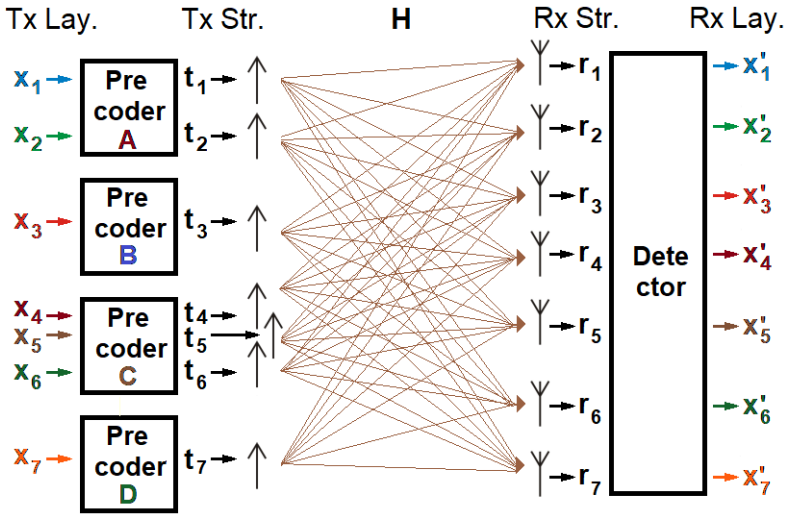
When using **multipoint-to-point MIMO** for **space-division multiple access** (click 11.9), it is done with **precoding** and **detecting** (click 2.4.2), that is mathematically modeled as:

$$\mathbf{r}^{v} = \mathbf{H}^{vxm}\,\mathbf{x}^{m} + \mathbf{n}^{v}$$

Since the **precoders** belong to **different transmitters**, it is **not** possible to perform an **overall precoding** of **all** the **layers** (click 2.4.3.5) **sent** by **all** the **transmitters**. There will therefore be **cross-interference** between the **different transmitters**. This **cross-interference** is **mitigated** by the **receiver detector**, but it **cannot** be **completely suppressed**.

The **layers** may be **disjoint among transmitters**, or they may **rather** be **shared** by **several transmitters**. When there are **shared layers**, the **quantity** of **Tx antennas must** be **larger** than the **quantity** of **layers**. When there are **shared layers**, they may be used for **unicast** communication by using some **multiplexing** technique within **each shared layer**. For example, in **5G each Uplink layer** is **multiplexed** in **frequency** and **time** with **SC-FDMA** (click 11.16), or in some specific scenarios, with **OFDMA** (click 11.15).

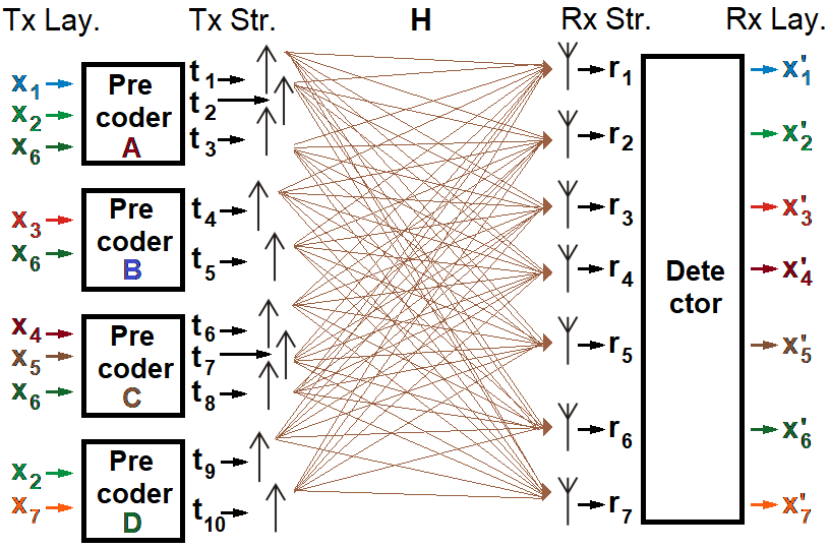
The following figure illustrates **multipoint-to-point MIMO** with **disjoint layers among transmitters**:




Some **relevant remarks** on this figure are:

- "**H**" is the **channel state information matrix** (click 2.4.5).
- This example has **4 transmitters**, labeled **A** to **D**, **each** with its corresponding **precoder**.
- Notice that **different transmitters** have the **same** or **different quantity** of **Tx antennas**. In this example, **transmitter A** has **2 Tx antennas**, **transmitters B** and **C** have **1 Tx antenna**, and **transmitter C** has **3 Tx antennas**.
- The **Tx layers** are "$x_1$" to "$x_7$" and the **Rx layers** are "$x'_1$" to "$x'_7$".
- The **Tx streams** are "$t_1$" to "$t_7$" and the **Rx streams** are "$r_1$" to "$r_7$".
- This example has **1 Tx stream**, **1 Tx antenna**, **1 Rx antenna** and **1 Rx stream** for **each layer**, but there may be **many other cases**.
- The **channel addition** of **all** the **Tx streams** "$t_i$" is **received** by **each Rx antenna** "**j**" as the **Rx stream** "$r_j$".
- **Noise exists**, but it is **not shown** in the figure.

The following figure illustrates **multipoint-to-point MIMO** with **shared layers among transmitters**:




Some **relevant remarks** on this figure are:

- **All** of the **relevant remarks** on the **previous figure apply** to **this figure**, **except** the one on **1 Tx stream**, **1 Tx antenna**, **1 Rx antenna** and **1 Rx stream** for **each layer** (the sixth remark).
- This example has **2 shared layers**: a) **layer** "$x_2$" is **shared** by **transmitters A** and **D**; b) **layer** "$x_6$" is **shared** by **transmitters A**, **B** and **C**.
- The **quantity** of **Tx antennas** (**10**) is **larger than** the **quantity** of **layers** (**7**).
- **Noise exists**, but it is **not shown** in the figure.

### 2.4.3.4 Multipoint-to-Multipoint MIMO

It consists of **MIMO** with **several transmitters** and **several receivers**.

**Different transmitters** may have a **different quantity** of **Tx antennas** and **different receivers** may have a **different quantity** of **Rx antennas**. The **total quantity** of **Tx antennas** may be **different** to the **total quantity** of **Rx antennas**.

**Multipoint-to-multipoint MIMO** combines the **drawbacks** of **point-to-multipoint** and **multipoint-to-point**. Additionally, **channel estimation** may be **very challenging**.

### 2.4.3.5 Layers, streams, antennas, channel rank and codewords

**Precoding layer or MIMO layer**

A **layer** is a **bitstream** whose **throughput** is **similar** to the one **achieved** through the **channel** with **only one Tx antenna** and **only one Rx antenna** (**SISO**). **Each layer** corresponds to an **independent bitstream**.

**Layers** are an element of **space-division multiplexing** (or **multiple access**) to increase the **throughput** by **simultaneously delivering multiple layers** over the **same frequency segment**.

The **quantity** of **layers cannot exceed** the **lesser** of the **Tx antennas** or the **Rx antennas**. A couple examples: with **5 Tx antennas** and **3 Rx antennas**, the **maximum** quantity of

**layers** is **3**; with **3 Tx antennas** and **5 Rx antennas**, the **maximum** quantity of **layers** is also **3**.

The term "**layer**" is also **widely used** to denote the hierarchical "**levels**" (with their associated functions) of **protocol architectures**. Since this **different meaning** is used in **different context**s, in most cases it is clear what the meaning of "**layer**" is.

**Precoding stream or MIMO stream**

A **stream** is a **signal either fed** to **one Tx array antenna or received** from **one Rx array antenna**. A **transmitted stream** is what **physically carries** the **data** after having been processed (e.g., **precoded** and **mapped**). A **received stream** is the **addition** of **all** the **transmitted streams after** having **propagated** through the **channel** and adding the **noise**.

**Tx streams** are **derived** from **Tx layers** during **precoding**. **Rx layers** are **derived** from **Rx streams** during **detection.**

The **quantity** of **Tx** or **Rx streams** is the **same** as the **quantity** of **Tx** or **Rx antennas**, respectively.

**Antennas**

**Each Tx array antenna radiates one** and **only one Tx stream**. **Each Rx array antenna receives one** and **only one Rx stream**

**Channel Rank**

The **channel rank** is the **maximum quantity** of **layers** that can be **supported**. The **channel rank** may be **below** the **lesser** of the **quantity** of **Tx** or **Rx antennas** (e.g., due to **limited spatial diversity**, click 11.8).

In **5G** (and also in **LTE**) the **channel rank** is reported by the **UE** to the **NR node**, as part of the **UE capability** (click 4.9.1).

**Codeword**

A **codeword** is a **data block** plus its **channel code**, that is **sent** over **one layer**.

For more detail on **codewords**, click 4.12.7.

**Relationships between element quantities**

The **relationships** between the **quantities** of the different **elements** analyzed in this section are:

**(Codewords = Layers) ≤ (Lesser streams = Lesser antennas) ≤ Channel rank**

### 2.4.4 Relationships between diversity, beamforming, precoding and MIMO

**Diversity** (click 11.8), **beamforming** (click 11.12), **precoding** (click 2.4.2) and **MIMO** (click 2.4.3) are **different**, but **related**, concepts.

**Diversity vs. Tx beamforming**

**Both diversity** (click 11.8) and **beamforming** (click 11.12) **increase** the **reliability** of a **transmitted bitstream**.

However, **diversity** and **beamforming** have significant **differences**:

- **Diversity transmits** and **receives different signals** (**each** of which **encodes all or part** of the **only intended bitstream**) while **beamforming only transmits** and receives **one signal**.
- **Diversity sends** its **signals** over **several** (**partly**) **independent channels**, while **analog beamforming sends** its **signal** over **only one channel**. However, **digital beamforming** may be considered to **send** a **signal** over **several channels** because the **signal** of **each main beam** follows a **different path**.
- **Diversity techniques require** to perform **diversity combining** at the **receiver**, while **Tx beamforming** does **not require** any **processing** at the **receiver**. Notice though that **additive combining** does **not** require **any processing**.
- **Some diversity techniques** work with **only one transmission antenna**, while **beamforming always works** with an **antenna array**.

For the **specific case** of **space diversity** with **additive combining**, it may be considered **somehow similar** to **digital Tx beamforming** with **no Rx beamforming**.

**Beamforming vs. Precoding**

**Both beamforming** (click 11.12) and **precoding** (click 2.4.2) **feed one different signal** to **each antenna** of the **array**.

However, **beamforming** and **precoding** have significant **differences**:

- **Beamforming only transmits one signal** while **precoding transmits different signals**.
- **Beamforming** can be applied to **transmission** and to **reception**, while **precoding only** applies to **transmission**.
- When the **receiver** has **more than one Rx antenna**, **beamforming cannot maximize** the **signal level** at **all** of the **Rx antennas**, while **precoding can** do it.
- **Beamforming** may be (**semi**)**static or dynamic**, while **precoding** is **always dynamic**.
- **Analog beamforming feeds slightly different signals** (a given **signal** with **different phasings** and **scalings**), while **precoding feeds considerably different signals** (the **streams**). This **difference** does **not** apply to **digital beamforming**.

**MIMO vs. beamforming, diversity and precoding**

The concept of **MIMO only** refers to **several Tx antennas** and/or **several Rx antennas**, and it is **not** associated to any **specific** processing **technique**. **MIMO** may therefore be used for **beamforming** (click 11.12), for **spatial diversity** (click 11.8), for **space-division multiplexing** (click 11.6), or for a **combination** of them.

However, in **many texts MIMO** is **implicitly** associated to **space multiplexing**.

### 2.4.5 Channel State Information

The **channel state information** (**CSI**) is a characterization of the **propagation characteristics** of **each** of the **different spatial channels** (**1 spatial channel** with **each UE**) that the **NR node manages**. **CSI** describes **how** a **transmitted signal arrives** at the **receiver after** the **combined effect** of, for example, *Attenuation*, *Doppler* and *Fading*.

Knowledge of the **CSI** at the **transmitter** allows it to adapt the **transmitted signal** (**precoding**, click 2.4.2) to **optimize** the **received signal** to **increase its reliability** (**diversity**, click 11.8) or for **space-division multiplexing** (**MIMO**, click 2.4.3).

There are basically **two** levels of **CSI**:

- **Instantaneous CSI**

  Consists of the **current channel characteristics**, which can be viewed as knowing the **impulse response** of a **digital filter**. **Instantaneous CSI** is sometimes called **short-term CSI**.

- **Statistical CSI**

  Consists of a **statistical characterization** of the **channel**. **Statistical CSI** for a channel can include, for example, its **fading distribution**, its **average gain**, its **line-of-sight component** and its **spatial correlation**.

**CSI** is usually **estimated** by the **receiver**, **quantized** and **feedback** to the **transmitter**. **CSI estimation** by the **receiver** may be done over a **known pilot signal** sent by the **transmitter**. In **TDD systems** it is possible for the **transmitter** itself to determine the **CSI** by doing **reverse-link channel estimation**. In this case, the **transmitter** and the **receiver** can have different **CSI values**. The **CSI** at the **transmitter** and at the **receiver** are sometimes referred to as **CSIT** and **CSIR** (respectively).

The **CSI acquisition** is limited in practice by **how fast** the **channel characteristics change**. In **fast fading** channels, the **channel characteristics vary rapidly** under the transmission of a **single information symbol**. In this **type** of **channels only statistical CSI** is useful. However, in **slow fading channels** the **instantaneous CSI** can be estimated with **reasonable accuracy** and used for **precoding** for some time before being **outdated**. Many systems use a combination of **instantaneous CSI** and **statistical CSI** to interoperate with a **wide range** of **channel types**.

The **CSI** may be **modeled** as the **channel matrix H** that **transforms** the **transmitted streams** into the **received streams**, as shown in **precoding** (click 2.4.2).

Source web page:

- https://en.wikipedia.org/wiki/Channel_state_information

## 2.5 NR signals

**Each transmitted signal** is the **electro-magnetic energy radiated** from **1 antenna**.

**Each received signal** is the **electro-magnetic energy received** from **1 antenna**.

These **signals** are **processed** in **several stages**, along which the **signals** are **represented** in **different ways**.

For an **overview** of **NR signal processing**, click 2.5.1.

For a **more detailed** description of the **NR signal processing** in **transmission** (**Tx**), click 2.5.2.

For the **complex-valued baseband signal** representation in the **frequency domain**, click 2.5.3.

For the **change** of **representation** of the **complex valued baseband signal** from/to the **frequency domain** from/to the **time domain** (in **both** cases with **digital representation**) using the (**inverse**) **discrete Fourier transform**, click 11.19.

### 2.5.1 Overview of NR signal processing

**Each transmitted signal** and **each received signal** is a **real-valued multicarrier** (click 11.3.2) **passband signal** in the **time domain** with **analog representation** "$PS(t_a)$".

**Each real-valued multicarrier** (click 11.3.2) **passband signal** in the **time domain** with **analog representation** "$PS(t_a)$" is converted **from** (in **Tx**) or **to** (in **Rx**) a **complex-valued multicarrier baseband signal** in the **time domain** with **analog representation** "$BS(t_a)=I(t_a)+jQ(t_a)$" by means of an **up/down converter** (click 4.5.8.12).

**Each complex-valued multicarrier baseband signal** in the **time domain** with **analog representation** "$BS(t_a)=I(t_a)+jQ(t_a)$" is converted **from** (in **Tx**) or **to** (in **Rx**) a **complex-valued multicarrier baseband signal** in the **time domain** with **digital representation** "$BS(t_d)=I(t_d)+jQ(t_d)$" by means of a **digital-analog converter** (click 4.5.8.13).

**Each complex-valued multicarrier baseband signal** in the **time domain** with **digital representation** "$BS(t_d)=I(t_d)+jQ(t_d)$" is converted **from** (in **Tx**) or **to** (in **Rx**) a **complex-valued multicarrier baseband signal** in the **frequency domain** "$BS(f)$" by using (respectively) an **inverse discrete Fourier transform** (**iDFT**, click 11.19) or a **DFT**.

For the **mathematical representation** of the **complex-valued multicarrier baseband signal** in the **frequency domain**, click 2.5.3.

The **Uplink** (or **SUL**) **signals transmitted** by **each** and **every UE** in the **same bandwidth part** conform to the **same carrier**, and are **time-synchronized** and **frequency-aligned**, so their **cross interference** at the **NR node** should be **sufficiently low**.

### 2.5.2 5G NR Tx signal processing

**Each signal** is subject to **multiple processing stages** along the different **NR node subsystems** (click 4.7) or of the **UE**. Initially, **each Tx signal** is **1 baseband complex-valued multicarrier signal** (click 11.3.2) in the **frequency domain** "$BS(f)$".

**Each** such **baseband complex-valued signal** in the **frequency domain** "$BS(f)$" conforms to the **characteristics** of the **corresponding carrier** (click 2.2.5.1). Some of these **characteristics** are the **numerology** (click 2.1), the **quantity** of **subcarriers** (which are the **coefficients** for the **inverse discrete Fourier transform** (**iDFT**, click 11.19), the **frequency location** and the **location** of the **DC subcarrier** (click 2.2.10.2). The **baseband complex-valued signal** in the **frequency-domain** "$BS(f)$" is the **materialization** of the **resource grid** (click 2.2.5.6) that is **determined** by the **carrier** and within the **currently active** (click 2.2.7.4) **bandwidth part** (click 2.2.7).

The **Tx signal bandwidth** is the **complete bandwidth** determined by the **carrier**. The **subcarriers** that are **outside** of the **currently active bandwidth part** have **zero power**, so their corresponding **frequency bins** have a **zero coefficient** in the **iDFT** (i.e., they are **Null subcarriers**, click 2.2.10.1). Notice that even though these **subcarriers** have **zero coefficient**, they **still belong** to the **Tx signal**.

**Each Downlink Tx complex-valued baseband signal** in the **frequency domain** "$BS(f)$" is an **OFDM** (click 11.14) **multicarrier signal**.

**Each Uplink** (or **SUL**) **Tx complex-valued baseband signal** in the **frequency domain** "$BS(f)$" is a **DFT-s-OFDMA** (click 11.16) **multicarrier signal** or an **OFDMA** (click 11.15) **multicarrier signal**.

In **SISO** (click 2.4.3) **systems**:

- The **Downlink signal** that is **received** at the **single Rx antenna** of **each** and **every UE** in the **cell** (click 3.1.1) is the **transmitted signal** from the **single Tx antenna** of the **NR node**, after the **signal** has **transversed** the **specific channel** from the **Tx antenna** to the **single Rx antenna** of **each** of the **UEs** in the **cell**.
- The **Uplink** (or **SUL**) **signal** that is **received** at the **single Rx antenna** of the **NR node** is the **superposition** of **several signals**, where **each signal** is **transmitted** by the **single Tx antenna** of **each** and **every** of the different **active UEs** in the **cell**, after **each such signal** has **transversed** its **specific channel** from the **single Tx antenna** of **each** of the **UEs** to the **single Rx antenna**.

In **MIMO** (click 2.4.3) **systems**:

- The **Downlink signal** that is **received** at **each** of the **Rx antenna**(**s**) of **each** of the **UEs** in the **cell** is the **superposition** of the **transmitted signals** from **each** and **every Tx antenna** of the **NR node**, after **each transmitted signal** has **transversed** its **specific channel** from **each** of the **Tx antennas** to **each** of the **Rx antennas**.
- The **Uplink** (or **SUL**) **signal** that is **received** at **each** of the **Rx antennas** of the **NR node** is the **superposition** of **several signals**, where **each signal** is **transmitted** by **each** of the **Tx antennas** of **each** and **every** of the different **active UEs** in the **cell**, after **each such signal** has **transversed** its **specific channel** from **each** of the **Tx antennas** to **each** of the **Rx antennas**.

When **space-division multiplexing** (click 11.6.3) is used, there is **1 initial Tx complex-valued baseband signal** in the **frequency domain** "$BS_{lay}(f)$" for **each Tx layer** "$lay$". These "$N_{layer}$" **Tx complex-valued baseband signals** in the **frequency domain** "$BS_{lay}(f)$" **are** later transformed into "$N_{port}$" **Tx complex-valued baseband signals** in the **frequency domain** "$BS_{aps}(f)$". This complete transformation is performed with the then current **precoding matrix** "$W$". **Each** resulting **signal** "$BS_{aps}(f)$" is a **stream**, and it may also be called an "**antenna port sequence**".

**Each** of the "$N_{port}$" **Tx complex-valued baseband signal** in the **frequency domain** "$BS_{aps}(f)$" is later transformed into **1 pair** of **Tx real-valued baseband signals** in the **time domain** with **digital representation** (i.e., **time samples**) "$I_{aps}(t_d)+jQ_{aps}(t_d)$". The function "$I(t_d)$" is the **in-phase component**, "$Q(t_d)$" is the **quadrature component** and "$j$" is the **imaginary** "$\sqrt{-1}$". **Each such** transformation is performed by **computing 1 iDFT** (click 2.5.3) with the **carrier characteristics**.

**Each pair** of **Tx real-valued baseband signals** in the **time domain** with **digital representation** "$I_{aps}(t_d)+jQ_{aps}(t_d)$" is later transformed into **1 pair** of **Tx real-valued baseband signals** in the **time domain** with **analog representation** "$I_{aps}(t_a)+jQ_{aps}(t_a)$". **Each such** transformation is performed with **1 dual digital-to-analog converter** (**D/A**, click 4.5.8.13).

**Each pair** of **Tx real-valued baseband signals** in the **time domain** with **analog representation** "$I_{aps}(t_a)+jQ_{aps}(t_a)$" is later transformed into **1 Tx real-valued passband**

**signal** in the **time domain** with **analog representation** "$PS_{aps}(t_a)$". **Each such** transformation is performed with **1 upconverter** (click 4.5.8.12).

For more detail on the **signal generation** and **processing** in **PDSCH** and **PUSCH**, click 4.12.

For more detail on **OFDM baseband signal generation**, refer to clause 5.3 of TS 38.211.

### 2.5.3 Baseband signal representation

The **mathematical representation** of the **complex-valued multicarrier baseband signal** in the **frequency domain** is done in the following way.

For **digital modulations** (click 11.2.1), **each symbol** is usually represented as a **complex number** "$a \cdot e^{j\varphi}$". Recall that **Euler**'s **formula** for **complex numbers** states:

$$e^{jx} = cos(x) + j\ sin(x)$$

The complex number **magnitude** "$a$" represents the **symbol amplitude** and the complex number **phase** "$\varphi$" (i.e., its **argument**) represents the **symbol phase**. The **amplitudes** "$a_s$" of the different **symbols** from **each digital modulation constellation** are usually **normalized** to **1**.

**One symbol interval** of a **complex-valued OFDM baseband signal** is represented in the **frequency domain** as a **summatory** of **symbols**, **each** of them adjusted to its **power level**, and placed in its specific **OFDM subcarrier** and **symbol interval**:

- **Power level**

  **Each symbol** is adjusted to its **specific power level** by **multiplying** its complex representation by a **scalar power level** "$\beta$". The result is "$\beta \cdot a \cdot e^{j\varphi}$".

- **Subcarrier placement**

  **Each symbol** is placed in its **specific OFDM subcarrier** (i.e., its specific **DFT** (click 11.19) **frequency bin** (click 11.19.3)) by **multiplying** its complex representation by the **phasor** (click 11.19.6) "$e^{j2\pi fs}$", where "$f_s$" is the **center frequency** of **subcarrier** "$s$" within the **baseband signal**. The result is "$\beta \cdot a \cdot e^{j(2\pi fs+\varphi)}$". Recall that the **frequency** of **phasors** may take **negative** values (as well as **positive** values).

- **Symbol interval placement**

  **Each symbol** is placed in its **specific symbol interval** "$(t_1, t_2)$", by adding the **variable time** "$t$" to the exponent, for "$t$" within "$(t_1, t_2)$". The result is "$\beta \cdot a \cdot e^{j(2\pi fst+\varphi)}$". It **cyclic prefix** (click 2.3.2) is added **later** to the time-domain signal by the **cyclic prefixer** (click 4.5.7.3).

The resulting signal during 1 symbol interval is converted into the

Source web pages:

- https://electronics.stackexchange.com/questions/260172/is-a-baseband-signal-single-sided

# 3. Cellular networks and cells

This chapter contains the description of mobile networks, nodes and cells.

## 3.1 Cell, antenna site and sector

Source web pages:

- https://www.cenrf.net/info/what-is-the-difference-between-a-cell-and-a-se-84804871.html

### 3.1.1 Cell

The **term** "**cell**" can have the following **2 related interpretations**:

- **Physical cell**

  **Each physical cell** is a **given geographical area** (click 3.10.3) **illuminated** by **1** to **3 antennas** (or **antenna arrays**), that are located at **1 antenna site**, that belong to **1 NR node** (click 4.1), such that the **UEs located** in that **area** can access to **5G services**, over **1** to **3 related radio channel(s)** (click 2.2.3).

  **Each physical cell** can be characterized by its **angle** (see **sector** below) and its **reach** (click 3.10.1), **both** in respect to the **antenna site** (see below) and by its **1** to **3 radio channels**.

- **Logical cell**

  **Each logical cell** is a **set** of **parameters** that the **NR node offers** to the **UEs**, and that the **UEs** recognize as a **serving cell**.

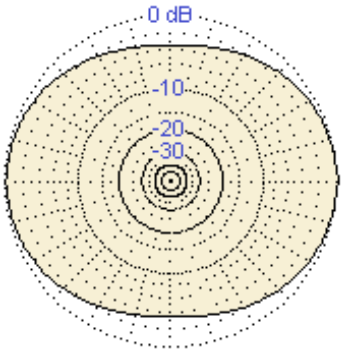
  **Each logical cell** can be characterized by its **physical cell identifier** (**PCI**, click 3.5.3.2), by its **1** to **3 radio channels**, by its **SIB1** (click 5.3.4), by its **other SIBs** and **posSIBs** (click 5.3.5) and possibly by its **synchronization** and **signal block** (**SSB**, click 6.1). The "**physical**" **cell identifier** refers to the **cell identifier** at the **physical layer** of the protocol stack, to distinguish it from **cell identifier** in **higher layers**.

The **1** to **3 related channel(s)** and **antennas** of a **cell** may be **either**:

- **1 channel**: **1 TDD channel**, with **Downlink** and **Uplink carriers** and usually with **1 antenna** (or **antenna array**).
- **2 channels**:
  - **1 pair of FDD channels** (**1 Downlink channel** and **1 Uplink channel**), and usually with **1 antenna**.
  - **1 TDD channel**, with **Downlink** and **Uplink carriers** plus **1 SUL channel**, and usually with **2 antennas** (or **antenna arrays**): **1** for **TDD** and **1** for **SUL**.
- **3 channels**: **1 pair of FDD channels** (**1 Downlink channel** and **1 Uplink channel**) plus **1 SUL channel** and usually with **2 antennas** (or **antenna arrays**): **1** for **FDD** and **1** for **SUL**.

Be aware that some texts call "**cell**" to the **node** whose **antennas** illuminate the **cell(s)**. This **terminology misuse** should be avoided because it has a **high risk** of creating confusion.

The following figure shows the **radiation pattern** of an **omnidirectional antenna** that **illuminates** a certain **cell area**:




**Final remarks on cells**

**Cells** are sometimes called "**serving cells**" to emphasize that they provide communication **services**, but this book will mainly use the term "**cells**".

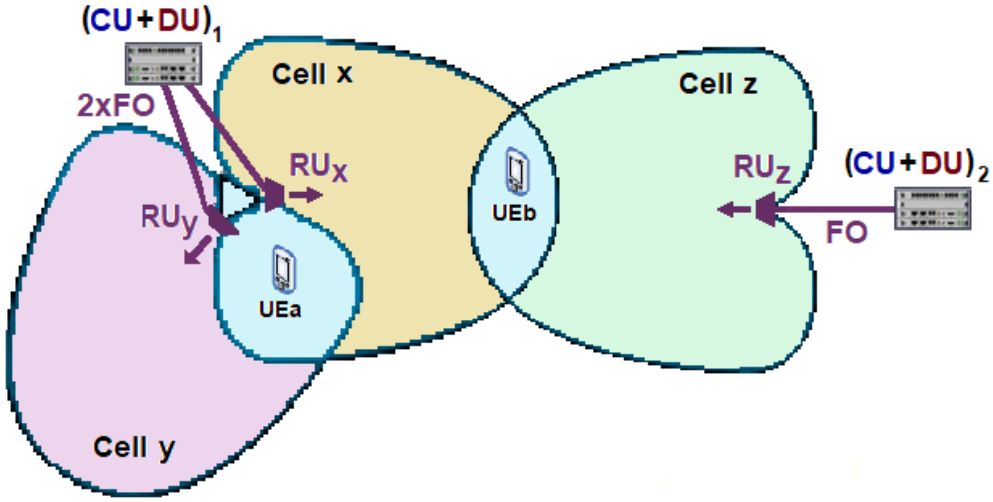
Whenever the **cell area** is a **sector** (click 3.1.4), the **cell** is sometimes called a "**sector**"

For more detail on **different types of cells** and **cell groups**, click 3.5.3.

For the relationship between **cells**, **nodes** and **channels**, click 3.2.

## 3.1.2 Neighbor cells

**Two cells** whose **areas** (click 10.3) **overlap** or share a **frontier** are called **neighbor cells**. The following figure shows **two pairs** of **neighbor cells**: (**Cell$_x$**, **Cell$_y$**) and (**Cell$_x$**, **Cell$_z$**):




Some **relevant remarks** on this figure are:

- **Each cells** is illuminated by a **120º antenna** (i.e., **each** of them is a **120º sector**). The **orientation** of **each** of the **120º antennas** is indicated by the **arrows**
- The **cells** "**Cell$_x$**", "**Cell$_y$**" and "**Cell$_z$**" are illuminated (respectively) by the **antennas** of **RUs RUx**, **RUy** and **RUz**.
- There are **two antenna sites**, **1** and **2**. **Antenna site 1** hosts the **antennas** from **RUx** and **RUy**. **Antenna site 2** hosts the **antennas** from **RUz**.
- The **RU** at **antenna site 1** belong to **gNB1**, while the **RU** at **antenna site 2** belongs to **gNB2**.

- **Each** of **both antenna sites** could host **other antennas** (not shown), that may belong to **these gNBs** and/or to **other gNBs**.
- In this example, **each gNB** has **only one DU** co-located with its **CU**, but in general other configurations are possible (click 4.3).
- **UEa** is under **coverage** of **cells** "**Cell$_x$**" and "**Cell$_x$**". In case these **two cells** operate on **different channels**, then **UEa** could **add** its **capacity** through **each** of the **two cells** if **carrier aggregation** (click 10.2) is **supported**.
- **UEb** is under **coverage** of **cells** "**Cell$_x$**" and "**Cell$_z$**" that belong (respectively) to **gNBs** "**1**" and "**2**". In case these **two cells** operate on **different channels**, then **UEb** could **add** its **capacity** through **each** of the **two cells** if **dual connectivity** (click 10.3) is supported.
- For more detail on the **labels** in the figure, click 4.8.

Since an **operator** might have **several different channels** in the **same area**, **cells** may **overlap** in **many different ways** (nested cells, cells with a large overlap, with a small overlap, …). Consequently, **each cell** might have a **substantial quantity** of **neighbor cells** (e.g., **12**), **each** of which **also** might have a **substantial quantity** of **neighbor cells** (some **shared**, some **not**). This is why a **large quantity** of **PCI** (click 3.5.3.2) values is required.

**Neighbor cells** that **operate** on **different channels** do **not interfere**. However, **neighbor cells** that operate on the **same channel** experience a **significant co-channel interference within**, and **around**, their **overlapping areas**. **Neighbor cells** that operate in the **same channel** are called **same-channel neighbor cells**. For **each cell**, its **quantity** of **same-channel neighbor cells** is small (e.g., **1** to **4**, see **coverage** (click 3.10.5)). The **co-channel interference** between **same-channel neighbor cells** might be lar**gely mitigated** by **scheduling coordination** (click 4.11.4.7).

### 3.1.3 Antenna site

**Each antenna site** (also called just "**site**") is **one geographical location** that **hosts one or more antennas**. An **antenna site may** host **antennas** belonging to: **different nodes** and/or **different technologies** (e.g., **5G**, **LTE**, **UMTS**, **GSM**, …) and/or different **telcos**. The **antenna site** for a **macrocell** (click 3.2.3) is usually a **telecommunication tower** or the **rooftop** of a tall building and hosts a substantial quantity of antennas.

An **antenna site** is sometimes called just "**site**".

### 3.1.4 Sector

**Each sector** is a **cell** whose **area** is **not distributed all around** the **antenna site**, and **rather** it **approximately** covers 1 **geometrical circle sector** of a circle that is **centered** in the **antenna site**.

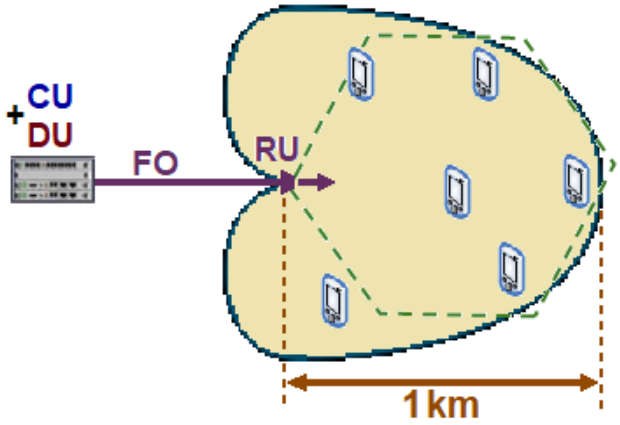
**Each sector** can be characterized by its **beamwidth**, by its **elevation** and by its **reach** (click 3.10.1).

The **initial** cellular technologies mainly used **omnidirectional antennas**, and **each cell** was therefore a **rough circular area** (frequently depicted as a **hexagon**) **centered** in the **antenna site**. **LTE** frequently used **directional antennas** with **120º beamwidth**, and **each cell** was therefore a **rough 120º sector** of a **circle centered** in the **antenna site**. This is frequently depicted as a **hexagon** with a **vertex** located at the **antenna site**. **5G** uses mainly **120º** and **60º sectors**.

**Sectors** are essential for planning **coverage** (click 3.10.5) and **sector densification** (click 3.10.6.3).

An example of a **cell sector** is the **area** illuminated by a **120º antenna**, with a **cell reach** (click 3.10.1) of **1 km**, where the **antenna** belongs to a **gNB**. This **gNB** provides service over **1 TDD channel**, with a bandwidth of **80 MHz**, located from **3,600 MHz** to **3,680 MHz**, within **TDD band n78**. This cell is a **120º sector**.

The following figure shows this example, with **six UEs** camped in this **cell**:




Some **relevant remarks** on this figure are:

- In this example, there is **1 antenna site** that hosts **1 antenna**. The **antenna** has a **120º beamwidth** and its **orientation** is indicated by the **arrow**.
- The **cell** is illuminated by the **120º antenna**.
- The "real" **area** of the **cell** is depicted with **mauve color**, based on the **theoretical radiation patterns** of the **120º antenna**.
- The usual representation of a **120º cell** as a **hexagon** is depicted with **green dotted** lines.
- This **antenna site** could host **other antennas** (not shown), that may belong to **this gNB** and/or to other **gNBs**.
- In this example, there is **only one DU** co-located with its **CU**, but in general other configurations are possible (click 4.3).
- For more detail on the **labels** in the figure, click 4.8.

## 3.2 Cell types and cell groups

### 3.2.1 Cell types under the network point of view

Under the **network point of view**, depending on whether `SIB1` is **broadcast** or **not**, you may have:

- `SIB1` **cell** (**SIB1-Cell**)

  A **cell** where `SIB1` is periodically **broadcast** through **PDSCH**, and therefore, where **CSS-Type 0 configuration** is periodically **broadcast** within **SSB**. A **cell** of this type may **become** the **primary cell** (**PCell**) of a **UE** through **initial access** (click 8). See below the **functions** performed by a **SIB1-Cell**.

- **Non-SIB1 cell (NSIB-Cell)**

  A **cell** where **SSB** (click 6.1) is **broadcast**, but where **neither SIB1 nor CSS-Type 0 configuration** are **broadcast**. A **cell** of this type **cannot become** the **primary cell** (**PCell**) of a **UE** through **initial access**.

  If a **UE** attempts **initial access** on a **NSIB-Cell**, its **SSB** usually indicates **either** the **GSCN** where the **UE** may find an **SSB** that provides the **CSS-Type 0 configuration or** a **GSCN range** where the **UE** will **not** find an **SSB** that provides the **CSS-Type 0 configuration**. This information is **very useful** to **speed-up** the **successful cell search** of the **UE**. For more detail, see the description of **ssb-SubcarrierOffset** (click 8.7.1) from **MIB** (click 8.12.3).

- **Non-SSB cell (NSSB-Cell)**

  A **cell** where **SSB** is **not broadcast**. Notice that **every NSSB-Cell** is **also** an **NSIB-Cell**.

  In a **NSSB-Cell**, the **UE** acquires **time** and **frequency synchronization** based on **SSBs** from a **special cell**, or from another **secondary cell** of the **same cell**, if applicable as described in TS 38.133.

A **SIB1-Cell** performs the following functions:

1. **Periodically** (click 6.1.2.2.3) **broadcasts SSB** (click 6.1).

   Reception of **SSB** allows the **UE** to perform **beam selection** (click 6.1.2.2.3), get the **physical cell identity** (**PCI**, click 3.5.3.2) and acquire **Downlink synchronization** (click 2.3.6).

2. The **SSB** contains the **CSS-Type 0 configuration** (click 8.4)

   Reception of **CORESET#0** allows the **UE** to determine where to locate the **SIB1** transmission within **PDSCH**.

3. **Periodically** (click 5.3.4) **broadcasts SIB1** (click 5.3.4)

   Reception of **SIB1** allows the **UE** to perform the **random access procedure** (click 9) (to achieve **Uplink transmission**) and to get many relevant **system information parameters** (click 5.3.4).

4. **Establishes an RRC connection** (click 5.8) **with the UE**

   The **RRC connection** allows the **UE** to get **all** the **common** and **dedicated configuration parameters** (click 5.1.3) for this cell, that has become the **UE**'s **primary cell**. In case **carrier aggregation** is used, the **UE** also gets the **configuration** for **all** the **cells** in its **master cell group**. In case **dual connectivity** is used, the **UE** also gets the **configuration** for **all** the **cells** in its **secondary cell group**.

Under the **network point of view**, depending on whether the network is **public** or **private**, you may have:

- **Public network cell**

  A **cell** belonging to a **public network**.

- **Non-public network only cell** (**NPN-only cell**)

  A **cell** that is **only** available for normal service for the **specific subscribers** of the **non-public network** of the **cell**. An **NPN-capable UE** determines that a **cell** is **NPN-only** when parameter `cellReservedForOtherUse` is `true` **and** parameter `npn-IdentityInfoList-r16` is **present** (both parameters from `CellAccessRelatedInfo`, click 3.6.4).

The **cell types** under the **network point of view** are interpreted **in the same way** by **all UEs**. In other words, **each and every UE** considers a **SIB1-Cell**, or an **NPN-only cell**, as a **SIB1-Cell**, or an **NPN-only cell**, (respectively): there is **no** margin of "**interpretation**" by the **UE**. However, for **cell types** under the **UE point of view** (click 3.2.2), a **given cell** may be considered a **primary cell** by **one UE** and a **secondary cell** by **another UE**.

### 3.2.2 Cell types and cell groups under the UE point of view

Under the **UE point of view**, there are the following **cell types** and **cell groups**:

- **Primary cell** (**PCell**)

  It is the **cell** hosting the **UE**'s **RRC connection** with the **network**. The **cell** where a **UE completes** its **initial access** (click 8) becomes its **primary cell**. The **cell** where a **UE completes** the **connection re-establishment** becomes its **primary cell**. Therefore, the **UE**'s **primary cell** usually **changes** if the **UE conditions** change (**mobility**, **handover**, …).

  In case of **dual connectivity** (**DC**), the **primary cell** is **associated** to the **DC master node**, it belongs to the **DC master cell group** (**MCG**) and its operation frequency (channel) is the **primary frequency**.

  In the context of **carrier aggregation** (**CA**, click 10.2), they call **primary cell** to the **cell** that conveys the **carrier aggregation signaling** for the **cell group**. Notice that this **second** definition is **different** from the **first** definition (first sentence in this bullet point), which may lead to some **confusion**. The **two** following cases may occur:

  - **CA without DC**

    There is **only one CA primary cell**, which is the same as the "**actual**" **primary cell** based on the **first definition**, so there is no ambiguity.

  - **CA with DC**

    There are **two** "**CA primary cells**" but **only one** "**actual**" **primary cell** based on the **first definition**. To avoid ambiguity in this case, the following terminology is used: a) the **CA primary cell** of the **master cell group** is called "**primary cell**" and it is the same as the "**actual**" **primary cell** based on the **first definition**; b) the **CA primary cell** of the **secondary cell group** is called "**primary-secondary cell**" (see the next bullet) and it is a **different cell** from the "**actual**" **primary cell** based on the **first definition**.

  In case there is **neither carrier aggregation**, **nor dual connectivity**, the **primary** is the **only cell** of the **UE**.

- **Primary-secondary Cell** (**PSCell**)

  It **only** exists in case of **dual connectivity**. It is **always different** to the "**actual**" **primary cell** based on the **first definition** from the previous bullet point**.** It is

**associated** to the **DC secondary node**, and it belongs to the **DC secondary cell group** (**SCG**). Depending on the **usage**, or **not**, of **CA** in the **secondary cell group**, the **two** following **cases** may occur:

- **DC without CA in the secondary cell group**

  The **DC secondary cell group** (**SCG**) **only** has **once cell**, which is the **primary-secondary cell**. **primary cell**, which is the same as the "**actual**" **primary cell** based on the **first definition**, so there is no ambiguity.

- **DC with CA in the secondary cell group**

  The **CA primary cell** of the **secondary cell group** is called "**primary-secondary cell**".

- **Secondary cell(s)** (**SCell(s)**)

  They **only** exist in case of **carrier aggregation**. They are the **carrier aggregation cells other** than the **special cells** (see the next bullet). In case there is **dual connectivity** in addition to **carrier aggregation**, they may belong to the **DC master cell group and/or** to the **DC secondary cell group**. They may be activated or deactivated according to traffic, following the **carrier aggregation** procedures.

  Within the **secondary cells**, you may have the following specific types:

  - **PUCCH secondary cell**

    A **secondary cell** configured **with PUCCH** by `PUCCH-Config` (click 7.9.4).

  - **PUCCH-less secondary cell**

    A **secondary cell** configured **without PUCCH**.

- **Special cell(s)** (**SpCell(s)**)

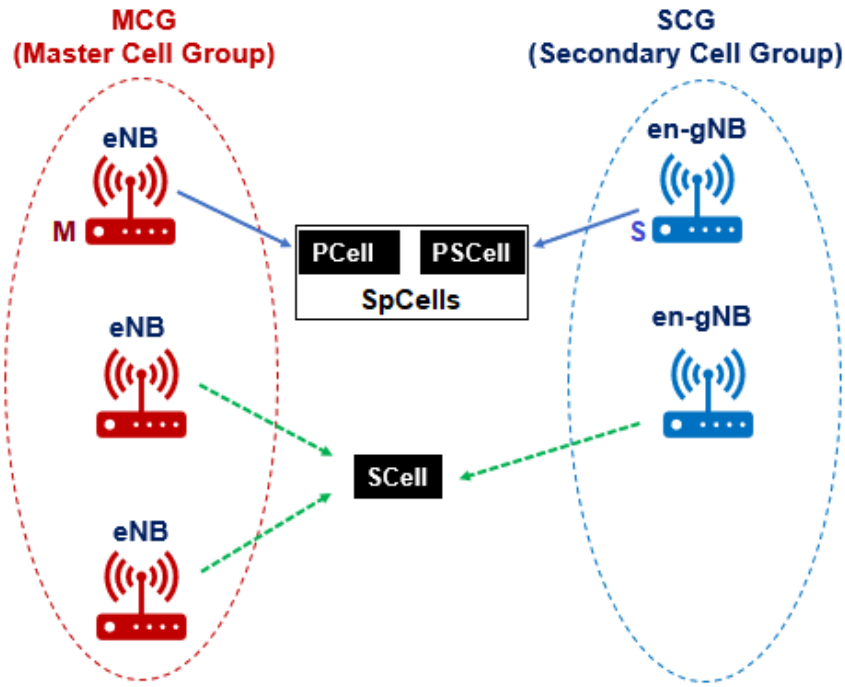
  **A special cell** is **either** the **primary cell** or the **primary-secondary cell**. The **special cells** are the **primary cell** and the **primary-secondary cell**. As you may see, a **special cell** is **not** a **specific type** of **cell** and it is just a **term** to **jointly** denote the **primary and/or** the **primary-secondary cell**s. This term is mainly used in the **context** of **carrier aggregation**.

- **Cell(s)** (or **serving cell(s)**)

  This is a term to **jointly** denote **all** the **cells** providing **service** to a **UE** at any **given time**.

**Example of dual connectivity and carrier aggregation**

The following figure shows an example of **combined dual connectivity** and **carrier aggregation**, showing the **two cell groups** and the different **types** of **nodes** and **cells**:




Some **relevant remarks** on the **particular example** of this figure are:

- This is a **5G deployment NSA option 3**.
- The **master node** (marked with "**M**") is an **eNB**, while the **secondary node** (marked with "**S**") is an **en-gNB**.
- The **primary cell** (**PCell**) is associated to the **master node** (an **eNB**) and belongs to the **master cell group**. The **primary-secondary cell** (**PSCell**) is associated to the **secondary node** (e.g., an **en-gNB**) and belongs to the **secondary cell group**. This applies not only to this example, but to **every case** of **DC plus CA** operation.
- The **master cell group** has **two secondary cells** (**SCells**). It is therefore doing **carrier aggregation** over **three carriers** (channels).
- The **secondary cell group** has **one secondary cell** (**SCell**). It is therefore doing **carrier aggregation** over **two carriers** (channels).
- Recall that the **nodes** may be part of the **same physical equipment** and/or may be placed in the **same location**.

Source web pages:

- http://telcosought.com/5g-ran/5g-nr-mcg-scg-spcell-pcell-scell-node/#:~:text=or%20MR%2DDC.-,5G%20Cell%20Groups%20Types,Groups%20defined%20for%20Dual%20Connectivity.&text=Master%20Cell%20Group%20%E2%80%93%20this%20is,more%20SCell%20(Secondary%20Cell)

### 3.2.3 Cell types based on their size

There is **not** a **formal definition** of **cell sizes**, **nor** a **widely accepted terminology**. The following is therefore a **reasonably frequent classification**, but other texts may perform a different one.

**Cells** can be categorized by their **size** into:

- **Macrocell**

  **Most frequently**, their **cell range** (click 3.10.1.1) is between **2 km** and **5 km** and their **cell area** (click 3.10.3) is between **3 km²** and **40 km²**. **On some cases**, their **cell range** can be up to **60 km** and their **cell area** up to **100 km²**. The **theoretical cell range limit** is **100 km**, but in actual deployments **different real-world effects** reduce this. In **coastal antenna sites** it is **not infrequent** a **cell range** around **60 km inside** the **sea** or **lake**.

  Their **radiated power** is around **20** to **40 W**.

  They operate in **low frequency**: **FR1-1 bands below 1 GHz**.

  They provide **outdoors coverage** with **high indoor penetration**. They provide **some degree** of **coverage** over **large geographic areas**.

  The **antennas** of **macrocells** are located on **high places** (hills, tall buildings, …) or on top of **high antenna masts** (also called telecommunication towers). Standalone **antenna masts** can be as high as **30m**. The 2023 report of the Wireless Infrastructure Association (**WIA**) indicates that the **US** has **209,500 macrocells**. In the same year, Spain has around **60,000 macrocells**.

- **Microcell**

  Their **cell range** is between **200 m** and **2 km** and their **cell area** is between **.15 km²** to **12 km²**. **On some cases**, their **cell range** can be up to **10 km**

  Their **radiated power** is around **1** to **6 W**.

  They operate in **medium frequency**: **FR1-1 bands** above **1 GHz** "1 GHz < $f$ < 2.6 GHz" and **FR1-2 bands** "3.5 GHz < $f$ < 6.7 GHz".

  They provide **outdoors coverage** with **low indoor penetration**. They provide **high-performance coverage** on hotspots with **high user density**, and they **offload macrocells**.

  The **antennas** of **microcells** are located on already existing **infrastructure**, such as **buildings**, **lamp posts** or **urban furniture**.

- **Picocell**

  Their **cell range** is between **10 m** and **200 m** and their **cell area** is between **300 m²** to **0.15 km²**

  Their **radiated power** is around **100** to **250 mW**.

  They operate in **high frequency**: **FR2-1 bands** "24 GHz < $f$ < 47 GHz". They can also operate in **very high frequency**: **FR2-2 bands** "$f$ = 60 GHz".

  They provide **indoors coverage** or **small outdoor areas**. They provide **high-performance coverage** within **office buildings**, **malls** or **railway stations**.

  The **antennas** of **picocells** are located on the ceilings or **walls**.

- **Femtocell**

  Their **cell range** is between **10 m** and **30 m** and their **cell area** is between **300 m²** to **1,000 m²**.

  Their **radiated power** is around **10** to **100 mW**.

  They operate in **very high frequency**: **FR2-2 bands** "$f$ = 60 GHz". They can also operate in **high frequency**: **FR2-1 bands** "24 GHz < $f$ < 47 GHz".

  They provide **indoors coverage** for **one home** or **one small office**.

  The **antennas** of **femtocells** are located wherever the user wants to place them, as it is the case for a **WiFi access point**.

The 2023 WIA report cited above also indicates that the **US** has **452,500 outdoor** "**small-cells**" and **747,400 indoor** "**small-cells**". The term "**small-cell**" is somehow vague and can refer to any of the **last 3 types** of cell cited above or to a **mix** of them.

Source web pages:

- https://www.techtarget.com/searchnetworking/feature/Macrocell-vs-small-cell-vs-femtocell-A-5G-introduction
- https://www.essentracomponents.com/en-gb/news/industries/telecoms-data/a-guide-to-5g-small-cells-and-macrocells
- cellmapper.net/map

## 3.3 Relationship between cells, NR nodes, channels and UEs

**Each cell** belongs to **only one NR node**. **Each cell** may host **several UEs**. **Two neighbor cells** may belong to the **same NR node** or to **two different NR nodes**. **Two neighbor cells** with **low overlap** may operate in the **same channels**. **Two neighbor cells** with **substantial overlap** should operate in **different channels**.

The **antennas** of **each NR node** may illuminate **several cells**. **Different cells** of the **same NR node** may have the **same channels** and/or **different channels**. **Two neighbor NR nodes** are involved in the **handover** of a **UE** when the **UE moves out** of the coverage of one of the **nodes** and **into** the coverage of the **other node**.

**Each channel** may be **used** in **very many cells** from **very many different nodes**. It is **quite frequent** that **all** the **cells** that use a **given channel jointly cover** the **whole network coverage area**, or a **large part** of it.

**Each active UE** is **camped** on **one cell**, **connected** to **one NR node**. If a **UE** uses **carrier aggregation** (click 10.2), it may be **simultaneously connected** to **many cells** from the **same NR node**. If the **UE** uses **dual connectivity** (click 10.3), it may be **simultaneously connected** to **cells** from **two neighbor NR nodes**. A **UE** may simultaneously use **carrier aggregation** and **dual connectivity**.

For example, an **NR node** may be a **3-sector site**, handling **3 cells** for **each TDD channel**. In case the **NR node** had **4 TDD channels**, the **NR node** would be handling **12 cells**. The **4 cells** within **each sector** would have a **significant coverage overlap**.

Although somehow **unusual**, the following configurations are valid.

- **Moving cells**. **Cells** can **move**, if the **antenna** that **illuminates** them is **moving** (e.g., a **NR node** onboard a train) or by using **beamforming** techniques.

- **2-node cells**: As an **exception** to the **rule** that **each cell** is **illuminated** by **only one node**, **two neighbor nodes** may **collaborate** to create a "**2-node cell**" with **coordinated multipoint** (click 10.11).

## 3.4 Network types

Several **5G identifiers** and some **procedures** depend on the **type** of **5G network**. This is why this section presents several **types** of **5G networks**.

A **mobile communications network** may offer **public service** to **many** individual or corporate **customers** or provide **private service** for the **internal use** of **only one organization**. A **mobile communications network** may provide **radio access** to **UEs** based on **land infrastructure** and/or on **satellite infrastructure**.

The following **types** of **mobile communications networks** can be identified:

- **Public Land Mobile Network** (**PLMN**)

  It provides **public service** with radio access over **land infrastructure**. It is managed by **1 telecommunications operator company** (or "**operator**" in short) that most frequently **includes all**, or many, of the mobile technology **generations** (**1G** to **5G**, click 13.2). **Each operator** may manage **many** different **PLMNs** of different technology generations and in different geographical areas (e.g., in **several** different **countries**).

- **Non-Terrestrial Network** (**NTN**)

  It provides radio access based on **satellite infrastructure**. It **usually** (but not always) provides **public service**. It is **usually** (but not always) operated by **1 telecommunications operator company**.

- **Non-Public Network** (**NPN**)

  It provides **private service**. It **usually** (but not always) provides the radio access service based on **land** infrastructure. There are **2 types** of **non-public networks**:

  - **Public Network Integrated Non-Public Network** (**PNI-NPN**)

    It is a **non-public network** that **relies** on **network functions** provided by a **public land mobile network**. It is provided by **PLMN** using a **private network slice** (i.e., a **network slice** for the **sole use** of **1 corporate customer**), **Closed Access Group** (**CAG**) **cells** or a **combination** of both.

    Regarding infrastructure, on **one extreme**, it may be deployed as **only** a **private slice**, **without** any customer-specific **infrastructure**. On **the other extreme**, it may be deployed based on **almost all private infrastructure** (e.g., all the RAN and transport networks, plus all edge computing, and some core network functions). There may also be a **wide range** of **in-between cases**.

  - **Standalone Non-Public Network** (**SNPN**)

    It is a **non-public network** that does **not rely** on **network functions** provided by a **public land mobile network**. It is therefore a **private network**, and it involves substantial **investment** and **commitment** from the **owner**.

Notice though that the **deployment** and **operation** of a **standalone NPN** may be **outsourced** by the owner to an **operator** and/or a **manufacturer** and/or an **engineering** company.

For more detail on **non-public networks**, refer to clause 4.8 of TS 38.300.

Source web pages:

- https://www.3gpp.org/technologies/npn
- https://www.linkedin.com/pulse/5g-non-public-network-dharmesh-yadav

## 3.5 Network, node and cell identity

The **network identity** (click 3.5.1), **node identity** (click 3.5.2) and **cell identity** (click 3.5.3) are **related** among themselves. The **very specific** concept of **system information area identity** (click 3.5.4) is also related with the other ones.

The following table summarizes the relationships between the **PLMN identity**, the **node identity** and the **network layer cell identities**:

| **New radio cell global identity (NCGI)**<br>5 or 6 decimal digits plus 36 bits | | | |
|---|---|---|---|
| **Public land mobile network identity (PLMN ID)**<br>5 or 6 decimal digits | | **New radio cell identity (NCI)**<br>36 bits | |
| **Mobile country code (MCC)** | **Mobile network code (MNC)** | **gNode B identity (gNB ID)** | **Cell identity (CI)** |
| 3 decimal digits | 2 or 3 decimal digits | 22 to 32 bits | 4 to 14 bits |

The following table summarizes the **physical layer cell identity**:

| **Physical cell identity (PCI)**<br>(1,008) | |
|---|---|
| **PCI group**<br>(336) | **PCI identity**<br>(3) |

The **number** between parentheses indicates the **quantity** of **different identifiers**.

### 3.5.1 Network identity

**Each 5G network** is **identified**, depending on the **type of network**, as follows:

- **Public land mobile network identity (PLMN ID)**

  It **uniquely identifies each PLMN** in the **world**.

  It is composed of **3 decimal digits** for the **mobile country code** (**MCC**) plus **2 or 3 decimal digits** for the **mobile network code** (**MNC**). Notice that for the **MNC** the quantity of digits matters: for example, the **MNC** "`02`" is **different** from the **MNC** "`002`". The **PLMN IDs** are used to determine if the **cell selection** procedure is **successful** (click 8.1).

  The **network**'s **PLMN IDs** are **conveyed** to the **UE** in `plmn-IdentityInfoList` from `CellAccessRelatedInfo` (click 3.6.4), within `SIB1` (click 5.3.4). The **data type** of `plmn-IdentityList` is `PLMN-IdentityInfoList` (click 3.6.5), which contains a **list of lists** of data type

`PLMN-Identity` (click 3.6.6). **Each instance** of `PLMN-Identity` contains **1 PLMN ID**.

- **Standalone non-public network identity** (**SNPN ID**)

  It **uniquely identifies each SNPN** in the **world**.

  It is composed of **1 PLMN ID** (**1** instance of `PLMN-Identity`) plus **1** instance of `NID-r16` (click 3.6.7).

- **Public network integrated non-public network identity** (**PNI-NPN ID**)

  It **uniquely identifies each PNI-NPN** in the **world**.

  It is composed of **1 PLMN ID** (**1** instance of `PLMN-Identity`) plus **1** `cag-Identity-r16` from `CAG-IdentityInfo-r16` (click 3.6.7). **CAG** stands for "**Closed Access Group**".

For the **relationship** between the **different types** of **network**, **node** and **cell identities**, click 3.4.

### 3.5.2 RAN node identity

**Each NR node** is **identified** as follows:

- **NR node Identity** (**NR node ID**)

  It **uniquely identifies each NR node within** its **public land mobile network** (**PLMN**). It is a **static** identifier with **variable** length, from **22** to **32 bits**.

  Together with the **cell identity** (**CI**, click 3.5.3) constitutes the **36-bit new cell identity** (**NCI**, click 3.5.3).

  The **UE extracts** the **NR node ID** from the **NCI** using `gNB-ID-Length-r17` from `PLMN-IdentityInfo` (click 3.6.5), sent within `SIB1` (click 5.3.4). Parameter `gNB-ID-Length-r17` indicates the **quantity of bits** of **NR node ID** within the **NCI**.

For the **relationship** between the **different types** of **network**, **node** and **cell identities**, click 3.4.

### 3.5.3 Cell identity

**Each cell** is assigned **several different identifiers**, with **different values** and **uniqueness**, depending on the **scope**:

- **Network layer cell identity** (click 3.5.3.1).
- **Physical layer cell identity** (click 3.5.3.2).
- **Cell group cell identity** (click 3.5.3.3).

As you may see, there are **several cell identities** and cell **identity components** that should **not** be **mistaken** among themselves.

For the **relationship** between the **different types** of **network**, **node** and **cell identities**, click 3.4.

### 3.5.3.1 Network layer cell identity

Within the **network layer** scope, **each serving cell** is **identified** as follows:

- **New radio cell global identity** (**NCGI**)

  It **uniquely identifies each serving cell globally** (i.e., in the **whole world**). It is composed of the **public land mobile network identifier** (**PLMN ID**, click 3.5.1) to which the **cell** belongs to, **and** the **new radio cell identity** (**NCI**, see below) of the **cell**.

  The **PLMN ID** for **NCGI** is in the **first** `PLMN-Identity` within the **`SEQUENCE OF`** in parameter `plmn-IdentityList` from `PLMN-IdentityInfo` (click 3.6.5), sent within `SIB1` (click 5.3.4). Refer to the definition of "***Global cell identity***" in clause 3.1 of TS 38.331, because in TS 38.331 the **NCGI** is called "***Global cell identity***".

  The **NCGI** is similar to the **EUTRA cell global identifier** (**ECGI**) used in **LTE**.

- **New radio cell identity** (**NCI**)

  It **uniquely identifies each public serving cell within** its **PLMN**. It is a **static** identifier with **fixed length** of **36 bits**. It is composed of the **NR node Identity** (**NR node ID**, click 3.5.2), which has from **22** to **32 bits**, **and** the **cell identity** (**CI**) which has from **4** to **14 bits**. Together with the **PLMN ID** (click 3.5.1) constitutes the **NCGI** (see above).

  The **NCI** is **conveyed** to the **UE** in parameter `cellIdentity` from `PLMN-IdentityInfo` (click 3.6.5), sent within `SIB1` (click 5.3.4).

- **Cell identity** (**CI**)

  It **uniquely identifies each serving cell within** its **NR node**. It is a **static** identifier with **variable length** from **4** to **14 bits**.

  Together with the **NR node Identity** (**NR node ID**, click 3.5.2) constitutes the **36-bit new radio cell identity** (**NCI**, see above).

  The **UE extracts** the **CI** from the **NCI** (see above), using parameter `gNB-ID-Length-r17` from `PLMN-IdentityInfo` (click 3.6.5), sent within `SIB1` (click 5.3.4). Parameter `gNB-ID-Length-r17` indicates the **quantity of bits** of **NR node ID** within the **NCI**, and the **remaining bits** are the **CI**.

### 3.5.3.2 Physical layer cell identity

Within the **physical layer** scope (including **handover**), **each serving cell** is **identified** as follows:

- **Physical cell identity** (**PCI**)

  It **uniquely identifies each and every cell among all neighboring cells**. It is a **static** identifier with **1,008 different values**. The **PCI** is sometimes called the **physical layer cell identity**. The **PCI** is **sometimes denoted as** "$N_{ID}^{cell}$". Each cell has **only one PCI**.

  The **PCI** is **not unique** within the **whole network**. However, for **each** and **every** serving **cell**, its **PCI** and the **PCIs** of **all** of **its neighbor cells must all** be **different**. In another words, for **each** and **every** serving **cell**, its **PCI must** be **different** to the one of **all** of its **neighbor cells** (**no collision**), and also, **all** of **its**

**neighbor cells must** have **different PCIs** (**no confusion**). The **first** condition is required to prevent that a **handover command** to move to a **neighbor cell** is **mistaken** by a **command** to **remain** in the **current cell**. The **second** condition is required to prevent that a **handover command** to move to a given **neighbor cell** is **mistaken** by a **command** to move to a **different neighbor cell**.

To assign a **unique identity** to **neighboring countries**, only **four identifiers** are required (refer to the four color theorem). However, **cells** may **overlap** in **many different ways**. Consequently, the **PCI range** of values has to be **substantially larger than 4** in order to assign a **unique identity** to **all neighboring cells**.

The **PCI** (denoted "$N_{ID}^{cell}$") has the following **two parts**:

- **PCI group** (denoted "$N_{ID}^{(1)}$")

  It ranges from **0** to **335**. It is conveyed in **SSS** (click 6.19.3), within **SSB**.

- **PCI identity** (denoted "$N_{ID}^{(2)}$")

  It ranges from **0** to **2**. It is conveyed in **PSS** (click 6.19.1), within **SSB**.

The **PCI** is calculated as:

$$PCI = 3 \bullet \text{PCI group} + \text{PCI identity}$$

$$N_{ID}^{Cell} = 3 \bullet N_{ID}^{(1)} + N_{ID}^{(2)}$$

where "$N_{ID}^{(1)} \in \{0..335\}$" and "$N_{ID}^{(2)} \in \{0..2\}$".

In **5G** there are **1,008** unique **PCIs** (double than the **504** in **LTE**).

The **physical cell identity allows** to distinguish **PSS/SSS** coming from **different cells**, in order for the **UE** to select the **most suitable one**(**s**) (usually the one(s) arriving with higher power).

The **PCI** is also contained in **each instance** of the `PhysCellId` **data type**, which is an `INTEGER (0..1007)`. This **data type** is used in several **RRC** parameters.

For more detail on **PCI**, refer to clause 7.4.2.1 of TS 38.211.

### 3.5.3.3 Cell group cell identity

Within the scope of **each** of the **UE**'s **cell groups**, **each cell** is identified with a value from `ServCellIndex` (click 3.9.9) and **each secondary cell** is identified with a value from `SCellIndex` (click 3.9.9).

The **special cell always** has identifier `0` and **each secondary cell always** has an identifier from `1` to `31`.

## 3.5.4 System information area identity

For the **very specific** purpose of defining the **scope** of **other SIBs** and **posSIBs**, the network may define **system information areas**. A **system information area** is a **set** of **cells**. A **cell** may **only** belong to **one SI area**.

Every **SI area** is assigned a **24 bit SI area identity**, which is **unique** within the **network**. The **SI area identity** may be assigned to each cell with the optional `systemInformationAreaID` from `SI-SchedulingInfo` (click 5.5.3.1), within `SIB1` (click 5.3.4).

## 3.6 Network, node and cell identity parameters

The **UE**'s **PLMN**, **NR node** and **cell identity** (click 3.5) are **configured** with:

- **Common parameters** that are **broadcast** (or **unicast**) within `SIB1`, as described in section 3.6.1.
- **Common** and **dedicated parameters** that are **unicast** within the **Non-SI parameter group**, as described in section 3.6.2.
- **Common** and **dedicated parameters** sent in **RRC inter-node messages**, as described in section 3.6.3.

**PLMN**, **NR node** and **cell identity** configuration does **not** have **any parameter** within `MIB` (click 5.3.3).

Possibly existing **network, node and cell identity parameters** within **other SIBs** and **posSIBs** (click 5.3.5) are **not addressed** in this book.

For an overall explanation on **UE configuration parameters** click 5. For a brief overview of **ASN.1**, click 5.9.

### 3.6.1 Network, node and cell identity parameters within `SIB1`

For **network**, **node** and **cell identity** parameters within `SIB1` (click 5.3.4), the **most relevant related data types** are:

- `CellAccessRelatedInfo` (click 3.6.4): common parameters
- `PLMN-IdentityInfoList` (click 3.6.5): instantiated as common
- `PLMN-IdentityInfo` (click 3.6.5): instantiated as common

The **ancestor tree** (click 5.9.8) of these **relevant data types** up to `SIB1` is:

```
                                  SIB1 (click 5.7.4)
cellAccessRelatedInfo               CellAccessRelatedInfo (click 3.6.4)
  plmn-IdentityInfoList               PLMN-IdentityInfoList (click 3.6.5)
                                        PLMN-IdentityInfo (1..maxPLMN) (click 3.6.5)
      plmn-Identity                       PLMN-Identity (1..maxPLMN) (click 3.6.6)
```

Some **relevant remarks** on this **ancestor tree** are:

- For the **ancestor trees** of `SIB1` **up to** the **top**, click 5.3.4.
- The **data type** `CellAccessRelatedInfo`, also appears on **Sidelink** messages, but **Sidelink** is **not covered** in this book.
- The constant `maxPLMN` is `12`, which is the **maximum quantity** of **PLMNs** broadcast and reported by **each UE** at **establishment**.

### 3.6.2 Network, node and cell identity parameters within the Non-SI parameter group

For **network**, **node** and **cell identity** parameters within the **Non-SI parameter group** (click 5.3.6), the **most relevant related data types** are:

- `PLMN-IdentityInfoList` (click 3.6.5): instantiated as dedicated
- `PLMN-IdentityInfo` (click 3.6.5): instantiated as dedicated

The **ancestor tree** (click 5.9.8) of these **most relevant data types**, up to **UL-DCCH-Message** is:

```
                              UL-DCCH-Message (click 5.8.5.9)
message                         UL-DCCH-MessageType
  c1                              CHOICE
    measurementReport           MeasurementReport (0,1)
      measurementReport           MeasurementReport-IEs
        measResults                 MeasResults
          measResultServingMOList   MeasResultServMOList
                                      MeasResultServMO
                                                (1..maxNrofServingCells)
              measResultServingCell     MeasResultNR
                cgi-Info                  CGI-InfoNR (0,1)
                  plmn-IdentityInfoList     PLMN-IdentityInfoList (click 3.6.5)
                                              PLMN-IdentityInfo (1..maxPLMN)
                                                (click 3.6.5)
                      plmn-Identity             PLMN-Identity (1..maxPLMN)
                                                (click 3.6.6)
              measResultBestNeighCell   MeasResultNR (0,1)
                cgi-Info                  CGI-InfoNR (0,1)
                  plmn-IdentityInfoList     PLMN-IdentityInfoList (click 3.6.5)
                                              PLMN-IdentityInfo (1..maxPLMN)
                                                (click 3.6.5)
                      plmn-Identity             PLMN-Identity (1..maxPLMN)
                                                (click 3.6.6)
          measResultServFreqListEUTRA-SCG
                                      MeasResultServFreqListEUTRA-SCG (0,1)
            measResultServFreqListNR-SCG
                                      MeasResultServFreqListNR-SCG
                                        MeasResult2NR
                                                  (1..maxNrofServingCells)
                  cgi-Info                CGI-InfoNR (0,1)
                    plmn-IdentityInfoList   PLMN-IdentityInfoList (click 3.6.5)
                                              PLMN-IdentityInfo (1..maxPLMN)
                                                (click 3.6.5)
                        plmn-Identity           PLMN-Identity (1..maxPLMN)
                                                (click 3.6.6)
      ueInformationResponse-r16 UEInformationResponse-r16 (0,1)
        ueInformationResponse-r16 UEInformationResponse-r16-IEs
          rlf-Report-r16            RLF-Report-r16 (0,1)
            measResultListNR-r16      MeasResultList2NR-r16 (0,1)
                                        MeasResult2NR-r16 (1..maxFreq)
              measResultList-r16        MeasResultListNR
                                          MeasResultNR (1..maxCellReport)
                  cgi-Info                  CGI-InfoNR (0,1)
                    plmn-IdentityInfoList     PLMN-IdentityInfoList (3.6.5)
                                                PLMN-IdentityInfo
                                                        (1..maxPLMN) (3.6.5)
                        plmn-Identity             PLMN-Identity (1..maxPLMN)
                                                (click 3.6.6)
          connEstFailReport-r16     ConnEstFailReport-r16
            measResultNeighCellListNR MeasResultList2NR-r16 (0,1)
                                        MeasResult2NR-r16 (1..maxFreq)
              measResultList-r16        MeasResultListNR
                                          MeasResultNR (1..maxCellReport)
                  cgi-Info                  CGI-InfoNR (0,1)
                    plmn-IdentityInfoList     PLMN-IdentityInfoList (3.6.5)
                                                PLMN-IdentityInfo
                                                      (1..maxPLMN) (click 3.6.5)
                        plmn-Identity             PLMN-Identity (1..maxPLMN)
                                                (click 3.6.6)
```

```
    nonCriticalExtension      UEInformationResponse-v1700-IEs (0,1)
      connEstFailReportList-r17 ConnEstFailReportList-r17 (0,1)
                                  ConnEstFailReport-r16
                                                  (1..maxCEFReport-r17)
          measResultNeighCellListNR MeasResultList2NR-r16 (0,1)
                                      MeasResult2NR-r16 (1..maxFreq)
              measResultList-r16        MeasResultListNR
                                          MeasResultNR
                                                  (1..maxCellReport)
                  cgi-Info                    CGI-InfoNR (0,1)
                    plmn-IdentityInfoList       PLMN-IdentityInfoList
                                                          (click 3.6.5)
                                                  PLMN-IdentityInfo
                                                    (1..maxPLMN) (click 3.6.5)
                            plmn-Identity           PLMN-Identity
                                                      (1..maxPLMN) (3.6.6)
scgFailureInformation     SCGFailureInformation (0,1)
  scgFailureInformation     FailureInformation-IEs
    failureReportSCG          FailureReportSCG (0,1)
      measResultFreqList        MeasResultFreqList (0,1)
                                  MeasResult2NR (1..maxFreq)
          measResultServingCell     MeasResultNR (0,1)
            cgi-Info                  CGI-InfoNR (0,1)
              plmn-IdentityInfoList     PLMN-IdentityInfoList (click 3.6.5)
                                          PLMN-IdentityInfo (1..maxPLMN)
                                                          (click 3.6.5)
                plmn-Identity               PLMN-Identity (1..maxPLMN)
                                                          (click 3.6.6)
      measResultSCG-Failure     OCTET STRING (CONTAINING
                                  MeasResultSCG-Failure)
          measResultPerMOList       MeasResultList2NR
                                      MeasResult2NR (1..maxFreq)
              cgi-Info                  CGI-InfoNR (0,1)
                plmn-IdentityInfoList     PLMN-IdentityInfoList (3.6.5)
                                            PLMN-IdentityInfo
                                                  (1..maxPLMN) (click 3.6.5)
                  plmn-Identity               PLMN-Identity (1..maxPLMN)
                                                          (click 3.6.6)
mcgFailureInformation-r16 MCGFailureInformation-r16 (0,1)
  mcgFailureInformation-r16 MCGFailureInformation-r16-IEs
    failureReportMCG-r16      FailureReportMCG-r16
      measResultFreqList-r16    MeasResultList2NR
                                  MeasResult2NR (1..maxFreq)
          cgi-Info                  CGI-InfoNR (0,1)
            plmn-IdentityInfoList     PLMN-IdentityInfoList (click 3.6.5)
                                        PLMN-IdentityInfo (1..maxPLMN)
                                                          (click 3.6.5)
              plmn-Identity               PLMN-Identity (1..maxPLMN)
                                                          (click 3.6.6)
ulDedicatedMessageSegment-r16
                          ULDedicatedMessageSegment-r16 (0,1)
  dedicatedSIBRequest-r16   DedicatedSIBRequest-r16
    mcgFailureInformation-r16 MCGFailureInformation-r16
      mcgFailureInformation-r16 MCGFailureInformation-r16-IEs
        failureReportMCG-r16      FailureReportMCG-r16 (0,1)
          measResultSCG-r16         OCTET STRING (CONTAINING
                                      MeasResultSCG-Failure)
            measResultPerMOList       MeasResultList2NR
                                        MeasResult2NR (1..maxFreq)
              cgi-Info                  CGI-InfoNR (0,1)
                plmn-IdentityInfoList     PLMN-IdentityInfoList
                                                          (click 3.6.5)
                                            PLMN-IdentityInfo
                                              (1..maxPLMN) (click 3.6.5)
                    plmn-Identity               PLMN-Identity
                                                  (1..maxPLMN) (3.6.6)
```

Some **relevant remarks** on this **ancestor tree** are:

- You may see that the corresponding **RRC NR messages** are `MeasurementReport`, `UEInformationResponse-r16`, `SCGFailureInformation`, `MCGFailureInformation-r16`, and `ULDedicatedMessageSegment-r16`.
- Notice that the **data type** `CellAccessRelatedInfo` does not exist in this **ancestor tree**, and it is only present in the **relevant PLMN**, **NR node** and **cell identity** parameters **broadcast** in a `SIB1` **message**.
- The constant `maxNrofServingCells` is `32`, which is the **maximum quantity** of **serving cells** (**special cells** plus **secondary cells**) in **each** of the **UE**'s **cell groups**.
- The constant `maxFreq` is `8`, which is the **maximum quantity** of **frequencies**.
- The constant `maxCellReport` is `8`.
- The constant `maxCEFReport-r17` is `4`, which is the **maximum quantity** of **CEF** reports by **each UE**.
- The constant `maxPLMN` is `12`, which is the **maximum quantity** of **PLMNs** broadcast and reported by **each UE** at establishment.

Recall that **Sidelink** channels (and consequently **Sidelink data types** and parameters) are **not** covered in this book.

### 3.6.3 Network, node and cell identity parameters sent in RRC inter-node messages

**PLMN**, **NR node** and **cell identity** are configured with **parameters sent** in **RRC inter-node messages** (click 5.8.6) for **inter-node procedures**.

**PLMN**, **NR node** and **cell identity** parameters sent in **RRC inter-node messages**, the **most relevant related data types** are:

- `PLMN-IdentityInfoList` (click 3.6.5): instantiated as dedicated
- `PLMN-IdentityInfo` (click 3.6.5): instantiated as dedicated

The **ancestor trees** (click 5.9.8) of **these relevant data types** up the **top** are:

```
                                HandoverPreparationInformation
handoverPreparationInformation
                                  HandoverPreparationInformation-IEs (0,1)
  rrm-Config                        RRM-Config (0,1)
    candidateCellInfoList             MeasResultList2NR
                                        MeasResult2NR (1..maxFreq)
      cgi-Info                            CGI-InfoNR (0,1)
        plmn-IdentityInfoList               PLMN-IdentityInfoList (click 3.6.5)
                                              PLMN-IdentityInfo (1..maxPLMN)
                                                                  (click 3.6.5)
          plmn-Identity                         PLMN-Identity (1..maxPLMN) (3.6.6)
```

```
                          CG-Config
cg-Config                   CG-Config-IEs
  candidateCellInfoListSN     OCTET STRING (CONTAINING
                              MeasResultList2NR) (0,1)
      cgi-Info                  CGI-InfoNR (0,1)
        plmn-IdentityInfoList     PLMN-IdentityInfoList (click 3.6.5)
                                    PLMN-IdentityInfo (1..maxPLMN) (3.6.5)
            plmn-Identity             PLMN-Identity (1..maxPLMN) (3.6.6)

                          CG-ConfigInfo
cg-ConfigInfo               CG-ConfigInfo-IEs
  candidateCellInfoListMN     MeasResultList2NR (0,1)
    cgi-Info                    CGI-InfoNR (0,1)
      plmn-IdentityInfoList       PLMN-IdentityInfoList (click 3.6.5)
                                    PLMN-IdentityInfo (1..maxPLMN) (3.6.5)
          plmn-Identity               PLMN-Identity (1..maxPLMN) (click 3.6.6)
  candidateCellInfoListSN     OCTET STRING (CONTAINING
                              MeasResultList2NR) (0,1)
    cgi-Info                    CGI-InfoNR (0,1)
      plmn-IdentityInfoList       PLMN-IdentityInfoList (click 3.6.5)
                                    PLMN-IdentityInfo (1..maxPLMN) (3.6.5)
          plmn-Identity               PLMN-Identity (1..maxPLMN) (click 3.6.6)
```

Some **relevant remarks** on these **ancestor trees** are:

- You may see that the **roots** of the **trees** are the **RRC inter-node messages** `HandoverPreparationInformation`, `CG-Config` and `CG-ConfigInfo`.
- Notice that the **data type** `CellAccessRelatedInfo` does not exist in these **ancestor trees**, and it is only present in the **relevant PLMN**, **NR node** and **cell identity** parameters (click 3.6.1) within a `SIB1` **message**.
- The constant `maxFreq` is `8`, which is the **maximum quantity** of **frequencies**.
- The constant `maxPLMN` is `12`, which is the **maximum quantity** of **PLMN** broadcast and reported by **each UE** at **establishment**.

## 3.6.4 `CellAccessRelatedInfo` data type

It is **only broadcast** (or **unicast**) within `SIB1` (click 5.3.4).

For its **ancestor tree up to** `SIB1`, click 2.2.6.2.

**All** its parameters are **common** (click 5.1.3).

**Each** of its **instances** contains the **cell access related information** for the corresponding cell.

### 3.6.4.1 `CellAccessRelatedInfo` parameters

Its **five parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `cellReservedForFutureUse-r16` **ENUMERATED** `{true}`: optional, common

  It indicates whether the **cell** is **reserved** for **future use**, as defined in TS 38.304. It is **applicable** to **all PLMNs** and **NPNs**. It is **ignored** by **IAB-MT**.

- `cellReservedForOtherUse` **ENUMERATED** `{true}`: optional, common

  It indicates whether the **cell** is **reserved**, as defined in TS 38.304. It is **applicable** to **all PLMNs**. It is **ignored** by **IAB-MT** for **cell barring** determination, **but still considered** by **NPN capable IAB-MT** for determination of an **NPN-only cell**.

- `npn-IdentityInfoList-r16` `NPN-IdentityInfoList-r16`: optional, common

  Its **data type** `NPN-IdentityInfoList-r16` is a **`SEQUENCE (SIZE (1..maxNPN-r16)) OF`** `NPN-IdentityInfo-r16`, where `maxNPN-r16` is `12` which is the **maximum quantity** of **non-public networks** (**NPNs**) **broadcast** and **reported** by each **UE** at **establishment**.

  **Each** of these **nameless** parameters (that has **data type** `NPN-IdentityInfo-r16`) configures a **list** of `NPN-Identity-r16` (i.e., the **identity** of **1 NPN**) and **additional information** associated to **each such list**.

  A given **value** of `NPN-Identity-r16` (i.e., the **identity** of **either 1 specific PNI-NPN** Mobile communications networks or **1 specific SNPN**) can be **included only once** within a given **instance** of `NPN-IdentityInfoList-r16`.

  In case of **NPN-only cells**, `plmnIdentityList` from `PLMN-IdentityInfo` (click 3.6.5) contains a **single parameter** (that has data type `PLMN-Identity`) which does **not count** for the **limit** of `12` and the `cellIdentity` from `PLMN-IdentityInfo` (click 3.6.5) of the **first entry** of `plmn-IdentityInfoList` (see above) is set to the same value as the `cellIdentity-r16` from `NPN-IdentityInfo-r16` of the **first entry** of `npn-IdentityInfoList-r16` (see above).

  The **NPN index** is defined as "$B+c_1+c_2+\ldots+c_{n-1}+d_1+d_2+\ldots+d_{m-1}+e(i)$" for the **NPN identity** included in the **n-th instance** of `NPN-IdentityInfoList` and in the **m-th instance** of `npn-Identitylist` within that `NPN-IdentityInfoList` instance, and the **i-th instance** of its corresponding `NPN-Identity-r16`, where:

  - "$B$" is the index used for the **last PLMN** in the `PLMN-IdentityInfoList`; in **NPN-only cells** $B$ is considered `0`.
  - "$c_j$" is the quantity of **NPN index** values used in the **j-th** entry of `NPN-IdentityInfoList-r16` entry.
  - "$d_k$" is the quantity of **NPN index** values used in the **k-th** `npn-IdentityList` entry within the **n-th** `NPN-IdentityInfoList-r16` entry.
  - "**e(i)**" is:
    - **i** if the **n-th** entry of `NPN-IdentityInfoList-r16` entry is for **SNPN(s)**.
    - **1** if the **n-th** entry of `NPN-IdentityInfoList-r16` entry is for **PNI-NPN(s)**.

- `plmn-IdentityInfoList` `PLMN-IdentityInfoList` (click 3.6.5): mandatory, common

  Its **data type** `PLMN-IdentityInfoList` is a **`SEQUENCE (SIZE (1..maxPLMN)) OF`** `PLMN-IdentityInfo` (click 3.6.5), where `maxPLMN` is `12`, which is the **maximum quantity** of **PLMN broadcast** and **reported** by **each UE** at **establishment**.

**Each** of these **nameless** parameters (that has **data type** `PLMN-IdentityInfo`) configures a **list** of `PLMN-Identity` (i.e., the **identity** of **1 PLMN**) and **additional information** associated to **each such list**.

A given **value** of `PLMN-Identity` (i.e., the **identity** of **1 specific PLMN**) can be **included only once** within a given **instance** of `PLMN-IdentityInfoList`.

The **PLMN index** is defined as "$b_1+b_2+\ldots+b_{n-1}+i$" for the **i-th instance** of `PLMN-Identity` **within** the **n-th instance** of `PLMN-IdentityInfo` **within** `PLMN-IdentityInfoList`. **Each** quantity "$b_j$" is the **total quantity** of **instances** of `PLMN-Identity` within the **j-th instance** of `PLMN-IdentityInfo` **within** `PLMN-IdentityInfoList`.

- `snpn-AccessInfoList-r17 SEQUENCE (SIZE (1..maxNPN-r16)) OF SNPN-AccessInfo-r17`: optional, common

  **Each** of its **nameless** parameters with data type `SNPN-AccessInfo-r17` contains the **access related information** of the **SNPN** of the same **sequential position** within `npn-IdentityInfoList-r16` (see above).

  Constant `maxNPN-r16` is `12` which is the **maximum quantity** of **non-public networks** (**NPNs**) **broadcast** and **reported** by **each UE** at **establishment**.

  For more detail on `snpn-AccessInfoList-r17`, refer to TS 23.501.

The **total quantity** of **PLMNs** (identified by an instance of `PLMN-Identity` within `plmn-IdentityList`, see above), plus **PNI-NPNs** (each identified by a `PLMN-Identity` instance and a `cag-Identity-r16` from `CAG-IdentityInfo-r16` (click 3.6.7)) plus **SNPNs** (each identified by a `PLMN-Identity` instance and a `NID-r16` instance (click 3.6.7)) **jointly contained** in `plmn-IdentityInfoList` (see above) and in `npn-IdentityInfoList-r16` (see above) **cannot exceed** `12`, except for **NPN-only cells**.

For more detail on these parameters, refer to the table "*CellAccessRelatedInfo field descriptions*" of TS 38.331.

### 3.6.4.2 `CellAccessRelatedInfo` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
CellAccessRelatedInfo ::= SEQUENCE {
  plmn-IdentityInfoList PLMN-IdentityInfoList,
  cellReservedForOtherUse ENUMERATED {true} OPTIONAL,          -- Need R
  ...,
  [[
  cellReservedForFutureUse-r16 ENUMERATED {true} OPTIONAL,     -- Need R
  npn-IdentityInfoList-r16 NPN-IdentityInfoList-r16 OPTIONAL   -- Need R
  ]],
  [[
  snpn-AccessInfoList-r17 SEQUENCE (SIZE (1..maxNPN-r16)) OF
              SNPN-AccessInfo-r17 OPTIONAL                     -- Need R
  ]]
}
```

### 3.6.5 PLMN-IdentityInfoList and PLMN-IdentityInfo data types

They can be **broadcast** (or **unicast**) within SIB1 (click 5.3.4), **unicast** within the **Non-SI parameter group** (click 5.3.6) and **sent** within an **RRC inter-node message** (click 3.6.3).

For their **ancestor tree up to** SIB1, click 3.6.1.

For their **ancestor tree up to** CellGroupConfig (click 3.7.3), click 3.6.2.

**Each** of their **instances** contains **either all common parameters or all dedicated parameters** (click 5.1.3), depending on where it is **instantiated**.

PLMN-IdentityInfoList is a SEQUENCE (SIZE (1..maxPLMN)) OF PLMN-IdentityInfo, where maxPLMN is 12, which is the **maximum quantity** of **PLMNs broadcast** and **reported** by **each UE** at **establishment**. **Each** of these **nameless** parameters configures a **list** of PLMN-Identity (click 3.6.6) (i.e., the **identity** of **1 PLMN**) and **additional information** associated to **each such list**. A given **value** of PLMN-Identity can be **included only once** within a given **instance** of PLMN-IdentityInfoList.

#### 3.6.5.1 PLMN-IdentityInfo parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- cellIdentity CellIdentity: mandatory, common **or** dedicated

  It does **not** have a **field description** in the table "*PLMN-IdentityInfo field descriptions*" of TS 38.331, but its **same-name parameter** from CGI-Info-Logging has a field description.

  It is then **assumed** that it **unambiguously identifies** a cell within a given **PLMN**. It is the **new radio cell identity** (**NCI**, click 3.5.3) of the **cell**. Its **data type** CellIdentity is a BIT STRING (SIZE (36)), this is, a **bit string** of 36 bits.

  Recall from 3.4 that the **NCGI** of the **cell** within a **given PLMN** is determined by the **PLMN identity** of that **PLMN** and the **cells**'s **NCI within** that **PLMN**. Recall also that **each cell** may have **more than one NGCI**, in case the **cell** belongs to **more than one PLMN**.

  The **PLMN identity** to determine the **cell**'s **NCGI** is conveyed in the **first** PLMN-Identity from the **list** contained in plmn-IdentityList (see above). The cell's **NCI** within that **PLMN** is conveyed in this parameter.

- cellReservedForOperatorUse ENUMERATED {reserved, notReserved}: mandatory, common **or** dedicated

  It indicates whether the **cell** is **reserved** for **operator use** (**per PLMN**), as defined in TS 38.304. This parameter is **ignored** by **IAB-MT**.

- gNB-ID-Length-r17 INTEGER (22..32): optional, common **or** dedicated

  It is the **bit length** of the **NR node Identity** (**NR node ID**, click 3.5.2) out of the **36-bit** long cellIdentity (see above).

- `iab-Support-r16 ENUMERATED {true}`: optional, common **or** dedicated

  It indicates **both** the **support** of **integrated access and backhaul** (**IAB**) **and** the **cell status** for **IAB**. If it is **present**, the **cell supports IAB and** the **cell** is also considered as a **candidate** for cell (re)selection for **IAB-node**. If it is **absent**, the **cell** does **not support IAB and/or** the **cell** is **barred** for **IAB-node**.

- `plmn-IdentityList SEQUENCE (SIZE (1..maxPLMN)) OF PLMN-Identity` (click 3.6.6): mandatory, common **or** dedicated

  It does **not** have a **field description** in TS 38.331, but there is some information about it in the **running text**.

  It is a list of **PLMN identities** (click 3.5.1). The constant `maxPLMN` is `12`, which is the **maximum quantity** of **PLMNs broadcast** and **reported** by **each UE** at **establishment**.

  Recall from 3.4 that the **NCGI** of the **cell** within a **given PLMN** is determined by the **PLMN identity** of that **PLMN** and the **cells**'s **NCI within** that **PLMN**. Recall also that **each cell** may have **more than one NGCI**, in case the **cell** belongs to **more than one PLMN**.

  The **PLMN identity** to determine the **cell**'s **NCGI** is conveyed in the **first** `PLMN-Identity` from the **list** contained in this parameter. The cell's **NCI** within that **PLMN** is conveyed in parameter `cellIdentity` (see below).

- `trackingAreaCode TrackingAreaCode`: optional, common **or** dedicated

  It is the **tracking area code** to which the **cell** identified by `cellIdentity` belongs.

  If it is **absent**, this indicates that the **cell only** supports **PSCell/SCell** functionality (**per PLMN**) or it is a **non-terrestrial network** (**NTN**) **cell**. Its data type `TrackingAreaCode` is a `BIT STRING (SIZE (24))`.

For more detail on these parameters, refer to the table "*PLMN-IdentityInfo field descriptions*" of TS 38.331.

### 3.6.5.2 `PLMN-IdentityInfoList` and `PLMN-IdentityInfo` ASN.1 specifications

The **complete** specification of `PLMN-IdentityInfoList` is:

```
PLMN-IdentityInfoList ::= SEQUENCE (SIZE (1..maxPLMN)) OF PLMN-IdentityInfo
```

where the `maxPLMN` is `12`, which is the **maximum** quantity of **PLMNs** broadcast and reported by **each UE** at establishment.

The **complete** specification of `PLMN-IdentityInfo` is:

```
PLMN-IdentityInfo ::= SEQUENCE {
  plmn-IdentityList SEQUENCE (SIZE (1..maxPLMN)) OF PLMN-Identity,
  trackingAreaCode TrackingAreaCode OPTIONAL,                    -- Need R
  ranac RAN-AreaCode OPTIONAL,                                   -- Need R
  cellIdentity CellIdentity,
  cellReservedForOperatorUse ENUMERATED {reserved, notReserved},
  ...,
```

```
    [[
    iab-Support-r16 ENUMERATED {true} OPTIONAL                    -- Need S
    ]],
    [[
    trackingAreaList-r17 SEQUENCE (SIZE (1..maxTAC-r17))
                         OF TrackingAreaCode OPTIONAL,            -- Need R
    gNB-ID-Length-r17 INTEGER (22..32) OPTIONAL                   -- Need R
    ]]
}
```

where **maxPLMN** is **12**, which is the **maximum quantity** of **PLMNs** broadcast and reported by **each UE** at establishment.

### 3.6.6 PLMN-Identity data type

It can be **broadcast** (or **unicast**) within **SIB1** (click 5.3.4), **unicast** within the **Non-SI parameter group** (click 5.3.6) and **sent** within **RRC inter-node messages** (click 3.6.3).

For its **ancestor tree up to SIB1**, click 3.6.1.

For its **ancestor tree up to CellGroupConfig** (click 3.7.3), click 3.6.2.

For its **ancestor tree up to** the **RRC inter-node messages** (click 3.6.3).

**Each** of its **instances** contains **either all common parameters or all dedicated parameters** (click 5.1.3), depending on where it is **instantiated**.

**Each** of its **instances** contains one **public land mobile network** (**PLMN**) **identity** (click 3.5.1).

Its **complete ASN.1** (click 5.9) specification is:

```
PLMN-Identity ::= SEQUENCE {
  mcc MCC OPTIONAL,                                               -- Cond MCC
  mnc MNC
}
```

where

```
MCC ::= SEQUENCE (SIZE (3))     OF MCC-MNC-Digit
MNC ::= SEQUENCE (SIZE (2..3)) OF MCC-MNC-Digit
MCC-MNC-Digit ::= INTEGER (0..9)
```

### 3.6.7 NPN-Identity-r16 data type

It can be **broadcast** (or **unicast**) within **SIB1** (click 5.3.4), **unicast** within the **Non-SI parameter group** (click 5.3.6) and **sent** within **RRC inter-node messages** (click 3.6.3).

For its **ancestor tree up to SIB1**, click 3.6.1.

For its **ancestor tree up to CellGroupConfig** (click 3.7.3), click 3.6.2.

For its **ancestor tree up to** the **RRC inter-node messages** (click 3.6.3).

**Each** of its **instances** contains **either all common parameters or all dedicated parameters** (click 5.1.3), depending on where it is **instantiated**.

**Each** of its **instances** contains one **non-public network** (**NPN**) **identity**.

Its **complete ASN.1** (click 5.9) specification is:

```
NPN-Identity-r16 ::= CHOICE {
  pni-npn-r16 SEQUENCE {
    plmn-Identity-r16 PLMN-Identity,
    cag-IdentityList-r16 SEQUENCE (SIZE (1..maxNPN-r16))
                         OF CAG-IdentityInfo-r16
  },
  snpn-r16 SEQUENCE {
    plmn-Identity-r16 PLMN-Identity,
    nid-List-r16 SEQUENCE (SIZE (1..maxNPN-r16)) OF NID-r16
  }
}
```

where `maxNPN-r16` is `12`, which is the **maximum quantity** of **NPNs** broadcast and reported by **each UE** at establishment, and:

```
CAG-IdentityInfo-r16 ::= SEQUENCE {
  cag-Identity-r16 BIT STRING (SIZE (32)),
  manualCAGselectionAllowed-r16 ENUMERATED {true} OPTIONAL      -- Need R
}

NID-r16 ::= BIT STRING (SIZE (44))
```

Notice that **CAG** stands for "**Closed Access Group**".

## 3.7 Cell group parameters

**Each** of the **UE**'s **cell groups** is configured with:

- **Common and dedicated parameters** that are **unicast** within the **Non-SI parameter group**, as described in section 3.7.1.
- **Common and dedicated parameters sent** in **RRC inter-node messages**, as described in section 3.7.2.

**Cell group** configuration does **not** have **any parameter** within `MIB` (click 5.3.3), **nor** within `SIB1` (click 5.3.4).

Possibly existing **cell group** parameters within **other SIBs** and **posSIBs** (click 5.3.5) are **not addressed** in this book.

The configuration of the **master cell group** is done with **either** of the three optional **same-name parameters** `masterCellGroup` from `RRCSetup-IEs` (click 5.8.5.11), from `RRCResume-IEs` or from `RRCReconfiguration-v1530-IEs`. In case **dual connectivity** is used, the configuration of the **secondary cell group** is done with the optional parameter `secondaryCellGroup` from `RRCReconfiguration-IEs` (click 5.8.5.4). The **data type** of **these four** parameters is `CellGroupConfig` (click 3.7.3).

**Each cell group** is **specific** of **each UE**, according to its specific **capability**, **traffic** demand and other **characteristics**. Regarding the **configuration** of **each cell** within the **cell group**, **some parameters** are **common** and **others** are **dedicated**.

For the related configuration **parameters** for **one individual serving cell**, click 3.9.

For an overall explanation on **UE configuration parameters** click 5. For a brief overview of **ASN.1**, click 5.9.

### 3.7.1 Cell group parameters within the Non-SI parameter group

A **cell group** is configured with **parameters unicast** within the **Non-SI parameter group** (click 5.3.6) in every case, except in **handover** and **other inter-node procedures**.

For **cell group** parameters within the **Non-SI parameter group**, the **most relevant related data types** are:

- `CellGroupConfig` (click 3.7.3): common **and** dedicated parameters
- `MAC-CellGroupConfig` (click 3.7.4): dedicated parameters
- `PhysicalCellGroupConfig` (click 3.7.5): common **and** dedicated parameters

The **ancestor trees** (click 5.9.8) of these **most relevant related data types up to** the **top** are:

```
                                DL-CCCH-Message (click 5.8.5.3)
message                           DL-CCCH-MessageType (click 5.8.5.3)
  c1                                CHOICE
    rrcSetup                          RRCSetup (0,1) (click 5.8.5.11)
      criticalExtensions                RRCSetup-IEs
        masterCellGroup                   CellGroupConfig (click 3.7.3)
          mac-CellGroupConfig               MAC-CellGroupConfig (click 3.7.4)
          physicalCellGroupConfig           PhysicalCellGroupConfig (click 3.7.5)

                                DL-DCCH-Message (click 5.8.5.4)
message                           DL-DCCH-MessageType (click 5.8.5.4)
  c1                                CHOICE
    rrcReconfiguration                RRCReconfiguration (0,1) (click 5.8.5.4)
      rrcReconfiguration                RRCReconfiguration-IEs (click 5.8.5.4)
        secondaryCellGroup                CellGroupConfig (click 3.7.3)
        nonCriticalExtension              RRCReconfiguration-v1530-IEs (0,1)
          masterCellGroup                   CellGroupConfig (click 3.7.3)
            mac-CellGroupConfig               MAC-CellGroupConfig (click 3.7.4)
            physicalCellGroupConfig           PhysicalCellGroupConfig (click 3.7.5)
    rrcResume                         RRCResume (0,1) (click 5.8.5.4)
      rrcResume                         RRCResume-IEs
        masterCellGroup                   CellGroupConfig (click 3.7.3)
```

Some **relevant remarks** on these **ancestor trees** are:

- `CellGroupConfig` is included in **3 RRC NR messages**: `RRCReconfiguration`, `RRCResume` and `RRCSetup`. **Each** of these **NR RRC messages** may be used to **(re)configure** a **UE** via **dedicated signaling**.
- `CellGroupConfig` is **always** referred to within the **ASN.1** statement "**`OCTET STRING (CONTAINING CellGroupConfig)`**".
- The **root data types** are the **NR RRC message classes** `DL-DCCH-Message` and `DL-CCCH-Message`. In addition to these, there may be other **roots** arising from the first bullet point. Notice that the **top data type never** has a **field name**.

Be aware that the **ASN.1** statement "**`OCTET STRING (CONTAINING RRCReconfiguration)`**" may be found in **7 RRC data types**. This means that `RRCReconfiguration` (click 5.8.5.4) is the **son** of the said **seven data types**, and therefore, there can be **other ancestor trees** in addition to the **3 ones** shown above.

Recall that **Sidelink** channels (and consequently **Sidelink data types** and parameters) are **not** covered in this book.

### 3.7.2 Cell group parameters sent in RRC inter-node messages

A **cell group** is configured with **parameters** that are **sent** in **RRC inter-node messages** (click 5.8.6) in **handover** and in **other inter-node procedures**.

For **cell group** parameters **sent** in **RRC inter-node messages**, the **most relevant related data types** are:

- `CellGroupConfig` (click 3.7.3): common **and** dedicated parameters
- `MAC-CellGroupConfig` (click 3.7.4): dedicated parameters
- `PhysicalCellGroupConfig` (click 3.7.5): common **and** dedicated parameters

The **ancestor tree** (click 5.9.8) of these **most relevant related data types** up to the `HandoverCommand` **RRC inter-node message** (click 5.8.6) is:

```
                                HandoverCommand (click 5.8.6)
handoverCommand                   HandoverCommand-IEs
  handoverCommandMessage            RRCReconfiguration (0,1) (click 5.8.5.4)
    rrcReconfiguration                RRCReconfiguration-IEs (click 5.8.5.4)
      secondaryCellGroup                CellGroupConfig (click 3.7.3)
        mac-CellGroupConfig               MAC-CellGroupConfig (click 3.7.4)
        physicalCellGroupConfig           PhysicalCellGroupConfig (click 3.7.5)
      nonCriticalExtension              RRCReconfiguration-v1530-IEs (0,1)
        masterCellGroup                   CellGroupConfig (click 3.7.3)
          mac-CellGroupConfig               MAC-CellGroupConfig (click 3.7.4)
          physicalCellGroupConfig           PhysicalCellGroupConfig (click 3.7.5)
```

Some **relevant remarks** on this **ancestor tree** are:

- Be aware that the **ASN.1** statement "`OCTET STRING (CONTAINING RRCReconfiguration)`" may be found in **seven RRC data types**. This means that `RRCReconfiguration` (click 5.8.5.4) is the **son** of the said **seven data types**. However, the corresponding **ancestor trees** have **not** been **shown** for conciseness's sake.
- The **root data type** is the **RRC inter-node message** `Handover-Command`. In addition to it, there may be other **roots** arising from the first bullet point. Notice that the **top data type never** has a **field name**.

Recall that **Sidelink** channels (and consequently **Sidelink data types** and parameters) are **not** covered in this book.

### 3.7.3 `CellGroupConfig` data type

It can be **both unicast** within the **Non-SI parameter group** (click 5.3.6) and **sent** within **RRC inter-node messages** (click 3.6.3).

For its **ancestor tree up to** the **RRC inter-node messages**, click 3.7.2.

It contains **both common parameters** and **dedicated parameters** (click 5.1.3).

**Each** of its **instances** configures a **UE** with a **master cell group** (**MCG**) or a **secondary cell group** (**SCG**). A **cell group** comprises of **one MAC entity**, a **set** of **logical channels** with associated **RLC** entities, a **special cell** (**SpCell**) and **one or more secondary cells** (**SCells**).

**Each** of its **instances** provide the **UE** with the **parameaters** for **most** of the **physical layer elements**, for **all** its **bandwidth parts** (**initial** and/or **additional**), for **all** its **carrier types** (**Downlink**, **Uplink** and/or **SUL**), for **common** and **dedicated configuration**, for

**all** its **cells** (**special** and/or **secondary**), of its corresponding **cell group** (**master** or **secondary**). The "**corresponding**" elements are determined by the **specific place**, within the overall **ASN.1 specification**, where this **data type** has been **instantiated**.

It is **always** referred to within the **ASN.1** statement "`OCTET STRING (CONTAINING CellGroupConfig)`".

### 3.7.3.1 `CellGroupConfig` parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `cellGroupId` `CellGroupId`: mandatory, dedicated

  It is the **identifier** of this **cell group**.

  Its **data type** `CellGroupId` is an `INTEGER (0..maxSecondaryCellGroups)`. The value of `maxSecondaryCellGroups` is `3`. It allows to **identify one master cell group plus up to 3 secondary cell groups**. Even though this **identifier** allows to distinguish **among 4 cell groups**, the **RRC ASN.1 specification only** allows for **1 master cell group** plus **1 secondary cell group** You may check this in the **ancestor trees** from 3.7.1 and 3.7.2.

- `mac-CellGroupConfig` `MAC-CellGroupConfig` (click 3.7.4): optional, dedicated

  It contains the **MAC parameters applicable** to **all** the **cells** in the corresponding **entire cell group**.

- `physicalCellGroupConfig` `PhysicalCellGroupConfig` (click 3.7.5): optional, common

  It does **not** have a **field description** in TS 38.331. The literal description of its **data type** is that it "*is used to configure cell-group specific L1 parameters*".

  Its configuration parameters **apply** to **all** the **cells** in the corresponding **cell group**.

- `spCellConfig` `SpCellConfig` (click 3.9.5): optional, common **and** dedicated

  It contains the **parameters** for the **special cell** of **this cell group**. Recall that a **special cell** is a **primary cell** of a **master cell group** or primary-secondary cell of a secondary cell group)

  The description of its **data type** is that it provides the parameters for the **special cell** of the corresponding **cell group** (i.e., for the **primary cell** of a **master cell group** or for the **primary-secondary cell** of a **secondary cell group**).

- `sCellToAddModList` `SEQUENCE (SIZE (1..maxNrofSCells)) OF` `SCellConfig` (click 3.9.4): optional, common **and** dedicated

  It **adds** or **modifies up to** `31` **secondary cells** to the **cell group**, providing the **configuration** for **each** of them. **Secondary cells** may be **dynamically added**, **modified** or **removed** to **each** **UE** as required by **carrier aggregation** (click 10.2).

  The constant `maxNrofSCells` is `31`, which is the **maximum quantity** of **secondary cells** in **each** of the **UE**'s **cell groups**.

- `sCellToReleaseList` **SEQUENCE (SIZE (1..maxNrofSCells)) OF** `SCellIndex` (click 3.9.9): optional, common **and** dedicated

  It **removes up to** `31` **secondary cells** from the **cell group**. **Secondary cells** may be **dynamically added**, **modified** or **removed** to **each UE** as required by **carrier aggregation** (click 10.2).

  The constant `maxNrofSCells` is `31`, which is the **maximum quantity** of **secondary cells** in **each** of the **UE**'s **cell groups**.

  The data type `SCellIndex` (click 3.9.9) is an **INTEGER (1..31)**.

For more detail on these parameters, refer to the table "*CellGroupConfig field descriptions*" of TS 38.331.

### 3.7.3.2 `CellGroupConfig` ASN.1 specification

**Part** of its **ASN.1** (click 5.9) specification is:

```
CellGroupConfig ::= SEQUENCE {
  cellGroupId CellGroupId,
  other parameters (not shown)
  mac-CellGroupConfig MAC-CellGroupConfig OPTIONAL,              -- Need M
  physicalCellGroupConfig PhysicalCellGroupConfig OPTIONAL,      -- Need M
  spCellConfig SpCellConfig OPTIONAL,                            -- Need M
  sCellToAddModList SEQUENCE (SIZE (1..maxNrofSCells)) OF SCellConfig
                    OPTIONAL,                                    -- Need N
  sCellToReleaseList SEQUENCE (SIZE (1..maxNrofSCells)) OF SCellIndex
                     OPTIONAL,                                   -- Need N
  other parameters (not shown)
}
```

The value of `maxNrofSCells` is `31`, which is the **maximum quantity** of **secondary cells** in **each** of the **UE**'s **cell groups**.

## 3.7.4 `MAC-CellGroupConfig` data type

It can be **both unicast** within the **Non-SI parameter group** (click 5.3.6) and **sent** within **RRC inter-node messages** (click 3.6.3).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 3.7.1.

For its **ancestor tree up to** the **RRC inter-node messages**, click 3.7.2.

Its **only parent data type** is `CellGroupConfig` (click 3.7.3).

**Each** of its **instances** configures a **UE** with the **MAC layer parameters** corresponding to **all** the **cells** of **one** of its **cell groups** (**either** the **UE**'s **master cell group** or the **UE**'s **secondary cell group**).

**Each** of its **instances** is a **large data type** (its **ASN.1 specification** has **45 lines**), containing many different **parameters**.

It contains **groups of parameters** to configure the **UE** with the **scheduling** of **PUCCH scheduling resources**, **skip Uplink transmission**, **beam failure recovery**, **discontinuous reception** (**DRX**), several **RNTI values** and **other MAC elements**.

#### 3.7.4.1 `MAC-CellGroupConfig` parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `schedulingRequestConfig` `SchedulingRequestConfig` (click 7.9.14): optional, dedicated

  It contains a **list** of **instances** of `SchedulingRequestToAddMod` (click 7.9.15) to be **added/removed/modified** to/from/in the **dedicated configuration** (for **all** the **cells** of **one** of the **UE**'s **cell groups**) of the **PUCCH scheduling requests**.

For more detail on these parameters, refer to the table "*MAC-CellGroupConfig field descriptions*" of TS 38.331.

#### 3.7.4.2 `MAC-CellGroupConfig` ASN.1 specification

**Part** of its **ASN.1** (click 5.9) specification is:

```
MAC-CellGroupConfig ::= SEQUENCE {
  Other parameters (not shown)
  schedulingRequestConfig SchedulingRequestConfig OPTIONAL,          -- Need M
  Other parameters (not shown)
  schedulingRequestID-LBT-SCell-r16 SchedulingRequestId OPTIONAL,    -- Need R
  lch-BasedPrioritization-r16 ENUMERATED {enabled} OPTIONAL,         -- Need R
  schedulingRequestID-BFR-SCell-r16 SchedulingRequestId OPTIONAL,    -- Need R
  Other parameters (not shown)
  schedulingRequestID-BFR-r17  SchedulingRequestId OPTIONAL,         -- Need R
  schedulingRequestID-BFR2-r17 SchedulingRequestId OPTIONAL,         -- Need R
  schedulingRequestConfig-v1700 SchedulingRequestConfig-v1700
                                OPTIONAL,                            -- Need M
  Other parameters (not shown)
}
```

### 3.7.5 `PhysicalCellGroupConfig` data type

It can be **both unicast** within the **Non-SI parameter group** (click 5.3.6) and **sent** within **RRC inter-node messages** (click 3.6.3).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 3.7.1.

For its **ancestor tree up to** the **RRC inter-node messages**, click 3.7.2.

Its **only parent data type** is `CellGroupConfig` (click 3.7.3).

It contains **both common parameters** and **dedicated parameters** (click 5.1.3).

**Each** of its **instances** configures a **UE** with the **physical layer parameters** corresponding to **all** the **cells** of **one** of its **cell groups** (**either** the **UE**'s **master cell group** or the **UE**'s **secondary cell group**).

**Each** of its **instances** contains **groups** of **parameters** to configure the **UE** with **enhanced HC-Type 3 HARQ-ACK codebook feedback**, **triggering** of **HARQ-ACK re-transmission** on a **PUCCH resource**, **PUCCH Cell switching**, several **RNTI values** and **other physical layer elements**.

It is a **very large data type** (its **ASN.1 specification** has **90 lines**), containing many different **parameters**.

#### 3.7.5.1 `PhysicalCellGroupConfig` parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `ackNackFeedbackMode-r16 ENUMERATED {joint, separate}`: optional, dedicated

  It indicates which among the **joint** and **separate ACK/NACK feedback modes** to use **within** a **slot** (click 4.5.6.7 and refer to clause 9 of TS 38.213).

- `uci-MuxWithDiffPrio-r17 ENUMERATED {enabled}`: optional, dedicated

  When present, it **sets UCI reporting of different priorities** (click 7.6.8.2).

For more detail on these parameters, refer to the table "*PhysicalCellGroupConfig field descriptions*" of TS 38.331.

#### 3.7.5.2 `PhysicalCellGroupConfig` ASN.1 specification

**Part** of its **ASN.1** (click 5.9) specification is:

```
PhysicalCellGroupConfig ::= SEQUENCE {
  harq-ACK-SpatialBundlingPUCCH ENUMERATED {true} OPTIONAL,          -- Need S
  harq-ACK-SpatialBundlingPUSCH ENUMERATED {true} OPTIONAL,          -- Need S
  p-NR-FR1 P-Max OPTIONAL,                                           -- Need R
  pdsch-HARQ-ACK-Codebook ENUMERATED {semiStatic, dynamic},
  Other parameters (not shown)
  ackNackFeedbackMode-r16 ENUMERATED {joint, separate} OPTIONAL,     -- Need R
  Other parameters (not shown)
  uci-MuxWithDiffPrio-r17 ENUMERATED {enabled} OPTIONAL,             -- Need R
  Other parameters (not shown)
}
```

## 3.8 Cell identification, types, UE configuration and quantity

**Cell identification**

For the **purpose** of **UE**'s **physical layer configuration**, **each cell** is **identified** with the **Physical Cell Identity** (**PCI**, click 3.5.3.2), which is **broadcast** in **SSB**. The **PCI** uniquely identifies **each cell** in respect to **all** its **neighboring cells** (within the operator's network).

The **PCI** is also contained in **each instance** of the `PhysCellId` **data type**, which is an `INTEGER (0..1007)`.

For more detail on **cell identities**, click 3.5.3.

**Cell types**

For the **purpose** of **UE physical layer configuration**, there may be **primary**, **primary-secondary** and **secondary cells**.

For more detail on **cell types**, click 3.2.

**Cell UE configuration and quantity**

**Each UE** is configured with the following **cell types** and **quantities**:

- **1 primary cell** (**PCell**)

  It is the **cell** where the **UE** performs its **initial access** (click 8).

**After initial access**, it is the **cell** whose **PCI** is contained in `physCellId` that has the following **ancestor tree**:

```
masterCellGroup                 CellGroupConfig (click 3.7.3)
  spCellConfig                    SpCellConfig (0,1) (click 3.9.5)
    reconfigurationWithSync         ReconfigurationWithSync (0,1) (click 3.9.8)
      spCellConfigCommon              ServingCellConfigCommon (0,1) (click 3.9.6)
        physCellId                      PhysCellId
```

The **UE**'s **primary cell** is configured with `spCellConfig` that has the following **ancestor tree**:

```
masterCellGroup                 CellGroupConfig (click 3.7.3)
  spCellConfig                    SpCellConfig (0,1) (click 3.9.5)
```

- **Up to 1 primary-secondary cell (PSCell)**

  It is the **primary cell** of the **Secondary Cell Group** (click 3.2.2).

  The **UE** is configured with a **primary-secondary cell only** in case of **dual connectivity** (click 10.3).

  It is the **cell** whose **PCI** is contained in `physCellId` that has the following **ancestor tree**:

```
secondaryCellGroup              CellGroupConfig (click 3.7.3)
  spCellConfig                    SpCellConfig (0,1) (click 3.9.5)
    reconfigurationWithSync         ReconfigurationWithSync (0,1) (click 3.9.8)
      spCellConfigCommon              ServingCellConfigCommon (0,1) (click 3.9.6)
        physCellId                      PhysCellId
```

  The **UE**'s **primary-secondary cell** is configured with `spCellConfig` that has the following **ancestor tree**:

```
secondaryCellGroup              CellGroupConfig (click 3.7.3)
  spCellConfig                    SpCellConfig (0,1) (click 3.9.5)
```

- **Up to `31` or `62` secondary cells (SCells)**

  The **UE** is configured with **secondary cell(s) only** in case of **carrier aggregation** (click 10.2). The **maximum quantity** is `31` if **dual connectivity** (click 10.3) is **not** used, and it is `62` otherwise.

  The **secondary cells** in **each** of the **UE**'s **cell groups** (click 3.2.2) are configured with `sCellToAddModList`. Its **ancestor trees** for the **Master** and **Secondary Cell Groups** (click 3.2.2) are (respectively):

```
masterCellGroup                 CellGroupConfig (click 3.7.3)
  sCellToAddModList               SEQUENCE

secondaryCellGroup              CellGroupConfig (click 3.7.3)
  sCellToAddModList               SEQUENCE
```

  The **data type** of `sCellToAddModList` is a `SEQUENCE (SIZE (1..maxNrofSCells)) OF SCellConfig`. The value of `maxNrofSCells` is `31`, which is the **maximum quantity** of **secondary cells** in **each cell group**.

  **Each** of the **UE**'s **secondary cells** is configured with **one instance** of `SCellConfig` (click 3.9.4).

Notice that the **maximum quantity** of **physical layer elements** (**cells**, **bandwidth parts**, **CORESETs**, **search spaces**, etc) that can be **configured** to **each UE** is **not** the same as the **maximum quantity** that a **gNB** can handle.

## 3.9 Cell parameters

**Each** of the **UE**'s **cells** is configured with:

- **Common parameters** that are **broadcast** (or **unicast**) within `SIB1` (click 5.3.4), as described in section 3.9.1.
- **Common** and **dedicated parameters** that are **unicast** within the **Non-SI parameter group** (click 5.3.6), as described in section 3.9.2.

**Cell** configuration does **not** have **any** specific **parameter** within `MIB` (click 5.3.3).

Possibly existing **cell parameters** within **other SIBs** and **posSIBs** (click 5.3.5) are **not addressed** in this book.

For the related configuration **parameters** for a **cell group**, click 3.7.

For an overall explanation on **UE configuration parameters** click 5. For a brief overview of **ASN.1**, click 5.9.

### 3.9.1 Serving cell parameters within `SIB1`

**During initial access**, the **primary cell** is configured with **common parameters** that are **broadcast** (or **unicast**) within `SIB1` (click 5.3.4).

For **serving cell** parameters within `SIB1`, the **most relevant related data type** is:

- `ServingCellConfigCommonSIB`: common parameters

The **ancestor tree** (click 5.9.8) of `ServingCellConfigCommonSIB` up to `SIB1` is:

```
                          SIB1
servingCellConfigCommon     ServingCellConfigCommonSIB (0,1)
```

For the **ancestor trees** of `SIB1` **up to** the **top** click 5.3.4.

### 3.9.2 Serving cell parameters within the Non-SI parameter group

**Each serving cell** is configured with **common** and **dedicated parameters** that are **unicast** within the **Non-SI parameter group** (click 5.3.6) in the following cases:

- **The primary-secondary cell**.
- **Each secondary cell**.
- **The primary cell**, if configuration changes are done **after initial access**.

For **serving cell** parameters within the **Non-SI parameter group**, the **most relevant related data types** are:

- `SCellConfig` (click 3.9.4): common **and** dedicated parameters
- `SpCellConfig` (click 3.9.5): common **and** dedicated parameters
- `ServingCellConfigCommon` (click 3.9.6): common parameters
- `ServingCellConfig` (click 3.9.7): common **and** dedicated parameters

The **ancestor tree** (click 5.9.8) of these **most relevant related data types** up to `CellGroupConfig` is:

```
                                  CellGroupConfig (click 3.7.3)
spCellConfig                        SpCellConfig (0,1) (click 3.9.5)
  servCellIndex                       ServCellIndex (0,1) (click 3.9.9)
  reconfigurationWithSync             ReconfigurationWithSync (0,1) (click 3.9.8)
    spCellConfigCommon                  ServingCellConfigCommon (0,1) (click 3.9.6)
  spCellConfigDedicated               ServingCellConfig (0,1) (click 3.9.7)
sCellToAddModList                   SCellConfig (0..maxNrofSCells) (click 3.9.4)
  sCellIndex                          SCellIndex (click 3.9.9)
  sCellConfigCommon                   ServingCellConfigCommon (0,1) (click 3.9.6)
  sCellConfigDedicated                ServingCellConfig (0,1) (click 3.9.7)
```

Some **relevant remarks** on this **ancestor tree** are:

- For the **ancestor trees** of `CellGroupConfig`, **up to** some **RRC NR messages** click 3.7.1 and **up to** some **RRC inter-node messages** click 3.7.2.
- The constant `maxNrofSCells` is `31`, which is the **maximum quantity** of **secondary cells** in **each** of the **UE**'s **cell groups**.
- Notice that there is one **asymmetry** between the **configuration structure** of the **special cell** and of **each secondary cell**. This **asymmetry** is that the **common configuration** structure of the **special cell** has the **additional intermediate** parameter `reconfigurationWithSync` (that has **data type** `ReconfigurationWithSync`) between `SpCellConfig` and `ServingCellConfigCommon`, while this intermediate parameter does **not exist** in the configuration structure of **each secondary cell** (therefore, in **each secondary cell** `SCellConfig` is the **parent** of `ServingCellConfig`).
- The data types `ServCellIndex` and `SCellIndex` (click 3.9.9 for both) have been included in the **ancestor tree**, although they do **not** actually belong to **this ancestor tree**, for completeness' sake.

Recall that **Sidelink** channels (and consequently **Sidelink** data types and parameters) are **not** covered in this book.

## 3.9.3 `ServingCellConfigCommonSIB` data type

It is **only broadcast** (or **unicast**) within `SIB1` (click 5.3.4).

For its **ancestor tree up to** `SIB1`, click 3.9.1.

**All** its parameters are **common** (click 5.1.3).

**Each** of its **instances** configures **all** the **UEs** in a **primary cell** with a set of **common parameters**, when the **UEs** are accessing the **cell** from the **IDLE** state.

### 3.9.3.1 `ServingCellConfigCommonSIB` parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `downlinkConfigCommon` `DownlinkConfigCommonSIB`: optional, common

  It does **not** have a **field description** in the table "*DownlinkConfigCommonSIB field descriptions*" of TS 38.331.

It then **assumed** that it has the **same semantics** of its **same-name parameter** from `ServingCellConfigCommon` (click 3.9.6).

- `n-TimingAdvanceOffset` **ENUMERATED** {`n0`, `n25600`, `n39936`}: optional, common

  It is the "$N_{TA,offset}$" (click 2.3.6.1) to be applied for **random access** on this **serving cell**. If it is absent, the **UE** applies the value defined for the **duplex mode** and **frequency range** of this **serving cell** from table 7.1.2-2 of TS 38.133.

- `ssb-PeriodicityServingCell` **ENUMERATED**: mandatory, common

  It does **not** have a **field description** in the table "*DownlinkConfigCommonSIB field descriptions*" of TS 38.331.

  It then **assumed** that it t it has the **same semantics** as its **case-different** parameter `ssb-periodicityServingCell` from `ServingCellConfigCommon` (click 3.9.6).

  For an explanation on **same-name** and **version-different parameters**, click 5.7.2.2.

- `ssb-PositionsInBurst` **SEQUENCE**: optional, common

  For **shared spectrum**, it indicates the **time locations** (in a **half frame**) of the **SSB transmissions** (click 6.1.2.1.3) within **each SSB burst**.

  Its **data type** is a **SEQUENCE** that contains two **bit strings** (`inOneGroup` and `groupPresence`) **each** of length `8` bits, as defined in clause 4.1 of TS 38.213. The **first/leftmost bit** corresponds to **SSB index 0**, the **second bit** corresponds to **SSB index 1**, and so on. **Value** `0` in the **bitmap** indicates that the corresponding **SSB** is **not** transmitted while **value** `1` indicates that the corresponding **SSB** is **transmitted**.

  The network configures the **same pattern** in this parameter as in the **same-name parameter** from `ServingCellConfigCommon` (click 3.9.6).

  For **shared spectrum** in **FR1**, only `inOneGroup` is used and the **UE** interprets this parameter same as `mediumBitmap` from `ServingCellConfigCommon`. The **UE** assumes that a bit in `inOneGroup` at position "**k** > $N_{SSB}^{qCL}$" is `0`.

  For **shared spectrum** in **FR2-2**, the **m-th** bit in `groupPresence` is set to `0` for "**m** > $N_{SSB}^{qCL}$ / **8**".

  The value of "$N_{SSB}^{qCL}$" is obtained from `MIB` as specified in clause 4.1 of TS 38.213, using table 4.1-1.

- `tdd-UL-DL-ConfigurationCommon` `TDD-UL-DL-ConfigCommon` (click 2.3.5.3): optional, common

  It does **not** have a **field description** in the table "*DownlinkConfigCommonSIB field descriptions*" of TS 38.331.

  It then **assumed** it has the **same semantics** as its **same-name** parameter from `ServingCellConfigCommon` (click 3.9.6).

- **supplementaryUplinkConfig** **UplinkConfigCommonSIB**: optional, common

  Same as **uplinkConfigCommon** (see below) but referred to the **SUL channel**.

- **uplinkConfigCommon** **UplinkConfigCommonSIB** (click 2.2.6.5): optional, common

  It does **not** have a **field description** in the table "*DownlinkConfigCommonSIB field descriptions*" of TS 38.331.

  It is then assumed that it has the **same semantics** as its **same-name parameter** from **ServingCellConfigCommon** (click 3.9.6).

Comparing **ServingCellConfigCommonSIB** with **ServingCellConfigCommon** (click 3.9.6), they **share most** of their **parameters**. One difference is that the **former** does **not have** the parameters **physCellId**, **ssbSubcarrierSpacing** and **dmrs-TypeA-Position**. The reasons for this are:

- **physCellId** conveys the **PCI** (click 3.5.3.2) of the **cell** to apply the other parameters in the data type. However, **ServingCellConfigCommonSIB** **only** applies to the **cell where** it is **received**, so there is **no** need to indicate the **PCI**.
- **ssbSubcarrierSpacing** conveys the **SSB subcarrier spacing** (click 6.1.4) of the **cell**. However, when receiving **ServingCellConfigCommonSIB** the **UE already knows** the **SSB subcarrier spacing** (otherwise it would **not** have been able to retrieve **MIB** and **SIB1**).

For more detail on these parameters, refer to the table "*ServingCellConfigCommonSIB field descriptions*" of TS 38.331.

### 3.9.3.2 ServingCellConfigCommonSIB ASN.1 specification

**Part** of its **ASN.1** (click 5.9) specification is:

```
ServingCellConfigCommonSIB ::= SEQUENCE {
  downlinkConfigCommon DownlinkConfigCommonSIB,
  uplinkConfigCommon UplinkConfigCommonSIB OPTIONAL,                -- Need R
  supplementaryUplink UplinkConfigCommonSIB OPTIONAL,               -- Need R
  n-TimingAdvanceOffset ENUMERATED {n0, n25600, n39936} OPTIONAL,   -- Need S
   ssb-PositionsInBurst SEQUENCE {
     inOneGroup    BIT STRING (SIZE (8)),
     groupPresence BIT STRING (SIZE (8)) OPTIONAL           -- Cond FR2-Only
   },
  ssb-PeriodicityServingCell ENUMERATED {ms5, ms10, ms20, ms40,
                                         ms80, ms160},
  tdd-UL-DL-ConfigurationCommon TDD-UL-DL-ConfigCommon OPTIONAL, -- Cond TDD
  other parameters (not shown)
}
```

## 3.9.4 SCellConfig data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** **CellGroupConfig** (click 3.7.3), click 3.9.2.

It contains **both common parameters** and **dedicated parameters** (click 5.1.3).

**Each** of its **instances** configures a **UE** with a **secondary cell** of the corresponding **cell group**: (the **master** or the **secondary cell group**).

**Each** of its **instances** configures a **UE** with **most** of the **physical layer elements**, for **all** its **bandwidth parts** (**initial** and/or **additional**), for **all** its **carrier types** (**Downlink**, **Uplink** and/or **SUL**), for **common** and **dedicated configuration**, of its corresponding **secondary cell**, of its corresponding **cell group** (**master** or **secondary**). The "**corresponding**" elements are determined by the **specific place**, within the overall **ASN.1 specification**, where this **data type** has been **instantiated**.

#### 3.9.4.1 `SCellConfig` parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `sCellConfigCommon ServingCellConfigCommon`: optional, common

  It contains the **common** parameters to configure the corresponding **secondary cell**.

- `sCellConfigDedicated ServingCellConfig`: optional, dedicated

  It contains the **dedicated** parameters to configure the corresponding **secondary cell**.

- `sCellSIB20-r17 SCellSIB20-r17`: optional, common

  It contains the `SIB20-r17` for this **secondary cell**, in order to allow the **UE** for **MBS broadcast reception** on this **secondary cell**. The network configures this parameter **only** for a **single secondary cell at a time**. The specification of its **data type** is:

  ```
  SCellSIB20-r17 ::= OCTEST STRING (CONTAINING SystemInformation)
  ```

  where `SystemInformation` (click 5.7.5) contains `SIB20-r17`.

#### 3.9.4.2 `SCellConfig` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
SCellConfig ::= SEQUENCE {
  sCellIndex SCellIndex,
  sCellConfigCommon ServingCellConfigCommon OPTIONAL,    -- Cond SCellAdd
  sCellConfigDedicated ServingCellConfig OPTIONAL,       -- Cond SCellAddMod
  ...,
  [[
  smtc SSB-MTC OPTIONAL                                  -- Need S
  ]],
  [[
  sCellState-r16 ENUMERATED {activated} OPTIONAL,        -- Cond SCellAddSync
  secondaryDRX-GroupConfig-r16 ENUMERATED {true} OPTIONAL
                                                         -- Cond DRX-Config2
  ]],
  [[
  preConfGapStatus-r17 BIT STRING (SIZE (maxNrofGapId-r17)) OPTIONAL,
                                                         -- Cond PreConfigMG
  goodServingCellEvaluationBFD-r17 GoodServingCellEvaluation-r17 OPTIONAL,
                                                         -- Need R
  sCellSIB20-r17 SetupRelease {SCellSIB20-r17} OPTIONAL -- Need M
  ]]
}
```

where:

```
SCellSIB20-r17 ::= OCTET STRING (CONTAINING SystemInformation)
```

### 3.9.5 `SpCellConfig` data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 3.9.2.

It contains **both common parameters** and **dedicated parameters** (click 5.1.3).

**Each** of its **instances** configures a **UE** with the **special cell** of the corresponding **cell group**: either the **primary cell** of a **master cell group** or the **primary-secondary cell** of a **secondary cell group**.

These parameters for the **special cell configure** the **UE** with **most** of the **physical layer elements**, for **all** its **bandwidth parts** (**initial** and/or **additional**), for **all** its **carrier types** (**Downlink**, **Uplink** and/or **SUL**), for **common** and **dedicated configuration**, of its **special cell**, of its corresponding **cell group** (**master** or **secondary**). The "**corresponding**" elements are determined by the **specific place**, within the overall **ASN.1 specification**, where this **data type** has been **instantiated**.

#### 3.9.5.1 `SpCellConfig` parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `reconfigurationWithSync ReconfigurationWithSync` (click 3.9.8): optional, common **and** dedicated

  It contains the **parameters** for the **synchronous** reconfiguration of the **target SpCell**.

- `spCellConfigDedicated ServingCellConfig` (click 3.9.7): optional, dedicated

  It contains the **dedicated** parameters to configure the **special cell**.

For more detail on these parameters, refer to the table "*SpCellConfig field descriptions*" of TS 38.331.

#### 3.9.5.2 `SpCellConfig` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
SpCellConfig ::= SEQUENCE {
  servCellIndex ServCellIndex OPTIONAL,                                  -- Cond SCG
  reconfigurationWithSync ReconfigurationWithSync OPTIONAL,
                                                    -- Cond ReconfWithSync
  rlf-TimersAndConstants SetupRelease {RLF-TimersAndConstants} OPTIONAL,
                                                                         -- Need M
  rlmInSyncOutOfSyncThreshold ENUMERATED {n1} OPTIONAL,                  -- Need S
  spCellConfigDedicated ServingCellConfig OPTIONAL,                      -- Need M
  ...,
  [[
  lowMobilityEvaluationConnected-r17 SEQUENCE {
    s-SearchDeltaP-Connected-r17 ENUMERATED {dB3, dB6, dB9, dB12,
                                 dB15, spare3, spare2, spare1},
    t-SearchDeltaP-Connected-r17 ENUMERATED {s5, s10, s20, s30, s60,
                            s120, s180, s240, s300, spare7, spare6,
                            spare5, spare4, spare3, spare2,
                            spare1}
  } OPTIONAL,                                                            -- Need R
  goodServingCellEvaluationRLM-r17 GoodServingCellEvaluation-r17 OPTIONAL,
                                                                         -- Need R
  goodServingCellEvaluationBFD-r17 GoodServingCellEvaluation-r17 OPTIONAL,
                                                                         -- Need R
  deactivatedSCG-Config-r17 SetupRelease {DeactivatedSCG-Config-r17}
                            OPTIONAL                          -- Cond SCG-Opt
 ]]
}
```

### 3.9.6 `ServingCellConfigCommon` data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 3.9.2.

**All** its parameters are **common** (click 5.1.3).

**Each** of its **instances** contains parameters which a **UE** would typically acquire from **SSB**, `MIB` or **SIBs** when accessing the cell from **IDLE**. It configures a **UE** with several **SCells** or with an additional cell group (**SCG**), and also for **SpCells** (in **MCG** and **SCG**) upon `ReconfigurationWithSync` (click 3.9.8).

**Each** of its **instances** configures a **UE** with **most** of the **physical layer elements**, of its **initial bandwidth parts**, for **all** its **carrier types** (**Downlink**, **Uplink** and/or **SUL**), of **common configuration**, of its corresponding **cell** (**special** or **secondary**), of its corresponding **cell group** (**master** or **secondary**). The "**corresponding**" elements are determined by the **specific place**, within the overall **ASN.1 specification**, where this **data type** has been **instantiated**.

#### 3.9.6.1 `ServingCellConfigCommon` parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `dmrs-TypeA-Position` **ENUMERATED** {`pos2`, `pos3`}: mandatory, common

  It is the **slot symbol index** of the **first DM-RS symbol** both for **PDSCH** and for **PUSCH** when using **DM-RS time-domain mapping** `typeA` (click 7.11.5.1). Value `pos2` indicates **slot symbol index 2** and `pos3` indicates **3** (i.e., the **third or** the **fourth** symbol in the slot, respectively).

In case it is **present** for the **primary cell**, it **supersedes** the **same-name parameter** from `MIB`.

For more detail on the location of **PDSCH-DM-RS**, refer to clause 7.4.1.1.1 of TS 38.211. For more detail on the location of **PUSCH-DM-RS**, refer to clause 6.4.1.1.3 of TS 38.211.

- `downlinkConfigCommon DownlinkConfigCommon` (click 2.2.6.8): optional, common

  It contains the **common parameters** for the **Downlink carriers** and for the **Downlink initial bandwidth part**.

  The **value** of its internal parameters should match the value of the **corresponding** parameters configured with `MIB` and with `SIB1` (if provided). As an exception, its internal parameters `controlResourceSetZero` and `searchSpaceZero` from `PDCCH-ConfigCommon` (click 6.12.4) can have its specific value defined within `downlinkConfigCommon` even if their two **same-name parameters** are **absent** in `MIB`. Their two **same-name parameters** are from `PDCCH-ConfigSIB1` (click 8.12.3.1), within `MIB` (click 5.3.3).

  Notice that there is a **same-name parameter** from `ServingCellConfigCommonSIB` (click 3.9.3).

- `n-TimingAdvanceOffset ENUMERATED {n0, n25600, n39936}`: optional, common

  It is the "$N_{TA,offset}$" (click 2.3.6.1) to be applied for **random access** on this **serving cell**. If it is absent, the **UE** applies the value defined for the **duplex mode** and **frequency range** of this **serving cell** from table 7.1.2-2 of TS 38.133.

- `physCellId PhysCellId`: optional, common

  It does **not** have a **field description** in the table "*ServingCellConfigCommon field descriptions*" of TS 38.331.

  It is then **assumed** that it is the **Physical Cell Identity** (**PCI**, click 3.5.3.2) of the **serving cell** to which corresponds the **other parameters** in this same data type **instance**.

  Its **data type** `PhysCellId` is an `INTEGER (0..1007)`.

- `ssb-periodicityServingCell ENUMERATED`: optional, common

  In TS 38.331 it is indicated that this is the **SSB periodicity** in **ms** for the rate-matching purpose. It may take the values `5`, `10`, `20`, `40`, `80` and `160` ms.

  In clause 4.1 of TS 38.213 it is indicated that this parameter is the **actual SSB burst periodicity** and also that if this parameter is **absent**, the **UE** applies the value `5` ms, except for **initial access**, where the **UE** may assume that the value is `20` ms.

  Notice that there is a **case-different parameter**, with the **same data type** and the **same semantics**, from `ServingCellConfigCommonSIB` (click 3.9.3).

- `ssb-PositionsInBurst` **CHOICE**: optional, common

  **For operation with licensed spectrum**

  It indicates the **time locations** (in a **half frame**) of the **SSB transmissions** (click 6.1.2.1.3) within **each SSB burst**.

  It is a **CHOICE** between **bit strings** of length `4`, `8` or `64` bits, as defined in clause 4.1 of TS 38.213. The **first/leftmost bit** corresponds to **SSB index 0**, the **second bit** corresponds to **SSB index 1**, and so on. **Value 0** in the **bitmap** indicates that the corresponding **SSB** is **not** transmitted while **value 1** indicates that the corresponding **SSB** is **transmitted**.

  The network configures the **same pattern** in **this parameter** as in the **same-name parameter** from `ServingCellConfigCommonSIB` (click 3.9.3).

  **For shared spectrum**

  The **UE** assumes that **one or more SSBs** indicated by `ssb-PositionsInBurst` may be transmitted within the discovery burst transmission window and have candidate **SSB indexes** corresponding to **SSB indexes** provided by `ssb-PositionsInBurst` (refer to clause 4.1 of TS 38.213). If the **k-th bit** of `ssb-PositionsInBurst` is set to **1**, the **UE** assumes that **one or more SSBs** within the discovery burst transmission window with candidate **SSB indexes** corresponding to **SSB index** equal to **k-1** may be transmitted; if the **k-th bit** is rather set to **0**, the **UE** assumes that the corresponding **SSBs** are not transmitted. The **k-th** bit is set to **0**, where "**k >** `ssb-PositionQCL-*`" and the quantity of **actually transmitted SSBs** is not larger than the quantity of **1**'s in the bitmap.

  The network configures the **same pattern** in this parameter as in the **same-name parameter** from `ServingCellConfigCommonSIB`.

  For **shared spectrum** in **FR1**, only `mediumBitmap` is used, and for **FR2-2**, `longBitmap` is used (see the **ASN.1 specification** below).

- `ssbSubcarrierSpacing` `SubcarrierSpacing` (click 2.2.10.5.1): optional, common

  It is the **cell**'s **SSB subcarrier spacing** (click 6.1.4). Only the following values are **applicable** depending on the **frequency range** where the **band** is located:

  - **FR1**: **15** or **30 kHz**
  - **FR2-1**: **120** or **240 kHz**
  - **FR2-2**: **120**, **480**, or **960 kHz**

- `supplementaryUplinkConfig` `UplinkConfigCommon` (click 2.2.6.9): optional, common

  Same as the previous parameter but referred to the **SUL channel**.

- `tdd-UL-DL-ConfigurationCommon` `TDD-UL-DL-ConfigCommon` (click 2.3.5.3): optional, common

  It is the **common TDD Uplink/Downlink configuration** for **all** the **UEs** in the **cell**.

Notice that **ServingCellConfigCommonSIB** (click 3.9.3) has a **same-name** parameter with the **same data type** and the **same semantics**.

- **uplinkConfigCommon UplinkConfigCommon** (click 2.2.6.9): optional, common

  It contains the **common parameters** for the **Uplink carriers** and for the **Uplink initial bandwidth part**.

Comparing **ServingCellConfigCommonSIB** (click 3.9.3) with **ServingCellConfigCommon**, they **share most** of their **parameters**. For an explanation of some differences, click 3.9.3.1.

For more detail on these parameters, refer to the table "*ServingCellConfigCommon field descriptions*" of TS 38.331.

### 3.9.6.2 ServingCellConfigCommon ASN.1 specification

**Part** of its **ASN.1** (click 5.9) specification is:

```
ServingCellConfigCommon ::= SEQUENCE {
  physCellId PhysCellId OPTIONAL,                    -- Cond HOAndServCellAdd
  downlinkConfigCommon DownlinkConfigCommon OPTIONAL,
                                                     -- Cond HOAndServCellAdd
  uplinkConfigCommon UplinkConfigCommon OPTIONAL,                  -- Need M
  supplementaryUplinkConfig UplinkConfigCommon OPTIONAL,           -- Need S
  n-TimingAdvanceOffset ENUMERATED {n0, n25600, n39936} OPTIONAL,
                                                                   -- Need S
  ssb-PositionsInBurst CHOICE {
    shortBitmap  BIT STRING (SIZE  (4)),
    mediumBitmap BIT STRING (SIZE  (8)),
    longBitmap   BIT STRING (SIZE (64))
    } OPTIONAL,                                          -- Cond AbsFreqSSB
  ssb-periodicityServingCell ENUMERATED {ms5, ms10, ms20, ms40, ms80,
                                ms160, spare2, spare1} OPTIONAL,  -- Need S
  dmrs-TypeA-Position ENUMERATED {pos2, pos3},
  other parameters (not shown)
  ssbSubcarrierSpacing SubcarrierSpacing OPTIONAL,
                                                  -- Cond HOAndServCellWithSSB
  tdd-UL-DL-ConfigurationCommon TDD-UL-DL-ConfigCommon OPTIONAL, -- Cond TDD
  other parameters (not shown)
}
```

## 3.9.7 ServingCellConfig data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** **CellGroupConfig** (click 3.7.3), click 3.9.2.

**Most** of its parameters are **dedicated** (but a few are **common**) (click 5.1.3).

**Each** of its **instances** configures the **UE** with:

- **Most** of the **physical layer elements**, of its **initial bandwidth parts**, for **all** its **carrier types** (**Downlink**, **Uplink** and/or **SUL**), of **dedicated configuration**.
- **Most** of the **physical layer elements**, of **all** its **additional bandwidth parts**, for **all** its **carrier types** (**Downlink**, **Uplink** and/or **SUL**), **dedicated** (**and some common**) **configuration**.
- The **Downlink default bandwidth part** and the **first active bandwidth parts**, for **all** its **carrier types** (**Downlink**, **Uplink** and **SUL**).

all of them for its corresponding **cell** (**special** or **secondary**), of its corresponding **cell group** (**master** or **secondary**). The "**corresponding**" elements are determined by the **specific place**, within the overall **ASN.1 specification**, where this **data type** has been **instantiated**.

#### 3.9.7.1 `ServingCellConfig` parameters

Notice that the **parameters** of `ServingCellConfig` that are **reciprocal** to the `UplinkConfig` parameters are **not presented here**, and they are **rather** presented as the ***`DownlinkConfig`*** **parameter set** in 2.2.8.3.

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- ***`DownlinkConfig`*** **parameters** (click 2.2.8.3): common **and** dedicated

  This is **not** a parameter. It is a **set of parameters** that is defined for reciprocity to `uplinkConfig UplinkConfig`, in the next bullet point. It contains the **dedicated** configuration for the **Downlink carriers** and **bandwidth parts**.

- `supplementaryUplink UplinkConfig` (click 2.2.8.4): optional, common **and** dedicated

  It contains the **dedicated** configuration for the **SUL carriers** and **bandwidth parts**.

  It is **only provided** when **either** `supplementaryUplinkConfig` from `ServingCellConfigCommon` or `supplementaryUplink` from `ServingCellConfigCommonSIB` is **present**.

- `tdd-UL-DL-ConfigurationDedicated TDD-UL-DL-ConfigDedicated` (click 2.3.5.5): optional, dedicated

  It does **not** have a **field description** in TS 38.331. It is then **assumed** that it is the **dedicated** (**UE-specific**) **TDD Uplink/Downlink configuration**.

- `uplinkConfig UplinkConfig` (click 2.2.8.4): optional, common **and** dedicated

  It contains the **dedicated** configuration for the **Uplink carriers** and **bandwidth parts**.

  It is **only provided** when `uplinkConfigCommon` from **either** `ServingCellConfigCommon` or `ServingCellConfigCommonSIB` is **present**. **Addition** or **release** of this parameter can **only** be done **upon SCell addition** or **release** (respectively).

For more detail on these parameters, refer to the table "*ServingCellConfig field descriptions*" of TS 38.331.

#### 3.9.7.2 `ServingCellConfig` ASN.1 specification

Notice that the **parameters** of `ServingCellConfig` that are **reciprocal** to the `UplinkConfig` parameters are **not presented here**, and they are **rather** presented as the ***`DownlinkConfig`*** **parameter set** in 2.2.8.3.

**Part** of its **ASN.1** (click 5.9) specification is:

```
ServingCellConfig ::= SEQUENCE {
  tdd-UL-DL-ConfigurationDedicated TDD-UL-DL-ConfigDedicated
                                   OPTIONAL,                    -- Cond TDD
  other parameters presented in DownlinkConfig
  uplinkConfig             UplinkConfig OPTIONAL,                -- Need M
  supplementaryUplink      UplinkConfig OPTIONAL,                -- Need M
  other parameters presented in DownlinkConfig
  csi-MeasConfig SetupRelease {CSI-MeasConfig} OPTIONAL,         -- Need M
  sCellDeactivationTimer ENUMERATED {ms20, ms40, ms80, ms160, ms200, ms240,
                                     ms320, ms400, ms480, ms520, ms640,
                                     ms720, ms840, ms1280, spare2, spare1}
                         OPTIONAL,          -- Cond ServingCellWithoutPUCCH
  other parameters (not shown)
  downlinkChannelBW-PerSCS-List SCS-SpecificCarrier (1..maxSCSs)
                                OPTIONAL                         -- Need S
  other parameters (not shown)
  intraCellGuardBandsDL-List-r16 SEQUENCE (SIZE (1..maxSCSs)) OF
                           IntraCellGuardBandsPerSCS-r16 OPTIONAL, -- Need S
  intraCellGuardBandsUL-List-r16 SEQUENCE (SIZE (1..maxSCSs)) OF
                           IntraCellGuardBandsPerSCS-r16 OPTIONAL, -- Need S
  other parameters (not shown)
}
```

### 3.9.8 `ReconfigurationWithSync` data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 3.9.2.

It contains **both common parameters** and **dedicated parameters** (click 5.1.3).

Its **only parent** data type is `SpCellConfig` (click 3.9.5). Its **only grand parent** data type is `CellGroupConfig` (click 3.7.3).

**Each** of its **instances** contains the **parameters** for the **synchronous** reconfiguration of the **target SpCell**.

#### 3.9.8.1 `ReconfigurationWithSync` parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `newUE-Identity RNTI-Value`: mandatory, common

  It is the **C-RNTI** (click 6.9.10.2) used in the procedure of `ReconfigurationWithSync` (click 3.9.8) (e.g., **handover**).

- `rach-ConfigDedicated CHOICE`: mandatory, dedicated

  It is the **random access configuration** used in the procedure of `ReconfigurationWithSync` (click 3.9.8) (e.g. **handover**). The **UE** performs the **random access procedure** according to these parameters in the **first-active bandwidth part** (i.e., in `firstActiveUplinkBWP-Id` from `UplinkConfig` (click 2.2.8.4)).

  Its **data type** is a `CHOICE` between `uplink` and `supplementaryUplink`, **both** of them with data type `RACH-ConfigDedicated` (click 9.5.10).

- `spCellConfigCommon ServingCellConfigCommon` (click 3.9.6): optional, common

  It contains the **common** parameters to configure the **target special cell**.

For more detail on these parameters, refer to the table "*ReconfigurationWithSync field descriptions*" of TS 38.331.

#### 3.9.8.2 `ReconfigurationWithSync` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
ReconfigurationWithSync ::= SEQUENCE {
  spCellConfigCommon ServingCellConfigCommon OPTIONAL,              -- Need M
  newUE-Identity RNTI-Value,
  t304 ENUMERATED {ms50, ms100, ms150, ms200, ms500, ms1000, ms2000,
                                                                    ms10000},
  rach-ConfigDedicated CHOICE {
    uplink              RACH-ConfigDedicated,
    supplementaryUplink RACH-ConfigDedicated
  } OPTIONAL,                                                       -- Need N
  ...,
  [[
  smtc SSB-MTC OPTIONAL                                             -- Need S
  ]],
  [[
  daps-UplinkPowerConfig-r16 DAPS-UplinkPowerConfig-r16 OPTIONAL  -- Need N
  ]],
  [[
  sl-PathSwitchConfig-r17 SL-PathSwitchConfig-r17 OPTIONAL
                                        -- Cond DirectToIndirect-PathSwitch
  ]]
}
```

### 3.9.9 `ServCellIndex` and `SCellIndex` data types

`ServCellIndex` is an **`INTEGER`** that serves as **cell identifier** in the context of **each UE**'s **cell group**. The identified **cell** may be **either** the **special cell** (that **always** has identifier `0`) or **one secondary cell** (that **always** have identifiers from `1` to `31`). It **must** be **unique** within **each** of the **UE**'s **cell groups**.

Its **complete ASN.1** (click 5.9) specification is:

```
ServCellIndex ::= INTEGER (0..maxNrofServingCells-1)
```

where the constant `maxNrofServingCells-1` is `31`, which is the **maximum quantity**, minus 1, of **cells** in **each** of the **UE**'s **cell groups**.

`SCellIndex` is an **`INTEGER`** that serves as **secondary cell identifier** in the context of **each UE**'s **cell group**. It **must** be **unique** within **each** of the **UE**'s **cell groups**. Its value is the **index** for every parameter with data type "**`SEQUENCE OF SCellConfig`**". Therefore, its value serves to reference **each instance** of `SCellConfig` within **each** such "**`SEQUENCE OF`**".

Its **complete ASN.1** (click 5.9) specification is:

```
SCellIndex ::= INTEGER (1..31)
```

The value `31` is the **maximum quantity** of **secondary cells** in **each** of the **UE**'s **cell groups** (as indicated by the constant `maxNrofSCells`, whose value is `31`).

## 3.10 Deployment planning

**Deployment planning** consists of the **radio access network planning** and **deployment** in order to **cover** the **desired geographical area**, providing service above the **performance threshold** at **each specific place** (rural areas, cities, business district, airports, roads, etc).

**Deployment planning** is a **permanent activity**, because the network infrastructure is **continuously being updated**. **Fixed cells**, **new technologies** (e.g., **5G**), **upgraded equipment** (e.g., **DSS**, click 10.7), **cells** on **wheels**, etc, are **permanently added to**, and **removed from**, the **radio access network**.

In **deployment planning** there can be **2 different** objectives:

- **Increase coverage**

  **Increasing** the **coverage** (click 3.10.5) is typically based on deploying additional **macrocells** (click 3.2.3).

- **Densification**

  **Densification consists** of **increasing** the **cell area capacity** (click 3.10.4). This might allow to increase **performance threshold** (click 3.10.1.1).

  There are **several densification techniques** (click 3.10.6), **each** of them with its specific characteristics.

**Deployment planning** is **very complex**. Even with **excellent computer-based channel models**, the "**theoretical**" **coverage** has to be **complemented** with the **operator**'s **experience** and **heuristic techniques**.

Consequently, this section **only** provides a **preliminary approach** to **deployment planning**.

### 3.10.1 Cell reach

The **cell reach** is the **distance** from the **antenna site** to the **furthest point** in a **given direction** where the **service** is above the **performance threshold** (click 3.10.1.1). **Beyond** that **furthest point**, the **service** is below the **performance threshold**. The **cell range** (click 3.10.1.1) and the **cell area** (click 3.10.3) are two relevant concepts **derived** from the **cell reach**.

The **cell reach** depends on the **geographical direction** and on the **transmission direction** in which it is **calculated** or **measured**:

- The **geographical direction** is expressed as the **azimuth** (i.e., the clockwise **angle** from **North**) and the **elevation** (i.e., the **angle** from **horizontal**)

  **Cell reach** is **different** in **different geographical directions** because of the **orography** (hills, buildings, vegetation, tunnels, etc) and of the differences in **environment** (humidity, reflections (e.g., because of water bodies), noise, interference, etc).

- The **transmission direction** is **either Downlink** or **Uplink**.

  **Cell reach** is **different** in **Downlink** or **Uplink** because: **a)** the radio capacities of the **NR node** and the **UEs** are **very different**; **b)** for **FDD bands**, the **Downlink** and **Uplink radio channels** are **different** and **c)** in case the **cell** includes a **SUL channel**, the transmission in the **Uplink direction** may be done over the **main channel** (a **TDD** or **Uplink channel**) and/or over the **SUL** channel.

The **cell reach** is **not static** and it experiments variations because of:

- **Atmospheric conditions**

  **Rain**, **fog**, **hail**, **pressure**, **dust** (among others) **change** with **time** and modify the channel conditions, in **particular** at **high frequencies**. These modifications in channel conditions result in a **variation** of the **cell reach**.

- **External noise and interference**

  **External noise** (click 12.1.10) and **external interference** (click 12.1.13) **change** with **time** and modify the **signal** to **noise** and **interference ratio** (**SNIR**). These modifications in **SNIR** result in a **variation** of the **cell reach**.

- **Cyclic patterns**

  **Atmospheric conditions** usually follow a **daily** and **yearly** (seasonal) **cycle**. **External noise** and **external interference** usually follow a **daily**, **weekly** and **yearly** (seasonal) **cycle**. Their combination results in a **daily**, **weekly** and **yearly cycle** of the **cell reach**.

The variations on the **cell reach** propagate to the **cell range** and to the **cell area**.

#### 3.10.1.1 Performance threshold

The **UE performance threshold** is the performance **below which** the **UE** is considered to be **out** of **coverage**.

The **UE performance threshold** can be established in terms of:

- **Downlink UE-in-cell capacity** (click 4.9.3.3).
- **Uplink UE-in-cell capacity**.
- **Packet-loss probability**.
- **Round-trip latency** to the **edge-computing node**.
- **Received power**.
- **Traffic load**.

The **performance threshold** is defined in a different way for **individual customers** and **corporate customers**.

##### 3.10.1.1.1 Performance threshold for individual customers

The **performance threshold** for **individual customers** is typically defined by the **national authorities** (i.e., the **regulator**).

For **LTE**, the **individual customers** typically understood the **performance threshold** as the one that allows to have a **voice call** at any moment (i.e., even in the **peak hour**). This can be technically defined as the level of **bidirectional UE-in-cell throughput** for a **voice call**.

However, **operators** and **regulators** typically defined the **LTE performance threshold** as a certain **level** of **outdoors power received** at the **UE**. The advantages of this approach is that it is **objective**, **independent** of the **UE capabilities** (click 4.9.1) and **easy** to **check**.

When determining which is the **received power level threshold**, it is convenient to take into account:

- **Uplink** is the **most restrictive** transmission **direction**. Consequently, the **received power level threshold** should be increased above the one required for **Downlink** reception, in order to provide a **certain guarantee** that **Uplink** communications will also be **possible**.
- **More power** is required for **indoors communication**. It is not infrequent that customers in **rural areas** complain because they have **coverage outside** their **home**, but **not inside**. Consequently, the **received power level threshold** should be increased to provide a **certain guarantee** that **indoors** communications will also be **possible**.

The disadvantages of this approach are that **received power** does **not guarantee** a **certain throughput, nor a certain latency, and it does not take into account the network** load. The **5G** technology was designed to offer a plethora of **new services** to **consumers**: self-driven cars, on-line gaming, **HD** or even **UHD** streaming, etc.

Consequently, it is required that **operators** and **regulators redefine** the **performance threshold** to incorporate guarantees for:

- **UE-in-cell capacity**

  This is required to enable **consumers**, while in **coverage**, to access **high throughput eMBB services**, such as **HD** streaming.

- **Latency**

  This is required to enable **consumers**, while in **coverage**, to access **low-latency eMBB services**, such as **on-line gaming**.

- **Peak-hour**

  This is required to enable **consumers**, while in **coverage**, to receive service **above** the **performance threshold**, not only under low traffic but also during the **peak-hour** of traffic.

- **Location provision**

  This is required to enable **consumers**, while in **coverage**, to receive **location-based services**.

#### 3.10.1.1.2 Performance threshold for corporate customers

**Corporate customers** include specific **performance thresholds** in the **service contract** that they agree with the **operators**. In this context, the definition of the **performance thresholds** might be called **service level agreement** (**SLA**), **quality** of **service** (**QoS**) or **quality** of **experience** (**QoE**).

**Different performance thresholds** may be agreed for different services, different geographical areas, different seasons or different weekly/daily patterns. The **great control flexibility** of **5G** and the **slicing capability** enable the operators to engage in **contracts** with detailed and complex definitions of **performance thresholds**.

The **definition** of the **performance thresholds** may be based on **all** the **metrics** indicated in 3.10.1.1.1, and also on **additional ones** (e.g., packet-loss probability or power

consumption for **IoT** devices). Some specific cases of **performance threshold definition** are:

- **Performance threshold in own premises**

  Corporate customers that have an **on-premises specific 5G infrastructure** define very detailed **performance thresholds** to guarantee all their required services. The **standard services classes eMBB**, **URLLC**, **mMTMC** may be parametrized with different values, and **other service classes** can be defined. Some examples are large industrial facilities (e.g., Airbus), oil refineries (e.g., Repsol in Puertollano), ports (e.g., port of Málaga), shipyards (e.g., Navantia) and military facilities.

- **Performance threshold of specific UEs through the RAN**

  Corporate customers may **contract specific performance thresholds** for **specific UEs** throughout the **RAN** (possible with some **geographical** or **time-based restrictions**). In this case, the operator's flexibility to define **complex** and **diverse service classes** is **lower** than in the **customer**'s **own premises**, but still m**uch higher** than what was available with **LTE**. The "**specific UEs**" whose **performance threshold** is guaranteed might belong to the **corporate customer itself** and/or to their **customers** and/or to their **suppliers** and/or to their **partners**.

  An example is the **emergency services** of the region of **Extremadura** (Spain). They have advantageously replaced their **legacy TETRA network** by a corporate **5G slice**. They have benefited from **lower cost**, **much higher throughput**, **wider service range**, **more integration** and **openness**, **software-based flexibility**, **increased coverage** and **better reliability**. A **very specific characteristic** of this case is to include a **performance threshold** not only in the peak-hour, but in particular under the heaviest possible traffic load. This can be guaranteed by **traffic prioritization** and **weighted scheduling** within the **5G** network.

- **Performance threshold for every UE accessing service**

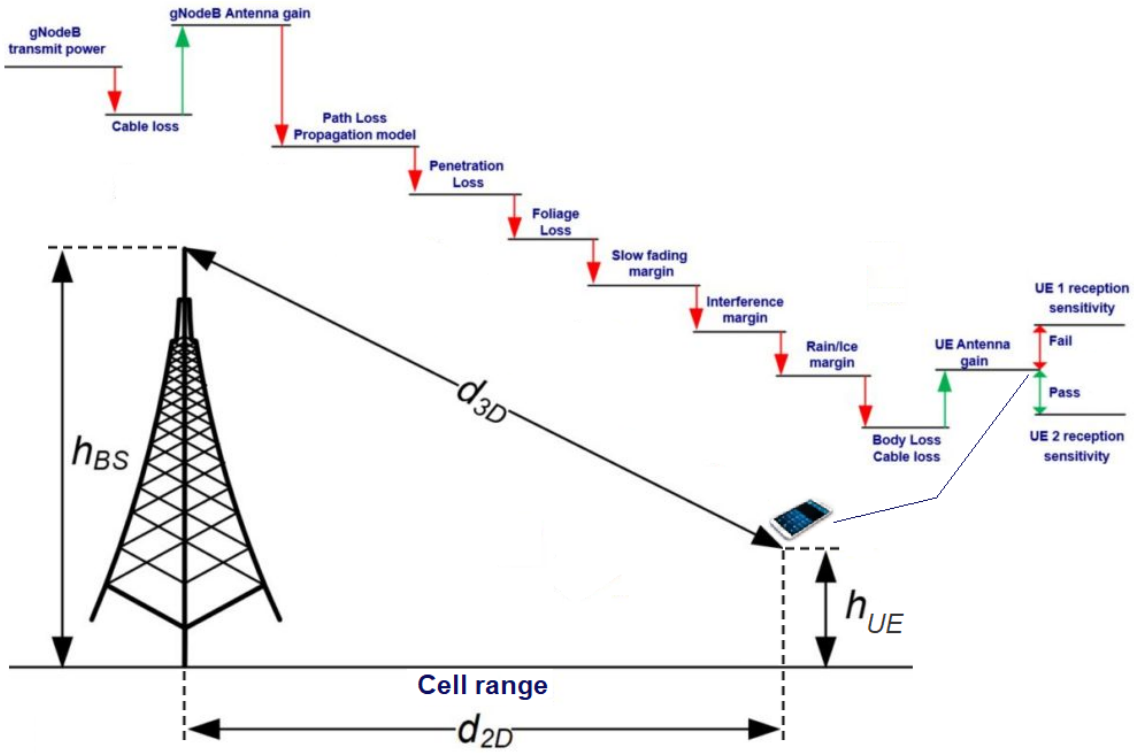
  Corporate customers may **contract specific performance thresholds** for **every UEs** that accesses a **specific service**. For example, a **top-end streaming platform** may contract that **every UE** that is accessing its service will have a certain **minimum UE-in-cell capacity**. Likewise, an **on-line videogame platform** may contract that **every UE** that is accessing its service will have a certain **maximum latency**.

#### 3.10.1.2 Cell reach estimation from Tx power

In terms of **power**, **both** the **NR node** and the **UE require** to **receive above** a certain **signal** to **noise** and **interference ratio** (**SNIR**), that is determined by the **performance threshold** (click 3.10.1.1).

This implies that the **Tx power** must be **enough** so that the **Rx power** is above the **receiver**'s **sensitivity threshold**. The relationship between the **Tx power** and the **Rx power** is determined by the **channel** and by the **channel conditions**. A **very basic transmission model** is provided by the **free-space path-loss** (**FSPL**) **formula** (click 12.2.5), but this **basic model** does **not** include many **relevant aspects**.

TR 38.901 contains a quite detailed **channel model** for frequencies from **500 kHz** up to **100 GHz**. The following figure shows the **different effects** that **reduce** the **Tx power** along its **path** to the **receiver**, according to this model:




The figure depicts the **Downlink** transmission **direction** , but the **channel model** works in a similar way in the **Uplink** transmission **direction** .

Notice that **path loss** includes **free-space path-loss** (click 12.2.5), **attenuation** (click 12.2.6) and **other channel effects**.

The following link contains an **on-line calculator** for the **link budget** according to this **channel model**:

- https://5g-tools.com/5g-nr-link-budget-calculator/

#### 3.10.1.2.1 Uplink cell reach estimation from Tx power

The **Uplink link budget** can be estimated with the expression:

$$P_{Rx} = P_{Tx} + G_{Tx} + G_{Rx} - PL \cdot C_{Reach} - L_{misc}$$

where:

- "$P_{Rx}$" is the **received power** at the **NR node**, in **dBm**.
- "$P_{Tx}$" is the **transmitted power** by the **UE**, in **dBm**.
- "$G_{Tx}$" and "$G_{Rx}$" are (respectively) the **antenna gains** of the **UE** and of the **gNB**, in **decibels** (**dB**).
- "$PL$" is the **total path loss**, in **dB/km**.
- "$C_{Reach}$" is the **cell reach**, which is the surface **distance** from the **UE** to the **NR node**, in **km**.

- "$L_{misc}$" represents **different miscellaneous losses** in addition to the **total path loss**, in **decibels** (**dB**). "$L_{misc}$" arises from body loss, feeder loss, polarization mismatch, penetration loss and other effects.

It is required that the **received power** is above the **NR node sensitivity** "$S_{Rx}$", and therefore:

$$S_{Rx} < P_{Tx} + G_{Tx} + G_{Rx} - PL \cdot D - L_{misc}$$

The **receiver sensitivity** of the **NR node** "$S_{Rx}$" can be estimated as:

$$S_{Rx} = N_{thermal} + NF + SN_{Min}$$

where:

- "$N_{thermal}$" is the **thermal noise** at the **NR node**, which is:

  $N_{thermal} = -174$ dBm/Hz $+ 10 \cdot log_{10}(W_{Min})$, expressed in **dBm**

  where "$W_{Min}$" is the **minimum bandwidth** to be **scheduled** (click 4.11.4.1) to the **receiver**, in **Hertz**. Its value is determined by the **definition** of the **performance threshold** (click 3.10.1.1).

- "$NF$" is the **noise figure** of the **NR node**, in **decibels** (**dB**).
- "$SN_{Min}$" is the **minimum signal** to **noise ratio** at the **receiver**, in **dB**. Its value is determined by the **definition** of the **performance threshold**.

The expression therefore becomes:

$$-174 + 10 \cdot log_{10}(W_{Min}) + NF + SN_{Min} < P_{Tx} + G_{Tx} + G_{Rx} - PL \cdot C_{Reach} - L_{misc}$$

Which can be reordered as:

$$C_{Reach} < [(P_{Tx} - (L_{misc} + NF - G_{Tx}) + G_{Rx} - (SN_{Min} + 10 \cdot log_{10}(W_{Min})) + 174] / PL$$

whose terms are determined by:

- The **regulation** mainly determines "$P_{Tx}$".
- The **implementation** of the **NR node** and of the **UE** mainly determines "$L_{misc}$", "$NF$" and "$G_{Tx}$". The **UE antenna** is largely **omnidirectional** (i.e., **isotropic**, click 11.21.3) and consequently "$G_{Tx}$" is close to **zero dB**.
- The **direction** mainly determines "$G_{Rx}$", departing from the **implementation** of the **NR node**, in particular of its **sectors** (click 3.1.4), which determines its horizontal **antenna radiation pattern** (click 11.21.5).
- The **definition** of the **performance threshold** (click 3.10.1.1) mainly determines "$SN_{Min}$" and "$W_{Min}$".
- The **channel frequency**, the **direction** and the **environment** mainly determine "$PL$".

For more detail, refer to TR 38.901. Its latest version (as of the publishing of this book) is V19.0.0 from June 2025.

Source web pages:

- https://www.telecomhall.net/t/receiver-sensitivity-in-5g-sub-6-ghz-and-mmwave/34029
- https://5g-tools.com/5g-nr-link-budget-calculator/

- https://hellofuture.orange.com/en/extreme-5g-coverage/

#### 3.10.1.2.2 Example of cell reach estimation from Tx power

The parameters for this example are:

- "$P_{Tx}$": the **maximum transmitted power** of **each UE** (click 4.9.1) is **23 dBm** on most cases, although on a few particular cases it may reach **35 dBm**. Assuming that the **UE** can transmit at the **maximum regulated power**, and therefore, **23 dBm** is selected as the most representative value.
- "$L_{misc}$": the **miscellaneous losses** are estimated at a **total** of **5 dB**.
- "$NF$": the **noise figure** at the **NR node** is estimated at **5 dB**.
- "$G_{Tx}$": the **UE antenna gain** can be around **1 dBi**. Since we are determining cell reach, we calculate it for **worst case UE**, and consequently, **0 dBi** is selected.
- "$G_{Rx}$": the **NR node antenna gain** is considered to be in a **high-gain direction**, and it is estimated at **17 dBi**.
- "$SN_{Min}$": the **performance threshold** in this example requires to provide **QPSK**, in **1 layer** which implies a value of **17 dB**.
- "$W_{Min}$": the **performance threshold** in this example requires to schedule **20 MHz**.
- "$PL$": the **channel frequency** is **700 MHz** and there are **no** significant **obstacles** in the corresponding **direction**. Consequently, "$PL$" is estimated at **4.67 dB/km**.

The expression from 3.10.1.2.1 is:

$$C_{Reach} < [(P_{Tx} - (L_{misc} + NF - G_{Tx}) + G_{Rx} - (SN_{Min} + 10 \cdot \log_{10}(W_{Min})) + 174] / PL$$

Introducing the above values in the expression:

$$C_{Reach} < [(23 - (5 + 5 - 0) + 17 - (-4 + 10 \cdot \log_{10}(20 \cdot 10^6)) + 174] / 4.67$$

$$C_{Reach} < 28.7 \text{ km}$$

### 3.10.2 Cell range

The **cell range** (also called **cell radius**) denotes the **maximum cell reach** of the **cell** in **every possible direction**. The **cell range** is **limited** by **many factors** (click 3.10.2.1), but the **main factor** is usually the **channel frequency** "$f$". A rough estimation on the **maximum cell range** depending on **frequency** is:

- **Low frequency**

  Channels in **FR1-1 bands below** 1 GHz.

  The **maximum cell range** is around **60 km** (a **macrocell**, click 3.2.3)

- **Medium frequency**

  Channels in **FR1-1 bands** above **1 GHz** "1 GHz < $f$ < 2.6 GHz" and in **FR1-2 bands** "3.5 GHz < $f$ < 6.7 GHz".

  The **maximum cell range** is around **10 km** (a **small macrocell** or a **large microcell**, click 3.2.3).

- **High and very high frequency**

  Channels in **FR2 bands** "24 GHz < $f$ < 60 GHz".

  The **maximum cell range** is around **200 m** (a **femtocell** or **picocell**, click 3.2.3).

#### 3.10.2.1 Cell range limits

Some of the **factors** that **limit** the **cell range** are:

- **Cell range limit** from **UE Tx power**

  The **UE Tx power** determines a **limit** to the **cell range**. Recall from **path loss** (click 12.2.5) that the **received power** is reduced with the **square** of the **distance** (**even** with **zero attenuation**) because of **energy spreading** (click 12.1.5).

  In practice, there are **many additional effects** (e.g., **atmospheric absorption**, **rain**, **fog**, **refraction**, …) that **reduce** the actual **received power**. For more detail on this, refer to the description on **cell reach** from **Tx power** (click 3.10.1.2.1).

- **Cell range limit** from **NR node Rx antenna gain**

  The **NR node Rx antenna gain** determines a **limit** to the **cell range**. This is the **reciprocal element** of the **UE Tx power** (see the previous bullet point). A **higher node Rx antenna gain** allows the **NR node** to **receive** the **Uplink radiated power** from a **larger distance**.

- **Cell range limit** from **cyclic prefix**

  The **cyclic prefix duration** (click 2.3.2) determines a **limit** to the **cell range**.

  The **symbol interference** (click 12.2.4) **increases** with **distance**. The **cyclic prefix technique** (click 11.13) is a very **effective countermeasure**, but its **effectiveness** depends on the **cyclic prefix duration**. Consequently, the **duration** of the **normal cyclic prefix** and of the **extended cyclic prefix** determine, **each** of them, a **specific limit** to the **cell range**.

- **Cell range limit** from **TDD**

  The **negative impact** of the **TDD** (click 2.3.4) **guard period** (click 11.5.2) on **bitrate** and on **latency** determines a **limit** to the **cell range**.

  At the same time, **TDD** is used in **bands** (click 2.2.13) of **higher frequencies** (**FR1-2** and **FR2**) because of **implementation difficulties** for the **filters** required by **FDD**. In these bands, attenuation is **high** to **very high**, and therefore, the **Tx power** (click 3.10.6.7) also imposes a **cell range limit**.

  Consequently, in **FR1-2** and **FR2 bands** the **cell reach limit** is **jointly** caused by **Tx power** and the **TDD guard period**.

- **Cell range limit** from **RA preamble**

  The characteristics of the **random access** (**RA**, click 9) **procedure** and of the **random access preamble** (click 7.3.1) determine a **limit** to the **cell range**. This is presented with some detail in 3.10.2.2.

**Signal reach** is solved by using **low frequency bands** for **large cells**. **Low frequency bands** (e.g., below **1 GHz**) have **low atmospheric attenuation** (click 12.2.6) and **high indoors penetration**.

### 3.10.2.2 Cell range limit from RA preamble

If the **UE** is **too far** from the **antenna site**, the characteristics of the **RA preamble** might **not** allow the **UE** to **complete successfully** the **random access procedure** (click 9). In this case, the **UE** will **not** be able to **receive service**.

**RA preambles** (click 7.3.1) with **long** preamble **Formats** (**0** to **3**) are designed for **large** and **very large cells** (above around **9 km reach**, and theoretically, up to **100 km reach**). They are called "**long**" for their **large time duration** (from **1** to **4.3 ms**, including the **guard period**). See below for a more detailed explanation on their design.

**RA preambles** with **short** preamble **Formats** (**A1** to **C2**) are designed for **medium**, **small** and **very small cells**. A **few** of these **Formats** support a **maximum reach** up to around **9 km** and many only support a **significantly smaller cell reach**. They are called "**short**" for their **small time duration** (from **2.23 µs** to **857 µs**, including the **guard period**). See below for a more detailed explanation on their design.

**Long preamble Formats**

Their **long PRACH-specific symbol duration** (**200** or **800 µs**) and their **large ZC sequence length** (**839 PRACH-specific symbol**) allows to cope with **larger signal distortion** associated with **large cells**. Their **long Psymbol duration** also allows for a **long PRACH-specific cyclic prefix**, because **each Psymbol** (except the **first one** in **time**) serves as a **cyclic prefix** of the **next Psymbol** in **time**.

Their **long PRACH-specific cyclic prefix** (~**103**, **153** or **684 µs**) and **long Guard Period** (~**97**, **716** or **953 µs**) allow to cope with the **larger** propagation **delay**, and the corresponding **larger multipath** (click 11.2.1) associated with **large cells**.

Their **small subcarrier spacing** (**1.25** or **5 kHz**) makes them more vulnerable to **frequency offsets**. **Frequency offsets** are usually caused by **poor quality oscillators** and/or by **Doppler shift** (click 12.1.8). **Long** preamble **Formats** are sometimes used to provide coverage for **high mobility** scenarios, which implies **high Doppler frequency offsets**. For these reasons, the **restricted Type A** and **restricted Type B sets** have been introduced. These **restricted sets** are based on specific **Zadoff-Chu** (click 11.20.4) **cyclic shifts** for **high Doppler** (click 12.1.8).

**Long** preamble **Formats** can only be used in **FR1 cells**. This is coherent with being designed for **large cells**, because of the **lower attenuation** and **higher penetration** of **lower frequency bands**.

**Short preamble Formats**

They can also use **three subcarrier spacings** (**15** to **960 kHz**) and **three ZC sequence lengths** (**$L_{RA}$=139**, **571** or **1,151 Psyms**). The different **allowed combinations** between one of the **subcarrier spacings** and one of the **ZC sequence lengths** make these preambles very **versatile**.

Their **large subcarrier spacings** makes them considerably **less vulnerable** to **frequency offsets**. This is why it has **not** been specified **restricted sets** to protect **short Formats** from **frequency offsets**. This also provides **protection** against **high Doppler**, allowing their use in **high-mobility scenarios**.

Because of their **small PRACH-specific cyclic prefix duration**, their **shortest** sequence (**$L_{RA}$=139**) can only be used for a **cell reach** up to around **4.7 km**. For **larger cells**, **degradation** occurs, and it is required to use one of the **two longer sequences** (**$L_{RA}$=571** or **1,151**).

The **TDD** frame configuration (click 2.3.4) has an impact on the relative **capacity split** between **Downlink** and **Uplink**, on the **latency**, and also on the **maximum cell reach**. For instance, with **30 kHz subcarrier spacing** and **14 Uplink symbol intervals** in the **frame**, **only** the **short** preamble **Formats** can be used. **Long** preamble **Formats** can only be used with at least **28 consecutive Uplink symbol intervals**.

The **maximum cell reach** with a **short** preamble **Format**, "$L_{RA}$=**139**" and **30 kHz subcarrier spacing** is around **5 km**. If **subcarrier spacing** is reduced to **15 kHz**, the **maximum cell reach** is **increased** to around **10 km**.

**Short** preamble **Formats** can be used in either **FR1** or **FR2 cells**, which is coherent with their design for **medium**, **small** and **very small cells**.

Source web pages:

- https://www.blacktechnoguys.com/2020/03/prach-basic-3.html

### 3.10.3 Cell area

The **cell area** is the **geographical area** where the **cell** provides **service** above the **performance threshold** (click 3.10.1.1).

The **cell area** is **determined** by the **cell reach** (click 3.10.1) at **surface level** in every **horizontal direction** (i.e., **azimuth**).

Since **both Downlink** and **Uplink transmission** are **essential** for **UE** communication, the **cell area** is usually considered the **intersection** between the [**Downlink cell area**] and the [**union** of the [**Uplink cell area**] and the [**SUL** (click 10.4) **cell area**, if configured]].

**5G** is designed **both** for **small cells** in zones with **high traffic density** (e.g., airports or stadiums) and for **large cells** in zones with **very low traffic density** (e.g., **rural** areas).

The **cell area** is usually represented with **clearly determined borders** of a very **neat geometrical shape** (e.g., a hexagon, a circle, a triangle, a radiation lobe, etc). However, the **cell area** is actually **not static** and **quite irregular**. It is **not static** because the **cell reach** is **not static**. It is quite **irregular** because the **cell range** takes **different values** depending on the **direction**. Some of the causes for the **different values** of the **cell reach** in **different directions** are:

- **Antenna radiation pattern**

  The **radiation pattern** (click 11.21.5) of **directional antennas** has **very different values** of **antenna gain** in the **different directions**. The result is usually a **quite irregular cell area**.

- **Natural topography**

  The **cell area** is **usually not flat**, and it may include **mountains**, **hills**, **creeks**, **lakes**, **rivers** and other natural irregularities. These create **shadow** areas and **reflections** that cause **diverse channel conditions** in different **3D directions** from the **antenna site**. The result is usually a **quite irregular cell area**.

- **Constructions**

  The **cell area** usually **includes** diverse **constructions** such as **buildings**, **houses**, **tunnels**, **roads** or **bridges**. These create **shadow** areas and **reflections** that cause **diverse channel conditions** in different **3D directions** from the **antenna site**. The result is usually a **quite irregular cell area**.

A few operators **cover Spain** with roughly **20,000 antenna sites** for **each** of them. The **LTE** coverage in **Spain** is around **97%** of its **surface**. **Spain** is **second EU largest country**, with **506,000 km²**. Dividing **both** results in an **average cell area** of **24.5 km²**, and an average **cell range** of **1.6 km**. This serves as an indicator of what is the **average LTE cell range** and **cell area**.

**A cell is a volume, not an area**

Strictly speaking, the **cell** does **not illuminate** an **area** and it rather **illuminates** a **volume**. In rural deployments it is not unusual that the **basement** of a house has **poor** or none **coverage** while the **top floor** has reasonably good **coverage**. Likewise, in deployments cases the coverage is **not** the **same** in the **different floors** of the same building. The **cell volume** is determined by the **cell reach** in **both azimuth** and **elevation horizontal**.

However, because of the **difficulties** in managing **volume** coverage, **many documents** and **analysis** consider a **cell** as being an **area** at **ground level** and **not** a **volume**.

### 3.10.4 Cell area capacity

The **area capacity** "$C_{cell}^{area}$" of a **cell** is its **total capacity** "$C_{cell}$" divided by its **total area** "$A_{cell}$" (click 3.10.3). Its units are **bits** per **second**, per **square meter** (**bps/m²**):

$$C_{cell}^{area} = C_{cell} / A_{cell}$$

This can be applied for **either Downlink** or **Uplink**.

For example, a **cell** has a **total Downlink capacity** of **2 Mbps** and its **area** is **4 km²**. Then, the **cell**'s **Downlink area capacity** is **500 kbps/km²**.

The **cell area capacity** is **easy** to **define** or **understand**, but it is **difficult** to **compute** or **measure**.

On the one hand, the **total capacity** "$C_{cell}$" of a **given cell** is **not static** and depends on the **location** of the **UEs**, on their **speed**, on the **scheduling strategy**, and on the **environment** (temperature, humidity, noise, etc).

On the other hand, the **total area** "$A_{cell}$" of a **given cell** is also **not static** and depends on **multiple factors** that determine the **cell range** (click 3.10.2.1).

To compute **total cell capacity** "$C_{cell}$" let us start by **computing** the **capacity** "$C^{NR,nUE}$" of the communication between the **NR node antenna** and "$n$" **UEs** located at the **same distance** "$d$" from the **NR node antenna** that **illuminates** the **cell**. The value of "$n$" is the **minimum quantity** of **UEs** that can **absorb all** the **capacity** of the **NR node**.

The **capacity** "$C^{NR,nUE}$" may be modeled using **Shannon**'s **limit** for **each** of the **total quantity** "$N_{layer}$" **layers** that are supported by the **NR node** in this **cell**, resulting in:

The **total capacity** "$C^{NR,nUE}$" of the communication between the **NR node antenna** and "$n$" **UEs** located in the **same place** may be modeled using **Shannon**'s **limit** for **each** of the **total quantity** "$N_{layer}$" **layers** that are supported by the **NR node** in this **cell**, resulting in:

$$C^{NR,nUE} \approx N_{layer} \bullet W_{cell} \bullet log_2(S/NI)$$

where "$W_{cell}$" is the **cell**'s **bandwidth**, "$S$" is the **signal power** at the **receiver** (the **NR node** or **each** of the **UEs**) and "$NI$" is the **noise** plus **interference power** at the **receiver**.

The **received signal power** "$S$" may be approximated with the **beamwidth FSPL** formula (click 12.2.5.3) for a **sphere wedge** and an **omnidirectional antenna**. The **noise**

plus **interference** "$NI$" can be approximated as being a **constant** "$K_{NI}$" multiplied by the **cell**'s **bandwidth** "$W_{cell}$". The result is:

$$C^{NR,nUE} \approx N_{layer} \bullet W_{cell} \bullet log_2([P_t \bullet \lambda^2/8\pi\varphi d^2] / [K_{NI} \bullet W_{cell}])$$

$$C^{NR,nUE} \approx N_{layer} \bullet W_{cell} \bullet log_2(P_t \bullet \lambda^2/8\pi K_{NI} W_{cell} \varphi d^2)$$

$$C^{NR,nUE} \approx N_{layer} \bullet W_{cell} \bullet [log_2(P_t \bullet \lambda^2/8\pi K_{NI} W_{cell} \varphi) - 2log_2(d)]$$

where "$P_t$" is the **transmitted power** (by the **NR node** or **adding** the one of the "$n$" **UEs**), "$\lambda$" is the **wavelength**, "$\varphi$" is the **NR node antenna beamwidth** and "$d$" is the **distance** (see above).

The **cell area capacity** can be **modeled** as the **average** of the **above expression** over **all** the valid **cell area**. Notice that a more accurate average could be computed by taking into account the **scheduling algorithm**, which may assign **more resources** to **UEs** that are **closer** to the **antenna site**.

The distance "$d$" **ranges** from "$d_{min}$" to the **cell range** "$cr$". The value of "$d_{min}$" can be derived from the following **condition** (click 12.2.5.1) of the **FSPL** formula:

$$(4\pi d/\lambda)^2 > G_t^v \bullet G_r^v$$

From the **beamwidth FSPL** formula indicated above we know that "$G_t^v \bullet G_r^v = 2\pi/\varphi$", and consequently:

$$(4\pi d/\lambda)^2 > 2\pi/\varphi$$

$$d > \lambda \bullet (1/8\pi\varphi)^{1/2}$$

$$d_{min} = 5\lambda \bullet (1/8\pi\varphi)^{1/2}$$

The **average** of "$log_2(d)$" over the **surface** of a **circular ring** between "$d_{min}$" and the **cell range** "$cr$" is:

$$Avg_{dmin}^{cr}(log_2(d), annulus) = (cr^2 log_2(cr) - d_{min}^2 log_2(d_{min})) / (cr^2 - d_{min}^2) - 1/2ln(2)$$

Since "$cr >> d_{min}$" and "$log_2(cr) >> 1/2ln(2)$" we can **approximate** the **average** to:

$$log_2(cr)$$

The **cell area capacity** is therefore:

$$C_{cell}^{area} \approx N_{layer} \bullet W_{cell} \bullet [log_2(P_t \bullet \lambda^2/8\pi K_{NI} W_{cell} \varphi) - 2log_2(cr)]$$

$$\approx N_{layer} \bullet W_{cell} \bullet log_2(P_t \bullet \lambda^2/8\pi K_{NI} W_{cell} \varphi cr^2)$$

Recall that the **FSPL formula** (click 12.2.5.1) is for **vacuum**, and therefore, this model **only** serves for **very low attenuation** (i.e., "$f \leq 1$ GHz").

The behavior of a **real radio system** is considerably **more complex** than this model because of its **multiple characteristics** (i.e., cross-interference between **layers**, cross-interference between neighbor cells, self-interference, fading, implementation issues, ...). Even so, this model is quite useful to illustrate the **effect** of **each factor** ("$N_{layer}$", "$W_{cell}$", "$P_t$", etc) in the **cell area capacity**. This model is also quite useful to illustrate the **densification techniques** (click 3.10.6).

Notice that for a **quantity** of **UEs** that is **larger** than "$n$" the **cell** can receive a **higher Uplink throughput** than the one computed above, because their **joint transmission power** is higher than the one considered above.

If the operator wants to serve **more customers** (**UEs**) in a given **area** and/or to provide **more capacity** per **customer** (**UEs**), the operator has to perform **densification** (i.e., to **increase** the **area capacity**). The **area capacity** "$C_{cell}^{area}$" of **1 cell** is the **total cell capacity** "$C_{cell}$" divided by the **cell**'s **coverage area** "$A_{cell}$":

$$C_{cell}^{area} = C_{cell} / A_{cell}$$

As a **simple model**, the cell **capacity** "$C_{cell}$" may be modeled using **Shannon**'s **capacity equation** for **each** of the "$N_{layer}$" **layers**, resulting in:

$$C_{cell}^{area} \approx N_{layer} \cdot W_{cell} \cdot log_2(S/NI) / A_{cell}$$

where "$W_{cell}$" is the **cell**'s **bandwidth**, "$S$" is the **signal power** at the **receiver** (the **NR node** or the **UE**) and "$NI$" is the **noise** plus **interference power** at the **receiver**. Replacing **both** the **received signal** "$S$" (click 12.2.5) and the **noise** plus **interference** "$NI$" (click 4.9.3.1) we get:

$$C_{cell}^{area} \approx N_{layer} \cdot W_{cell} \cdot log_2([P_t \cdot G_t \cdot G_r / k cr^2] / [K_{NI} \cdot W_{cell}]) / A_{cell}$$

where "$P_t$" is the **transmitted power**, "$G_t$" and "$G_r$" are (respectively) the **transmitter antenna gain** (click 11.21.2) and the **receiver antenna gain**, and "$cr$" is the **cell range** (click 3.10.2) that we have used instead of the distance "$d$". The factors "$k$" and "$K_{NI}$" are **constants** to adjust the **proportions**, among others, the replacement of "$d$" by "$cr$". The **dependency** from "$\lambda$" (click 12.2.5) is **not considered** because the **operators cannot** modify significantly the cell's **central frequency** "$f=3 \cdot 10^8/\lambda$" without **very major implications**.

Based on the **above model**, increasing the **cell**'s **area capacity** "$C_{cell}^{area}$" can be achieved by:

- **Increase** the **quantity** of **layers** "$N_{layer}$".
- **Increase** the **bandwidth** "$W_{cell}$".
- **Increase** the **transmitted power** "$P_t$".
- **Increase** the **transmitter gain** "$G_t$".
- **Increase** the **receiver gain** "$G_r$".
- **Reduce** the **cell range** "$\sim d$".
- **Reduce** the **cell area** "$A_{cell}$".

**increasing** the **bandwidth** "$BW_{cell}$" and/or the **signal power** "$S$" and/or by **reducing** the **cell area** "$A_{cell}$". The **operator cannot** (in the general case) **reduce** "$NI_{Rx}$" because it is due to **external causes** that the **operator cannot control**.

## 3.10.5 Coverage

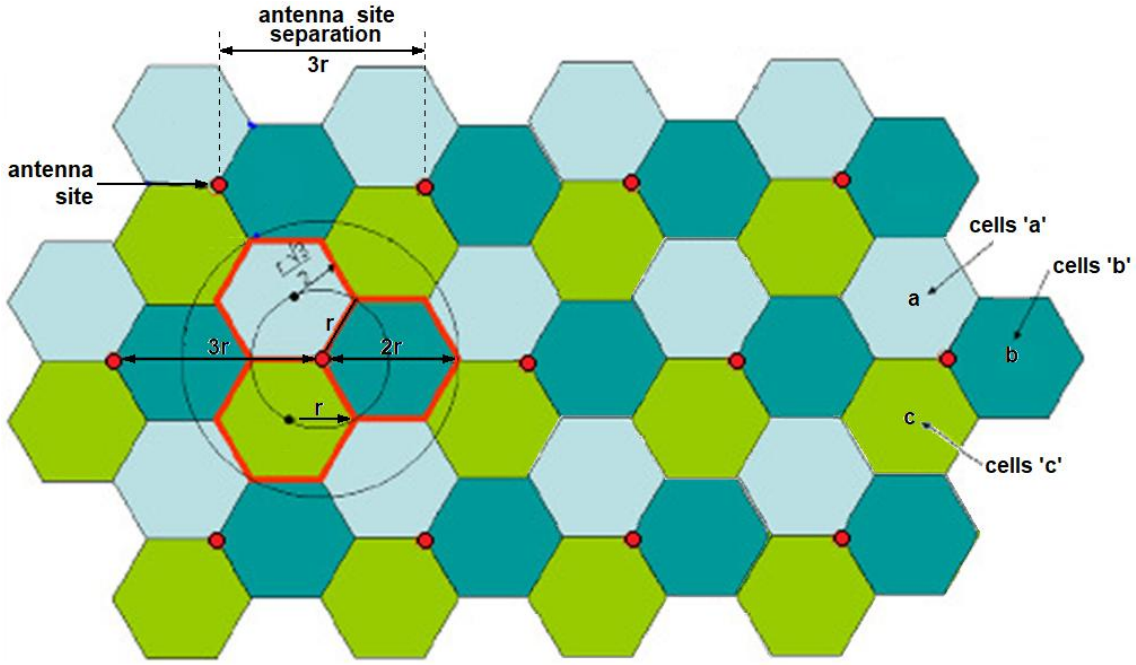
The **coverage** of the **radio access network** is the **union** of **all** the **cell areas** (click 3.10.3). The **desired** (or **regulated**) **coverage** is typically achieved by **deploying macrocells** that **jointly** cover the **corresponding area** above the **corresponding performance threshold** (click 3.10.1.1). This type of **coverage planning** can be performed by **tessellation**.

**Tessellation** consists of defining **1** or **more shape**(s) (called **tiles**) whose **combination** allows to **cover** a **plane surface** with **no overlaps** and **no gaps**. An example of **tile** is a **square**: it is possible to cover a **plane surface** with **no overlaps and no gaps** using **equally sized square tiles**.

In coverage planning a **frequently used tile** is an **hexagon**: it is possible to cover a **plane surface** with **no overlaps and no gaps** using **equally sized hexagonal tiles**.

For the deployments of **legacy** mobile communication systems, **each antenna site** defined **1 cell all around it**. **Each cell** was modeled as **1 hexagonal tile**, with the **antenna site** at its **center**, and the **target area** was covered with **such tiles**.

In **LTE**, **tri-sector sites** with **120º beamwidth antennas** were used very frequently. In this case, **each sector** was modeled as an **hexagonal tile**, with the **antenna site** at the **center** of **three hexagonal tiles**. Although the **equipment cost** of a **tri-sector site** is higher than the one of an **omnidirectional site**, the **cost different** is **low**, and the **area capacity** (click 3.10.4) is **almost tripled** (click 3.10.6.3) The **cell range** (click 3.10.2) is also usually higher. These aspects explain why **tri-sectorial site deployment** is a **very frequent** deployment technique to achieve the desired **coverage**. The following figure shows a simple example of this approach:




Some **relevant remarks** on this figure are:

- **Each red dot** is an **antenna site**.
- **Each antenna site** illuminates **3 sectors** (i.e., **cells**) around it. An **antenna site** and its **3 cells** has been **highlighted** with a **red border**.
- **Each** of the **3 cells** around **each antenna site** is shown with a **different color** (**cells** "**a**" in light blue, **cells** "**b**" emerald and **cells** "**c**" in light green).
- "***r***" is the **cell range** (click 3.10.2), and "2***r***" is the **cell diameter**.
- "3***r***" is the **distance** between **every 2 adjacent antenna sites**.

The situation in practice is obviously **much** more **diverse**: you have cells of **different sizes**, of **different technologies**, with **different capacities**, with **different geometry** (e.g., two narrow opposing sectors for road covering, or only one sector to illuminate a town from a hilltop) and **different characteristics** (e.g., **SILO**/**MIMO**, **FDD**/**TDD**, …). However, the **tessellation** basic principle **still holds**.

**Each operator** indicates its **overall coverage** as the **fraction** of the **target area** (e.g., **one country**) where there is **coverage**. The **fraction** of the **target area** is usually measured **both** in **population** and in **surface**, because **both metrics** are **relevant**. **Population coverage** is (almost) **always higher** than **surface coverage** because deployments (almost) **always leave out** the **less populated areas**.

Additionally, **each operator** usually offers a public on-line **coverage map**, that indicates the **specific areas** where it **provides coverage**.

### 3.10.6 Densification techniques

The **main objective** of **densification techniques** is the **increase** of the **area capacity** of the **RAN deployment**. The **area capacity** is increased in **urban** areas **and** in areas of **high customer** (**UE**) **density**, like business districts, railway stations and sport stadiums.

If the operator wants to serve **more customers** (**UEs**) in a given **area** and/or to provide **more capacity** per **customer** (**UEs**), the operator has to perform **densification** (i.e., to **increase** the **area capacity**). The **area capacity** "$C_{cell}^{area}$" of **1 cell** can be **modelled** (click 3.10.4) for "$f \leq 1$ GHz" as:

$$C_{cell}^{area} \approx N_{layer} \cdot W_{cell} \cdot log_2(P_t \cdot \lambda^2 / 8\pi K_{NI} W_{cell} \varphi cr^2)$$

On the one hand, the **wavelength** is mainly determined by the **bands** that are **allocated** to **5G** by the **national authorities** (click 2.2.2.1) and **each operator** can just **opt** to **license channels** in the corresponding **bands**. Moreover, this **model only** serves for **low frequencies**, **not** for every value of "$\lambda$". On the other hand, the **operators** do **not** have **direct control** over the **noise** and **interference factor** "$K_{NI}$". Consequently, for the purpose of **densification**, "$\lambda$" and "$K_{NI}$" are **not** considered **variables**. The resulting **cell area capacity model** that we are going to use is:

$$C_{cell}^{area} \approx N_{layer} \cdot W_{cell} \cdot log_2(k \cdot P_t / W_{cell} \varphi cr^2)$$

where "$k = \lambda^2 / 8\pi K_{NI}$" is considered constant.

Based on this **model**, **increasing** the **cell**'s **area capacity** "$C_{cell}^{area}$" can be achieved by:

- **Increasing** the **quantity** "$N_{layer}$" of **layers**.
- **Increasing** the **bandwidth** "$W_{cell}$".
- **Increasing** the **transmitted power** "$P_t$".
- **Reducing** the **sector width** (i.e., the **beamwidth**) **area** "$\varphi$".
- **Reducing** the **cell range** "$cr$".

There are several **densification techniques** that can be combined in order to **optimize** the **cost-capacity** of the **deployment**. These **densification techniques** can be **classified** as **either spectrum densification**, **space densification** or **power densification**:

- **Spectrum densification**

  It consists of **increasing** the **bandwidth** "$W_{cell}$" (i.e., **more spectrum**) at **each** of the **existing antenna sites**. **Spectrum** is expensive (click 2.2.2.1) but **antenna sites**, **equipment** and **operation** are also expensive (click 4.7.8).

  Notice that **spectrum mutualization** between **several operators** using some **RAN sharing** (click 10.16) technique increases the **area capacity only** because of **resource pooling** (unless of course part of the mutualized spectrum was previously unused). However, **spectrum mutualization increases** the **UE-in-**

**cell capacity** (click 4.9.3.3) which is greatly **advantageous** in **rural areas** where **cell load** might be **low**.

There are **2 spectrum densification techniques**:

- **Channel bandwidth densification** (click 3.10.6.1)

  It consists of **increasing** the **channel bandwidth** of the **existing cells**.

- **Carrier densification** (click 3.10.6.2)

  It consists of **deploying more carriers** (i.e., **more cells**) in additional **new channels**, in the **existing antenna sites**.

- **Space densification**

  It consists of **increasing** the **space reusability** of the **spectrum** in **each existing cell**. There are **4 space densification techniques**:

  - **Sector densification** (click 3.10.6.3)

    It consists of **increasing** the **quantity** of **sectors** in **each existing cell**, which is the same as **reducing** the **sector width** "**φ**".

  - **Beamforming densification** (click 3.10.6.4)

    It consists of **increasing** the **quantity** of **beams** per **cell** and/or of **narrowing** the **beams**.

  - **MIMO densification** (click 3.10.6.5)

    It consists of **increasing** the **total quantity** of **multiple input multiple output** (**MIMO**, click 2.4.3) **layers** in **each existing cell**.

  - **Site densification** (click 3.10.6.6)

    It consists of **increasing** the **quantity** of **antenna sites**, by **reducing** their **cell range** "***cr***", while **keeping** the **existing coverage area**.

- **Power densification**

  It consists of **increasing** the **transmitted power** "$P_t$" with the **existing cell range** "***cr***". There is **1 power densification technique**:

  - **Power densification** (click 3.10.6.7)

    It consists of **increasing** the **NR node transmission power** with the **existing cell range** "***cr***".

- **Combined space and spectrum densification**

  It consists of **increasing both** the **quantity** of **antenna sites** and the **available spectrum**. There is **1 combined space** and **spectrum densification technique**:

  - **Site and spectrum densification** (click 3.10.6.8)

    It consists of **deploying additional antenna sites** with **additional spectrum**, whose **cell areas overlap** with the **areas** of the **existing cells** with **existing spectrum**.

Most **densification techniques** also allow to **increase** the **UE-in-cell capacity** (i.e., the **maximum capacity** of the **UE** in a **cell**, click 4.9.3.3), in addition to the cell **area capacity** (click 3.10.4).

The following table summarizes the different **densification techniques** indicating what **factor** of the model **formula** is **modified** by them, what is the **cell area capacity growth**, and a rough estimation of the **relative cost among themselves**:

| Densification technique | Modifies | Area capacity growth | Cost |
|---|---|---|---|
| **Channel bandwidth densification** (3.10.6.1) | $\Delta W_{cell}$ | $\propto W_{cell} \cdot log_2(1/W_{cell})$ | €€ |
| **Carrier densification** (3.10.6.2) | $\Delta W_{cell}$ | $\propto W_{cell} \cdot log_2(1/W_{cell})$ | €€€ |
| **Sector densification** (3.10.6.3) | $\nabla \varphi$ | $\propto log_2(1/\varphi)$ | € |
| **Beamforming densification** (3.10.6.4) | $\nabla \Omega$ | $\propto log_2(1/\Omega)$ | € |
| **MIMO densification** (3.10.6.5) | $\Delta N_{layer}$ | $\propto N_{layer}$ | €€ |
| **Site densification** (3.10.6.6) | $\nabla cr$, $\Delta Sites$ | $\propto log_2(1/cr^2)$ | €€€€ |
| **Power densification** (3.10.6.7) | $\Delta P_t$ | DL: $\propto log_2(P_t)$ UL: 0 | € |
| **Site and spectrum densification** (3.10.6.8) | $\Delta W_{cell}$, $\Delta Sites$ | DL: $\propto W_{cell} \cdot log_2(k \cdot P_t/W_{cell}\varphi cr^2)$ UL: $\propto W_{cell} \cdot log_2(9/4 W_{cell})$ | €€€€€ |

Some relevant remarks on this table are:

- "**DL**" and "**UL**" stand for **Downlink** and **Uplink**. For the densification techniques where neither "**DL**" nor "**UL**" is indicated, the **growth** applies to **both Downlink** and **Uplink**.
- "**Δ**" is the symbol for "increases".
- "**∇**" is the symbol for "decreases".
- "**=**" is the symbol for "stays the same".
- "**∝**" is the symbol for "proportional to".

It is **possible** to **combine several densification techniques** in each deployment re-planning instance.

### 3.10.6.1 Channel bandwidth densification

**Channel bandwidth densification** consists of **increasing** the **channel bandwidth** "$W_{cell}$" of the **existing cells**.

This technique does **not** introduce **new cells**.

**Cell area capacity growth**

Based on the **selected model** (click 3.10.6):

$$C_{cell}^{area} \approx N_{layer} \cdot W_{cell} \cdot log_2(k \cdot P_t/W_{cell}\varphi cr^2)$$

the **cell area capacity** "$C_{cell}^{area}$" (click 3.10.4) grows **proportionally** to "$W_{cell} \cdot log_2(1/W_{cell})$". This is **below** a **direct proportion** to the **channel bandwidth** "$W_{cell}$" because **more bandwidth** also implies a **higher noise** and **interference** "$K_{NI} \cdot W_{cell}$".

Notice that we are **assuming** that the **total radiated power** "$P_t^{site}$" by the **antenna site** stays the **same**. In case that "$P_t^{site}$" can be increased, this would be a combined case with **power densification** (click 3.10.6.7).

**Restrictions**

A **very severe restriction** is that the **additional spectrum** must be **contiguous** to the **channel** of the **existing cells**.

**Growth limit**

One **limit** arises from the **channel transmission bandwidth** (click 2.2.3.6) in the corresponding **band**, beyond which it is **not possible** to **increase** the **cell**'s **channel bandwidth**.

Another **limit** arises from the **spectrum cap** to a **single operator** that might have been determined by the **national authorities** (click 2.2.2.1) in the corresponding **band**.

**Cost aspects**

The **main cost** arises from the **additional spectrum cost** (click 2.2.2.1).

The **additional cost** of the **CAPEX** of the required **equipment** is usually **low**, because on some cases it only requires **equipment** reconfiguration or some **RF equipment adjustments**.

**Increase of UE-in-cell capacity**

It increases the **UE-in-cell capacity** (click 4.9.3.3) for **every UE** that supports the additional **channel bandwidth**.

### 3.10.6.2 Carrier densification

**Carrier densification** consists of **deploying more carriers** (i.e., **more cells**) in additional **new channels**, in **existing antenna sites**.

Do **not confuse site** and **spectrum densification** (click 3.10.6.8) with **carrier densification**: **both** add **more spectrum**, but the **former** adds **more sites**, while the **latter** does **not**.

**Cell area capacity growth**

Based on the **selected model** (click 3.10.6):

$$C_{cell}^{area} \approx N_{layer} \cdot W_{cell} \cdot \log_2(k \cdot P_t / W_{cell} \varphi cr^2)$$

the **cell area capacity** "$C_{cell}^{area}$" (click 3.10.4) grows **proportionally** to "$W_{cell} \cdot \log_2(1/W_{cell})$". This is **below** a **direct proportion** to the **channel bandwidth** "$W_{cell}$" because **more bandwidth** also implies a **higher noise** and **interference** "$K_{NI} \cdot W_{cell}$".

Notice that we are **assuming** that the **total radiated power** "$P_t^{site}$" by the **antenna site** stays the **same**. In case that "$P_t^{site}$" can be increased, this would be a combined case with **power densification** (click 3.10.6.7).

**Restrictions**

The **new carrier must** belong to one of the allowed **carrier aggregation** (**CA**, click 10.2) combinations.

**Growth limit**

The growth is **limited** by the **maximum quantity** of **aggregated carriers**.

**Cost aspects**

The **main cost** arises from the **additional spectrum cost** (click 2.2.2.1).

An **additional cost** arises from the required **equipment CAPEX**: it typically requires to add a complete **Radio Unit** (**RU**, click 4.7.3) for **each sector** in the **each new cell**, and possibly also **Distributed Units** (**DUs**, click 4.7.2).

**Increase of UE-in-cell capacity**

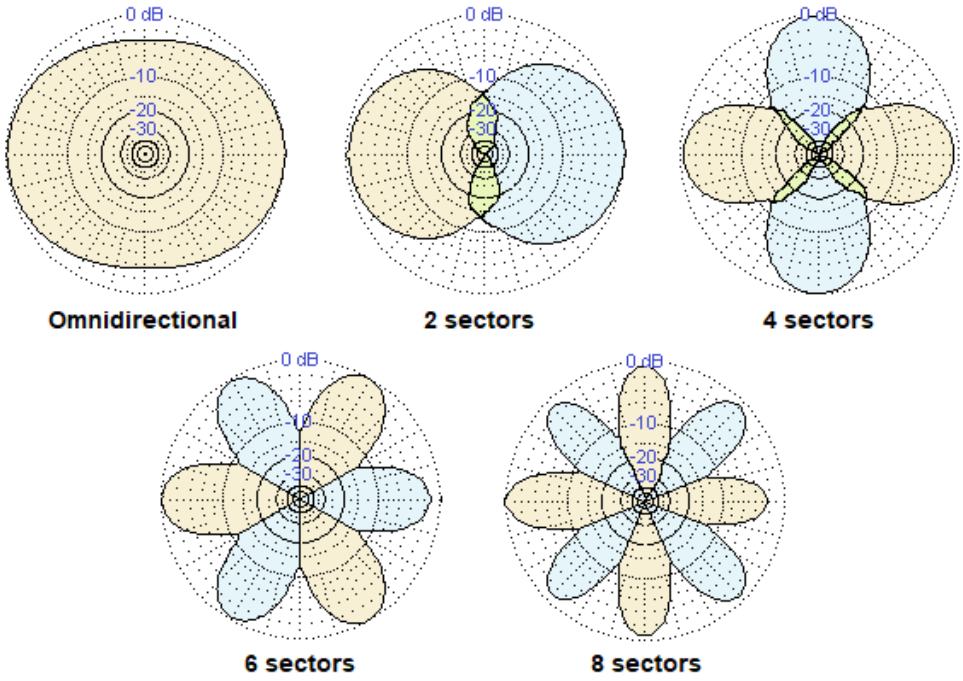
It increases the **UE-in-cell capacity** (click 4.9.3.3) for **every UE** that supports the additional **carrier**.

### 3.10.6.3 Sector densification

**Sector densification** consists of **increasing** the **quantity** of **sectors** in **existing antenna sites**.

**Sector densification** is sometimes called **sectorization**.

The following figure shows the **horizontal radiation patterns** (click 11.21.5) from a **single sector** (which is quite unusual) up to **eight sectors**:




**Cell area capacity growth**

Based on the **selected model** (click 3.10.6):

$$C_{cell}^{area} \approx N_{layer} \bullet W_{cell} \bullet \log_2(k \bullet P_t / W_{cell} \varphi c r^2)$$

the **cell area capacity** "$C_{cell}^{area}$" (click 3.10.4) grows **proportionally** to "$\log_2(1/\varphi)$", where "$\varphi$" is the **beamwidth** of **1 sector**:

**Restrictions**

No specific restrictions.

**Growth limit**

One **limit** arises from **co-channel interference** between **neighbor sectors**: if **sectors become too narrow**, the **co-channel interference becomes unmanageable**.

Another **limit** arises from **mobility**: if **sectors become too narrow**, a **mobile UE** crossing the **sectors** might **not** have enough time to perform **handover** and **cell access**.

**Sites** with **up to 8 sectors** are deployed in **5G RAN**. A higher quantity of sectors might not provide the expected benefits.

**Cost aspects**

The **main cost** arises from the required **equipment CAPEX**: it typically requires to add a complete **Radio Unit** (**RU**, click 4.7.3) for **each beam**, and possibly also additional **Distributed Units** (**DUs**, click 4.7.2).

**Increase of UE-in-cell capacity**

It increases the **UE-in-cell capacity** (click 4.9.3.3) for **every UE** whose **intrinsic capacity** (click 4.9.3.2) is **higher** than the **pre-existing UE-in-cell capacity**.

Source web pages:

- https://mastdatabase.co.uk/blog/2021/08/04/mast-sectorisation-radiation-and-deployment-patterns/
- https://www.commscope.com/blog/2014/cells-sectors-and-antenna-beamforming/
- https://www.sharetechnote.com/html/5G/5G_QCL.html

### 3.10.6.4 Beamforming densification

**Beamforming** (click 2.4.1) **densification** consists of **increasing** the **quantity** of **beams** per **cell** and/or of **narrowing** the **beams**.

It is possible to **dynamically** create **beams** with the **desired beamwidth** and **direction**, that selectively **illuminate** a **UE** or **group** of **nearby UEs**. **Each beam increases** the **antenna gain** in the **beam direction**.

**Cell area capacity growth**

For **vertical beams** (i.e., **sphere wedges**), based on the **selected model** (click 3.10.6):

$$C_{cell}^{area} \approx N_{layer} \cdot W_{cell} \cdot log_2(k \cdot P_t / W_{cell} \varphi c r^2)$$

the **cell area capacity** "$C_{cell}^{area}$" (click 3.10.4) grows **proportionally** to "$log_2(1/\varphi)$", where "$\varphi$" is the **beamwidth**:

For **circular beams** (i.e., **sphere sectors**), we adjust the **selected model** (click 3.10.6) based on the **beamwidth FSPL** formula (click 12.2.5.3) for a **sphere sector** of "$\Omega$" steradians as follows:

$$C_{cell}^{area} \approx N_{layer} \cdot W_{cell} \cdot log_2(2k \cdot P_t / W_{cell} \Omega c r^2)$$

With this adjusted model, the **cell area capacity** "$C_{cell}^{area}$" (click 3.10.4) grows **proportionally** to "$log_2(1/\Omega)$", where "$\Omega$" is the **beamwidth**.

**Restrictions**

No specific restrictions.

**Growth limit**

For **beam sweeping**, a **limit** arises from **beam synchronization** with **time scheduling**, in particular **with high-speed mobility**: if **beams** become **too narrow**, they might **lose synchronization** with **temporal scheduling**.

For **direct beams**, a **limit** arises from **beam positioning accuracy**, in particular **with high-speed mobility**: if **beams** become **too narrow**, they may **miss** the **target UE(s)**.

**Cost aspects**

The **main cost** arises from the **CAPEX** of the **analog** or **digital beamforming equipment** to be **placed** at **each** **Radio Unit** (**RU**, click 4.7.3).

**Increase of UE-in-cell capacity**

It increases the **UE-in-cell capacity** (click 4.9.3.3) for **every UE** whose **intrinsic capacity** (click 4.9.3.2) is **higher** than the **pre-existing UE-in-cell capacity**.

### 3.10.6.5 MIMO densification

**MIMO densification** consists of **increasing** the **total quantity** of **multiple input multiple output** (**MIMO**, click 2.4.3) **layers** in **each existing cell**.

**Cell area capacity growth**

Based on the **selected model** (click 3.10.6):

$$C_{cell}^{area} \approx N_{layer} \cdot W_{cell} \cdot log_2(k \cdot P_t / W_{cell} \varphi cr^2)$$

the **cell area capacity** "$C_{cell}^{area}$" (click 3.10.4) grows **linearly** with the **total quantity** "$N_{layer}$" of **layers** in the **cell**:

It is well known that **each** **layer** **cannot completely** deliver the **full capacity** of a **single** **layer** system, and consequently the **actual growth** is **below linear**. In spite of this, the **cell area capacity growth** of **MIMO densification** is **excellent**.

**Restrictions**

No specific restrictions.

**Growth limit**

One **limit** arises from the **technological capacity** to build **massive MIMO antenna arrays**, in particular for **low frequencies**.

Another **limit** arises from the **accuracy** of the **channel state information** (**CSI**, click 2.4.5) that can be **achieved**.

A third **limit** arises from the **quantity** of **active UEs** in the **cell**: beyond the **summatory** of the **quantity** of **layers** supported **by each active UE**, there is **no cell area capacity growth**. This is particularly relevant for **high frequencies**, where the **cell area** (click 3.10.3) is **quite small**.

**Cost aspects**

The **main cost** arises from the **CAPEX** of the deployment of more **Radio Unit** (**RU**, click 4.7.3) and possibly of more **Distributed Units** (**DUs**, click 4.7.2). Recall from 2.4.3.5 that **each** **layer** requires a **complete RF chain** (click 4.5.8.1).

**Increase of UE-in-cell capacity**

It increases the **UE-in-cell capacity** (click 4.9.3.3) for **every UE** whose supported **quantity** of **layers** (click 4.9.3.2) is **higher** than the **pre-existing quantity** of **layers** in the **cell**.

### 3.10.6.6 Site densification

**Site densification** consists of **increasing** the **quantity** of **antenna sites**, by **reducing** their **cell range** "$cr$", while **keeping** the **existing coverage area**. This is conceptually similar to **replanning** the **coverage** (click 3.10.5) in the **target area** with a **smaller tile**

**size**. Replanning does **not** mean that **all** the **antenna sites** are **new**: just to **deploy some additional antenna sites** such that the **new pattern** conforms to a **tessellation** with **smaller tile size**. Notice that there is **no bandwidth** "$W_{cell}$" **increase** (only the existing channels are used).

The **site densification** may be **permanent** or **temporary**. For **stational** (**touristic sites**) or **punctual reasons** (music festival, local or state fairs, disaster areas,…) the **site densification** may be **temporary**. These **temporary antenna sites** are typically deployed with an **all-comprised antenna site** (including a **telescopic antenna mast**) inside a **truck** or a **trailer**. This type of equipment is sometimes called **cell on wheels** (**COW**).

The following picture shows a **cells on wheels** by **Telefonica** (**Movistar**) with the folded antenna mast on top of the truck:




This picture is from Madrid (Spain).

The following picture shows a **cells on wheels** by **Telefonica** (**Movistar**) with the extended and secured extended telescopic antenna mast. The extended telescopic antenna mast of a second cell on wheels may be seen in the back.




This specific deployment was performed for a one-day special event in Madrid (Spain).

As a somehow extreme example of **seasonal demand fluctuations**, the town of **Noja** in north Spain has **100.000 inhabitants** in the **summer**, and **only 2.700** in the **winter**!

**Site densification** is also called "**tower densification**".

Do **not confuse site** and **spectrum densification** (click 3.10.6.8) with **site densification**: **both** add **more sites**, but the **former** adds **more spectrum**, while the **latter** does **not**.

Source web page:

- https://www.elmundo.es/papel/historias/2025/08/03/688ce854fdddffde518b4570.html

**Cell area capacity growth**

Based on the **selected model** (click 3.10.6):

$$C_{cell}^{area} \approx N_{layer} \cdot W_{cell} \cdot log_2(k \cdot P_t / W_{cell} \varphi cr^2)$$

the **cell area capacity** "$C_{cell}^{area}$" (click 3.10.4) grows **proportionally** to "$log_2(1/cr^2)$".

**Restrictions**

The **location** of the **existing antenna sites** may **not match** so well the **new tessellation**.

**Growth limit**

The main **limit** arises from **cost considerations**, because the **total quantity** of **antenna sites** is **proportional** to "$1/cr^2$". If the **cell range** "$cr$" is reduced to **one half** "$cr/2$", the **total quantity** of **antenna sites** is **quadrupled!** The **newly required antenna sites** would be **threefold** the **current quantity** of **antenna sites!**

For **outdoor sites**, another **limit** arises from the **minimum cell size** that can sustain **mobility**, because of the required **time** for **handover** and **cell access**.

**Cost aspects**

**Site construction**, including the **backhaul** connectivity, is **considerably more expensive** than the corresponding **telecom equipment**. This technique therefore requires a **considerable investment**.

**Increase of UE-in-cell capacity**

It increases the **UE-in-cell capacity** (click 4.9.3.3) for **every UE**.

### 3.10.6.7 Tx power densification

**Power densification** consists of **increasing** the **transmission power** with the **existing cell range** "$cr$".

**Cell area capacity growth**

Based on the **selected model** (click 3.10.6):

$$C_{cell}^{area} \approx N_{layer} \cdot W_{cell} \cdot log_2(k \cdot P_t / W_{cell} \varphi cr^2)$$

the **cell area capacity** "$C_{cell}^{area}$" (click 3.10.4) grows **proportionally** to "$log_2(P_t)$" in the **Downlink transmission direction** (**not** in **Uplink**).

**Restrictions**

**Power densification** is **only feasible** if the **increased NR node Tx power complies** with the **regulations** and with the **maximum interference** with **neighbor cells** in the **overlapping channels**.

**Growth limit**

The **limit** arises from the **power** emission **regulations**.

For **each UE**, the **maximum transmitted power** "$P_t$" is **23 dBm** (click 4.9.1) on most cases, although on a few particular cases it may reach **35 dBm**. The **UE transmits** at its **maximum possible power** (as instructed by the **NR node**). Consequently, **power densification cannot** be applied in the **Uplink transmission direction**. As a side comment, a **small** improvement on the **cell area capacity** "$C_{cell}^{area}$" might be obtained through the **scheduling algorithm**, by assigning the existing **Uplink resources** to a **higher quantity** of **UEs**, in which case the **total Uplink power** is **increased**.

For the **NR node**, the regulation is usually established in terms of **power density flux**, which has to be **computed** by taking into account **all** the **radiating equipment** in the **antenna site**, plus, also the one in **nearby antenna sites**. For **dynamic systems** (i.e., **beamforming** and **MIMO**), the **beamwidth** (i.e., **antenna gain**) and **power concentration** has also to be taken into account.

**Cost aspects**

The **main cost** arises from the **OPEX cost** of an **increased electricity bill**.

Some **CAPEX equipment cost** (i.e., upgrading the power amplifiers) might also be required.

**Increase of UE-in-cell capacity**

It increases the **Downlink UE-in-cell capacity** (click 4.9.3.3) for **every UE**.

Source web page:

- https://5ghub.us/5g-transmit-power-and-antenna-radiation/

### 3.10.6.8 Site and spectrum densification

**Site and spectrum densification** consists of **deploying additional antenna sites** with **additional spectrum**, whose **cell areas overlap** with the **areas** of the **existing cells** with **existing spectrum**.

Same as the **site densification**, the **site** and **spectrum densification** may be **permanent** or **temporary** (click 3.10.6.6).

Do **not confuse site** and **spectrum densification** with **carrier densification** (click 3.10.6.2): **both** add **more spectrum**, but the **former** adds **more sites**, while the **latter** does **not**.

Do **not confuse site** and **spectrum densification** with **site densification** (click 3.10.6.6): **both** add **more sites**, but the **former** adds **more spectrum**, while the **latter** does **not**.

**Cell area capacity growth**

Based on the **selected model** (click 3.10.6):

$$C_{cell}^{area} \approx N_{layer} \bullet W_{cell} \bullet log_2(k \bullet P_t / W_{cell} \varphi cr^2)$$

the **Downlink cell area capacity** "$C_{cell}^{area}$" (click 3.10.4) grows **proportionally** to the **complete Downlink cell area capacity** "$N_{layer} \bullet W_{cell} \bullet log_2(k \bullet P_t / W_{cell} \varphi cr^2)$" of **each newly deployed cell**, that is aggregated as an **additional carrier**. This is based on the **assumption** that the **new sites** do **not imply** a **power reduction** of the **existing sites**. Moreover, it is assumed that the **UE** is capable of **dual connectivity** (**DC**, click 10.3) and it can therefore **aggregate carriers** from **2 NR nodes**. Under these assumptions, the **new sites** transmit their corresponding **NR Tx power**, and consequently, the total **Downlink cell area capacity** in the **cell area** of the **newly deployed cells** is a **net increase**.

The **Uplink cell area capacity** grows due to **additional spectrum** "$W_{cell}$" and to the **reduced distance** between **some UEs** and the **newly deployed site**. This can be estimated as if the **cell range** were smaller. A **very rough estimation** is to consider the **cell range reduced** in "**2/3**". In this case, the **Uplink cell area capacity** grows **proportionally** to "$W_{cell} \bullet log_2(9/4 W_{cell})$"

For **UEs** without **DC capacity**, the **cell area capacity growth** depends on the **location** and **capacity** of **each new site** in respect to the **location** and **capacity** of the previously **existing neighbor sites**.

**Restrictions**

Its **full advantages** are **only** achieved for **UEs** with **dual connectivity**. If **not**, there is a **lower growth** in **cell area capacity**.

**Growth limit**

One **limit** arises from **maximum** available **spectrum** to **one operator**.

For **outdoor sites**, another **limit** arises from the **minimum cell size** that can sustain **mobility**, because of the required **time** for **handover** and **cell access**.

**Cost aspects**

There are **two relevant cost** elements: the **additional spectrum** and the **site construction** cost. This densification technique is therefore the **most expensive**.

**Increase of UE-in-cell capacity**

It increases the **UE-in-cell capacity** (click 4.9.3.3) for **every UE** that supports **dual connectivity**, and to a **lesser extent**, to the **other UEs**.

# 4. NR nodes and UEs

For more detail on **NR nodes**, refer to clauses 4.1 and 4.2 of TS 38.300.

## 4.1 NR node definition

A **NR node** is a **network function** whose **antennas** illuminate **1** or **more cells** (click 3.1.1) where the **UEs camp** to receive service. The **NR node** is responsible for **all radio-related** functions in the **cell**(s) **illuminated** by its **antenna**(**s**). The **NR node transmits Downlink frames** and **signals** and **receives Uplink frames** and **signals**.

Some examples of the **radio-related functions** of a **NR node** are **radio resource management**, **admission control**, **connection establishment**, **routing** of **user-plane data** to the **UPF**, **routing** of **control-plane information** to the **AMF** and **quality-of-service** (**QoS**) **flow management**.

The **union** of the **cell areas** (click 3.10.3) that are illuminated by the **antennas** of a **NR node** is the **coverage area** of the **NR node**.

**Two NR nodes** that have **at least one neighbor cell pair**, such that **each cell** in the **neighbor pair** belongs to **each** of the **two nodes**, are called **neighbor NR nodes**.

The **radio access network** (**RAN**) of a **PLMN** (click 3.4) contains **many NR nodes**.

## 4.2 NR node types

According to its **control** and **radio capabilities**, an **NR node** may be **either** of the **three** following **types**:

- **gNB** (or **gNodeB** or **next-generation Node B**)

  A **native 5G node** with **NR radio interface** that provides the **5G UEs** with **5G user plane** and **control plane protocol** terminations and interoperates with the **5G Core**.

- **ng-eNB** (or **next-generation enabled eNB**)

  An **LTE node** with **E-UTRA radio interface** that provides the **5G UEs** with **5G control plane** (but **not user plane**) **protocol** terminations and interoperates with the **5G Core**.

- **en-gNB** (or **EPC-connected gNB**)

  It is a **reduced capability gNB**. It has the **NR radio interface** that provides the **5G UEs** with **5G user plane protocol** terminations, but it does **not** provide **5G control plane protocol** terminations, and it does **not** interoperate with the **5G Core**. It **must** have a control connection to an **LTE eNB**, and it interoperates with a **5G-enabled LTE Core** (i.e., with a **5G-enabled EPC**).

These **three types** of **NR node** are quite related to the **different initial deployment scenarios** (click 10.15) for **5G**.

According to its **MIMO capability** (click 10.11), an **NR node** may be **either** of the **two** following **types**:

- **Non-MIMO NR node**

  It is an **NR node** that does **not** have any **MIMO capability**.

- **MIMO NR node**

  It is an **NR node** that has **MIMO capability**.

## 4.3 NR node functional blocks, physical functions, payload processing, protocol architecture and data flow

The **functional blocks** of the **NR node**, in particular the **physical functions**, the **payload processing stages** of the **physical binary channels**, the **protocol architecture** of the **New Radio** and the **data flow** through the **protocol architecture** are **all different views** of the **same aspect**. You may want to refer to the following specific topics:

- For the **functional blocks** of the **NR node**, click 4.5.
- For the **physical functional blocks** in the **NR node**, click 4.5.7.
- For the **PBCH payload processing**, click 6.5.
- For the **PDCCH payload processing**, click 6.10.
- For the **PDSCH** and **PUSCH Tx payload processing**, click 4.12.2.
- For an **overview** of the **PDSCH** and **PUSCH Rx payload processing**, click 4.12.16.
- For the **PUCCH Tx payload processing**, click 7.8.2.
- For the **PDSCH** and **PUSCH data flow** across the **user-plane protocol stack**, click 4.10.4.

## 4.4 NR node functional splits

The **NR node functional splits** are **logical locations** where the **NR node functionality** may be **split** into **subsystems**, such that the **subsystems** can be placed in **different physical locations**.

**Placing** the **NR node subsystems** (click 4.7) in **different physical locations** allows a **cost reduction** because **several complete NR nodes**, **each located** at an **antenna site**, may be **replaced** by **only one distributed NR node**. The **single distributed NR node** is **cheaper** because it reduces the **functionality** (and the **cost**) at **each** of the **several antenna sites**. The **functionality** removed from **each antenna site** is implemented in a **lower quantity** of **NR node subsystems** that can offer the **same** (or **better**) **performance** because of **resource pooling**.

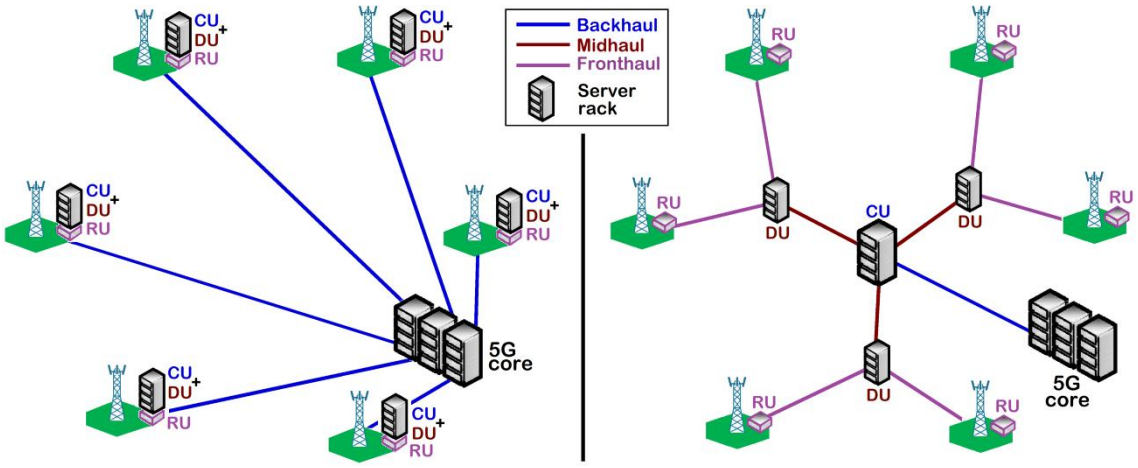
In **5G**, the **most frequent subsystem types** are the **Central Unit** (**CU**, click 4.7.1), the **Distributed Unit** (**DU**, click 4.7.2) and the **Radio Unit** (**RU**, click 4.7.3). These **subsystem types** may correspond to **different combinations** of **splits**.

The **backhaul** (click 4.6.1) is the **remote interface** between the **5G Core** (click 1.1.1) and **each** of its **Central Units** (**CUs**, click 4.7.1). The **backhaul connectivity** spans between **each CU** and the **closest point of presence** (**PoP**) of the **transport network**. The **midhaul** (click 4.6.2) is the **remote interface** and the **connectivity** between **each Central Unit** and **each** of its **Distributed Units** (**DUs**, click 4.7.2). The **fronthaul** (click 4.6.3) is the **remote interface** and the **connectivity** between **each Distributed Unit** and **each** of its **Radio Units** (**RUs**, click 4.7.3).

The **connectivity** in **backhaul** and **midhaul** is **most frequently** provided by a **converged packet network** (although it may **also** be provided by a **direct point-to-point link**).

**Fronthaul** connectivity is provided either as a **fiber link** (in **CPRI**) or as a **packet flow** (in **eCPRI**) over a **packet-switched network**.

The following figure shows the **same deployment** based on **six complete NR nodes** on the **left**, and based on **only one distributed NR node** on the **right**:

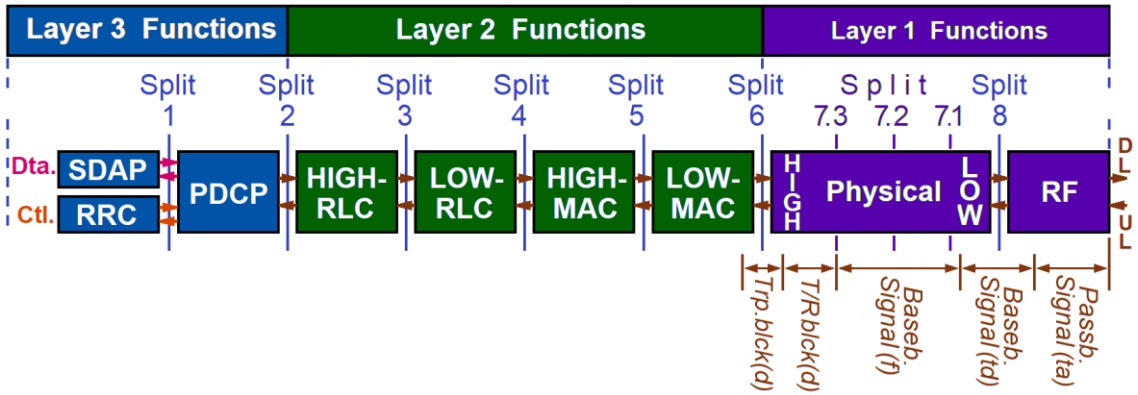



Some **relevant remarks** on this figure are:

- The **distributed NR node** is composed of **1 CU**, **3 DUs** and **6 RUs**.
- The figure shows the **backhaul**, **midhaul** and **fronthaul** **remote interfaces**, not the **respective connectivity**.
- For more detail on the **labels** in the figure, click 4.8.

To **physically distribute** the **NR node functionality**, it is divided into **9 functional blocks** (click 4.5) with **8 possible splits** between them. The **split 7.x** (click 4.5.7.2) is **different** to the **other splits** in that it may be done at **3 mutually exclusive locations** (i.e. **splits**) instead of at **only 1 location**, as is the case with the **other splits**. These **3 mutually exclusive splits** are **7.1**, **7.2** and **7.3**.

The following figure shows the **9 functional blocks** and the **8 splits** that have been defined for the **NR node**:




Some **relevant remarks** on this figure are:

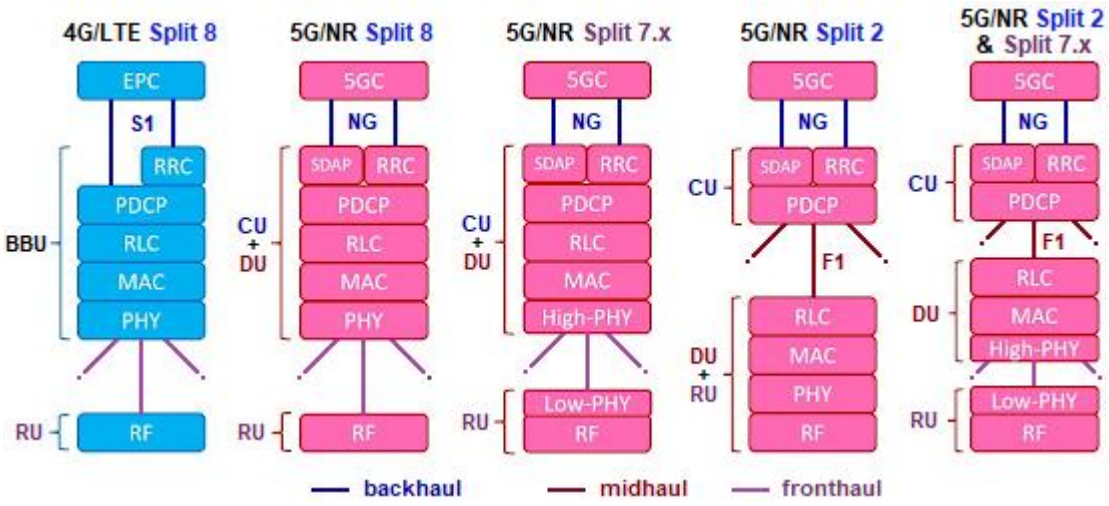
- The figure shows to **which layer** of the **protocol stack** belongs **each** of the **9 functional blocks**.

- The text "**Dta.**" stands for **user data**. The text "**Ctl.**" stands for **control**. The **protocol entities** of **both SDAP** and **RRC** are **directly** installed over the **PDCP protocol entity**.
- **Downlink (Tx) information flow** is **left to right**, while the **Uplink (Rx) information flow** is **right to left**.
- For more detail on the **labels** in the figure, click 4.8.

Some documents consider that the **RF functional block** is **not** part of the **Layer 1 functions**. However, the definition of **Layer 1** in **most protocol architectures** includes **all** the procedures and subsystems **down to** (and **not** including) the **physical medium**. This is why we consider that the **RF functional block** is included in **Layer 1 functions**.

**Zero** or **more splits** may be selected in **each implementation** of a **distributed NR node**. Regarding the **mutually exclusive splits 7.x**, **at most 1** of them may be selected for **each implementation**. The **quantity** of **subsystem types** that compose **each distributed NR implementation** is the **quantity** of **splits plus 1**. If **0 splits** are selected, the result is **0+1=1 system type**: a **complete non-distributed NR node** (click 4.1). If **1 split** is selected, the result is **1+1=2 subsystem types**. If **2 splits** are selected, the result is **2+1=3 subsystem types**.

The following figure shows some **frequent cases** (**1** for **LTE** and **4** for **5G**) of **functional splits** for **distributed NR node** implementations:




Some **relevant remarks** on this figure are:

- **NG** and **S1** are (respectively) the **standardized backhaul remote interfaces** for in **5G** and in **LTE**.
- **F1** is the **standardized midhaul remote interface** in **5G**.
- **CPRI** and **eCPRI** are the **standardized fronthaul remote interfaces** in **5G**.
- For more detail on the **labels** in the figure, click 4.8.

For the **data flow though** the **protocol architecture** shown in this figure, click 4.10.4.

The **functional split** names "**1**", "**2**", "**4**", "**6**" and "**8**" correspond to **O-RAN terminology**, which is widely accepted. However, these **splits** are called (respectively) "**A**", "**B**", "**C**", "**D**" and "**E**" in the **eCPRI** specification.

The **functional split** names "**7.1**", "**7.2**" and "**7.3**" correspond to **O-RAN terminology**, which is widely accepted. However, these **splits** are called (respectively) "**7.a**", "**7.b**" and "**7.c**" in some older **3GPP** documents (e.g., in TS 38.801). These **splits** are called (respectively) "**ID**", "**IID**" and "**IU**" in the **eCPRI** specification.

In **LTE**, the **most frequent subsystem types** were the **baseband unit** (**BBU**, click 4.7.4) and the **remote radio head** (**RRH**, click 4.7.5). The **RRH** is quite similar to the **Radio Unit** in **5G terminology**.

Source web pages:

- https://www.5gtechnologyworld.com/open-ran-functional-splits-explained/
- https://blog.3g4g.co.uk/2021/03/5g-ran-functional-splits.html
- https://www.mdpi.com/2076-3417/10/21/7429
- https://aviatnetworks.com/uncategorized/evolution-transport-5g/

## 4.5 NR node functional blocks

The **NR node functional blocks** are defined as part of the **NR functional splits** (click 4.4), and **each** of them is **described** in this section.

For the **NR node functional blocks**, the **NR node physical functions**, the **payload processing** of the **physical binary channels**, the **protocol architecture** of the **New Radio** and the **data flow** through the **protocol architecture**, click 4.3.

### 4.5.1 SDAP and RRC functional block

The **SDAP** and **RRC functional block** includes the **Service Data Adaptation Protocol** (**SDAP**) **entity** and the **RRC** (click 5) **protocol entity**.

The main services and functions of **SDAP** include:

- **Mapping** between a **QoS flow** and a **data radio bearer**.
- **Marking QoS flow ID** (**QFI**) in both **Downlink** and **Uplink** packets.

A **specific protocol entity** of **SDAP** is **instantiated** for **each** individual **PDU session**.

Some characteristics of the **SDAP** and **RRC functional block** are:

- It is usually implemented as **software** over a **general-purpose computer**.
- It manages **data** in the **digital domain**.
- It interfaces to the **5G core** (click 1.1.1) and to the **PDCP** (click 4.5.2) **functional block**.
- It belongs to the **Central Unit** (**CU**, click 4.7.1).

### 4.5.2 Packet Data Convergence Protocol (PDCP) functional block

The **PDCP functional block** transports **user/control data**, performs **IP header compression**, **encryption** and **integrity protection**. The main services and functions of the **PDCP entity** include:

- Transfer of data (user plane or control plane).
- Maintenance of **PDCP SNs**.

- **Header compression** and **decompression** using the **ROHC protocol**.
- **Header compression** and **decompression** using **EHC protocol**.
- **Ciphering** and **deciphering**.
- **Integrity protection** and **integrity verification**.
- **Timer** based **SDU discard**.
- For **split bearers**, **routing**.
- **Duplication**.
- **Reordering** and **in-order delivery**.
- **Out-of-order delivery**.
- **Duplicate discarding**.

Some characteristics of the **PDCP functional block** are:

- It is usually implemented as **software** over a **general-purpose computer**.
- It manages **data** in the **digital domain**.
- It interfaces to the **SDAP and RRC** (click 4.5.1) **functional block** and to the **High RLC** (click 4.5.3.1) **functional block**.
- It belongs to the **Central Unit** (**CU**, click 4.7.1).

#### 4.5.2.1 PDCP radio bearers

They are **channels** defined over the **packet data convergence protocol** (**PDCP**, click 4.5.2). There are the following **types** of **radio bearers**:

- **Signaling radio bearers** (**SRBs**)

  They are used for **unicast RRC messages** and for **unicast NAS messages**. They support the **packet data unit** (**PaDU**) **sessions**. For more detail, see below.

- **Data radio bearers** (**DRBs**)

  They are used for **unicast user data**. They support the **packet data unit** (**PaDU**) **sessions**. If a **UE** does **not** support **MRBs**, they can also be used for **IP multicast user data**.

- **Multicast radio bearers** (**MRBs**)

  They are used for **multicast signaling messages** and for **multicast user data**. The **MRBs** are **set up** as **required**, in order to **deliver** the agreed **QoS** for **multicast communication**.

  The **MRBs** were introduced in **R17** for **MBS services**.

**Signaling Radio Bearers**

They are **radio bearers** that are **only** used for **unicast RRC messages** and for **unicast NAS messages**. The following **SRBs** are defined:

- **SRB0**

  For **RRC messages** sent for **initial access** (click 8). These are mainly `RRCSetupRequest` (click 7.15.4) and `RRCSetup` (click 7.15.4).

- **SRB1**

  For **most RRC messages** (which **may** include a **piggybacked NAS message**). **SRB1** is also used for **NAS messages prior** to the establishment of **SRB2**.

- **SRB2**

  For **NAS messages** and for **RRC messages** which **include logged measurement information**. **SRB2** has a **lower priority** than **SRB1** and **may** be **configured** by the network **after access stratum security activation**.

- **SRB3**

  For **some specific RRC messages** when the **UE** is using (**NG**)**EN-DC** or **NR-DC** (click 10.3).

- **SRB4**

  For **RRC messages** which **include application layer measurement report information**. **SRB4** has a **lower priority** than **SRB1** and can **only** be **configured** by the network **after access stratum security activation**.

**SRB0** is **unsecured**, while **all** the **other SRBs** are **secured**.

**RRC messages** sent through **SRB0** are sent through the **CCCH logical** binary channel (click 4.11.1). **RRC messages** sent through **SRB1**, **SRB2** and **SRB3** are **all** sent through the **DCCH logical** binary channel. **NAS messages** sent through **SRB1** and **SRB2** are **also** sent through the **DCCH logical** binary channel.

In **Downlink**, **piggybacking** of **NAS messages** is used **only** for **one dependent** (i.e. with joint success/failure) **procedure**: **bearer establishment**/**modification**/**release**.

In **Uplink**, **piggybacking** of **NAS messages** is used **only** for **transferring** the **initial NAS message** during **connection setup** and **connection resume**.

Note that the **NAS messages** transferred via **SRB2** are **also contained** in **RRC messages**, which **however** do **not include** any **RRC protocol control information**.

Once the **access stratum security** is **activated**, **all RRC messages** on **SRB1**, **SRB2**, **SRB3** and **SRB4**, **including** those containing **NAS messages**, are **integrity protected** and **ciphered** by **PDCP**. **NAS independently** applies **integrity protection** and **ciphering** to the **NAS messages** (refer to TS 24.501).

**Split SRB is supported** for **all** the **MR-DC** options in **both SRB1** and **SRB2** (**split SRB** is **not supported** for **SRB0**, **SRB3** and **SRB4**).

For **shared spectrum** in **FR1**, **SRB0**, **SRB1** and **SRB3** are assigned with the **highest priority Channel Access Priority Class** (**CAPC**), (i.e. **CAPC** = **1**) while **CAPC** for **SRB2** is **configurable**.

For more detail on **PDCP signaling radio bearers**, refer to clause 4.2.2 of TS 38.331.

### 4.5.3 Radio link control sublayer

**Radio Link Control** (**RLC**) is a **Layer 2 logical-link control protocol**. **RLC** has **three modes of operation**:

- **Transparent Mode** (**TM**)

  This mode does **not** use **error detection** codes, so there is **no error control**. It is typically used for **real-time multimedia** (e.g., voice).

- **Unacknowledged Mode (UM)**

  This mode does **not** use **acknowledgements** so packets with **wrong CRC** are simply **discarded**. It is typically used for **multicast data**.

- **Acknowledged Mode (AM)**

  This mode is a conventional **ARQ protocol**. It is typically used for **unicast data**.

**RLC** may perform **Service Data Unit (SDU) segmentation** and/or **SDU concatenation**.

The **Maximum Transfer Unit (MTU**, i.e., maximum size of **SDUs**) of **RLC** in **LTE** and **5G** is configurable. For **LTE**, a usual value is around **4,000 bytes**, while for **5G** a usual value is around **8,000 bytes**.

The **functional split** between the **High RLC functional block** and the **Low RLC functional block** is indicated in the following **two subsections**.

#### 4.5.3.1 High RLC functional block

It performs the following **RLC functions** (click 4.5.2.1), that are **not** considered **time-critical (asynchronous)**:

- **Segmentation and reassembly**

  In **Tx**, **segmentation** of **higher-layer SDUs (Service Data Units)** into smaller **PDUs (Protocol Data Units)**.

  In **Rx**, **reassembly** of the **original higher-layer SDUs**.

- **Error recovery (ARQ based)**

  In **Tx**, **add** a **CRC**, await for **ACKs**, and if **not received** manage the **retransmission**.

  In **Rx**, **verify** the **CRC** and, if **correct**, accept the data and **send** an **ACK**.

Some characteristics of the **High RLC functional block** are:

- It is usually implemented as **software** over a **general-purpose computer**.
- It manages **data** in the **digital domain**.
- It interfaces to the **PDCP** (click 4.5.2) **functional block** and to the **Low RLC** (click 4.5.3.2) **functional block**.
- It belongs to the **Distributed Unit (DU**, click 4.7.2).

#### 4.5.3.2 Low RLC functional block

It performs the following **RLC functions** (click 4.5.2.1), that are considered **time-critical (synchronous)**:

- **PDU Formatting and Handling**

  Manages the final preparation of **PDUs** for transmission **over** the **MAC layer**.

- **Ensuring proper PDU timing**

  **Fast retransmission feedback** for **lower-latency ARQ**.

Some characteristics of the **Low RLC functional block** are:

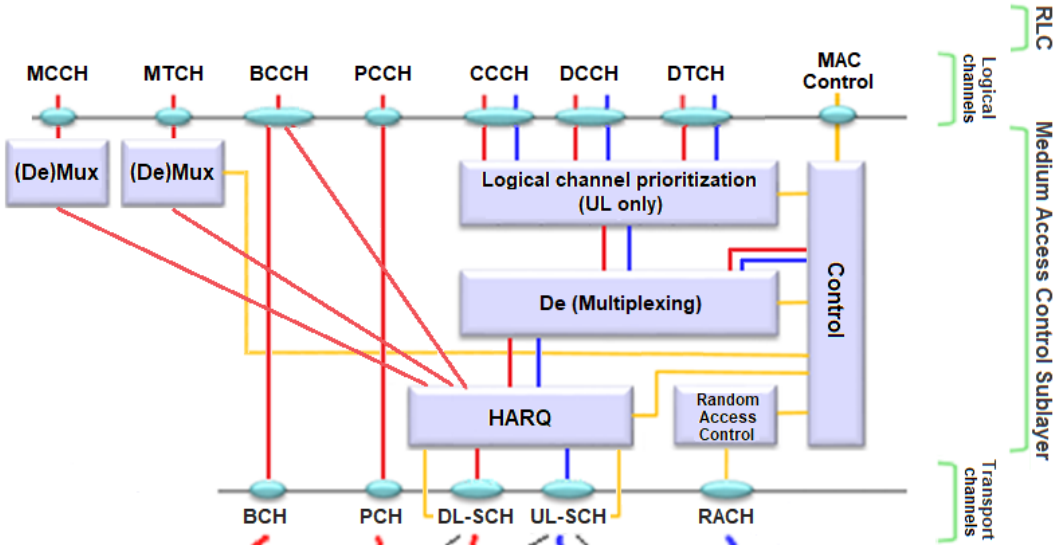
- It is usually implemented as **software** over a **general-purpose computer**, **possibly** with **hardware accelerators** (**ASIC**, **FPGA** or **GPU**).
- It manages **data** in the **digital domain**.
- It interfaces to the **High RLC** (click 4.5.3.1) **functional block** and to the **High MAC** (click 4.5.3.2) **functional block**.
- It belongs to the **Distributed Unit** (**DU**, click 4.7.2).

### 4.5.4 Medium access control sublayer

The **functional split** between the **High MAC functional block** and the **Low MAC functional block** is indicated in the following **two subsections**.

The **protocol data units** (**PDUs**) handled by the **MAC sublayer** are the **MAC PDUs** (click 4.5.5).

The following figure depicts the **internal structure** of the **Medium Access Control** sublayer and its relationship to the **logical binary channels** (click 4.11.1) and to the **transport binary channels** (click 4.11.2).




Some **relevant remarks** on this figure are:

- The **binary channels**, and **binary channel hierarchy**, are the same at **both ends** (**NR node** and **UE**) of the transmission link, with the **only difference** that the **direction** of information flows (**downwards** or **upwards** for **Downlink** or **Uplink**) is **reversed**.
- The figure depicts the **binary channels** at the **NR node**: **Downlink** (**red lines**) is **downwards information flow** and **Uplink** (**blue lines**) is **upwards information flow**. In a **UE** the **information flows** would be the **opposite**.
- **Black lines** do **not** indicate a **direct mapping** between channels. They **rather** indicate some type of **indirect relation** (for example, **scheduling** or **HARQ response**).
- **Yellow lines** do **not** indicate **information flow**. They **rather** indicate a **control relationship**.

- The **yellow lines** between **HARQ** and **DL-SCH** and **UL-SCH** represent the **HARQ control** for the **retransmission** of **PDSCH codewords** and of **PUSCH codewords**.

For an **extended figure** that shows the **complete RAN protocol architecture**, click 4.10.1.

For an overview of the **data flow** across the **user-plane protocol stack**, click 4.10.4.

For more detail on the **binary channels**, click 4.8.

For an **overview** of **MAC protocols**, click 11.11.

#### 4.5.4.1 High MAC functional block

The **High MAC functional block** handles the following **MAC** functions:

- **Logical Channel Management**

  **Mapping** between **logical channels** (click 4.11.1) and **transport channels**.

- **Logical Channel Prioritization**

  This is **only** done for **Uplink**.

- **(De)Multiplexing**

  **Multiplexing** and **demultiplexing** of **data flows**.

- **Scheduling Coordination**

  High-level **scheduling decisions**, such as determining **resource allocations** for **users** or **services**.

  Controls **dynamic scheduling** and prioritization **across multiple DUs** (click 4.7.2).

- **Non-Time-Critical Functions**

  Operations that are **less sensitive** to **latency**.

  **Communication** with **higher layers**, such as **RLC** (click 4.5.3) and **PDCP** (click 4.5.2).

Some characteristics of the **High MAC functional block** are:

- It is usually implemented as **software** over a **general-purpose computer**, **possibly** with **hardware accelerators** (**ASIC**, **FPGA** or **GPU**).
- It manages **data** in the **digital domain**.
- It interfaces to the **Low RLC** (click 4.5.3.2) **functional block** and to the **Low MAC**(click 4.5.4.2) **functional block**.
- It belongs to the **Distributed Unit** (**DU**, click 4.7.2).

Source web pages:

- https://devopedia.org/5g-nr-channels

#### 4.5.4.2 Low MAC functional block

The **Low MAC functional block** handles the following **MAC** functions:

- **Hybrid Automatic ReQuest (HARQ) protocol**

  The **HARQ protocol entity**. For more detail on **5G HARQ**, click 4.5.6.

- **Time-Critical Scheduling Execution**

  **Real-time**, **low-latency** aspects of **scheduling decisions**.

- **Random access procedure**

  Manage the **random access procedure** (click 9) in **PRACH** for **UE initial access**.

- **Close Interaction with Physical Layer**

  Direct **coordination** with the **physical layer** for **real-time** functions.

Some characteristics of the **Low MAC functional block** are:

- It is usually implemented as **software** over a **general-purpose computer**, **usually** with **hardware accelerators** (**ASIC**, **FPGA** or **GPU**).
- It manages **data** in the **digital domain**.
- It interfaces to the **High MAC** (click 4.5.4.1) **functional block** and to the **High Physical** (click 4.5.5) **functional block**.
- It belongs to the **Distributed Unit** (**DU**, click 4.7.2).

### 4.5.5 MAC data units: PDUs, subPDUs, subheaders, SDUs, CEs and payloads

The **MAC entities** of the **NR node** and of **each** of its **UEs** exchange **control information** for different purposes. **Most** of this **control information** is exchanged as **MAC protocol data units** (**MAC PDUs**, click 4.5.5.1).

**MAC PDUs** can contain **MAC subPDUs**, **MAC subheaders** (click 4.5.5.2), **MAC service data units** (**MAC SDUs**), **MAC control elements** (**MAC CEs**, click 4.5.5.3) and **MAC payloads** (click 4.5.5.4).

A **MAC subPDU** is a **byte string**. **Each MAC PDU** can contain **many MAC subPDUs**.

A **MAC SDU** is a **byte string**. **Each MAC PDU** can contain **many MAC SDUs**.

For more detail on **MAC control information**, refer to clause 6 of TS 38.321.

#### 4.5.5.1 MAC protocol data units (MAC PDUs)

A **MAC PDU** is a **byte string**. **Each MAC PDU** can contain **many MAC subPDUs**, **many MAC subheaders**, **many MAC SDUs**, **many MAC CEs** and **many MAC payloads**.

There are **5 types** of **MAC PDU**:

- **SCH MAC PDU** (click 4.5.5.5)

  It is used for **all** the **Uplink SCH** and **all** the **Downlink SCH signaling**, **except** the **signaling** that is sent with the "**Random access response**" **MAC PDU** (click 4.5.5.7).

- **Transparent MAC PDU** (click 4.5.5.6)

  It is used for **all** the **Uplink SCH data** and **all** the **Downlink SCH data**.

- **Random access response MAC PDU** (click 4.5.5.7)

  It is used in the **4-step random access procedures** (click 9.1.1).

- **MsgB MAC PDU** (click 4.5.5.8)

  It is used in the **2-step random access procedures** (click 9.1.1).

- **SL-SCH MAC PDU**

  It is used for **Sidelink**.

For more detail on **MAC PDUs**, refer to clause 6.1 of TS 38.321.

### 4.5.5.2 MAC subheaders

A **MAC subheader** is a **byte string**. **Each MAC PDU** (click 4.5.5.1) can contain **many MAC subheaders**. **Each MAC subheader** is placed **right before** the corresponding **MAC SDU**, **MAC CE**, or **padding**.

There are **4 types** of **MAC subheaders**:

- **SCH MAC subheaders**.
- **Random access response MAC subheaders** (click 4.5.5.7)

  They are used in the **4-step random access procedures** (click 9.1.1). For more detail,

- **MsgB MAC subheaders**.

  They are used in the **2-step random access procedures**. This includes the **MSGB fallback MAC subheader** (click 4.5.5.8.1), the **MSGB success MAC subheader** (click 4.5.5.8.2) and the **MSGB backoff indicator MAC subheader** (click 4.5.5.8.3).

- **SL-SCH MAC subheaders**.

  They are used for **Sidelink**.

For more detail on **MAC subheaders**, refer to clauses 6.2.1, 6.2.2, 6.2.2a and 6.2.4 of TS 38.321.

### 4.5.5.3 MAC control elements (MAC CEs)

A **MAC CE** is a **byte string**. **Each MAC PDU** (click 4.5.5.1) can contain **many MAC CEs**. **Each MAC CE** conveys a **specific type** of **control information** for a **specific control procedure**.

There are **65 types** of **MAC CEs**. Three of them are:

- **C-RNTI MAC CE** (click 9.3.4.1.1)

  It is sent by the **UE** as **Msg3** (click 9.3.4), in some cases of **4-step random access procedures** (click 9.1.1), to inform the **NR node** about which is its **C-RNTI**.

- **UE Contention Resolution Identity MAC CE** (click 9.3.5.2)

  It is sent by the **NR node** as **Msg4** (click 9.3.5), in some cases of **4-step random access procedures** (click 9.1.1), to convey the **unique identity** of the **UE** that has **won** the **contention resolution**.

- **Buffer status report MAC CEs**

  It is used by the **UE** to **inform** the **NR node** about **how much data** it has **buffered** awaiting to be **transmitted** through **PUSCH**.

For more detail on **MAC CEs**, refer to clause 6.1.3 of TS 38.321.

#### 4.5.5.4 MAC payloads

A **MAC payload** is a **byte string**. **Each MAC PDU** (click 4.5.5.1) can contain **many MAC payloads**.

There are **2 types** of **MAC payload**:

- **Random access response MAC payload** (click 4.5.5.7.1)

  It is used in the **4-step random access procedures** (click 9.1.1).

- **MsgB MAC payload**

  It is used in the **2-step random access procedures** (click 9.1.1).

For more detail on **MAC payloads**, refer to clauses 6.2.3 and 6.2.3a of TS 38.321.

#### 4.5.5.5 MAC PDU: SCH

The **SCH MAC PDU** (click 4.5.5) is used for **all** the **Uplink SCH** and **all** the **Downlink SCH signaling**, **except** the **signaling** that is done with the **transparent MAC PDU** and with the "**Random access response**" **MAC PDU** (click 4.5.5.7).

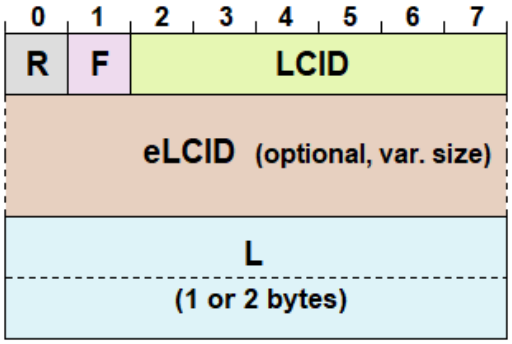
It consists of **1**, or **several**, **MAC subPDUs** (click 4.5.5). **Each MAC subPDU** consists of **either**:

- **1 SCH MAC subheader** (click 4.5.5.2) **only** (including padding).
- **1 SCH MAC subheader** and **1 MAC SDU** (click 4.5.5).
- **1 SCH MAC subheader** and **1 MAC control element** (click 4.5.5.3).
- **1 SCH MAC subheader** and padding.

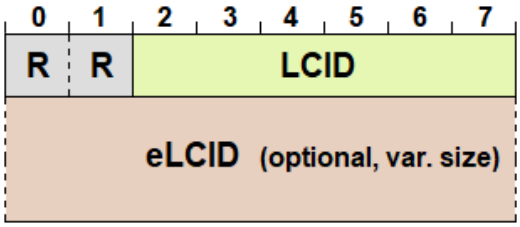
**All** of the **previous elements** are **octet aligned**.

**Each SCH MAC subheader** corresponds to **either** a **MAC SDU**, a **MAC CE**, or **padding**.

A **SCH MAC subheader** (**except** for **fixed sized MAC CE**, **padding**, and a **MAC SDU containing UL CCCH**) consists of the **4** or **5 header fields** "**R/F/LCID/(eLCID)/L**". The following figure shows this **SCH MAC subheader**:




A **SCH MAC subheader** for **fixed sized MAC CE**, **padding**, and a **MAC SDU containing UL CCCH** consists of the **2** or **3 header fields** "**R/LCID/(eLCID)**". The "**eLCID**" **field** is placed between parentheses because it is **optional**. The following figure shows this **SCH MAC subheader**:




In **both** figures the **octets** are **ordered top** (**first octet**) to **bottom** (**last octet**).

The **semantics** of **each** of the above **fields** is:

- **Reserved** "**R**" (1 bit)

  **Reserved** for **future use**.

- **Format Flag** "**F**" (1 bit)

  It indicates the size of the **Length** "**L**" **field**, which is "**F+1**" **bytes**.

- **Logical Channel ID** "**LCID**" (6 bits)

  It identifies the **type** of **payload** (e.g., **DTCH**, **DCCH**, **CCCH**) or of **MAC control element**. Values from **0 to 27** identify **dedicated logical channels**. Values from **28 to 31** are **reserved/padding/special**.

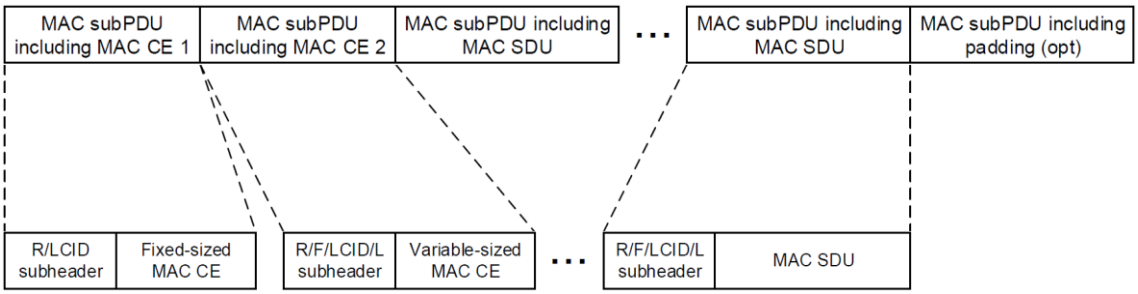
- **Extended LCID** "**eLCID**" (variable size and optional)

  It **replaces** or **extends** the **LCID field**, in order to accommodate **more values**. It can **only** be **present** if the **eLCID mode** is configured. It was introduced in **R17**.

- **Length** "**L**" (1 or 2 bytes)

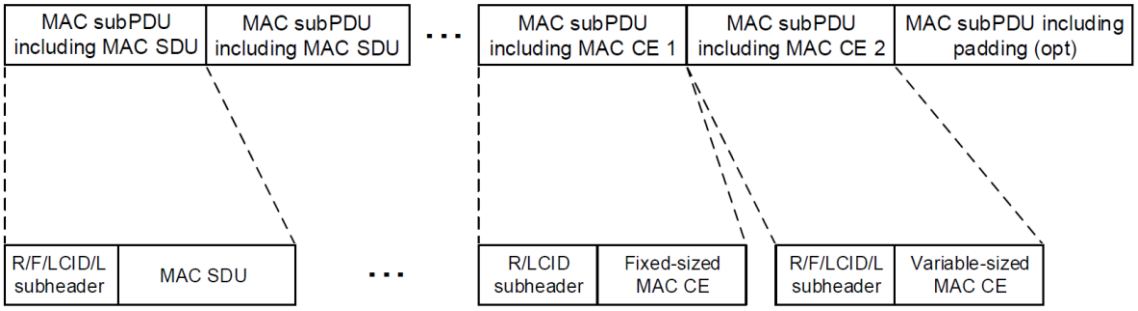
  It indicates the **length** in **bytes** of the **MAC SDU** or **MAC CE** that **follows** the **subheader**.

The following figure shows an **example** of a **Downlink SCH MAC PDU**.




In the figure, **octets** are **ordered left** (**first octet**) to **right** (**last octet**).

The following figure shows an **example** of an **Uplink SCH MAC PDU**.




In the figure, **byte** are **ordered left** (**first octet**) to **right** (**last byte**).

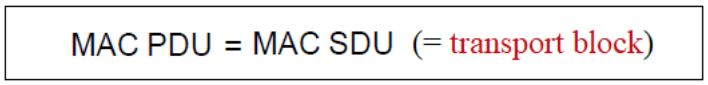
For more detail on the **SCH MAC PDU**, refer to clause 6.1.2 of TS 38.321.

For more detail on **MAC PDUs** and **CEs**, refer to clause 6 of TS 38.321.

### 4.5.5.6 MAC PDU: transparent

The **transparent MAC PDU** (click 4.5.5) **only** contains **1 MAC SDU** whose size is aligned to a **transport block**. This **MAC PDU** is used for transmissions on **PCH**, **BCH**, **DL-SCH** (including **BCCH**) and **SL-BCH**.

The following figure shows the **structure** of a **transparent MAC PDU**:



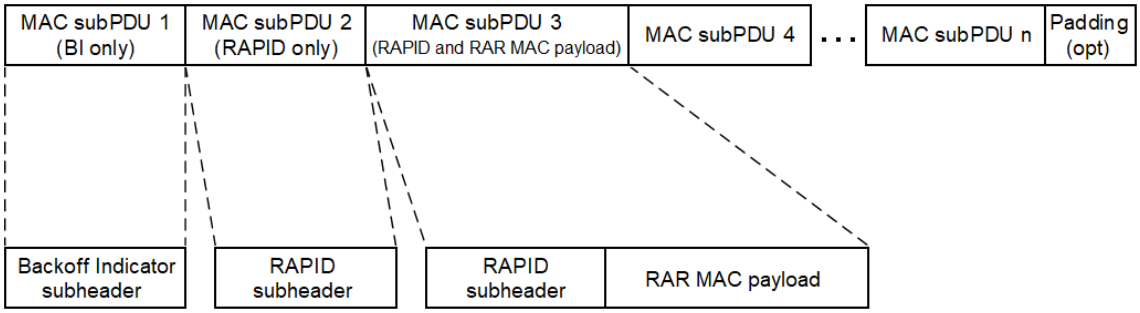

### 4.5.5.7 MAC PDU: RAR

**Each Random access response** (**RAR**) **MAC PDU** (click 4.5.5.1) consists of **1**, or **several**, **RAR MAC subPDUs** (click 4.5.5) and optionally **padding**. **Each RAR MAC subPDU** consists of **either**:

- **1 RAR Backoff Indicator MAC subheader** (click 4.5.5.7.1) **only**.
- **1 RAR RAPID MAC subheader** (click 4.5.5.7.1) **only** (i.e., an acknowledgment for **SI request**).
- **1 RAR RAPID MAC subheader** with **1 RAR MAC payload** (click 4.5.5.7.3).

**All** of the **previous elements** are **octet aligned**.

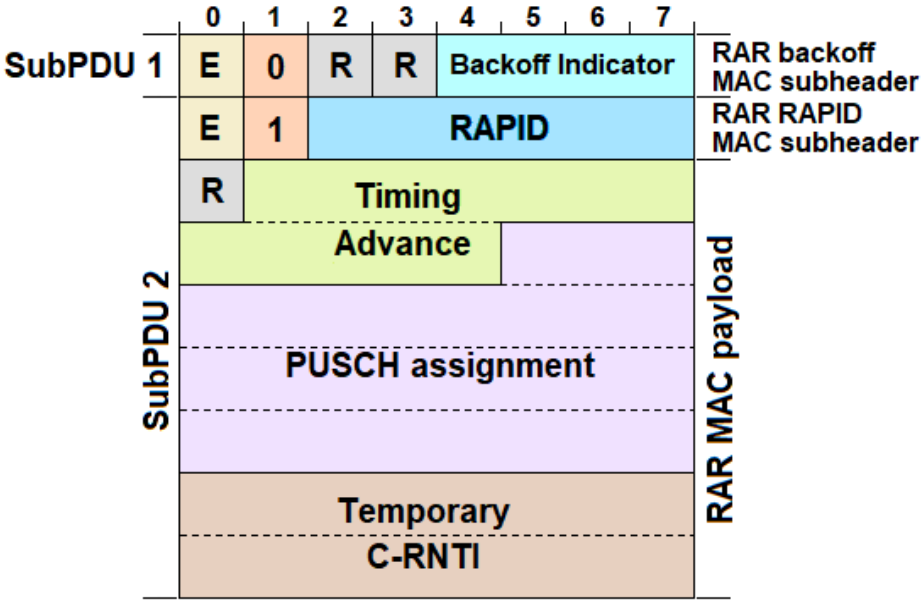
The following figure shows the **structure** of a **RAR MAC PDU**:




In the figure, **octets** are **ordered left** (**first octet**) to **right** (**last octet**).

With the **only exception** of an **RA procedure triggered** by **beam failure recovery** (**BFR**), the **RAR MAC PDU** used for **Msg2** (**random access response**, click 9.3.3) includes **1 RAR MAC payload**.

The following figure shows an **example** of **2 RAR MAC subPDUs** that could be used to **convey 1 Msg2**: **1 RAR MAC subPDU** contains **RAR Backoff Indicator MAC subheader** (click 4.5.5.7.1). The **second RAR MAC subPDU** contains **1 RAR RAPID MAC subheader** (click 4.5.5.7.1) and **1 RAR MAC payload** (click 4.5.5.7.3):



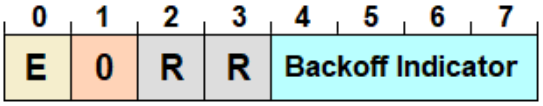

In the figure, **octets** are **ordered top** (**first octet**) to **bottom** (**last octet**).

For more detail on the "**Random access response**" **MAC PDU**, refer to clause 6.1.5 of TS 38.321.

For more detail on **MAC PDUs** and **CEs**, refer to clause 6 of TS 38.321.

#### 4.5.5.7.1 RAR "Backoff indicator" MAC subheader

The following figure shows a **RAR Backoff Indicator MAC subheader**:



The **semantics** of **each** of these **fields** is:

- **Extension** "**E**" (1 bit)

  Its value is"**0**" in the **last subPDU**, and it is "**1**" in **each** and **every previous subPDU**.

- **Type** "**T**" (1 bit)

  It is the **second bit**. Its value is "**0**" for the **Backoff Indicator MAC subPDU**, and it is "**1**" for the **RAPID MAC subPDU**.

- **Reserved** "**R**" (1 bit)

  **Reserved** for **future use**.

- **Backoff Indicator** (4 bits)

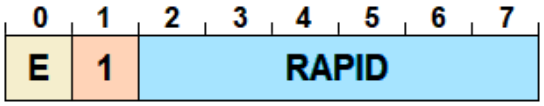
  It is the **index** for table 7.2.1 of TS 38.321, to obtain the **backoff** parameter value. The table values range from **5 ms** to **1,920 ms**. The **backoff** is the **time** that the **UE must wait before retrying** the **random access procedure**, in case it **fails**. This is the usual behavior of **contention protocols** (e.g., **CSMA/CD**) to **mitigate congestion**.

For more detail on the **RAR** "**Backoff Indicator**" **MAC subheader**, refer to clause 6.2.2 of TS 38.321.

For more detail on **MAC PDUs** and **CEs**, refer to clause 6 of TS 38.321.

#### 4.5.5.7.2 RAR RAPID MAC subheader

The following figure shows a **RAR random access preamble identifier** (**RAPID**) **MAC subheader**:




The **MSGB fallback MAC subheader** (click 4.5.5.8.1) is **identical** to the **RAR RAPID MAC subheader**.

The **semantics** of **each** of these **fields** is:

- **Extension** "**E**" (1 bit)

  Its value is"**0**" in the **last subPDU**, and it is "**1**" in **each** and **every previous subPDU**.

- **Type** "**T**" (1 bit)

  It is the **second bit**. Its value is "**1**" for the **RAPID MAC subPDU** (and **likewise** for the **MSGB fallback MAC subheader**).

- **Reserved** "**R**" (1 bit)

  **Reserved** for **future use**.

- **Random Access Preamble Identifier** (**RAPID**) (6 bits)

  It contains the **index** of the **RA preamble sequence** of the **UE** to which the **NR node** is **addressing this Msg2**. This **index** is the **same** as the **one** of the **RA preamble sequence** that the **addressed UE** sent within its **Msg1** (click 9.3.2).

For more detail on the "**RAPID**" **MAC subheader**, refer to clause 6.2.2 of TS 38.321.

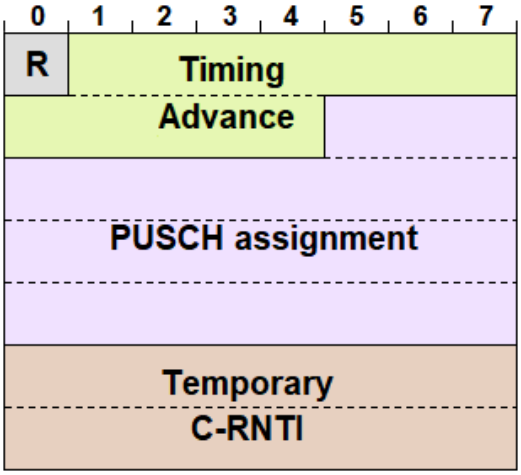
For more detail on **MAC PDUs** and **CEs**, refer to clause 6 of TS 38.321.

#### 4.5.5.7.3 RAR MAC payload

The **random access response** (**RAR**) **MAC payload** (click 4.5.5.4) has a **fixed size** of **7 bytes**. The **RAR MAC payload** is sometimes called the "**MAC RAR**".

The **MSGB fallback MAC payload** (**fallbackRAR**, click 4.5.5.8.4) is **identical** to the "**Random Access Response**" **MAC payload** (**MAC RAR**).

The following figure shows the **RAR MAC payload**:




In the figure, **octets** are **ordered top** (**first octet**) to **bottom** (**last octet**).

The **semantics** of **each** of its **fields** is:

- **Reserved** "**R**" (1 bit)

  **Reserved** for **future use**.

- **Timing Advance** (12 bits)

  It is the **index value** of the **Timing Advance** (click 2.3.6.1) that is used to control the amount of timing adjustment that the **MAC** entity has to apply. It is used to **synchronize** the **UE**'s **Uplink carrier** in respect to the **Downlink carrier**.

- **TC-RNTI** (i.e., a **temporary C-RNTI**) (16 bits)

  For more detail on the **TC-RNTI**, click 6.9.10.2.

- **PUSCH assignment** (27 bits)

  It contains the following **sub-fields**, as indicated in table 8.2-1 of TS 38.213:

| RAR grant field | Bitlength | Comments |
|---|---|---|
| **Frequency hopping flag** | **1** | Value **0** is **PUSCH without frequency hopping**.<br>Value **1** is **with frequency hopping**. |
| **PUSCH frequency resource allocation** (click 7.11.3) | **12** or **14** | **12**, for operation for shared spectrum in **FR1** or for **FR2-2** when `channelAccessMode2-r17` is **present**.<br>**14**, otherwise |
| **PUSCH time resource allocation** (click 7.11.4) | **4** | Index for table 6.1.2.1.1-**2** or **-3** of TS 38.214. |
| **MCS** (click 7.13) | **4** | Index for the corresponding **PUSCH** table from clause 6.1.4.1 of TS 38.214. |
| **TPC command for PUSCH** | **3** | Index for table 8.2-2 of TS 38.213. |
| **CSI request** | **1** | **1** Reserved |
| **ChannelAccess-CPext** | **2** or **0** | **2**, for operation for shared spectrum in **FR1** or for **FR2-2** when `channelAccessMode2-r17`= `semiStatic` is **present**<br>**0**, otherwise |

  The **ChannelAccess-CPext sub-field** indicates a **channel access type** and the **CP extension** for **shared spectrum** (TS 37.213) in **FR1** (as defined in table 7.3.1.1.1-4 of TS 38.212 or table 7.3.1.1.1-4A of TS 38.212) in case that "`channelAccessMode` = `semiStatic`" is **present**.

For more detail on the "**Random access response**" **MAC payload**, refer to clause 6.2.3 of TS 38.321.

For more detail on **MAC PDUs** and **CEs**, refer to clause 6 of TS 38.321.

### 4.5.5.8 MAC PDU: MSGB

The **MSGB MAC PDU** (click 4.5.5) is used to **convey MsgB** (click 9.3.7) in **2-step RA procedures** (click 9.1.1).

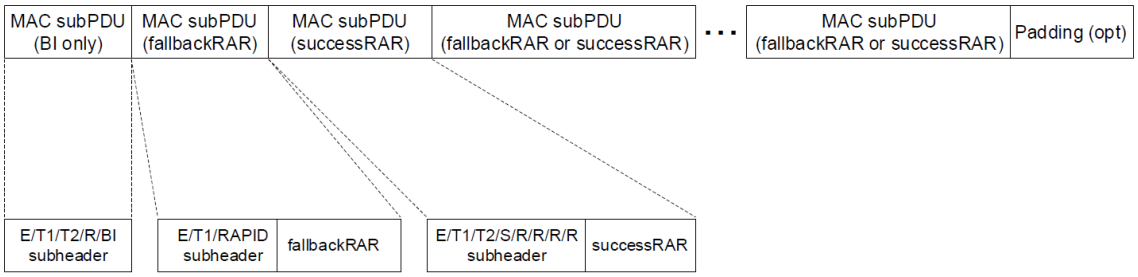
It consists of **1**, or **several**, **MAC subPDUs** (click 4.5.5) and **optionally padding**. **Each MAC subPDU** consists of **either**:

- **1 MSGB backoff indicator MAC subheader** (click 4.5.5.8.3).
- **1 MSGB fallback MAC subheader** (click 4.5.5.8.1) and **1 MSGM fallback MAC payload** (click 4.5.5.7.3).
- **1 MSGB success MAC subheader** (click 4.5.5.8.2) and **1 MSGM success MAC payload** (click 4.5.5.8.5).
- **1 MSGB MAC subheader** and **1 MAC SDU** for **CCCH**, **DCCH** or **DTCH**.
- **1 MSGB MAC subheader** and **1 MAC SDU** (click 4.5.5).
- **1 MSGB MAC subheader** and padding.

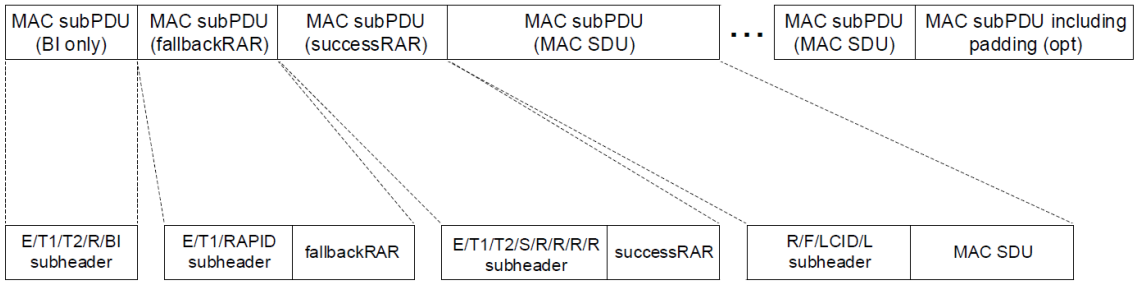
**All** of the **previous elements** are **octet aligned**.

A **MSGB MAC subPDU** with **only 1 MSGB fallback MAC subheader**, if present, **must** be **placed** at the **beginning** of the **MSGB MAC PDU**.

The following figure shows an **example** of **MSGB MAC PDU** that does **not include MAC SDUs**:




The following figure shows an **example** of **MSGB MAC PDU** that **includes MAC SDUs**:




For more detail on the **MSGB MAC PDU**, refer to clause 6.1.5a of TS 38.321.

For more detail on **MAC PDUs** and **CEs**, refer to clause 6 of TS 38.321.

#### 4.5.5.8.1 MSGB fallback MAC subheader

The following figure shows a **MSGB fallback MAC subheader**:

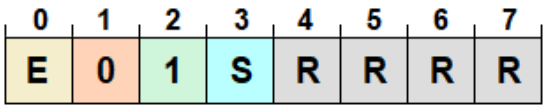
The **MSGB fallback MAC subheader** is **identical** to the **RAR RAPID MAC subheader** , click 4.5.5.7.1.

For more detail on the **MSGB fallback MAC subheader**, refer to clauses 6.1.5a and 6.2.2a of TS 38.321.

For more detail on **MAC PDUs** and **CEs**, refer to clause 6 of TS 38.321.

#### 4.5.5.8.2 MSGB success MAC subheader

The following figure shows a **MSGB success MAC subheader**:




The **semantics** of **each** of these **fields** is:

- **Extension** "**E**" (1 bit)

  Its value is"**0**" in the **last subPDU**, and it is "**1**" in **each** and **every previous subPDU**.

- **Type 1** "**T1**" (1 bit)

  It is the **second bit**. If it is "**1**", this is a **MSGB fallback MAC subheader**. If it is "**0**" this is **either** a **MSGB success MAC subheader** or a **MSGB backoff indicator MAC subheader**.

- **Type 2** "**T2**" (1 bit)

  It is the **third bit**. If "**T1**" is "**0**" and this is "**1**", this is a **MSGB success MAC subheader**.

- **SDU** "**S**" (1 bit)

  If it is "**1**" a **MAC subPDU** with a **MAC SDU** follows **this MAC subPDU**. If it is "**0**", there is **no MAC subPDU** with a **MAC SDU**.

- **Reserved** "**R**" (4 x 1 bit)

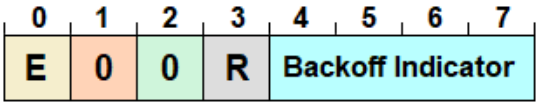
  **Reserved** for **future use**.

For more detail on the **MSGB success MAC subheader**, refer to clauses 6.1.5a and 6.2.2a of TS 38.321.

For more detail on **MAC PDUs** and **CEs**, refer to clause 6 of TS 38.321.

#### 4.5.5.8.3 MSGB "Backoff indicator" MAC subheader

The following figure shows a **MSGB Backoff Indicator MAC subheader**:




The **semantics** of **each** of these **fields** is:

- **Extension** "**E**" (1 bit)

  Its value is"**0**" in the **last subPDU**, and it is "**1**" in **each** and **every previous subPDU**.

- **Type 1** "**T1**" (1 bit)

  It is the **second bit**. If it is "**1**", this is a **MSGB fallback MAC subheader**. If it is "**0**" this is **either** a **MSGB success MAC subheader** or a **MSGB backoff indicator MAC subheader**.

- **Type 2** "**T2**" (1 bit)

  It is the **third bit**. If "**T1**" is "**0**" and this is "**0**", this is a **MSGB backoff indicator MAC subheader**.

- **Reserved** "**R**" (1 bit)

  **Reserved** for **future use**.

- **Backoff Indicator** (4 bits)

  It is the **index** for table 7.2.1 of TS 38.321, to obtain the **backoff** parameter value. The table values range from **5 ms** to **1,920 ms**. The **backoff** is the **time** that the **UE must wait before retrying** the **random access procedure**, in case it **fails**. This is the usual behavior of **contention protocols** (e.g., **CSMA/CD**) to **mitigate congestion**.

For more detail on the **MSGB** "**Backoff Indicator**" **MAC subheader**, refer to clauses 6.1.5a and 6.2.2a of TS 38.321.

For more detail on **MAC PDUs** and **CEs**, refer to clause 6 of TS 38.321.

#### 4.5.5.8.4 MSGB fallback MAC payload

The **MSGB fallback MAC payload** is sometimes called "**fallbackRAR**".

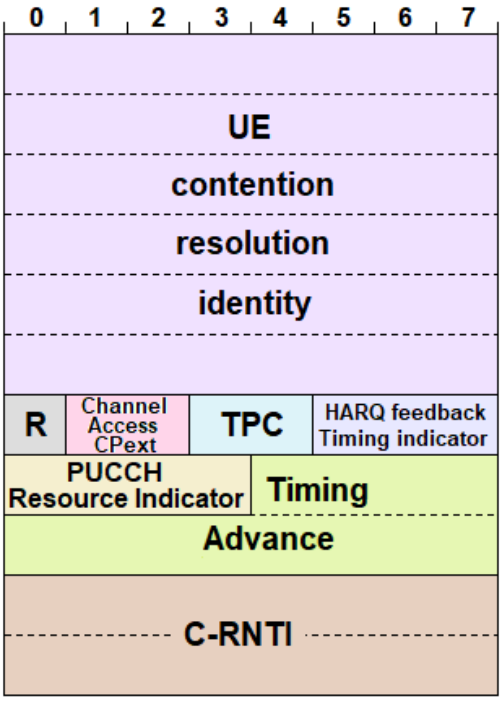
The **MSGB fallback MAC payload** (fallbackRAR) is **identical** to the **RAR MAC payload** (MAC RAR), click 4.5.5.7.3.

There are **2 different MAC payload formats** for **MSGB**:

#### 4.5.5.8.5 MSGB success MAC payload

The **MSGB success MAC payload** is sometimes called "**successRAR**".

The **MSGB success MAC payload** has a **fixed size** of **11 bytes**. The following figure shows the **MSGB success MAC payload**:




In the figure, **octets** are **ordered top** (**first octet**) to **bottom** (**last octet**).

The **semantics** of **each** of its **fields** is:

- **UE Contention Resolution Identity** (48 bits)

  It contains the UL CCCH SDU. If the UL CCCH SDU is longer than 48 bits, this field contains the first 48 bits of the UL CCCH SDU.

- **Reserved** "**R**" (1 bit)

  **Reserved** for **future use**.

- **ChannelAccess-CPext** (2 bits)

  The **channel access type** and **CP extension** for the **PUCCH resource** containing the **HARQ feedback** for **MsgB** in **shared spectrum** channel access as specified in TS 38.213. The field is **only** present when the **MSGB HARQ feedback** is to be transmitted with **shared spectrum** channel access as specified in TS 37.213. **Otherwise**, the field is **not present** and **2 Reserved** bits are present instead.

- **Transmit Power Control** (**TPC**) (2 bits)

  It is the **TPC command** for the **PUCCH resource** containing **HARQ feedback for MsgB**, as specified in TS 38.213.

- **HARQ Feedback Timing Indicator** (3 bits)

  It is the **PDSCH**-to-**HARQ feedback timing indicator** field for **MsgB HARQ** feedback, as specified in TS 38.213.

- **PUCCH Resource Indicator** (4 bits)

  It is the **PUCCH resource indicator** for **HARQ feedback** for **MsgB**, as specified in TS 38.213.

- **Timing Advance** (4 bits)

  It is the **index value** of the **Timing Advance** (click 2.3.6.1) that is used to control the amount of timing adjustment that the **MAC** entity has to apply. It is used to **synchronize** the **UE**'s **Uplink carrier** in respect to the **Downlink carrier**.

- **C-RNTI** (16 bits)

  It is the **identity** that is used by the **MAC entity** upon **completion** of the **random access procedure** (click 9).

For more detail on the **MSGB MAC payload**, refer to clause 6.2.3a of TS 38.321.

For more detail on **MAC PDUs** and **CEs**, refer to clause 6 of TS 38.321.

### 4.5.6 HARQ protocol in 5G

The **5G HARQ protocol** is **always** used for **unicast** and **multicast** delivery of **transport blocks** (click 4.12.1.1) in **PDSCH** and in **PUSCH**. The **5G HARQ protocol** is **not** used for **broadcast** delivery of **transport blocks** in **PDSCH**. Likewise, it is **not** used in **any** of the **other physical channels** (click 4.11.3). The way **PDSCH HARQ** and **PUSCH HARQ** work is **quite similar**, but it has a **few relevant differences**.

**Each transport block** (click 4.12.1.1) is **1 service data unit** of the **HARQ protocol**. **Each codeword** (click 4.12.1.2) is **1 protocol data unit** of the **HARQ protocol**. However, the **codewords** do **not** contain **any control field** for the **HARQ protocol**. The **control fields** for the **HARQ protocol** are **rather** delivered as **downlink control information** (**DCI**) **message** (click 6.9) **fields** and as **uplink control information** (**UCI**) (click 7.7).

The **NR node** has **1 PDSCH Tx HARQ protocol entity** for **each destination UE** (or **group** of **UEs**) and **cell**. **Each destination UE** and **cell** has **1 PDSCH Rx HARQ protocol entity**.

The **NR node** has **1 PUSCH Rx HARQ protocol entity** for **each source UE** and **cell**. **Each source UE** and **cell** has **1 PUSCH Tx HARQ protocol entity**.

**Each** of the **2 Tx/Rx HARQ protocol entities** in **Tx/Rx each pair** contains the **same quantity** of **HARQ processes** (click 4.5.6.1). The **HARQ protocol entities** in **2 different Tx/Rx pairs may** contain a **different quantity** of **HARQ processes**.

**Both** the **PDSCH HARQ protocol** and the **PUSCH HARQ protocol** are **asynchronous**, with **fractional retransmissions** (click 4.5.6.6.2), with **incremental redundancy** (click 4.5.6.6.3 and 11.12) and **adaptive retransmissions** (click 4.5.6.6.4).

**Asynchronous** means that the **time** between a **HARQ packet transmission** and the **reception** of its corresponding **(N)ACK** is **not** a **fixed quantity** of **slots**, and it is rather **variable**. The **timing** of the **(N)ACKs** of **PUSCH HARQ** is decided by the **NR node**, as part of **PDCCH Tx scheduling** (click 6.8). The **timing** of the **(N)ACKs** of **PDSCH HARQ** is signaled by the **NR node** to **each UE** as part of the **PUCCH scheduling** (click 7.6.1).

Being **asynchronous** and **adaptive** makes **5G HARQ especially suitable** for **eMBB** use cases. The **incremental redundancy** is based on **LDPC coding** (click 4.12.5).

More **detailed characteristics** of the **5G HARQ protocol** are indicated in the following subsections.

For an **overview** of **HARQ protocols** (including an overview of **LTE HARQ** and **UMTS HARQ**), click 11.12.

For more detail on **PDSCH HARQ**, refer to clause 5.3.2 of TS 38.321 and clause 5.1 of TS 38.214.

For more detail on **PUSCH HARQ**, refer to clause 5.4.2 of TS 38.321 and clause 6.1 of TS 38.214.

#### 4.5.6.1 5G HARQ processes

For **both PDSCH HARQ** and **PUSCH HARQ each 5G HARQ process** is **1 stop-and-wait protocol entity**. **Each Tx HARQ process handles** the **encoding**, **transmission** and **retransmission** of **1 transport block** (click 4.12.1.1). **Each Rx HARQ process handles** the **error detection**, **error correction** and **acknowledgement** of its **specific sequence** of **transport block(s)**. **Each Tx/Rx HARQ process pair** handles **each** of its **Tx/Rx transport blocks** in a **completely independent way** from the other **HARQ processes.**

**Acknowledging** and **retransmitting either** the **whole transport block** or **only** a **fraction** of it depends on the **5G HARQ mode** (click 4.5.6.4).

Since **5G HARQ** is **asynchronous**, its **HARQ processes** may be used in **any order** (i.e., they are **not obliged** to be used **round-robin**). It is then **required** to **explicitly identify** the **HARQ process** to which **each transport block**, **each NDI bit** (click 4.5.6.3) and **each (N)ACK** (click 4.5.6.7) corresponds. This is done with the **DCI message** (click 6.9) **field** "***HARQ process number***", whose value **ranges** from **0** to the **quantity**, minus 1, of **HARQ processes**. When a **DCI message** performs a **PDSCH** or **PUSCH assignment** for the **transmission** of a **transport block**, the **value** of this **field** in the **DCI message** identifies the **HARQ process** to which the **transport block belongs**. The following table shows, for **PDSCH HARQ** and for **PUSCH HARQ**, the **DCI message Formats** where

the said **field** is **present**, what is **assigned** by **each** of these **DCI Formats** and what is the **field**'s **bitlength** in **each** of these **DCI Formats**:

| | DCI message field "*HARQ process number*" | | |
|---|---|---|---|
| | **Assigns** | **DCI Format** | **Bitlength** |
| **PUSCH HARQ** | **PUSCH Unicast** | Format 0_0 (6.9.2) | 4 |
| | | Format 0_1 (6.9.3) | 4, 5 |
| | | Format 0_2 (6.9.4) | 0 to 5 |
| **PDSCH HARQ** | **PDSCH Unicast** | Format 1_0 (6.9.5) | 4 |
| | | Format 1_1 (6.9.6) | 4, 5 |
| | | Format 1_2 (6.9.7) | 0 to 5 |
| | **PDSCH MBS** | Format 4_1 (6.9.9.2) | 4 |
| | | Format 4_2 (6.9.9.3) | 4 |

**Formats 3_x** have **not** been included in the **table** because they correspond to **Sidelink**.

**Format 4_0** does **not** have the "***HARQ process number***" field because it **schedules PDSCH broadcast** data, and **HARQ** is **not** used with **broadcast** data.

For more detail on **PDSCH HARQ**, refer to clause 5.3.2 of TS 38.321 and clause 5.1 of TS 38.214.

For more detail on **PUSCH HARQ**, refer to clause 5.4.2 of TS 38.321 and clause 6.1 of TS 38.214.

### 4.5.6.2 Quantity of 5G HARQ processes

**Each** of the **2 Tx/Rx HARQ protocol entities** in **each Tx/Rx HARQ pair** contains the **same quantity** of **Tx/Rx HARQ processes** (click 4.5.6.1). The **HARQ protocol entities** in **2 different Tx/Rx HARQ pairs may** contain a **different quantity** of **Tx/Rx HARQ processes**.

In **PDSCH HARQ**, **each UE** has, in **each** of **its cells**, the following **quantity** of **HARQ Rx processes**:

- The **maximum quantity** is **16**, extended to **32** in **R17**.
- The **actual quantity** is configured to **each UE** with **either** `nrofHARQ-ProcessesForPDSCH` or `nrofHARQ-ProcessesForPDSCH-v1700`, **both** from `PDSCH-ServingCellConfig` (click 6.18.10).
- If **both** said parameters are **absent**, the **UE** assumes the **quantity** of **8**.

In **PUSCH HARQ**, **each UE** has, in **each** of **its cells**, the following **quantity** of **Tx HARQ processes**:

- The **maximum quantity** is **16**.
- The **actual quantity** is configured to **each UE** with `nrofHARQ-ProcessesForPUSCH-r17`, from `PUSCH-ServingCellConfig` (click 7.15.8).
- If the said parameter is **absent**, the **UE** assumes the **quantity** of **16**.

For more detail on **PDSCH HARQ**, refer to clause 5.3.2 of TS 38.321 and clause 5.1 of TS 38.214.

For more detail on **PUSCH HARQ**, refer to clause 5.4.2 of TS 38.321 and clause 6.1 of TS 38.214.

#### 4.5.6.3 5G HARQ new data indicator (NDI)

Following the **stop-and-wait procedure**, **each Tx HARQ process changes** the **value** (i.e., "**toggles**") of a **control bit** for **each newly transmitted transport block**. In **every retransmitted codeword** (that corresponds to an already transmitted **transport block**), the **bit value** is the **same** as in its **first transmission**. This is **required** for the corresponding **Rx HARQ process** to **distinguish** a **newly transmitted transport block** from a **retransmitted codeword** that corresponds to an already transmitted **transport block**. The **alternating** value of **each** such **bit** is why **stop-and-wait protocols** are sometimes called "**alternating bit**" **protocols**.

The **HARQ NDI bit(s)** is/are **not** sent as part of the **codeword** itself, and it/they is/are **rather** sent as follows:

- For **PDSCH HARQ**, the **NDI bit(s)** is/are sent by the **NR node** in the **DCI message** (click 6.9) **field** "***New data indicator***" (**NDI**) that **assigns** the **PDSCH** resources for the **UE**'s **reception** of the corresponding **transport block**. The following table shows, for **PDSCH HARQ**, the **DCI message Formats** where the said **field** is **present**, what is **assigned** by **each** of these **DCI Formats**, the **quantity** of **fields** in **each DCI message** and what is **each field**'s **bitlength** in **each** of these **DCI Formats**:

| | DCI message field "*New data indicator*" | | | |
|---|---|---|---|---|
| | **Assigns** | **DCI Format** | **# field(s)** | **Bitlength** |
| **PDSCH HARQ** | **PDSCH Unicast** | Format 1_0 (6.9.5) | 1 | 1 |
| | | Format 1_1 (6.9.6) | 1 or 2 | 1 to 8 |
| | | Format 1_2 (6.9.7) | 1 | 1 |
| | **PDSCH MBS** | Format 4_1 (6.9.9.2) | 1 | 1 |
| | | Format 4_2 (6.9.9.3) | 1 or 2 | 1 |

  **Formats 1_1** and **4_2** may have **1** or **2** "***New data indicator***" **fields**, because **each DCI message** with **either** of these **Formats may** schedule **1 or 2 transport block(s)** in **each PDSCH transmission**.

  **Format 4_0** does **not** have the "***New data indicator***" **field**, because it **schedules PDSCH broadcast** data, and **HARQ** is **not** used with **broadcast** data.

  The **bitlength** of **each** "***New data indicator***" **field** in **Formats 1_0**, **1_2**, **4_1** and **4_2** is **1 bit**, because **each DCI message** with **either** of these **Formats may** perform **only 1 PDSCH** or **PUSCH assignment** (click 6.9.6) that corresponds to **only 1 PDSCH** or **PUSCH transmission**.

  The **bitlength** of **each** "***New data indicator***" **field** in **Format 1_1** may be from **1** to **8 bits** because **each DCI message** with this **Format may** perform **1** to **8 PDSCH assignments** (click 6.9.6), **each** corresponding to **1 PDSCH transmission**.

- For **PUSCH HARQ**, this **bit** is **also** sent by the **NR node** in the **field** "***New data indicator***" (**NDI**) of the **DCI message** (click 6.9) that **assigns** the **PUSCH** resources for the **UE**'s **transmission** of the corresponding **transport block**. This is sent in the **opposite direction** that one would expect, because it is sent from the **NR node** (i.e., the **receiver** in **PUSCH HARQ**) to the **UE** (i.e., the **transmitter**

in **PUSCH HARQ**). The reason is that the **NR node** indicates the **UE** when it should transmit a **new transport block**, by "**toggling**" the value of the **NDI bit** in the **DCI message** that **assigns** the **UE** the resources to transmit a **PUSCH transport block**. Notice that in doing this, the **NR node** is also **acknowledging** (click 4.5.6.7) the **received transport block** as a **ACK** (**NDI is toggled**) or a **NACK** (**NDI not toggled**) in the corresponding **HARQ process**. The following table shows, for **PUSCH HARQ**, the **DCI message Formats** where the said **field** is **present**, what is **assigned** by **each** of these **DCI Formats** and what is the **field**'s **bitlength** in **each** of these **DCI Formats**:

| | DCI message field "*New data indicator*" | | |
|---|---|---|---|
| | **Assigns** | **DCI Format** | **Bitlength** |
| **PUSCH HARQ** | **PUSCH Unicast** | Format 0_0 (6.9.2) | 1 |
| | | Format 0_1 (6.9.3) | 1 to 8 |
| | | Format 0_2 (6.9.4) | 1 |

**Format 0_1** may have a "*New data indicator*" **bitlength** from **1** to **8 bits** because **1 DCI message** with this **Format may** perform **1** to **8 PUSCH assignments** (click 6.9.3), **each** corresponding to **1 PUSCH transmission**.

Notice that **all** the **Formats 0_x only** have **1** "*New data indicator*" field, because **PUSCH only** supports the **transmission** of **1 codeword for each PUSCH assignment**.

For more detail on **PDSCH HARQ**, refer to clause 5.3.2 of TS 38.321 and clause 5.1 of TS 38.214.

For more detail on **PUSCH HARQ**, refer to clause 5.4.2 of TS 38.321 and clause 6.1 of TS 38.214.

### 4.5.6.4 5G HARQ modes: transport block mode and CBG mode

**Both PDSCH HARQ** and **PUSCH HARQ** support **two operation modes**: **transport block** (**TB**) **mode** (same as **LTE**) and **code block group** (**CBG**) **mode**.

The **HARQ mode must** be the **same** for **all** the **PDSCH HARQ processes** of **every UE** and **cell**. The same applies to **all** the **PUSCH HARQ processes** of **every UE** and **cell**. The **HARQ mode** may be **different** between the **PDSCH HARQ processes** and the **PUSCH HARQ processes**.

**PDSCH HARQ** uses **transport block mode** if `codeBlockGroupTransmission` from `PDSCH-ServingCellConfig` (click 6.18.10) is **absent**. If it is **present**, it uses **CBG mode**.

**PUSCH HARQ** uses **transport block mode** if `codeBlockGroupTransmission` from `PUSCH-ServingCellConfig` (click 7.15.8) is **absent**. If it is **present**, it uses **CBG mode**.

#### Transport block mode

In **TB mode**, an **erroneous transport block** is retransmitted in **whole**. However, in **CBG mode only** the **erroneous** "**parts**" of an **erroneous transport block** are **retransmitted**. This requires that **each transport block** is divided in "**parts**", and that these "**parts**" may be individually **acknowledged**.

### Code block group mode

In **CBG mode**, **each transport block** is **divided** into **1** or **several code block group(s)**. **Each code block group** may be **individually checked** for **correctness** and **individually acknowledged**. **Only** the **unacknowledged code block group(s)** is/are **retransmitted**, which is **more efficient** than **always retransmitting** the whole **transport block**.

**Each code block group** is **considered** to be **1** or **several consecutive code block(s)** (click 4.12.4). **Each code block** is a (quasi) **fixed bitlength bit sequence** produced by the (possible) **segmentation** of a **transport block**. **Each transport block** is (possibly) **segmented** into **code blocks** to obtain **bit sequences** with a **suitable bitlength** for their **LDPC encoding** (click 4.12.5).

In **CBG mode**, **1** or **several consecutive code block(s)** are **considered** as being **grouped** into **1 CBG** such that **each acknowledgement** and **each retransmission** is done for **1 CBG**.

The **code block(s)** **not** actually **grouped** in **CBGs** in the sense that there is **no** actual **fields** nor any **encoding** that **implements** the **grouping**. The **code block(s)** are just **considered** to be **grouped** in **CBGs**, because of a **joint agreement** between **each HARQ transmitter process** and its corresponding **HARQ receiver process**.

**CBG mode** is **more efficient** for configurations with **large transport blocks** because it **retransmits only** the required **fraction** of a **transport block** instead of the **whole transport block**. Notice that a **5G transport block** **may** be **very large** (**up to ~156.0 KiB**, click 4.12.1), although in **most** configurations it is **smaller** than that.

**Maximum quantity** of **CBGs** in **each transport block**

The **maximum quantity** of **CBGs** for **each transport block** is configured with parameter `maxCodeBlockGroupsPerTransportBlock` (where its value is denoted "$N_{CBG,max}^{TB}$"). Its **data type** is `ENUMERATED {n2, n4, n6, n8}`. As you may see, the **maximum quantity** of **CBGs** in a **transport block** that can be configured is `8`.

For **PDSCH**, the said **parameter** is from `PDSCH-CodeBlockGroupTransmission`, which is instantiated within `PDSCH-ServingCellConfig` (click 6.18.10), and in case of **2 PDSCH transport blocks** the **maximum value** that can be configured is `4`.

For **PUSCH**, the said **parameter** is from `PUSCH-CodeBlockGroupTransmission`, which is instantiated within `PUSCH-ServingCellConfig` (click 7.15.8).

**Quantity** of **CBGs** in **each transport block** and **quantity** of **code block(s)** in **each CBG**

***If*** [[the **quantity** "$N_{CB}^{SCB(q)}$" of **code block(s)** of **transport block** "**q**"] is **larger** than ["$N_{CBG,max}^{TB}$" (the **maximum** configured **quantity** of **CBGs** in a **transport block**]],

> ***then***, the **quantity** of **CBG(s)** "$N_{CBG}^{TB(q)}$" is "$N_{CBG,max}^{TB}$" (the **maximum**). In this case, **every CBG** has "***ceiling***($N_{CB}^{SCB(q)}/N_{CBG,max}^{TB}$)" **code blocks**, **except** the **last** "$N_{CB}^{SCB(q)}$ ***mod*** $N_{CBG,max}^{TB}$" **CBG(s)**, that **each** has "***floor***($N_{CB}^{SCB(q)}/N_{CBG,max}^{TB}$)" **code block(s)**.
>
> ***else***, the **quantity** "$N_{CBG}^{TB(q)}$" of **CBG(s)** in **transport block** is the **quantity** of **code block(s)** "$N_{CB}^{STB(q)}$" in sequence "**q**". In this case, **every CBG** has **only 1 code block**.

As you may see, **all** the **code block group(s)** from a given **transport block** have **almost** the **same quantity** of **code block(s)** (the **maximum difference** is **1 code block**).

**CBG mode** is sometimes called "**code block group based transmission**".

The **HARQ CBG mode** in **PDSCH** and in **PUSCH** seems **incompatible** with the **multi-scheduling option** of **PDSCH** (click 6.9.6) and of **PUSCH** (click 6.9.3), respectively. The reason is that when the **multi-scheduling option** is **set** in **PDSCH** or in **PUSCH**, there are **not** enough **bits** in the "***CBG transmission information***" **field** to **signal** or to **acknowledge** (respectively) the required **quantity** of **CBGs**.

For more detail on **code block group based transmission**, refer to clauses 5.1.7 (for **PDSCH**) and 6.1.5 (for **PUSCH**) of TS 38.214.

For more detail on **PDSCH HARQ**, refer to clause 5.3.2 of TS 38.321 and clause 5.1 of TS 38.214.

For more detail on **PUSCH HARQ**, refer to clause 5.4.2 of TS 38.321 and clause 6.1 of TS 38.214.

### 4.5.6.5 5G HARQ channel coding

**Both PDSCH HARQ** and **PUSCH HARQ** use **CRC** for **error detection** and **Low-Density Parity Check** (**LDPC**) for **error correction** (i.e., **FEC**) in **both PDSCH HARQ** and **PUSCH HARQ**.

**Each transport block** is appended a **CRC** to produce a **protected block** (click 4.12.3). If the **protected block exceeds** a certain **bitlength**, it is **segmented** (click 4.12.4) into **several code blocks**. **Each code block** has a **CRC** and it is then **FEC encoded** (click 4.12.5) with **LDPC** and **rate-matched** (click 4.12.6).

The **specific 5G LDPC** code has **2 base code rates**: **1/3** and **1/5**, departing from which around **20** cases of different **code rates** are supported.

The **LDPC FEC redundancy** is **changed** in **every retransmission** (click 4.5.6.6) according to the **incremental redundancy** technique.

### 4.5.6.6 5G HARQ retransmissions

**HARQ retransmits** when receiving a **HARQ NACK** (click 4.5.6.7) or if the **(N)ACK** is **not received** according to the **scheduling**. The **HARQ protocol** does **not** use a **timer-based retransmissions**, as most link layer protocols do.

**Both** the **PDSCH HARQ protocol** and the **PUSCH HARQ protocol** use **fractional retransmissions** (click 4.5.6.6.2), with **incremental redundancy** (click 4.5.6.6.3 and 11.12) and **adaptive retransmissions** (click 4.5.6.6.4). As **most HARQ protocols do**, it has a **maximum quantity** of **retransmissions** (click 4.5.6.6.5).

**Both PDSCH HARQ** and **PUSCH HARQ** support **retransmitting** a **fraction** of a **transport block** and use **incremental redundancy** (click 11.12) plus **adaptive retransmissions**.

In **PUSCH HARQ**, the **UE** is **assigned** the **PUSCH resources** that the **UE requires** for the **retransmission** in the **same DCI message** where the **NR node** has signaled the **NACK**. The **UE** therefore does **not** send **scheduling requests** (click 7.7.1.2) for **retransmissions**.

#### 4.5.6.6.1 Retransmitting a complete transport block

The **bit sequence** of **each successive retransmission** of the **same transport block** is **different**, because of the usage of **incremental redundancy** (click 4.5.6.6.3). **HARQ retransmission** is therefore **more complex** than **just** resending the **same** initial **codeword**.

**Independently** of the **HARQ mode** (click 4.5.6.4), the **retransmission** of a **complete transport block** is signaled with the "***New data indicator***" **DCI message field**, as described in 4.5.6.3.

#### 4.5.6.6.2 Retransmitting a fraction of a transport block (CBGs)

**Recall** from 4.5.6.4 that in **CBG mode only** the **unacknowledged CBG(s)** are **retransmitted**, and **not** the **whole transport block**. In **CBG mode**, a **retransmission** corresponding to a given **transport block** by a given **HARQ process** contains **all** the **unacknowledged CBG(s)** from the **last retransmission**. The **quantity** of **CBG(s)** within **each successive retransmission** is then **successively reduced**. The **bit sequence** of **each successive retransmission** of the **same CBG** is **different**, because of the usage of **incremental redundancy** (click 4.5.6.6.3). **HARQ retransmission** is therefore **more complex** than **just** resending the **same** initial **encoded blocks** that correspond to the **CBG**.

**PDSCH HARQ signaling of retransmitted CBGs**

In **PDSCH HARQ CBG mode** (click 4.5.6.4), the **NR node explicitly informs** the **UE** about the **CBGs** of **each** of the **1 or 2 transport block(s)** that it is **retransmitting** with the "***CBG transmission information***" field of the **DCI Format 1_1 message** that schedules the corresponding **PDSCH assignment**. This **field** is a **bitmap** for the **1** or **2 CBG sequences** where **each** "**1**" indicates a **retransmitted CBG**. The **MSB half** of the **bitmap** corresponds to the **first transport block** and the **LSB half** corresponds to the **second transport block**. The **CBG indices** to which **each bit** corresponds **start** in "**0**" and continue **sequentially upwards** from **MSB** to **LSB**. The following table shows, for **PDSCH HARQ**, the **DCI message Format** where this **field** is **present**, what is **assigned** by this **DCI Format** and what is its **field**'s **bitlength**:

| | **DCI message field "*CBG transmission information*"** | | |
|---|---|---|---|
| | **Assigns** | **DCI Format** | **Bitlength** |
| **PDSCH HARQ** | **PDSCH Unicast** | Format 1_1 (6.9.6) | 0, 2, 4, 6, 8 |

The **field bitlength** is "**0**" if **CBG mode** (click 4.5.6.4) is **not configured** for **PDSCH**. Otherwise, the **field bitlength** is the **maximum quantity** of **transport blocks** per **PDSCH assignment multiplied** by the **maximum quantity** of **CBGs** per **transport block** in **PDSCH**. The **maximum quantity** of **transport blocks** per **PDSCH assignment** is configured with `maxNrofCodeWordsScheduledByDCI` from `PDSCH-Config` (click 6.18.6) and can take the values `1` or `2`. The **maximum quantity** of **CBGs** per **transport block** in **PDSCH** is configured with `maxCodeBlockGroupsPerTransportBlock` from `PDSCH-CodeBlockGroupTransmission` within `PDSCH-ServingCellConfig` (click 6.18.10), and can take the values `2`, `4`, `6` or `8`. However, the values `6` and `8` **cannot** be configured when `maxNrofCodeWordsScheduledByDCI` is `2`. Consequently, for a **maximum quantity** of **transport blocks** of **either** `1` or `2`, the **field**'s **bitlength** allows to signal **all** the possible **CBGs**.

Notice that this **field** has **different semantics** within **Format 1_1** for **PDSCH HARQ** and within **Format 0_1** for **PUSCH HARQ-ACKs** (click 4.5.6.7.2).

**Formats 1_0**, **1_2**, **4_1** and **4_2** do **not** have the "***CBG transmission information***" field because **transport blocks** assigned by them do **not** support the **HARQ CBG mode**.

**Format 4_0** does **not** have the "***CBG transmission information***" field because it **schedules PDSCH broadcast** data, and **HARQ** is **not** used with **broadcast** data.

For more detail on **signaling** the **retransmitted CBGs** in **PDSCH**, refer to clause 5.1.7.2 of TS 38.214.

#### 4.5.6.6.3 Incremental redundancy

This consists of sending **different FEC bits** in **each (re)transmission** of a **transport block**. Therefore, the **initial transmission** of a **transport block** and its **successive retransmissions** contain **different bit sequences** (i.e., the **transport block** is **not retransmitted as such**). **Each retransmission** of a **transport block** is based on the **LDPC encoded block(s)** that were **computed** for the **first transmission**. However, in **each successive retransmission** of the **transport block** a **different rate-matching** (click 4.12.6) is applied to **every encoded block**. This results in the **transmission** of a **sequence rate-matched blocks(s)** where **each rate-matched block** is **different** every time.

**Four** basic **cases** of **different FEC redundancy bits** are **supported**. These **four cases** are called **redundancy versions** the corresponding value is signaled in the **DCI field** "***Redundancy version***". This **field** is usually denoted as "$rv_{id}$" and its values may be {0, 1, 2, 3}. **Each value** of "$rv_{id}$" indicates which of the **4 rate-matched block** "**versions**" of **each same encoded block**, is sent. The **NR node** decides what **redundancy version** is used in the **initial transmission** and in **each successive retransmission** of a **transport block**, **both** for **PDSCH** and for **PUSCH**.

The **redundancy version** of a **PDSCH transport block** is indicated in the "***Redundancy version***" field of the **DCI message** that **assigns** the corresponding **PDSCH** resources to the **destination UE** (or **group** of **UEs**). The **redundancy version** that the **UE must** use for **each PUSCH transport block** is **signaled** by the **NR node** to the **UE** in the "***Redundancy version***" field of the **DCI message** that **assignments** the corresponding **PUSCH** resources to the **UE**. The following table shows, for **PDSCH HARQ** and for **PUSCH HARQ**, the **DCI message Formats** where the said **field** is **present**, what is **assigned** by **each** of these **DCI Formats**, the **quantity** of **fields** in **each DCI message** and what is **each field**'s **bitlength** in **each** of these **DCI Formats**:

| | DCI message field "*Redundancy version*" | | | |
|---|---|---|---|---|
| | **Assigns** | **DCI Format** | **# field(s)** | **Bitlength** |
| **PUSCH HARQ** | **PUSCH Unicast** | Format 0_0 (6.9.2) | 1 | 2 |
| | | Format 0_1 (6.9.3) | 1 | 2 to 8 |
| | | Format 0_2 (6.9.4) | 1 | 0 to 2 |
| **PDSCH HARQ** | **PDSCH U/B** | Format 1_0 (6.9.5) | 1 | 2 |
| | **PDSCH Unicast** | Format 1_1 (6.9.6) | 1 or 2 | 2 to 8 |
| | | Format 1_2 (6.9.7) | 1 | 0 to 2 |
| | **PDSCH MBS** | Format 4_0 (6.9.9.1) | 1 | 2 |
| | | Format 4_1 (6.9.9.2) | 1 | 2 |
| | | Format 4_2 (6.9.9.3) | 1 or 2 | 2 |

The text "**U/B**" stands for **unicast** or **broadcast**.

**Formats 1_1** and **4_2** may have **1** or **2** "***Redundancy version***" **fields**, because **each DCI message** with **either** of these **Formats may** schedule **1 or 2 transport block(s)** in **each PDSCH transmission**.

**Format 4_0 has** the "***Redundancy version***" **field even though** it **schedules PDSCH broadcast** data and **HARQ** is **not** used with **broadcast** data. The reason is that **rate-matching** is applied to **Format 4_0 broadcast transport blocks**, because they use **FEC** in spite of them **not** being processed by **HARQ**.

The **bitlength** of **each** "***Redundancy version***" **field** in **Formats 0_0**, **1_0**, **4_0**, **4_1** and **4_2** is **2 bits**, because **each DCI message** with **either** of these **Formats may** perform **only 1 PDSCH** or **PUSCH assignment** (click 6.9.6) that corresponds to **only 1 PDSCH** or **PUSCH transmission**, and **2 bits** are **required** to convey the "***Redundancy version***" value.

The **bitlength** of **each** "***Redundancy version***" **field** in **Formats 0_1** and **1_1** is **2** to **8 bits**. It is **2** ***if*** **only 1 PDSCH** or **PUSCH assignment** (click 6.9.6) is performed by the enclosing **DCI message**. **Else**, it is the **maximum quantity** of **PDSCH** or **PUSCH assignments** that can be performed by the enclosing **DCI message**, where **each bit** corresponds to **each PDSCH** or **PUSCH assignment**. **Bit value** "**0**" implies redu**ndancy version** "**0**" and **bit value** "**1**" implies **redundancy version** "**2**".

**PDSCH HARQ incremental redundancy in CBG mode**

For the case of **PDSCH HARQ incremental redundancy** in **CBG mode**, the **NR node may also signal** the **UE** if it can **compute incremental redundancy** or **not** with the **retransmitted CBGs**. This is done with the "***CBG flushing out information***" field of the **DCI Format 1_1 message** that schedules the corresponding **PDSCH assignment**. Value "**1**" indicates that the **UE** should **compute incremental redundancy**. Value "**0**" indicates that the **UE** should discard **all previously received CBGs** (for the current **transport block(s)**). The following table shows, for **PDSCH HARQ**, the **DCI message Format** where this **field** is **present**, what is **assigned** by this **DCI Format** and what is its **field**'s **bitlength**:

| | **DCI message field "*CBG flushing out information*"** | | |
|---|---|---|---|
| | **Assigns** | **DCI Format** | **Bitlength** |
| **PDSCH HARQ** | **PDSCH Unicast** | Format 1_1 (6.9.6) | 0, 1 |

The **field bitlength** is **0** if **CBG mode** is **not configured** for **PDSCH**. Otherwise, its **bitlength** is **1**.

**Formats 1_0**, **1_2**, **4_1** and **4_2** do **not** have the "***CBG flushing out information***" field because **transport blocks** assigned by them do **not** support the **HARQ CBG mode**.

**Format 4_0** does **not** have the "***CBG flushing out information***" field because it **schedules PDSCH broadcast** data, and **HARQ** is **not** used with **broadcast** data.

For more detail on **PDSCH HARQ incremental redundancy in CBG mode**, refer to clause 5.1.7.2 of TS 38.214.

#### 4.5.6.6.4 Adaptive retransmissions

This consists of the **Modulation and Coding Scheme** (**MCS**) possibly **changing** in **each successive retransmission** from an original **transport block**. This allows to **adapt MCS** to the **current channel conditions**, in order to **maximize** the **probability** that the **transport block** is **correctly received** in **each retransmission**.

When **2 transport blocks** are **scheduled** with **1 DCI message**, in **PDSCH each transport block** has its **specific value** of "***Modulation and coding scheme***", "***New data indicator***" and "***Redundancy version***" (refer to **DCI Format 1_1**, click 6.9.6). However, in **PUSCH both transport blocks must** have the **same value** for the said **3 DCI fields** (refer to **DCI Formats 0_x**, click 6.9).

#### 4.5.6.6.5 Maximum quantity of retransmissions

The **maximum quantity** of **retransmissions** "$N_{Rtx,max}^{HARQ}$" is **frequently** configured from **4** to **8**, but the **actual value** depends on each **deployment scenario**. **Lowering** the value of "$N_{Rtx,max}^{HARQ}$" **reduces delay** at the **cost** of **increased transport block loss**. A certain percentage of **lost** (and **duplicated**) **transport block(s)** are acceptable in **HARQ protocols**.

For more detail on **PDSCH HARQ**, refer to clause 5.3.2 of TS 38.321 and clause 5.1 of TS 38.214.

For more detail on **PUSCH HARQ**, refer to clause 5.4.2 of TS 38.321 and clause 6.1 of TS 38.214.

### 4.5.6.7 5G HARQ acknowledgements

**5G HARQ** supports **transport block acknowledgements**, **CBG acknowledgements** and **multi-UE acknowledgements**.

A **transport block acknowledgement** is an **ACK** that corresponds to a **whole received transport block**. This can be used **both** in **transport block mode** and in **CBG mode** (click 4.5.6.4).

A **CBG acknowledgement** is a (**N**)**ACK** that corresponds to **1** or **several received CBGs**. This can **only** be used in **CBG mode**.

A **multi-UE acknowledgement** is a **PUSCH HARQ acknowledgment** that corresponds to **more than one UEs**.

**CBG** and **multi-UE acknowledgements** are sometimes called "**structured acknowledgments**".

**Acknowledgements** work **differently** in **PDSCH HARQ** (click 4.5.6.7.1) and in **PUSCH HARQ** (click 4.5.6.7.2).

Some **complex scenarios** require **additional mechanisms** for **managing HARQ-ACKs** (click 4.5.6.7.3).

#### 4.5.6.7.1 PDSCH HARQ acknowledgements

In **PDSCH HARQ**, the **HARQ** (**N**)**ACKs** are **sent back** by the **UE** within a **UCI message** (click 7.7) transmitted through **either PUCCH** or **PUSCH**. The **scheduling** of the **transmission resources** required by the **UE** to **send back** the **HARQ-ACKs** of the **1** or **2 PDSCH codeword(s)** is performed with the same **DCI message** (click 6.9) that **assigns** the **UE** the **PDSCH resources** for the reception of the **1** or **2 PDSCH codeword(s)**.

**PDSCH HARQ acknowledgments in transport block mode**

In **PDSCH HARQ transport block mode** (click 4.5.6.4), the **UE acknowledges each** of the **1** or **2 transport blocks** that have been received in a given **slot** with **1 UCI bit**, where "**1**" indicates **ACK** and "**0**" indicates **NACK**.

**PDSCH HARQ acknowledgments in CBG mode**

In **PDSCH HARQ CBG mode** (click 4.5.6.4), the **UE acknowledges each CBG** with **1 UCI bit**, where "**1**" indicates **ACK** and "**0**" indicates **NACK**. If **all** the **pending CBGs** are **acknowledged**, then the whole **transport block** is **acknowledged**.

For more detail on **PDSCH HARQ acknowledgments** in **CBG mode**, refer to clause 9.1.2 of TS 38.213.

**PDSCH HARQ acknowledgment retransmission**

If the **NR node cannot** correctly **decode** the **HARQ-ACK** information from an **incoming UCI message**, it may **request** its **retransmission** with the **DCI message field** "***HARQ-ACK retransmission indicator***". When this field is **set** to "**1**", the **recipient UE retransmits** the corresponding **HARQ-ACK** information. This **reduces unnecessary transport block retransmissions** due to **lost ACKs**. The following table shows, for **PDSCH HARQ**, the **DCI message Formats** where the said **field** is **present**, what is **assigned** by **each** of these **DCI Formats** and what is the **field**'s **bitlength** in **each** of these **DCI Formats**:

| | DCI message field "*HARQ-ACK retransmission indicator*" | | |
|---|---|---|---|
| | **Assigns** | **DCI Format** | **Bitlength** |
| **PDSCH HARQ** | **PDSCH Unicast** | Format 1_1 (6.9.6) | 0, 1 |
| | | Format 1_2 (6.9.7) | 0, 1 |

This **field** is **optional**. It **only** exists if `pdsch-HARQ-ACK-Retx` is **present**.

**Formats 1_0** and **4_x** do **not** have the "***HARQ-ACK retransmission indicator***" field because they are **not** used to request **HARQ-ACK retransmission**.

When the **UE** is **transmitting back** a given **HARQ-ACK** for the **last time**, it signals this to the **NR node** using the **UCI Last Resource Retransmission** (click 7.7.1.4) indication.

**PDSCH one-shot HARQ acknowledgement**

The **NR node may request** the **UE** to **send back** a **single UCI message** with **complete HARQ-ACK feedback** for **several PDSCH assignments**. The **NR node** sends this **request** with the **DCI message field** "***One-shot HARQ-ACK request***". When this field is **set** to "**1**", the **recipient UE sends back** the **UCI message** with **complete HARQ-ACK feedback**. The following table shows, for **PDSCH HARQ**, the **DCI message**

**Formats** where the said **field** is **present**, what is **assigned** by **each** of these **DCI Formats** and what is the **field**'s **bitlength** in **each** of these **DCI Formats**:

| | DCI message field "*One-shot HARQ-ACK request*" | | |
|---|---|---|---|
| | **Assigns** | **DCI Format** | **Bitlength** |
| **PDSCH HARQ** | **PDSCH Unicast** | Format 1_1 (6.9.6) | 0, 1 |
| | | Format 1_2 (6.9.7) | 0, 1 |

The **one-shot HARQ-ACK feedback** is particularly **useful** for **large bandwidths** and **massive MIMO scenarios**, because it allows for more **efficient** and **compact HARQ-ACK reporting**.

**Formats 1_0** and **4_x** do **not** have the "***One-shot HARQ-ACK request***" field because they are **not** used to request **one-shot HARQ-ACK**.

**PDSCH HARQ acknowledgements for multicast**

By **default**, the **UE** does **not send back HARQ (N)ACKs** for a **received multicast transport block**. However, the **NR node** may configure the **UE** to **send** them **back**. If `harq-FeedbackEnablerMulticast-r17` from `MBS-RNTI-SpecificConfig-r17` is **present**, and **set** to `enabled`, the **UE always sends back HARQ (N)ACKs** for **received multicast/broadcast transport blocks**. If it is **present**, and **set** to `dci-enabler`, the **UE sends back HARQ (N)ACKs** for **received multicast transport blocks** if the **DCI message field** "***Enabling/disabling HARQ-ACK feedback indication***" is set to "**1**" in the **DCI message** that **assigns** the **UE** the corresponding **PDSCH reception resources**. The following table shows, for **PDSCH HARQ**, the **only DCI message Format** where this **field** is **present**, what is **assigned** by this **DCI Format** and what is its **field**'s **bitlength**:

| | DCI message field "*Enabling/disabling HARQ-ACK feedback indication*" | | |
|---|---|---|---|
| | **Assigns** | **DCI Format** | **Bitlength** |
| **PDSCH HARQ** | **PDSCH MBS** | Format 4_2 (6.9.9.3) | 0, 1 |

**Format 4_1** does **not** have the "***Enabling/disabling HARQ-ACK feedback indication***" field because **transport blocks** assigned by it does **not** support **DCI-enabled HARQ-ACK request**.

**Format 4_0** does **not** have the "***Enabling/disabling HARQ-ACK feedback indication***" field because it **schedules PDSCH broadcast** data, and **HARQ** is **not** used with **broadcast** data.

For more detail on **PDSCH HARQ**, refer to clause 5.3.2 of TS 38.321 and clause 5.1 of TS 38.214.

#### 4.5.6.7.2 PUSCH HARQ acknowledgements

In **PUSCH HARQ**, the **(N)ACKs** are **sent back** by the **NR node** within a **DCI message** (click 6.9) through **PDCCH**.

**PUSCH HARQ acknowledgments in transport block mode**

In **PUSCH HARQ transport block mode** (click 4.5.6.4), the **way** the **UE sends back** the **(N)ACKs** depends on the **type** of **PUSCH scheduling** configured to the **UE**:

- For **dynamic PUSCH scheduling**

  **Each HARQ (N)ACK** is **sent back** in **1** "***New data indicator***" **(NDI) field** (click 4.5.6.3) of the **DCI message** that **assignes** the **UE** the **transmission** of a **transport block** of the **same HARQ process** that **sent** the **acknowledged transport block**. If the **NDI** bit is "**toggled**", the **UE interprets** this as an **ACK**, because the **NR node only** "**toggles**" the **NDI bit** if the **last PUSCH (re)transmitted codeword** has been **correctly received**. If the **NDI** bit is **not** "**toggled**" it is interpreted as a **NACK**, and the **UE** will **retransmit** the **transport block**. Notice that **each HARQ process** behaves as **stop-and-wait protocol**, as was described in 4.5.6.1. Notice that whenever a **DCI message schedules 2 transport blocks**, it contains **2** "***New data indicator***" **(NDI) fields.**

- For **PUSCH semi-persistent scheduling**

  The **HARQ (N)ACKs** for **all** the **PUSCH HARQ processes** are **sent back** in the "***HARQ-ACK bitmap***" field of **1 DCI Format 0_1 message** for **configured grant Downlink feedback indication** (**CG-DFI**, click 6.9.3). **Each** "**1**" in the **bitmap** indicates **ACK**, and **each** "**0**" indicates **NACK**. The **identifiers** of the **PUSCH HARQ processes** to which **each bit** corresponds **start** in "0" and continue **sequentially upwards** from **MSB** to **LSB**.

  The following table shows, for **PUSCH HARQ** the only **DCI message Format** where this **field** is **present**, what is **assigned** by this **DCI Format** and what is its **field**'s **bitlength**:

| | DCI message field "*HARQ-ACK bitmap*" | | |
|---|---|---|---|
| | **Assigns** | **DCI Format** | **Bitlength** |
| **PUSCH HARQ** | **PUSCH Unicast** | Format 0_1 (6.9.3) | 16, 32 |

  The **field**'s **bitlength** is the **same** as the **quantity** of **PUSCH HARQ processes** (click 4.5.6.2).

  **Formats 0_0** and **0_2** do **not** have the "***HARQ-ACK bitmap***" field, because they do **not** support this **type** of **PUSCH HARQ-ACK delivery**.

**PUSCH HARQ acknowledgments in CBG mode**

In **PUSCH HARQ** in **CBG mode** (click 4.5.6.4), the **NR node sends back** the **HARQ (N)ACKs** for **all** the **CBGs** of the **PUSCH transport block** in the "***CBG transmission information***" field of the **DCI Format 0_1 message** that schedules the corresponding **PUSCH assignment**. This **field** is a **bitmap** where **each** "**1**" indicates **ACK**, and **each** "**0**" indicates **NACK**. The **CBG indices** to which **each bit** corresponds **start** in "**0**" and continue **sequentially upwards** from **MSB** to **LSB**. The following table shows, for

**PUSCH HARQ**, the **DCI message Format** where this **field** is **present**, what is **assigned** by this **DCI Format** and what its **field**'s **bitlength**:

| | DCI message field "*CBG transmission information*" | | |
|---|---|---|---|
| | **Assigns** | **DCI Format** | **Bitlength** |
| **PUSCH HARQ** | **PUSCH Unicast** | Format 0_1 (6.9.3) | 0, 2, 4, 6, 8 |

The **field bitlength** is "**0**" if **CBG mode** is **not configured** for **PUSCH** or if the **number** of **scheduled PUSCH** indicated by the "***Time domain resource assignment***" field is **larger** than **1**. Otherwise, the **field bitlength** is the **same** as the **maximum quantity** of **CBGs** per **transport block** in **PUSCH** (click 4.5.6.4).

Notice that this **field** has **different semantics** within **Format 0_1** for **PUSCH HARQ** and within **Format 1_1** for **PDSCH HARQ retransmissions** (click 4.5.6.6).

**Formats 0_0** and **0_2** do **not** have the "***CBG transmission information***" field, because they do **not** support **CBG ACK delivery**.

For more detail on **PUSCH HARQ acknowledgments** in **CBG mode**, refer to clause 6.1.5.2 of TS 38.214.

For more detail on **PUSCH HARQ**, refer to clause 5.4.2 of TS 38.321 and clause 6.1 of TS 38.214.

#### 4.5.6.7.3 Managing HARQ-ACKs in complex scenarios

The **management** of **HARQ-ACKs** may become **quite complex** in several scenarios, such as: **PDSCH HARQ** in **TDD mode**, **Uplink grant**, **Multi-transmission point scheduling**, or **group-based scheduling**.

This happens, for example, when the **UE** is **scheduled** with **multiple PDSCH transmissions** in the **same slot** or **across multiple slots** (e.g., in slot bundling scenarios). In this case, the **UE** requires **some** assistance to determine **how many HARQ-ACK bits** to report and in **what order**.

The information **required** by the **UE** is conveyed by the **NR node** in **1** to **3 DCI message field**(**s**) "***Downlink assignment index***" (**DAI**). The following table shows, for **PDSCH** and for **PUSCH**, the **DCI message Formats** where the said **field** is **present**, what is

**assigned** by **each** of these **DCI Formats**, what is the **field**'s **bitlength** in **each** of these **DCI Formats** and what is the **field**'s **semantics** in **each case:**

| DCI message field "*Downlink assignment index*" | | | |
|---|---|---|---|
| **Assigns** | **DCI Format** | **Bitlength** | **Semantics** |
| **PUSCH Unicast** | Format 0_1 (6.9.3) | 1, 2, 4 | There may be up to **3 DAI fields**. **Each** of them indicates **how many HARQ-ACK bits** to include. |
| | | 0, 2, 4 | |
| | | 0 to 2 | |
| | Format 0_2 (6.9.4) | 1, 2 | |
| | | 0, 2 | |
| | | 0 to 2 | |
| **PDSCH Unicast** | Format 1_0 (6.9.5) | 2 | Tracks **Downlink assignments** for **HARQ-ACK feedback**. |
| | Format 1_1 (6.9.6) | 0, 2, 4, 6 | |
| **PDSCH Multi-TRP** | Format 1_2 (6.9.7) | 0, 1, 2, 4 | Tracks **Downlink assignments** across **multiple TRPs**. |
| **PDSCH Multicast** | Format 4_1 (6.9.9.2) | 2 | Tracks **HARQ-ACK feedback** for **multiple UEs**. |
| | Format 4_2 (6.9.9.3) | 0, 2 | |

The **DCI field** "***Downlink assignment index***" may be used an **index** to the corresponding **HARQ-ACK codebook**. For more detail on **HARQ-ACK codebooks**, refer to clause 9.1 of TS 38.213.

For more detail on **PUSCH HARQ**, refer to clause 5.4.2 of TS 38.321 and clause 6.1 of TS 38.214.

## 4.5.7 High and Low physical functional blocks

The **specific** functions performed by the **High physical functional block** and by the **Low physical functional block** at the **NR node** depend on the **specific 7.x split option** (click 4.5.7.2) that is taken, among the **3 possible ones**: **7.1**, **7.2** and **7.3**.

For the **NR node functional blocks**, the **NR node physical functions**, the **payload processing** of the **physical binary channels**, the **protocol architecture** of the **New Radio** and the **data flow** through the **protocol architecture**, click 4.3.

Source web pages:

- https://www.researchgate.net/publication/331947823_Functional_Split_Perspectives_A_Disruptive_Approach_to_RAN_Performance_Improvement
- https://www.5gtechnologyworld.com/functional-splits-the-foundation-of-an-open-5g-ran/
- https://destevez.net/2022/02/demodulation-of-the-lte-uplink/

### 4.5.7.1 Characteristics of the high and low physical functional blocks

Irrespective of the **7.x split option** (click 4.5.7.2) that is taken, there are some **characteristics** of the **High** and **Low Physical functional blocks** that are now presented.

Some characteristics of the **High Physical functional block** are:

- It is usually implemented as **hardware** and **software**.
- It manages **data** in the **digital domain**, and in **splits 7.1** or **7.2 also baseband signals** in the **frequency domain**

- It interfaces to the **Low MAC functional block** (click 4.5.3.1) and to the **Low Physical** (click 4.5.3.2) **functional block**.
- It belongs to the **Distributed Unit** (**DU**, click 4.7.2).

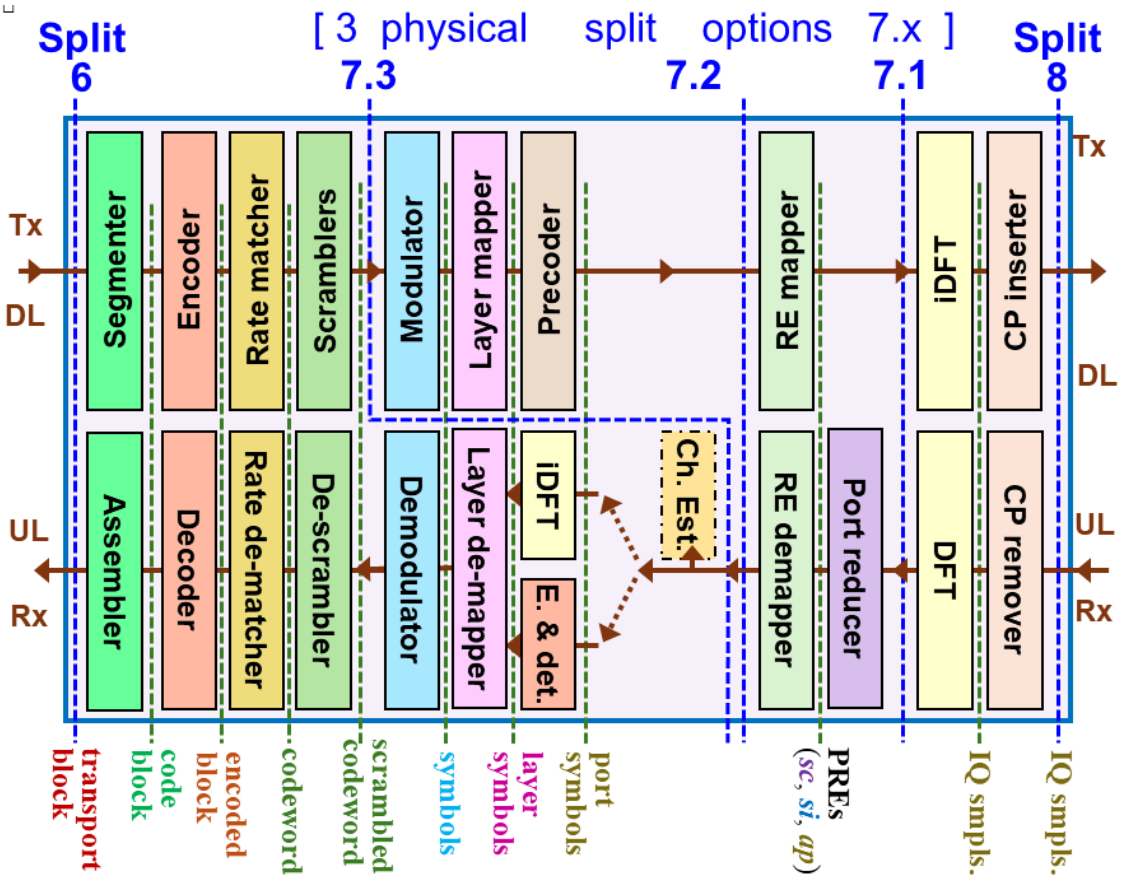
Some characteristics of the **Low Physical functional block** are:

- It is usually implemented as **hardware** and **software**.
- It manages **baseband signals both** in the **frequency domain** and in the **time domain**. In the (**quite frequent**) case that the **Radio Frequency** (**RF**) **functional block** and the **Low Physical functional block** are in **different locations**, it also manages **baseband signals** in the **digital domain**.
- It interfaces to the **High Physical functional block** and to the **RF** (click 4.5.8) **functional block**.
- It can belong **either** to the **Distributed Unit** (**DU**, click 4.7.2) or to the **Radio Unit** (**RU**, click 4.7.3).

For the **NR node functional blocks**, the **NR node physical functions**, the **payload processing** of the **physical binary channels**, the **protocol architecture** of the **New Radio** and the **data flow** through the **protocol architecture**, click 4.3.

### 4.5.7.2 Functional splits 7.1, 7.2 and 7.3

The following figure shows the **different functions** of the **High** and **Low Physical functional blocks** as well as the **3 mutually exclusive split options 7.1**, **7.2** and **7.3**:



Some **relevant remarks** on this figure are:

- The **Downlink** (**Tx**) **data flow** is **left to right**, while the **Uplink** (**Rx**) **data flow** is **right to left**.
- In a given **NR implementation**, the **selection** in **each case** of **one** (or none) **7.x split** (among the **3 possible** ones) determines **which functions** are contained **within** the **Low Physical functional block** and the **High Physical functional block**.
- **Split 7.3 only** affects the **Downlink** and its **Uplink** is the same as in **split 7.2**.
- When **two aligned Tx** and **Rx functions** have the **same color**, this indicates that they perform **reciprocal operations**.
- The **Tx** "**Precoder**" function is **optional** and is **only** performed if **Downlink space-division MIMO** and/or **Downlink space diversity** is used.
- The **Rx** "**E. & det.**" is the **equalization** and **optional detection**. **Detection** is only performed if **precoding** is used in **Uplink transmission**.
- The **Rx** "**Channel estimator**" function is **not** part of the **data processing path**, although it processes **received signals** in order to perform **channel estimation**.
- The **Rx** functions "**iDFT**" and "**Equalization and detection**" are shown in **parallel** because **Uplink SC-FDMA** and **space-division MIMO** are **mutually exclusive**. Consequently, **only one** of both functions is **used** at any **given time**. Also, **both functions** are **optional**, so it is possible that **none** of them is **used**.

For an **overview** of the different **functions** within the **Physical high** and the **Physical low functional blocks**, click 4.5.7.

The **split 7.3** is applied **Downlink** and **not Uplink** because in **Uplink** it would require **excessive processing load**. Notice that in **Uplink** the **NR node** handles the **signals** from **multiple UEs**, **each** requiring an **(i)DFT**, **channel estimation**, possibly combining signals from **multiple array antennas** (**MIMO**), etc. This implies a **very significant processing load**. If this were done at the **RU**, it would **significantly increase** its **complexity** (and **cost**) and would **deviate** from the intended simplicity of **split 7.3**.

For the **NR node functional blocks**, the **NR node physical functions**, the **payload processing** of the **physical binary channels**, the **protocol architecture** of the **New Radio** and the **data flow** through the **protocol architecture**, click 4.3.

#### 4.5.7.3 Functions within the high and low physical functional blocks

The **different functions** of the **High** and **Low Physical functional blocks** of the **NR node** are:

- **Segmenter/Assembler**

  In **Tx** it **implements** the **Downlink attachment** of a **CRC** to **1** or **2 transport block(s)** to produce **1** or **2 protected block(s)** (**PDSCH Tx stage 1**, click 4.12.3) and the **segmentation** of the **1** or **2 protected block(s)** into a **1** or **2 code block sequence(s)** (**PDSCH Tx stage 2**, click 4.12.4).

  In **Rx** it **implements** the **Uplink assembly** of **1 code block sequence** into **1 protected block** and the **verification** and **removal** of the **CRC** to produce **1 transport block** (**PUSCH Rx stage 10**).

- **Encoder/Decoder**

  In **Tx** it **implements** the **Downlink encoding** of **each code block into** an **encoded block** (**PDSCH Tx stage 3**, click 4.12.5).

  In **Rx** it **implements** the **Uplink decoding** of **each encoded block into** a **code block** (**PUSCH Rx stage 10**).

- **Rate (de)matcher**

  In **Tx** it **implements** the **Downlink rate-matching** of **each encoded block** into a **rate-matched block** (**PDSCH Tx stage 4**, click 4.12.6) and the concatenation of **1** or **2 rate-matched block sequence(s)** into **1** or **2 codeword(s)** (**PDSCH Tx stage 5**, click 4.12.7).

  In **Rx** it **implements** the **Uplink segmentation** of **1 codeword** into **1 rate-matched block sequence** (**PUSCH Rx stage 8**) and the **rate de-matching** of **each rate-matched block** into an **encoded block** (**PUSCH Rx stage 9**).

- **(De)Scrambler**

  In **Tx** it **implements** the **Downlink scrambling** of **1** or **2 codeword(s)** into **1** or **2 scrambled codeword(s)** (**PDSCH Tx stage 7**, click  and 4.12.9).

  In **Rx** it **implements** the **Uplink de-scrambling** of **1 scrambled codeword** into **1 codeword** (**PUSCH Rx stage 6**). This **Uplink Rx function** might also include the **PUSCH/PUCCH codeword de-multiplexing** (**PUSCH Rx stage 7**).

- **(De)modulator**

  In **Tx** it **implements** the **Downlink modulation** of **1** or **2 scrambled codeword(s)** into **1** or **2 modulated sequence(s)** (**PDSCH Tx stage 8**, click 4.12.10).

  In **Rx** it **implements** the **Uplink de-modulation** of **1 modulated sequence** into **1 scrambled codeword** (**PUSCH Rx stage 5**).

  For a **summary** of **modulations** used in different **physical binary channels** and **signals**, click 4.11.3.

- **Layer (de)mapper**

  In **Tx** it **implements** the **Downlink mapping** of the **1** or **2 modulated sequence(s)** to "$N_{layer}$" **layer sequence(s)** (**PDSCH Tx stage 9**, click 4.12.11).

  In **Rx** it **implements** the **Uplink de-mapping** from "$N_{layer}$" **layer sequence(s)** to **1 modulated sequence** (**PUSCH Rx stage 4**).

- **Tx precoder (optional)**

  It **implements** the **PDSCH Tx precoding** (**space-division MIMO**) of the "$N_{layer}$" **layer sequence(s)** into "$N_{port}$" **antenna port sequence(s)** (**PDSCH Tx stage 10**, click 4.12.13), **only** in case that **Tx PDSCH space-division MIMO** is used.

  It also may implement **space diversity** (click 11.8).

  The **location** of the **precoder** within the successive processing steps is **implementation dependent**.

- **Rx channel estimator**

  It **implements** the **Uplink Rx channel estimation** (click 2.4.5) for **each spatial channel**, based on the **received reference signals**. Notice that this **function** is **not** in the **data processing path**, and it **rather** belongs to the **physical layer control**.

- **Rx iDFT (optional)**

  It **implements** the **Uplink Rx inverse Digital Fourier Transform** in case that **transform-precoding** (click 11.16) is used in **Uplink transmission**.

  **Rx iDFT** and **Rx detection** are **both optional** and **mutually exclusive**.

- **Rx Equalization and detection (optional)**

  It **implements** the **Uplink Rx signal equalization**. It can also implement **PUSCH Rx detection** in case that **space-division MIMO** and/or **space diversity** (click 11.8) are used in **PUSCH transmission**.

  **Rx iDFT and Rx detection** are **both optional** and **mutually exclusive**.

- **RE (de)mapper**

  In **Tx** it **implements** the **Downlink mapping** of **port sequences** to **VREs** (**PDSCH Tx stage 12**, click 4.12.14) and from **VREs** to **PREs** (**PDSCH Tx stage 13**, click 4.12.15).

  In **Rx** it **implements** the **Uplink mapping** of **PREs** to **VREs** (**PUSCH Rx stage 1**) and from **VREs** to **port sequences** (**PUSCH Rx stage 2**).

- **Tx digital beamformer (optional)**

  In **Rx** it implements the **digital beamforming** (click 2.4.1.2) processing.

- **Rx port reducer**

  It implements the **Rx port reduction** (click 4.5.7.4) processing.

- **(inverse) Digital Fourier Transformer**

  In **Tx** it **implements** the **inverse DFT** (click 11.19) that converts the **Downlink signal representation** from the **frequency domain** to the **time domain**. It is adjusted according to the **PDSCH numerology** of the **Downlink currently active bandwidth part** (click 2.2.7) that corresponds to the **Downlink carrier** (click 2.2.5). The **output signal** is represented as a **series** of **IQ samples** (a **digitized time-domain signal**).

  In **Rx** it **implements** the **DFT** that converts the **Uplink signal representation** from the **time domain** to the **frequency domain**. It is adjusted according to the **PUSCH numerology** of the **Uplink currently active bandwidth part**, that corresponds to the **Uplink carrier**.

  For more detail on the **(i)DFT transformer** click 4.5.7.5.

- **Cyclic Prefix inserter/remover**

  In **Tx** it **inserts** the **cyclic prefix** (click 2.3.2, 7.3.1 and 11.13) right before **each Downlink symbol**, within the **symbol interval**.

  In **Rx** it **removes** the **cyclic prefix** that is **received** right before **each Uplink symbol**.

It operates in the **digitized time domain** (over a **series** of **IQ samples**).

#### 4.5.7.4 Port reduction

**Port reduction** consists of several techniques to **reduce** the **quantity** of **antenna ports** (click 4.11.5) that are **required** for **signaling** in **MIMO** systems. **Port reduction** is therefore extremely important for **5G massive MIMO** (arrays with 64 or more antennas), for **beamforming** and for **dynamic TDD**.

**Massive MIMO**, **beamforming** and **dynamic TDD** require **multiple Tx** and **Rx antenna ports**. **Each antenna port** typically requires its own **specific signaling**, such as **Demodulation Reference Signals** (**DM-RS**), for **channel estimation**. **Without port reduction**, the **required quantity** of **antenna ports** in **large antenna arrays** can lead to an **extremely large signaling overhead**. This is **largely mitigated** with the following functions of **port reduction**:

- **Combining ports**

  It consists of **grouping** or **combining antenna ports** to reduce the **quantity** of **signaling resources** (e.g., **reference signals**) needed to represent **multiple physical antennas**. This is achieved through **precoding techniques** that m**ap signals** from a reduced **quantity** of **virtual antenna ports** to a larger **quantity** of **physical antennas**.

- **Simplifying Channel Estimation**

  Instead of transmitting a **specific reference signal** for **each** and **every antenna port**, the **simplified channel estimation** can achieve a **sufficiently accurate** representation of the **channel characteristics** with **significantly fewer reference signals**. This reduces the complexity of channel estimation and feedback while still enabling efficient beamforming.

- **Dynamic Resource Allocation**

  It consists of the **dynamic assignment** of **antenna ports** based on **transmission requirements**, such as the **quantity** of **active users**, **channel conditions**, and **spatial degrees** of **freedom** needed for **beamforming**.

The **key benefits** of **port reduction techniques** are:

- **Reduced Overhead**

  **Reducing** the **quantity** of **transmitted reference signals minimizes signaling overhead** and **improves spectral efficiency**.

- **Efficient Scaling:**

  **Port reduction** enables the practical implementation of **massive MIMO** by limiting the **resource cost** of managing **large antenna arrays**.

- **Energy Efficiency:**

  **Fewer antenna ports** and **reduced signaling** translate into **lower power consumption**, an increasingly important element in mobile networks.

- **Maintained Performance**

  Despite **reducing** the **quantity** of **antenna ports**, **advanced precoding** ensures that **performance** (e.g., **beamforming gain**, **channel estimation accuracy**) is **preserved**.

#### 4.5.7.5 Discrete Fourier Transformer (DFT)

The **Discrete Fourier Transformer** (**DFT**) computes **both** the **inverse Discrete Fourier Transform** (**iDFT**, click 11.19) and the **Discrete Fourier Transform** (**DFT**).

The **iDFT** transforms a **complex-valued baseband signal** from the **frequency domain into** the **time domain** (in **both** cases with **digital representation**). The **iDFT** is **performed** in **transmission**.

The **DFT** transforms a **complex-valued baseband signal** from the **time domain** with **digital representation** into the **frequency domain** (in **both** cases with **digital representation**). The **DFT** is performed in **reception**.

When **DFT-s-OFDM** (click 11.16) is used, an **additional DFT** is performed by the **UE** in **Uplink transmission** is used. In this case, the **NR node** applies an **iDFT** in **Uplink reception**. The reason is that **5G Uplink** supports **either OFDMA** (click 11.15) or **DFT-s-OFDM** (click 11.16). **DFT-s-OFDM** requires applying, **transform-precoding**, click 4.12.12).

Notice that in the **NR node** there is **no optional transform-precoding** (i.e., a **DFT function**) in **Downlink Tx direction** because the **NR node only** uses **OFDM**.

The (**i**)**DFT size** "$N$" has to be **larger** (click 11.19.4.2) than **twice** the **product** of the **maximum frequency** "$F_{max}$" of the **baseband signal multiplied** by the **total sampled time** "$T_{to}$" (click 11.19.2).

$$N > 2F_{max} \bullet T_{to}$$

The **maximum frequency** "$F_{max}$" of the **baseband signal** is **half** of the **channel transmission bandwidth**. The **channel transmission bandwidth** is the **quantity** of **transmission resource blocks** "$N_{RB,\mu}$" (click 2.2.3.6.1), for the corresponding **channel bandwidth** (click 2.2.3.6) and **numerology** "**μ**", multiplied by the **bandwidth** of **1 resource block**:$2F_{max} = N_{RB,\mu} \bullet 12 \bullet \mathbf{SCS}_{\mu}$ Hz

$$2F_{max} = N_{RB,\mu} \bullet 12 \bullet 2^{\mu} \bullet 15 \bullet 10^3 \text{ Hz}$$

The **sampled time** is the **duration** of **1 symbol** (click 2.3.2) which **depends** on the **numerology** "**μ**". The **duration** of a **1 symbol** is approximately:

$$T_{to} = T_{symb} = 71 \bullet 10^{-6} \bullet 2^{-\mu} \text{ sg}$$

Consequently:

$$N > 2F_{max} \bullet T_{to}$$

$$N > (N_{RB,\mu} \bullet 12 \bullet 2^{\mu} \bullet 15 \bullet 10^3) \bullet (71 \bullet 10^{-6} \bullet 2^{-\mu})$$

$$N > 12.78 \bullet N_{RB,\mu}$$

The **DFT size** "$N$" is typically a **power** of **2**.The **actual DFT size** used is **implementation dependent**. **Each vendor usually** configures the **minimum** required **DFT size** depending on "$N_{RB}$" and other factors (band, oversampling requirements, etc) in order to **save computing power** and **reduce consumption**.

For example, a **20 MHz channel** in a **FR1 band** with "**μ**=2" has a **maximum transmission bandwidth** "$N_{RB}$=24" (refer to table 5.3.2-1 of TS 38.101-1). Consequently:

$$N > 12.78 \bullet N_{RB,\mu}$$

$$N > 306.7$$

For this example, "***N*=512**" could be used.

The **minimum value** of "***N***" that can is required by **any 5G case** can be computed using the **maximum** "$N_{RB}$" value in **5G**. The **channel transmission bandwidth** "$N_{RB}$" for each **pair** of values of **channel bandwidth** and **channel**'s **control/shared subcarrier spacing** is indicated in tables 5.3.2-1 of TS 38.101-**1** (for **FR1 terrestrial**), of TS 38.101-**2** (for **FR2**) and of TS 38.101-**5** (for **FR1 satellite**). Scanning through these tables we get:

| FR | $max(N_{RB})$ | $min(N)$ |
|---|---|---|
| **FR1** | 273 | 3,489 |
| **FR2** | 264 | 3,374 |

This shows that a **DFT size** "***N*=4,096**" is enough to cover **every 5G cannel configuration**.

### 4.5.7.6 Channel Estimator

It computes the **Channel State Information** (**CSI**, click 2.4.5) based on the received **Channel State Information Reference Signals** (**CSI-RS**, click 6.20) and other signals. The **Channel State Information** is reported by the **UE to** the **NR node** in a **UCI message** (click 7.7). The **Channel State Information** sent by the **UE** may contain the following **reports**:

- **PMI**: Precoding Matrix Indicator.
- **RI**: Rank Indicator.
- **LI**: Layer Indicator.
- **CQI**: Channel Quality Indicator.
- **CRI**: **CSI-RS** Resource Indicator, where "**CSI-RS**" stands for "Channel State Information - Reference Signal".
- **SSBRI**: **SSB** Resource Indicator. It indicates which **SSB instance** (click 6.1.2.2.3) has been **received** by the **UE** with the **best signal quality**.
- **RSRP**: Reference Signal Received Power. It conveys the received power level of a specific reference signal (**CSI-RS** or **SSB**).
- **PTI**: Precoding Type Indicator.

### 4.5.7.7 Equalizer

In **OFDM(A)**, as well as in **SC-FDMA**, **equalization** is achieved on the **receiver side**, after the **DFT**, by **multiplying each Fourier coefficient** (i.e., **subcarrier**) by a **complex number**. Thus, **frequency-selective fading** and **phase distortion** can be mitigated.

**Frequency-domain equalization** using **fast Fourier transforms** requires **less computation** than conventional **time-domain equalization**, which requires **multi-tap FIR** or **IIR-filters**. **Less computation reduces power consumption** less power and also **reduces numerical noise** (**lower compounded round-off error**).

Source web page:

- https://en.wikipedia.org/wiki/Single-carrier_FDMA

### 4.5.8 Radio Frequency (RF) functional block

In **Tx**, the **RF functional block** transforms **one**, or **several**, **baseband signal(s)** in the **time domain** with **digital representation** into **one**, or **several**, **high-power passband signal(s)** in the **time domain** with analog **representation**.

In **Rx**, the **RF functional block** transforms **one**, or **several**, **very-low power passband signal(s)** in the **time domain** with **analog representation** into **one**, or **several**, **baseband signal(s)** in the **time domain** with **digital representation**.

Some books consider that the **antennas** are **not** part of the **RF functional block**. However, in this book it is considered that the **antennas** are part of the **RF functional block**, **both** for **completeness'** sake and because the **antennas** are conceptually part of the **Layer 1 functions**. However, we do make a **distinction** between the **RF chains** (click 4.5.8.1) and the **antennas** (click 4.5.8.5).

For **Tx**, the **RF functional block** contains **1 or more Tx RF chains** and **1 or more antennas**.

For **Rx**, the **RF functional block** contains **1 or more Rx RF chains** and **1 or more antennas**.

The **Tx RF chain** and the **Rx RF chain** may **share** a **duplexing subsystem** (click 4.5.8.8).

Some characteristics of the **RF functional block** are:

- It manages **both baseband** and **passband signals** in the **time domain**, both with **analog representation**. In the (**quite frequent**) case that the **RF functional block** and the **Low Physical functional block** are in **different locations**, it also manages **baseband signals** in the **digital domain**.
- It interfaces to the **Low Physical functional block** (click 4.5.5) and to the **air** (physical medium).
- It belongs to the **Radio Unit** (**RU**, click 4.7.3).

An **RF functional block implementation** consists of a **combination** of **different RF components** and **RF subsystems**.

**Each RF component** is a **passive** or **active hardware device**. **Each RF component** is characterized (among other traits) by the **frequency segment(s)** in which it can operate. **Each RF component** can **only** be used for **channels within** its **frequency segment**. Whenever a **new channel** is configured (e.g., due to a frequency refarming by the national authorities), in case it is **outside** of the **frequency segment** of a given **RF component**, that **RF component** has to be **physically replaced** by another one. When an **operator** faces this problem, it **may** attempt to sell the corresponding **RF components** to the **new licensee** of its **current channel**.

It is therefore desirable that **RF components** have as **many frequency segments** as possible, **each** of them as **wide as possible**. This **increases flexibility**, but it obviously comes at a **higher cost**. A **balance** between **flexibility** and **cost** is **carefully** analyzed by **each operator** company for **each acquisition** of **RF components** and **subsystems**.

The **most frequent RF components** and **subsystems** are described in the **following subsections**.

#### 4.5.8.1 RF chain

**Each Tx RF chain** is composed of (input to output) **possibly 1 RF digital preprocessor** (click 4.5.8.14), **1 D/A converter** (click 4.5.8.13), **1 upconverter** (click 4.5.8.12), **1 Tx channel filter** (click 4.5.8.8) and **1 power amplifier** (click 4.5.8.10).

**Each Rx RF chain** is composed of (input to output) **1 low-noise amplifier** (click 4.5.8.11), **1 Rx channel filter** (click 4.5.8.8), **1 downconverter** (click 4.5.8.12), **1 A/D converter** (click 4.5.8.13) and **possibly 1 RF digital postprocessor** (click 4.5.8.13).

In case a **Tx RF chain** performs **Tx analog beamforming** (click 2.4.1.1), it also has **1 Tx analog beamformer** (click 4.5.8.7) **after** the **Tx channel filter**. **After** the **Tx analog beamformer**, the **Tx RF chain** has **as many power amplifiers** as **streams** are produced by the **Tx analog beamformer**. The **Tx analog beamforming** allows the **NR node** to perform **Tx beam sweeping** (click 6.1.2.3) and the **UE** to perform **Rx beam selection**.

In case a **Rx RF chain** performs **Rx analog beamforming**, it also has **1 Rx analog beamformer before** the **Rx channel filter**. **Before** the **Rx analog beamformer**, the **Rx RF chain** has **as many low-noise amplifiers** as **streams** are managed by the **Rx analog beamformer**. The **Rx analog beamforming** allows the **NR node** to perform **Rx beam sweeping** (click 2.4.1.5), **even** if the **UE** does **not** perform **Tx analog beamforming**.

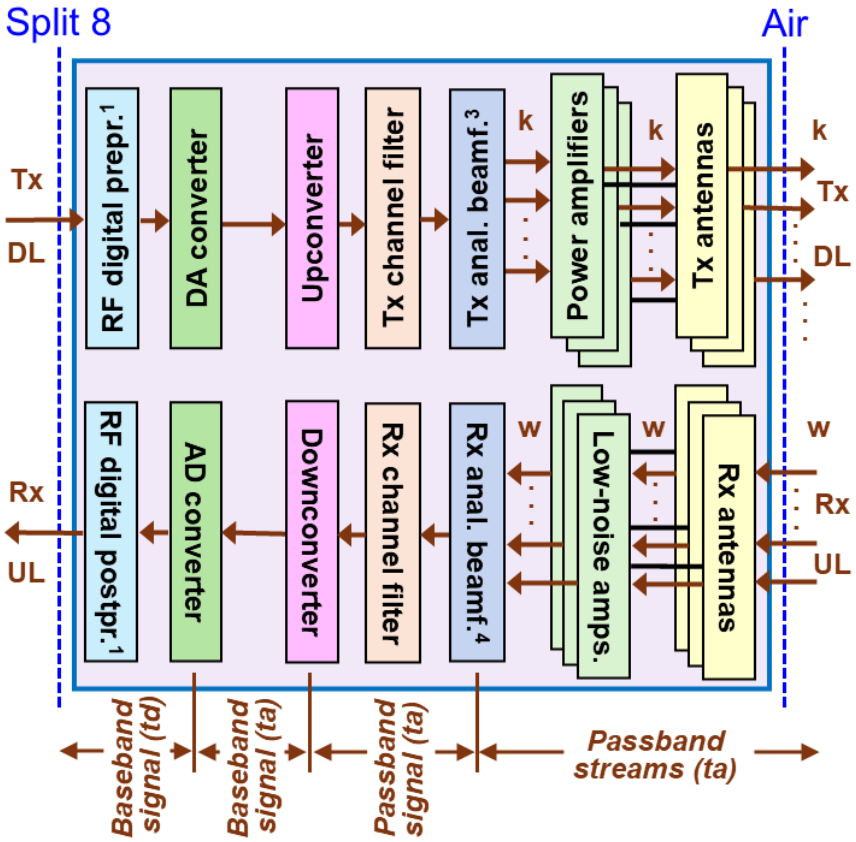
**One Tx RF chain** and its corresponding **Rx RF chain** form a **RF chain pair**.

The connection between the **RF chains** and the **antennas** may be done in **either** of the **two** following ways:

- **Each Tx** or **Rx RF chain** is connected to **1 Tx** or **Rx antenna** with **1 direct** and **exclusive RF cable** (respectively).
- **Each RF chain pair** is connected to its **antenna**(**s**) by means of **duplexer**(**s**) (click 4.5.8.8).

**RF chain pair with analog beamforming**

The following figure shows **one RF chain pair** with **analog beamforming** and connected to its **antennas** by means of **direct RF cable**s:



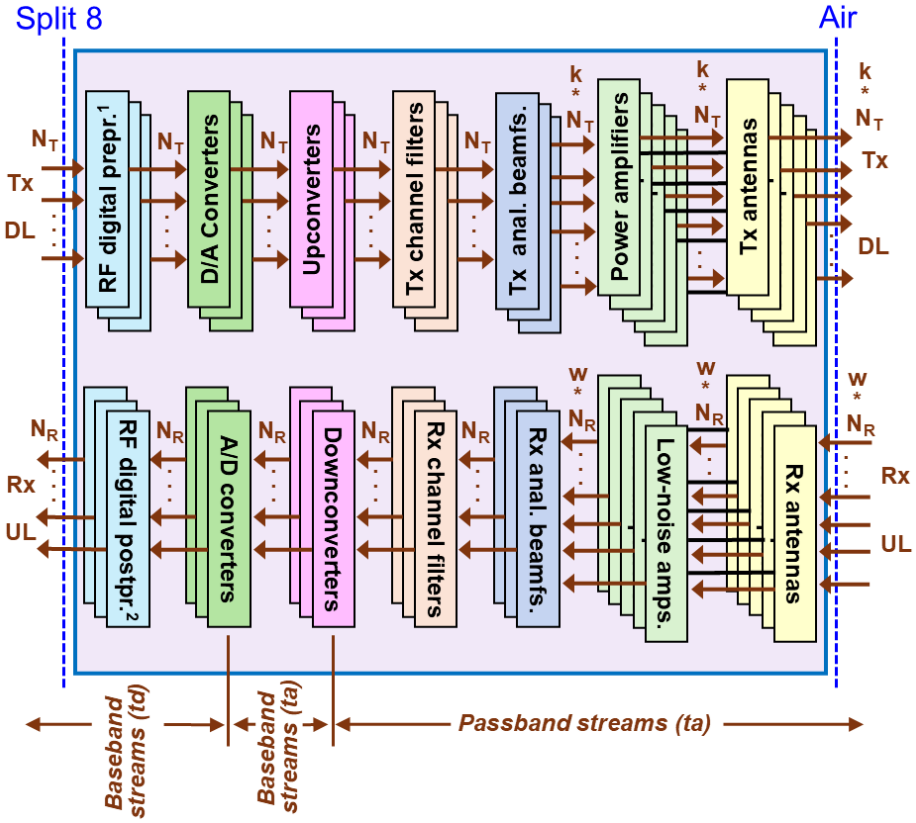

Some relevant remarks on this figure are:

- **Each box** represents an **RF component**. For their descriptions see the subsections further below.
- **Each black line** represents **1 RF cable**.
- **Tx analog beamforming** implies that "**k**" **weighted** and **phased versions** of the **single input Tx signal** are radiated to the **physical media** (i.e., air or vacuum). This is indicated in the figure as "**k**" **Tx arrows** (i.e., "**k**" **Tx passband streams** (in **time-domain analog** "***ta***" representation). The result from **Tx analog beamforming** is **only one transmitted signal** with a **main lobe** in the **desired direction**.
- **Each** of the "**k**" **weighted** and **phased versions** of the **single input Tx signal** requires **1 exclusive power amplifier** and **1 exclusive Tx antenna**.
- **Rx analog beamforming** implies that "**w**" **Rx signals** are **weighted** and **phased** to achieve the corresponding **gain** in the **desired direction**. This is indicated in the figure as "**w**" **Rx arrows** (i.e., "**w**" **Rx passband streams** in **time-domain analog** "***ta***" representation). The result from **Rx analog beamforming** is **only one received signal** with a **main lobe** in the **desired direction**.
- **Each** of the "**w**" **Rx** signals for **Rx analog beamforming** requires **1 exclusive Rx antenna** and **1 exclusive low-noise amplifier**.

- For more detail on the **labels** in the figure, click 4.8.

**Several RF chain pairs with analog beamforming**

The following figure shows "$N_T$" **Tx RF chains** and "$N_R$" **Rx chains**, **both** with **analog beamforming**, and connected to their **antennas** by means of **direct RF cable**s:

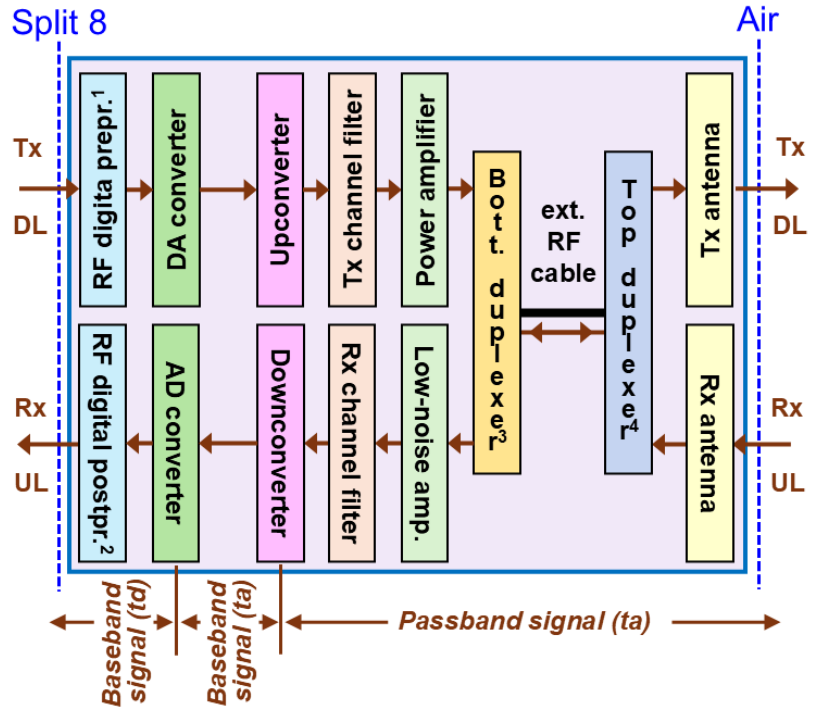



Some relevant remarks on this figure are:

- **Each box** represents an **RF component**. For their descriptions see the subsections further below.
- The **black lines** represent **RF cables**, that can be **external** or **internal**.
- The "$N_T$" **Tx streams** coming from the **Low Physical functional block** (click 4.5.5) are originated by **Tx space multiplexing** (click 11.6.3) and/or by **Tx digital beamforming** (click 2.4.1.2) and/or by **Tx space diversity** (click 11.8).
- The "$N_R$" **Rx streams** going the **Low Physical functional block** (click 4.5.5) are destined to **Rx space multiplexing** (click 11.6.3) and/or to **Tx digital beamforming** (click 2.4.1.2) and/or to **Rx diversity combining** (click 11.8)
- For more detail on the **labels** in the figure, click 4.8.

**One RF chain pair without analog beamforming and connected with duplexers**

The following figure shows **1 RF chain pair**, **without analog beamforming**, and connected to their **antennas** by means of **duplexers**:




Some **relevant remarks** on this figure are:

- **Each box** represents an **RF component**. For their descriptions see the subsections further below.
- The **black line** represents **one shared RF cable**, which is most frequently **external**.
- Since there is **no analog beamforming**, there is **only one Tx** and **one Rx signal** exchanged with the **physical media** (i.e., the air).
- This example is for a **Tx/Rx antenna pair** (click 4.5.8.5), and this is why a **top duplexer[4]** (click 4.5.8.8) is included in addition to the **bottom duplexer[3]**.
- For more detail on the **labels** in the figure, click 4.8.

### 4.5.8.2 RF cable

**Each RF cable propagates** one **passband signal** between its **two endpoints**. It **propagates** equally in **both directions**, but in some cases it is used to **propagate** in **only one direction**.

**Each RF cable** can be characterized by its **length**, its **frequency segment**, its **maximum supported power** and its **power losses**.

**Signal power decreases** as it travels **along** an RF cable, so the **maximum length** of an **RF cable** is usually limited to a few **tens** of **meters**.

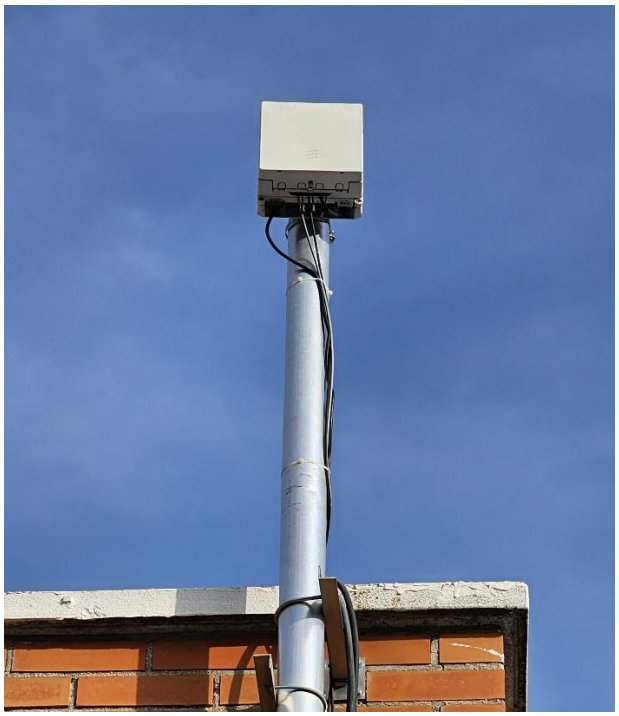
**RF cables** can be **internal** to an **RF component** (e.g., to connect the **antenna elements within** an **antenna**), or **external** (e.g., to connect an **RF chain** with an **antenna**).

In **each subsection** that describes an **RF component**, when listing its possible interconnections with **other RF components**, we will **omit internal RF cables** and will only list (when applicable) **external RF cables**. The reason is that an **internal RF cable** by itself is **not** usually considered by itself an **RF component**, but **rather** an internal part of an actual **RF component**.

Some characteristics of **each RF cable** are:

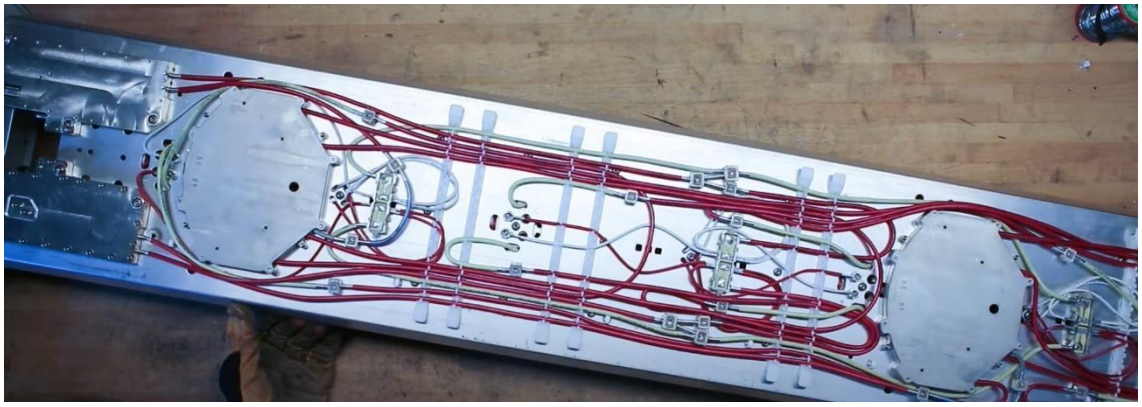
- It is a **passive hardware device**.
- It manages **1 passband signal** in the **time domain** with **analog representation**.
- An **external RF cable** its connected on **one end either** to an **RF chain** (click 4.5.8.1) **or** to a **bottom duplexer** (click 4.5.8.8) and on the **other end**, **either** to an **antenna** (click 4.5.8.5), to an **antenna array** (click 4.5.8.6), or to a **top duplexer** (click 4.5.8.8).
- An **external RF cable** is **not always** required.

The following picture shows the **external RF cables** (in black color) connecting the **RF chains** located inside the **equipment shelter** to the **antenna** at the **top** of the **mast**.



This picture is from the 5TONIC laboratory, at IMDEA Networks and U. Carlos III, Madrid (Spain).

The following picture shows the **internal RF cables** (in pink color) connecting the **antenna elements** within an **antenna**.



### 4.5.8.3 Antenna element

**Each antenna element** is an **individual radio frequency device** that **radiates** (**Tx**) **one radio signal** to the **air** (or **vacuum**) and/or **receives** (**Rx**) **one passband radio signal** from the **air** (or **vacuum**).

**Each antenna element** can be characterized by its **transmission direction(s)** (**Tx**, **Rx** or **Tx/Rx**), by the **frequency segment** in which it can operate, by its **horizontal** (i.e., the **top view**) **radiation pattern**, by its **vertical** (i.e., the **side view**) **radiation pattern**, by its **polarization** and by its **location** and **orientation** within the **antenna** (click 4.5.8.5).

Some characteristics of **each antenna element** are:

- It is a **passive hardware device**.
- It manages **1 passband signal** in the **time domain** with **analog representation**.
- It interfaces on its **one end** to the **air** (or **vacuum**) and on its **other end either** with the **antenna RF connector** or with a **combiner** (click 4.5.8.4).
- At least **1 antenna element** is **always** required for **each cell** (click 3.1.1).

An **antenna** may contain **several antenna elements**. **All** the **antenna elements** of a **given antenna radiate/receive exactly** the **same radio signal**.

At least **1 antenna element** is **always required** in **every antenna**. **Several antenna elements** are **required** in an **antenna** when a **specific radiation pattern** is **desired**.

**All** the **antenna elements** are **connected** through an **RF** distribution system inside the **antenna**, (e.g., several **RF cables**, click 4.5.8.2) to the **RF connector** of the **antenna**.

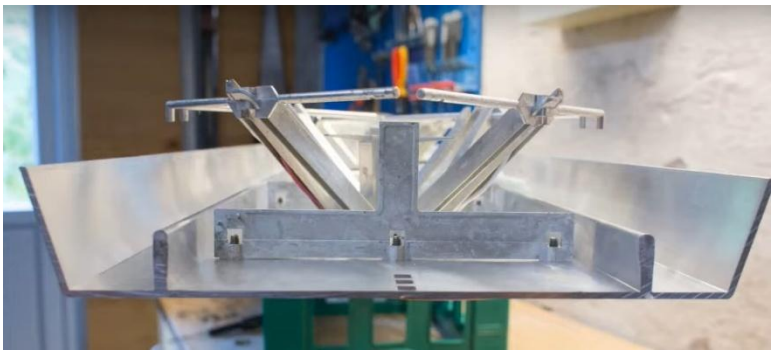
The **physical implementation** of an **antenna element** may be a **dipole** (click 11.21.8), a **PCB mounted component** or other formats.

The following **three** pictures show **three views** of a set of **4 antenna elements**, for **900 MHz**, where **each** antenna element is a **dipole** (click 11.21.8).

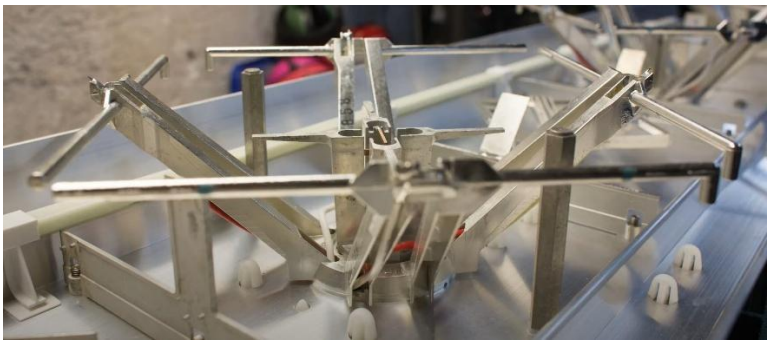
**Front** view of a set of **4 antenna elements**:



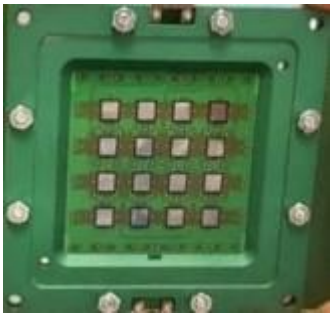
**Side** view of a set of **4 antenna elements**:



**Perspective** view of a set of **4 antenna elements**:



The following picture shows **16 antenna elements**, at **mmWave**, **each** of which is a **PCB** mounted **component**:



For an **introduction** to **antenna theory**, click 11.21.

Source web pages:

- https://www.nxp.com/company/about-nxp/smarter-world-blog/BL-DESIGNING-A-5G-MMWAVE

### 4.5.8.4 Combiner

**Each combiner transforms multiple input signals** into **one output signal**. It is **almost always** used in **reception**.

**Each combiner** can be characterized by its **frequency segment(s)**.

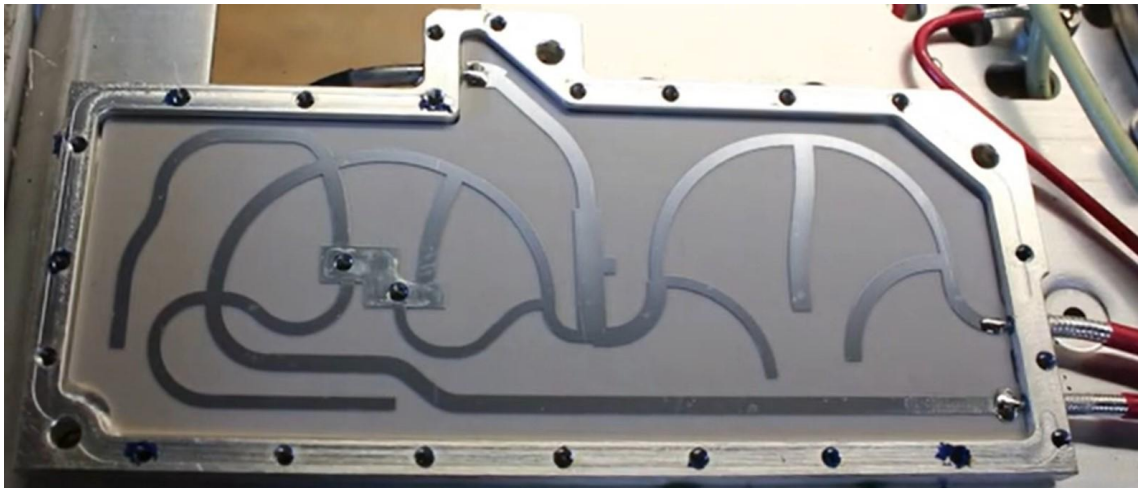
Some characteristics of **each combiner** are:

- It is a **passive** or **active hardware device**.
- It manages several **passband signals** in the **time domain** with **analog representation**.
- Its **input signals** (**Rx**) come **either** from **several antenna elements** (click 4.5.8.3) or from **several antennas** (click 4.5.8.5) and its **output signal** goes **either** to an **external RF cable** (click 4.5.8.2), to a **top duplexer** (click 4.5.8.8) or to a **power amplifier** (click 4.5.8.10).
- A **combiner** is **not always** required.

**Combiners** may be (among others) **antenna combiners** or **diversity combiners**.

**Antenna combiner**

An **antenna combiner** takes as **input** the **signals** from **different internal antenna components** and **outputs one signal**. The following picture shows a **900 MHz** and **1,800 MHz antenna combiner**:



Some **relevant remarks** on this picture are:

- This **combiner** takes as **inputs one 900 MHz signal** (from the **900 MHz antenna elements**) and **one 1,800 MHz signal** (from the **1,800 MHz antenna elements**) and its **output** is a **single combined signal**.
- The **two inputs** are the **two red RF cables** at the **bottom-right corner** of the **combiner**.

- The **output** is the **black RF cable** at the **top-middle** of the **combiner**.

**Diversity combiner**

A **diversity combiner** takes as **input** the **diversity signals received** from several (**partly**) **independent channels** and **outputs one Rx signal** with **improved reliability**. For more detail on **diversity combining**, click 11.8.

Source web pages:

- https://www.sannytelecom.com/what-is-an-antenna-combiner/

### 4.5.8.5 Antenna

**Each antenna radiates** (**Tx**) **one radio signal** to the **air** (or **vacuum**) and/or **receives** (**Rx**) **one passband radio signal** from the **air** (or **vacuum**).

**Each antenna** can be characterized by its **transmission direction(s)** (**Tx**, **Rx** or **Tx/Rx**), by the **frequency segment** in which it can operate, by its **maximum antenna gain**, by its **radiation pattern** (click 12.1.6) and by its **polarization**.

The **antenna radiation pattern** usually has **several secondary lobes** in addition to a **main lobe**. The **antenna radiation pattern** depends on **frequency**.

Some characteristics of **each antenna** are:

- It is a **passive hardware device**.
- It manages **1 passband signal** in the **time domain** with **analog representation**.
- For a **Tx/Rx antenna**, it interfaces on **one** of its **ends** with the **air** (or **vacuum**) and on its **other end either** with an **external RF cable** (click 4.5.8.2), or with a **combiner** (click 4.5.8.4).
- For a **Tx antenna**, it interfaces on its **one end** with the **air** (or **vacuum**) and its **other end either** with an **external RF cable**, or with a **top duplexer** (click 4.5.8.8), or with a **power amplifier** (click 4.5.8.10).
- For a **Rx antenna**, it interfaces on its **one end** with the **air** (or **vacuum**) and its **other end either** with an **external RF cable**, or with a **top duplexer**, or with a **low-noise amplifier** (click 4.5.8.11) or with a **combiner**.
- At least **1 antenna** is **always** required for **each cell** (click 3.1.1).

An **antenna** may be **standalone** or part of an **antenna array** (click 4.5.8.6).

**Standalone antenna**

**Each standalone antenna** is usually composed of **several antenna elements** (click 4.5.8.2), **all** of which **radiate** and/or **receive exactly** the **same passband radio signal**.

In **most** cases, the **same standalone antenna** acts **both** as **transmission** (**Tx**) and **reception** (**Rx**) **antenna** (a **Tx/Rx antenna**). In a **few** cases, **one standalone antenna** is required for **Tx** and a **different standalone antenna** is required for **Rx**, where **both antennas** jointly form a **single Tx/Rx standalone antenna pair**. One example of a **Tx/Rx standalone antenna pair** is when the **Tx antenna** and the **Rx antenna** work with **different polarizations** (e.g., in some **MIMO** systems).

**Either 1 Tx/Rx standalone antenna**, or **one Tx/Rx standalone antenna pair**, or **1 Tx/Rx antenna array** (click 4.5.8.6) is required for **each cell**. (i.e., for **each sector** in an

**antenna site**). For **SUL** (click 10.4) operation, an **additional Rx antenna** or **Rx antenna array** is **required** for the **cell**. For **SDL** (click 10.5) operation, an **additional Tx antenna** or **Tx antenna array** is **required** for the **cell**. At least **1 antenna** is **required** in **every NR node**.

The **external appearance** of **1 standalone antenna** is usually a **whitish rectangular box** connected to **1** or **2 RF cables** (click 4.5.8.7). The following picture shows an **antenna site** (click 3.1.3) with **4 standalone Tx/Rx antennas**, **each** connected to **2 RF cables**:



This picture is from Soweto, Johannesburg (South Africa).

**Array antenna**

**Each array antenna** is composed of **one** or a **few antenna elements** (click 4.5.8.2).

**Not all** the **array antennas** of an **antenna array** are used for **data**. **Some array antennas may** be used for **control signals**, such as **demodulation reference signals** (click 4.11.3.2) and/or for **physical signals** (click 4.11.3.2, 4.11.3.3).

**Antenna related concepts**

A **multiband antenna** (click 4.7.4) is **not** considered an **RF component**. It is **rather** considered an **RF subsystem**.

An **antenna port** (click 4.11.5) is **not** an **RF component**. It is **rather** an **identifier** of a **logical entity** in **space multiplexing** (click 11.6.3).

For an **introduction** to **antenna theory**, click 11.21.

### 4.5.8.6 Antenna array

**Each antenna array** is a **set** of **antennas** (click 4.5.8.5), with a **very precise physical disposition**, where **each Tx antenna radiates** a **different** (**even** if only **slightly**) **signal** and **each Rx antenna outputs** a different (**even** if only **slightly**) **signal**.

**Each antenna array** can be characterized by the characteristics of its **antennas**, by the **quantity** of **Tx antennas**, by its **quantity** of **Rx antennas**, by its **quantity** of **Tx and Rx antennas** and by the **physical disposition** of its **antennas**.

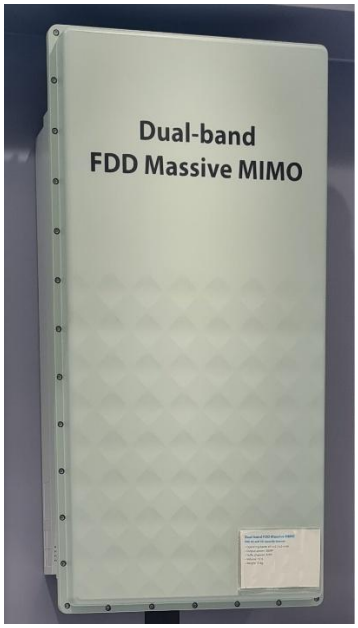
Some characteristics of **each antenna array** are:

- It is a **passive hardware device**.
- It manages **several passband signals** in the **time domain** with **analog representation**.
- On its **one end**, it interfaces with the **air** (or **vacuum**) and on its **other end either** with a **set** of **external RF cables** (click 4.5.8.2), or with a **set** of **channel filters** (click 4.5.8.9) or with a **combiner** (click 4.5.8.4)
- At least **1 antenna element** is **always** required for **each cell** (click 3.1.1).
- An **antenna array** is **not always** required.

**Either 1 Tx/Rx standalone antenna**, or **one Tx/Rx standalone antenna pair**, or **1 Tx/Rx antenna array** (click 4.5.8.6) is required for **each cell**. (i.e., for **each sector** in an **antenna site**). For **SUL** (click 10.4) operation, an **additional Rx antenna** or **Rx antenna array** is **required** for the **cell**. For **SDL** (click 10.5) operation, an **additional Tx antenna** or **Tx antenna array** is **required** for the **cell**. At least **1 antenna** is **required** in **every NR node**.

The **external appearance** of an **antenna array** is similar to the one of a standalone **antenna**: a **whitish rectangular box** connected to **several RF cables** (click 4.5.8.7).

The following picture shows a **dual-band FDD massive MIMO antenna array**:




This picture is from Mobile World Congress in Spain.

In **precoding**, the **signals** fed to the **antennas** in the **array** may be **extremely different** among themselves.

An **antenna array** is **required** for **MIMO operation** (**beamforming** and/or **diversity** and/or **precoding**).

For an **introduction** to **antenna theory**, click 11.21.

#### 4.5.8.7 Analog beamformer

An **analog beamformer produces one** (or more) **strong gain radiation pattern lobe(s)** in one (or more) **configurable direction(s)**. For more detail on **analog beamforming**, click 11.12.

Some characteristics of **each analog beamformer** are:

- It is usually a **passive hardware device**.
- It manages **several passband signals** in the **time domain** with **analog representation**.
- For a **Tx analog beamformer**, its **input signal** comes from **1 Tx channel filter** (click 4.5.8.9) and **each** of its **output signals** goes to **1 power amplifier** (click 4.5.8.10).
- For a **Rx analog beamformer**, **each** of its **input signals** comes from **1 Rx channel filter** (click 4.5.8.9) and its **output signal** goes to **1 downconverter** (click 4.5.8.4).
- At least **1 antenna element** is **always** required for **each cell** (click 3.1.1).
- An **analog beamformer** is **not always** required.

For the relationship between **analog beamforming** and **scheduling** in **5G**, click 10.11.

#### 4.5.8.8 Duplexer

**Each duplexer** converts from/to **one duplex passband signal** to/from **two simplex passband signals**. **Each duplexer** has **one duplex connector** (for the **duplex signal**) on **one end** and a **pair** of **simplex connectors** (for the **two simplex signals**) on the **other end**. A **duplexer** manages a **passband signal** in the **time domain** with **analog representation**.

A **bottom duplexer** is **required** when **one external RF cable** (click 4.5.8.2) is **shared** by **one RF chain pair** (click 4.5.8.1). An **additional top duplexer** is required in case a **Tx/Rx antenna pair** (click 4.5.8.5) is used.

For a **bottom duplexer**, its **duplex connector** interfaces with the **shared external RF cable**. **One** of its **2 simplex connectors** receives **1 input signal** from **1 power amplifier** (click 4.5.8.10) and its **other simplex connector outputs 1 signal** to **1 low-noise amplifier** (click 4.5.8.11).

For a **top duplexer**, its **duplex connector** interfaces with the **shared external RF cable**. **One** of its **two simplex connectors** receives **1 input signal** from **1 Rx antenna** and its **other simplex connector outputs 1 signal** to **1 Tx antenna**.

Some characteristics of **each duplexer** are:

- It manages **1 passband signal** in the **time domain** with **analog representation**.
- A **duplexer** is **not always** required.

For a figure showing how **both duplexers** are **connected**, see the **third RF chain** from 4.5.8.1.

There are **two types** of duplexers:

- **Frequency-division duplexers**, which are used in **FDD** (click 2.2.12).

- **Time-division duplexers**, which are used in **TDD** (click 2.3.4).

**Frequency-division duplexer**

**Each frequency-division duplexer** can be characterized by its **Downlink** and **Uplink frequency segments**, its **maximum supported power** and its **power losses**. It is a **passive hardware device** and consists of **filters** that separate the **Downlink channels** and the **Uplink channels** that are respectively located in the **Downlink** and **Uplink semi-bands**.

**Time-division duplexer**

**Each time-division duplexer** can be characterized by its **frequency segment**, by its **Tx-to-Rx transition time**, by its **Rx-to-Tx transition time**, by its **maximum supported power** and by its **power losses**. It is an **active hardware device** and consists of a **timed switch**, that alternatively connects the **RF connector** on **one end** to **either** of the **two RF connectors** on its **other end**. In case there is a **top duplexer**, it **must** be **perfectly synchronized** with the **bottom duplexer**.

### 4.5.8.9 Channel filter

**Each filter suppresses** the **signals outside** of its **frequency segment**. It can be characterized by its **frequency segment**. **Each channel filter** is used to **avoid out-of-channel emissions** and/or **receptions**.

Some characteristics of **each channel filter** are:

- It is a **passive hardware device**.
- It manages **1 passband signal** in the **time domain** with **analog representation**.
- For a **Tx channel filter**, its **input signal** comes from **1 upconverter** (click 4.5.8.12) and its **output signal** goes to **1 power amplifier** (click 4.5.8.10).
- For a **Rx channel filter**, its **input signal** comes from **1 low-noise amplifier** (click 4.5.8.11) and its **output signal** goes to **1 downconverter** (click 4.5.8.12).
- At least **1 Tx channel filter** and **1 Rx channel filter** is **always** required for **each cell** (click 3.1.1).

A **filter** is an **extremely common analog component**. In addition to the **channel filters** just indicated, **other filters** are **frequently present** in **different places** of the **RF functional block** (e.g., **after** a **power amplifier**, **inside** an **FDD duplexer**, …).

### 4.5.8.10 Power amplifier

**Each power amplifier** increases the **power** level of **1 signal** for its transmission through the **physical medium**. **Power amplifiers** are typically used in **transmission**, while **low-noise amplifiers** (**LNA**, click 4.5.8.11) are typically used in **reception**.

**Each power amplifier** can be characterized by its **frequency segment**, by its **amplification factor** and by its **maximum power level**.

**Power amplifiers** cause **non-linear distortions** in **OFDM**(**A**) systems, leading to a **high PAPR** (click 11.4) and to **out-of-band** (**OOB**) emissions. Some of these **negative effects** may be **mitigated** performing **digital pre-processing** (click 4.5.8.14) of the signal.

Some characteristics of **each power amplifier** are:

- It is an **active hardware device**.

- It manages **1 passband signal** in the **time domain** with **analog representation**.
- Its **input signal** comes **either** from **1 Tx channel filter** (click 4.5.8.9), or from **1 Tx analog beamformer** (click 4.5.8.7). Its **output signal** goes **either** to **1 bottom duplexer** (click 4.5.8.8), or to **1 external RF cable** (click 4.5.8.2), or to **1 antenna** (click 4.5.8.5).
- At least **1 power amplifier** is **always** required for **each cell** (click 3.1.1).

#### 4.5.8.11 Low-noise amplifier (LNA)

**Each low-noise amplifier** (**LNA**) increases the **amplitude** of a **signal** (and **not** necessarily its **power**) while introducing as **little noise** as possible. **Low-noise amplifiers** are typically used in **reception**, while **power amplifiers** (click 4.5.8.10) are typically used in **transmission**.

**Each low-noise amplifier** can be characterized by its **frequency segment**, by its **amplification factor** and by its **noise**.

Some characteristics of **each low-noise amplifier** are:

- It is an **active hardware device**.
- It manages **1 passband signal** in the **time domain** with **analog representation**.
- Its **input signal** comes **either** from **1 bottom duplexer** (click 4.5.8.8), or from **1 external RF cable** (click 4.5.8.2), or from **1 antenna** (click 4.5.8.5).
- At least **1 low-noise amplifier** is **always** required for **each cell** (click 3.1.1).

#### 4.5.8.12 Up/down converters

The **following subsections** present different aspects of **up/down converters**.

For more detail on **upconversion**, refer to clause 5.4 of TS 38.211.

##### 4.5.8.12.1 Analog up/down converters

In **Tx**, **each upconverter** transforms **1 complex-valued Tx baseband signal** "$\boldsymbol{I}(t_a)+\boldsymbol{jQ}(t_a)$" into **1 Tx real-valued passband signal** "$\boldsymbol{PS}(t_a)$", where **both** signals are in **analog representation** in the **time domain**.

In **Rx**, **each downconverter** transforms **1 Rx real-valued passband signal** into **1 Rx complex-valued baseband signal** "$\boldsymbol{I}(t_a)+\boldsymbol{jQ}(t_a)$", where **both** signals are in **analog representation** in the **time domain**.

**Each up/down converter** can be characterized by its **bandwidth** and by its **center frequency**.

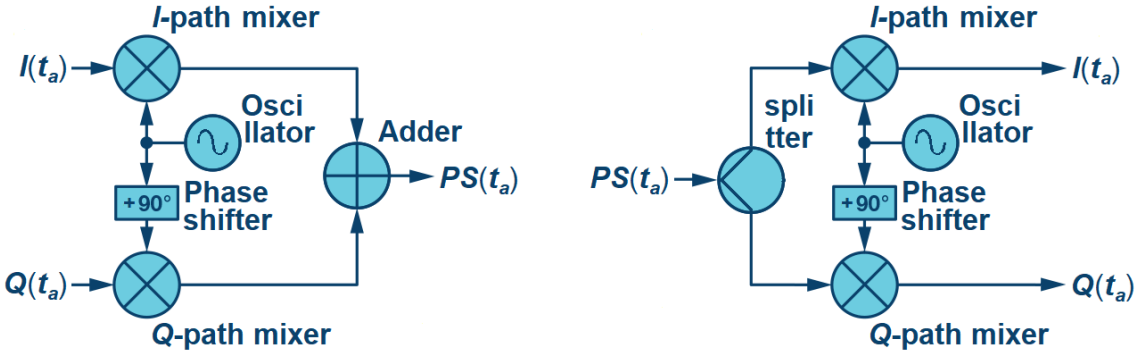
Some characteristics of **each up/downconverter** are:

- It is an **active hardware device**.
- For an **upconverter** (**Tx**), its **2 input real-valued baseband signals** in the **time domain** with **analog representation** "$\boldsymbol{I}(t_a)$" and "$\boldsymbol{Q}(t_a)$" come from **1 dual-channel D/A converter** (click 4.5.8.13). Its **single output real-valued passband signal** in the **time domain** with **analog representation** "$\boldsymbol{PS}(t_a)$" goes to **1 Tx channel filter** (click 4.5.8.9).
- For a **downconverter** (**Rx**), its **single input real-valued passband signal** in the **time domain** with **analog representation** "$\boldsymbol{PS}(t_a)$" comes from **1 Rx channel**

**filter** (click 4.5.8.9). Its **2 output real-valued baseband signals** in the **time domain** with **analog representation** "$I(t_a)$" and "$Q(t_a)$" go to a **dual-channel A/D converter** (click 4.5.8.13).

- At least **1 upconverter** and **1 downconverter** are **always** required for **each cell** (click 3.1.1).

A basic **upconverter** can be implemented with an **oscillator**, a **phase shifter**, **2 mixers** and an **adder**. A basic **downconverter** can be implemented with an **oscillator**, a **phase shifter**, **2 mixers** and a **splitter**. A **mixer** is a device that **outputs** the **product** (**multiplication**) of its **two input signals**. The **frequency** of the **oscillator** is the **frequency shift** resulting from the **up/down converter**. The following figure shows a basic diagram of an **upconverter** (**left**) and of a **downconverter** (**right**):




### 4.5.8.12.2 Mathematical formulation of upconversion

**Upconversion** of a **complex-valued baseband signal** in the **frequency domain** "$BS(f)$" is formulated by taking the **real part** of its **multiplication** by a **phasor** (click 11.19.6) of the **upconversion frequency** "$F_u$":

$$PS(t)=real(BS(f)\cdot e^{j2\pi Fut})$$

This may be seen quite intuitively by representing the **baseband signal** in terms of its frequency components:

$$BS(f)=a_1\, e^{j2\pi f1t}+a_2\, e^{j2\pi f2t}+a_3\, e^{j2\pi f3t}+\ldots+a_n\, e^{j2\pi fnt}$$

$$PS(t)=real(BS(f)\cdot e^{j2\pi Fut})$$

$$PS(t)=real([a_1\, e^{j2\pi f1t}+a_2\, e^{j2\pi f2t}+a_3\, e^{j2\pi f3t}+\ldots+a_n\, e^{j2\pi fnt}]\cdot e^{j2\pi Fut})$$

$$PS(t)=real(a_1\, e^{j2\pi f1t}\, e^{j2\pi Fut}+a_2\, e^{j2\pi f2t}\, e^{j2\pi Fut}+a_3\, e^{j2\pi f3t}\, e^{j2\pi Fut}+\ldots+a_n\, e^{j2\pi fnt}\, e^{j2\pi Fut})$$

$$PS(t)=real(a_1\, e^{j2\pi(f1+Fu)t}+a_2\, e^{j2\pi(f2+Fu)t}+a_3\, e^{j2\pi(f3+Fu)t}+\ldots+a_n\, e^{j2\pi(fn+Fu)t})$$

As you may see, **each** and **every** of the **phasors** that define "$BS(f)$" has its frequency **increased** in exactly the **upconversion frequency** "$F_u$".

The **upconversion** may also be formulated as the **multiplication** (respectively) of the **in-phase** component "$I(t)$" and the **quadrature** component "$Q(t)$" of "$BS(t)$" by "$cos(2\pi F_u t)$" and by "$-sin(2\pi F_u t)$". This **mathematical expression** corresponds **exactly** to the **basic upconverter** presented above:

$$BS(t)=I(t)+j\,Q(t)$$

$$PS(t)=I(t)\cdot cos(2\pi F_u t)-Q(t)\cdot sin(2\pi F_u t)$$

The **equivalence** between both formulations may be **proven** as:

$$PS(t)=real(BS(t)\cdot e^{j2\pi ft})$$

$$=real([I(t)+j\,Q(t)]\cdot[cos(2\pi F_u t)+jQ(t)\cdot sin(2\pi F_u t)])$$

$$=real(I(t)\cdot cos(2\pi F_u t)-Q(t)\cdot sin(2\pi F_u t)+j[I(t)\cdot cos(2\pi F_u t)+Q(t)\cdot sin(2\pi F_u t)])$$

$$=I(t)\cdot cos(2\pi F_u t)-Q(t)\cdot sin(2\pi F_u t)$$

#### 4.5.8.12.3 Mathematical formulation of downconversion

**Downconversion** of a **real-valued passband signal** in the **frequency domain** "$PS(f)$" is formulated by its **multiplication** by a **phasor** (click 11.19.6) of the **downconversion frequency** "$F_d$":

$$BS(f)=PS(f)\cdot e^{j2\pi t\,Fd}$$

This may **seem strange**, because this is **almost** the **same formulation** as for the **upconversion**. However, this formulation is correct because the **multiplication** by this **phasor** creates **both** the **downconverted baseband signal** "$BS(f)$" plus **high-frequency** "$2F_d$" components (which can be easily suppressed by low-pass filtering).

This may be shown with the **product** of **two cosine functions**, **one** representing the **passband signal** and the other the **downconversion sinusoid**:

$$BS(f)=PS(f)\cdot cos(2\pi F_u t)$$

$$BS(f)=cos(2\pi F_u t)\cdot cos(2\pi F_d t)$$

$$BS(f)=1/2[cos(2\pi(F_u-F_d)t)+cos(2\pi(F_u+F_d)t)]$$

$$BS(f)=cos(2\pi t)/2+cos(2\pi(2F_u)t)/2$$

The **first term** is the **baseband signal**. The **second term** is the **twice upconverted component** that is **suppressed** with a **low-pass filter**.

The **downconversion** may also be formulated as the **multiplication** of the **passband signal** in the **time domain** "$BS(t)$" by "$cos(2\pi F_d t)$" and by "$-sin(2\pi F_d t)$". This results (respectively) in the **in-phase** component "$I(t)$" and the **quadrature** component "$Q(t)$" of "$BS(t)$". This **mathematical expression** corresponds **exactly** to the **basic downconverter** presented above:

$$I'(t) = PS(t)\cdot cos(2\pi F_d t)$$

$$Q'(t)= -PS(t)\cdot sin(2\pi F_d t)$$

The proof of this formulation for the **in-phase** component "$I(t)$" is:

$$I'(t) = PS(t)\cdot cos(2\pi F_d t)$$

$$I'(t) = [I(t)\cdot cos(2\pi F_u t)-Q(t)\cdot sin(2\pi F_u t)]\cdot cos(2\pi F_d t)$$

$$I'(t) = I(t)\cdot cos(2\pi F_u t)\cdot cos(2\pi F_d t)-Q(t)\cdot sin(2\pi F_u t)\cdot cos(2\pi F_d t)$$

using the standard trigonometric identities:

$$I'(t) = I(t)\cdot cos(2\pi(F_u-F_d)t)/2+I(t)\cdot cos(2\pi(F_u+F_d)t)/2$$

$$-Q(t)\cdot sin(2\pi(F_u+F_d)t)/2-Q(t)\cdot sin(2\pi(F_u-F_d)t)/2$$

since "$F_u=F_d$":

$$I'(t) = I(t)/2 + I(t)\cdot cos(2\pi(2F_d)t)/2 - Q(t)\cdot sin(2\pi(2F_d)t)/2$$

The **second** and **third terms** are the **high frequency** "$2F_d$" components that are **suppressed** with a **low-pass filter**, resulting in "$I'(t) = I(t)/2$".

The proof of this formulation for the **quadrature** component "$Q(t)$" is:

$$Q'(t)= -PS(t)\cdot sin(2\pi F_d t)$$

$$Q'(t)= -[I(t)\cdot cos(2\pi F_u t)-Q(t)\cdot sin(2\pi F_u t)]\cdot sin(2\pi F_d t)$$

$$Q'(t)= -I(t)\cdot cos(2\pi F_u t)\cdot sin(2\pi F_d t)+Q(t)\cdot sin(2\pi F_u t)\cdot sin(2\pi F_d t)$$

using the standard trigonometric identities:

$$Q'(t)= -I(t)\cdot sin(2\pi(F_d+F_u)t)/2+I(t)\cdot sin(2\pi(F_d-F_u)t)/2 + Q(t)\cdot cos(2\pi(F_d-F_u)t)/2 - Q(t)\cdot cos(2\pi(F_d+F_u)t)/2$$

since "$F_u=F_d$":

$$Q'(t)= -I(t)\cdot sin(2\pi(2F_d)t)/2 + Q(t)/2 - Q(t)\cdot cos(2\pi(2F_d)t)/2$$

The **first** and **third terms** are the **high frequency** "$2F_d$" components that are **suppressed** with a **low-pass filter**, resulting in "$Q'(t) = Q(t)/2$".

#### 4.5.8.12.4 Up/down converters in the frequency-domain or with digital representation

The above explanation is for **up/down converters** in the **time domain** with **analog representation**. However, it is also possible to do the **upconversion** in the **frequency domain** and in the **time-domain** with **digital representation**.

**Up/down conversion** in the **frequency-domain** is performed, as has been shown in the **mathematical formulation** above, by **multiplying** "$BS(f)$" by the **phasor** "$e^{j2\pi t\,Fd}$".

**Up/down conversion** in the **time-domain** with **digital representation** is performed using **very high frequency digital signal processors** (**DSP**).

In **both** cases the **up/down conversion** is performed **before** the **analog-digital conversion**. Consequently, **both** cases require a **very high frequency analog-digital converter**.

### 4.5.8.13 Analog to/from Digital converters (A/D and D/A)

In **Tx**, **each D/A converter** transforms **1 baseband signal** in the **time domain** from a **digital** to an **analog** representation, by means of **sampling** and **binary codification** of **each sample**.

In **Rx**, **each A/D converter** transforms **1 baseband signal** in the **time domain** from an **analog** to a **digital** representation, by means of **sample reconstruction** and **filtering**.

**Each analog to/from digital converter** can be characterized by its **bandwidth** and by its **dynamic margin**.

Some characteristics of **each analog to/from digital converter** are:

- It is an **active hardware device**.

- For a **D/A converter** (**Tx**), its **input signal** comes **either** from the **Low Physical functional block** (click 4.5.5) or from an **RF digital preprocessor**. Its **output signal** goes to a **Tx channel filter** (click 4.5.8.9).
- For a **A/D converter** (**Rx**), its **input signal** comes from an **Rx channel filter** (click 4.5.8.9). Its **output signal** goes **either** to the **Low Physical functional block** or to an **RF digital postprocessor** (click 4.5.8.14).
- At least **1 A/D converter** and **1 D/A converter** are **always** required for **each cell** (click 3.1.1).

#### 4.5.8.14 RF digital processor

In **Tx**, **each digital preprocessor** may perform **one** or **more functions**, such as:

- **Channel filtering**
- **Interpolation**
- **Digital pre-distortion** (**DPD**), to **pre-correct** the **non-linearities** introduced by the **power amplifiers** (click 4.5.8.10).

In **Rx**, **each RF digital postprocessor** may perform **one** or **more signal conditioning functions**.

**Each RF digital processor** can be characterized by its **bandwidth**, and by the **specific function**(**s**) that it implements.

Some characteristics of **each digital processor** are:

- It is frequently implemented as **software** over a **general-purpose computer**, **possibly** with **hardware accelerators** (**ASIC**, **FPGA** or **GPU**).
- It manages **one** or **several baseband signal**(**s**) in the **time domain** with **digital representation**.
- For an **RF digital preprocessor** (**Tx**), its **input signal** comes from the **Low Physical functional block** (click 4.5.5) and its **output signal** goes to a **D/A converter** (click 4.5.8.13).
- For an **RF digital postprocessor** (**Rx**), its **input signal** comes from a **D/A converter** (click 4.5.8.13) and its **output signal** goes to the **Low Physical functional block** (click 4.5.5).
- An **RF digital processor** is **not always** required.

## 4.6 Backhaul, midhaul and fronthaul

The **backhaul** (click 4.6.1) is the **remote interface** between the **5G Core** (click 1.1.1) and **each** of its **Central Units** (**CUs**, click 4.7.1). The **backhaul connectivity** spans between **each CU** and the **closest point of presence** (**PoP**) of the **transport network**.

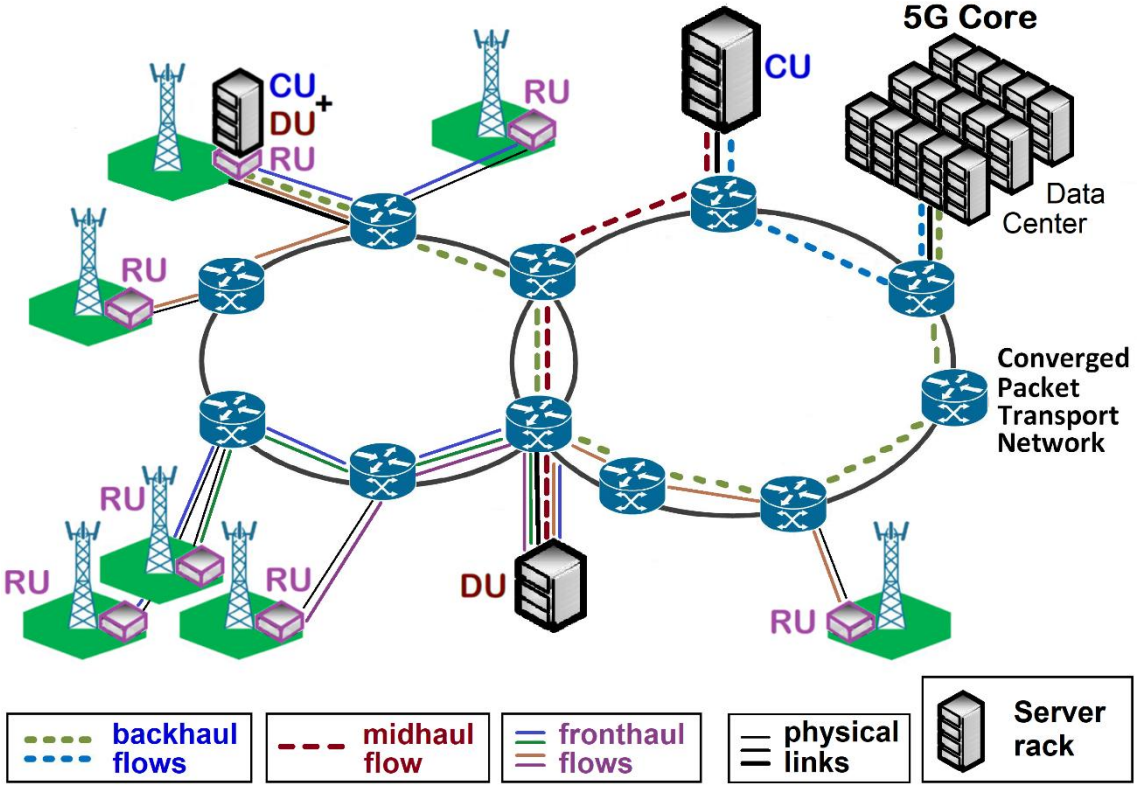
The **midhaul** (click 4.6.2) is the **remote interface** and the **connectivity** between **each Central Unit** and **each** of its **Distributed Units** (**DUs**, click 4.7.2).

The **fronthaul** (click 4.6.3) is the **remote interface** and the **connectivity** between **each Distributed Unit** and **each** of its **Radio Units** (**RUs**, click 4.7.3).

The **connectivity** in **backhaul** and **midhaul** is **most frequently** provided by a **converged packet network** (although it may **also** be provided by a **direct point-to-point link**).

**Fronthaul** connectivity is provided either as a **fiber link** (in **CPRI**) or as a **packet flow** (in **eCPRI**) over a **packet-switched network**.

The following figure shows an example of a **Radio Access Network** (**RAN**):




Some **relevant remarks** on this figure are:

- In this **RAN** example, the **fronthaul remote interface** is **eCPRI**. This is why **all** the **connectivity** is provided by a **converged packet network**.
- The figure shows the **information flows** corresponding to **backhaul** (**BH**), **midhaul** (**MH**) and **fronthaul** (**FH**). **All** these **information flows** are transported over the **converged packet network**.
- The figure shows **1 integrated NR node** with **1 CU**, **1 DU** and **1 RU** at the **same site**.
- The figure shows **1 distributed NR node** with **1 CU**, **1 DU** and **6 RUs**, **all** of them at **different site**s.
- The **transport network** might be the **right ring** and the **backhaul** might be the **left ring**. The **backhaul connectivity** of the **left CU** would therefore span from the **CU** to the **right ring** (e.g., rural **CU**). The **backhaul connectivity** of the right **CU** would be its link to the **right ring** (e.g., urban **CU**).

The **required transmission capacity** for **backhaul**, **midhaul** and **fronthaul** are **extremely different**. The **required transmission capacity** for **each** of them is also **extremely dependent** on the **cell characteristics**. The following table shows the required

transmission capacity for **1 cell** with **100 MHz DL/UL bandwidth**, **64QAM modulation** (**$Q_m$= 6 bits/symbol**), **8 MIMO layers** and **32 antenna ports**.

| | Func. Split | Capacity [Gbps] | |
|---|---|---|---|
| | | Downlink | Uplink |
| **Backhaul** | **N/A** | 4.00 | 3.00 |
| **Midhaul** | **2** | 4.02 | 3.02 |
| **Fronthaul** | **7.3** | 22.00 | 84.00 |
| | **7.2** | 86.00 | 84.00 |
| | **7.1** | 10.00 | 16.50 |
| | **8** | 157.20 | 157.20 |

Source web pages:

- https://accelleran.com/understanding-the-du-and-cu-in-open-ran-architecture-2/
- https://aviatnetworks.com/uncategorized/evolution-transport-5g/

### 4.6.1 Backhaul

The **backhaul** is the **remote interface** between the **5G Core** (click 1.1.1) and **each** of its **Central Units** (**CUs**, click 4.7.1). The **backhaul connectivity** spans between **each CU** and the **closest point of presence** (**PoP**) of the **transport network**.

**Backhaul remote interface**

The **backhaul remote interface** is the **NG remote interface**. The **NG remote interface** is composed of **2 parts**: the **NG control plane** (**NG-C interface**) and the **NG user plane** (**NG-U interface**). The **NG-C** is the **interface** between the **CU** and the **Core**'s **Access** and **Mobility Management Function** (**AMF**). The **NG-U** is the **interface** between the **CU** and the **Core**'s **User Plane Function** (**UPF**).

The **NG remote interface** tolerates a **maximum one-way latency** of **30** to **50 ms** (depending on service type), which determines the **maximum allowed distance** of the **backhaul connectivity** (see below).

The **NG remote interface** is quite complex, and it is specified in the following TS documents:

| TS Document | Title | Contents |
|---|---|---|
| **TS 38.401** | NG RAN; Architecture description | High-level architecture including **NG**, **Xn**, **F1**, etc. |
| **TS 38.410** | NG RAN; NG general aspects and principles | Architecture, layering, naming, versioning |
| **TS 38.412** | NG RAN; NG signaling transport | Transport network layer (e.g., SCTP, GTP) |
| **TS 38.413** | NG RAN; NG Application Protocol (NGAP) | Control plane signaling over **NG-C** |
| **TS 38.414** | NG RAN; NG data transport | **NG-U**, GTP-U transport |

**Backhaul connectivity**

The **backhaul connectivity** is **most frequently** provided by a **converged packet network**, but it may **also** be provided by a **direct point-to-point link**.

The **backhaul** **connectivity distance** depends on the **deployment type**:

- **Urban small cell**: **50 m** to **1 km**.
- **Urban macro**: **200 m** to **3 km**.
- **Suburban macro**: **1.5 km** to **10 km**.
- **Rural macro**: **2 km** to **20 km**.
- **Remote macro**: **10 km** to **80+ km**.

For the required **backhaul** **capacity**, you may see the table in section 4.4.

**New 5G deployments** are increasingly based in a **converged packet network**, where the backhaul and the **transport network** have converged into a **seamless packet-based infrastructure**. Consequently, the **frontier** between backhaul and the **transport network** is **blurred**, and it is based more on **operation** and **infrastructure** considerations than on a **technical difference**. The **backhaul** might be considered the **part** of the **converged packet network** that is **only** devoted to **mobile communications traffic**, while the **transport network** would be the **part** where **fixed communications traffic** (e.g., fiber to the home) is **also conveyed**.

The **backhaul** **connectivity** is based on **either fiber**, **radio-link(s)** or **satellite** (more recently).

### 4.6.2 Midhaul

The **midhaul** is the **remote interface** and the **connectivity** between **each** **Central Unit** (**CUs**, click 4.7.1) and **each** of its **Distributed Units** (**DUs**, click 4.7.2).

**Midhaul** **functional splits**

The **midhaul** **remote interface** typically corresponds to **functional split 2** (click 4.4). **Functional split 2** is the division between the **PDCP** (click 4.5.2) **functional block**, placed at the **CU**, and the **High RLC** (click 4.5.3.1) **functional block**, placed at the **DU**.

**Midhaul** **remote interface**

The **midhaul** **remote interface** for **functional split 2** is the **F1** **interface.**

The **F1** **remote interface** is composed of **2 parts**: the **F1** **control plane** (**F1-C** **interface**) and the **F1** **user plane** (**F1-U** **interface**). The **F1-C** **interface** includes the **Radio Resource Control** (**RRC**, click 5.8) **signaling** and the **UE context management**. Its **protocol stack** is composed of **SCTP** over **IP** and the **F1 Application Protocol** (**F1AP**). The **F1-U** **interface** carries the **Packet Data Convergence Protocol** (**PDCP**, click 4.5.2) **packets**. Its **protocol stack** is composed of the **user plane** of the **GPRS tunneling protocol** (**GTP-U**) over **IP**.

The **F1** **remote interface** tolerates a **maximum one-way latency** of **10** to **20 ms**, which determines the **maximum allowed distance** of the **midhaul** **connectivity**

The **F1 remote interface** is specified in the following TS documents:

| TS Document | Title | Contents |
|---|---|---|
| **TS 38.401** | NG-RAN; Architecture description | High-level architecture including **NG**, **Xn**, **F1**, etc. |
| **TS 38.470** | NG-RAN; F1 general aspects and principles | High-level architecture, interface functions, protocol stack overview |
| **TS 38.472** | NG-RAN; F1 signaling transport | **F1-C** transport over SCTP |
| **TS 38.473** | NG-RAN; F1 Application Protocol (F1AP) | **F1-C** signaling messages and procedures |
| **TS 38.474** | NG-RAN; F1 data transport | **F1-U** transport over GTP-U |

**Midhaul connectivity**

The **midhaul connectivity** is **most frequently** provided by a **converged packet network**, but it may **also** be provided by a **direct point-to-point link**.

The **midhaul connectivity distances** and **deployment types** are similar to the ones of **backhaul** (click 4.6.1). The reason is that the **centralization** of **CUs** and the **distribution** of **DUs** is **architecturally similar** to **backhaul planning**.

For the required **midhaul capacity**, you may see the table in section 4.4.

### 4.6.3 Fronthaul

The **fronthaul** is the **remote interface** and the **connectivity** between **each Distributed Unit** (**DU**, click 4.7.2) and **each** of its **Radio Units** (**RUs**, click 4.7.3).

**Fronthaul functional splits**

The **fronthaul remote interface** typically corresponds **either** to **functional split 8** (click 4.4) or to **functional split 7.2** (click 4.5.7.2). **Functional split 8** is the division between the **Low physical functional block**, placed at the **DU**, and the **Radio Frequency** (click 4.5.8) **functional block**, placed at the **RU**. **Either** of the **3 functional splits 7.1**, **7.2** and **7.3** is the division between the **High physical** (click 4.5.5) **functional block**, placed at the **DU**, and the **Low physical** (click 4.5.5) **functional block**, placed at the **RU**.

**Fronthaul remote interfaces**

The **fronthaul remote interface** for **functional split 8** is the **Common Public Radio Interface** (**CPRI**). It is specified in the document "**Common Public Radio Interface (CPRI); Interface Specification**", produced by the **CPRI cooperation group** (an industry consortium). Its latest version is **7.0** and it was published on October 2015. This version allows for a **bitrate** up to **24 Gbps**.

The **CPRI interface** tolerates a **maximum one-way latency** of **100** to **250 µs**, which determines the **maximum allowed distance** of the **fronthaul connectivity** with **functional split 8** (see below). The **maximum** tolerated **delay jitter** (i.e., **latency variability**) is around **10 ns**.

The **fronthaul remote interface** for **functional split 7.2** is the **enhanced Common Public Radio Interface** (**eCPRI**). It is specified in the document "**Common Public Radio Interface: eCPRI Interface Specification**", produced by the **CPRI cooperation group** (an industry consortium). Its latest version is **2.0** and it was published on May 2019. This version allows for a **bitrate** over **25 Gbps per link**.

The **eCPRI interface** tolerates a **maximum one-way latency** of **250 µs** to **1 ms**, which determines the **maximum allowed distance** of the **fronthaul connectivity** with **functional split 7.2** (see below). The **maximum** tolerated **delay jitter** (i.e., **latency variability**) is **50** to **100 µs**. The tolerance to **delay jitter** is **much larger** in **eCPRI** than in **CPRI** because **eCPRI** incorporates a **buffering** mechanism.

The **maximum** tolerated **latency** and **jitter** in **fronthaul** is due to the **real-time signal processing** requirements to maintain a **sufficiently precise radio timing** (e.g., for **beamforming** or **space-division MIMO**).

The specifications for **CPRI** and **eCPRI** may be downloaded from:

- https://www.cpri.info/spec.html

**Fronthaul connectivity**

The **fronthaul connectivity** with **CPRI** is provided by a **direct point-to-point fiber** or by a **direct point-to-point lambda** over **WDM**. The **maximum distance** is around **10 km** due to its **strict latency** and **delay jitter** constraints (see above).

The **fronthaul connectivity** with **eCPRI** can be provided by an **Ethernet packet converged network**, as it is done for **backhaul** (click 4.6.1) and **midhaul** (click 4.6.2). The **maximum distance** is around **40 km** due to its **less strict latency** and **delay jitter** constraints (see above).

For the required **fronthaul capacity**, you may see the table in section 4.4.

## 4.7 NR node subsystems

**Each NR node subsystem** is one of the **physical components** of an **NR node**.

**Each NR node subsystem** can be characterized by the **NR node functional blocks** (click 11.16) that it contains.

**Each equipment manufacturer** offers a **wide range** of commercial **NR node subsystems**, such that their **combinations** allow to **assemble NR nodes** with **specific characteristics**. Since **NR node subsystems** are **commercial equipment**, the **terminology** is **quite diverse** and **little standardized**. However, the **subsystems** presented in the following subsections are **quite widespread** and **well established**. In addition to the **subsystems** presented here, **many more exist** in the commercial offer by the different manufacturers.

### 4.7.1 Central Unit (CU)

The **Central Unit** (**CU**) consists of the **RRC/SDAP functional block** (click 4.5.1) and the **PDCP functional block** (click 4.5.2). Therefore, the **CU** consists of **protocol layer 3** and **control/management functions**. The **CU** conforms to **functional split 2** (click 4.4).

The main functions of the **CU** include:

- **Network Management**

  **Manages broader data flow** within the network, dictating how **data packets travel** through the infrastructure.

- **Supervision of Distributed Units**

  In order to maintain coherent communication between the **5G Core** and the **Radio Units** (**RUs**, click 4.7.3), the **CU** instructs the **Distributed Units** (**DUs**, click 4.11.1), guiding their data management tasks.

- **High-level Decision-making**

  Make **overarching network decisions**, such as **mobility management** and **user-specific communication bearers**.

Some characteristics of the **CU** are:

- It is usually **implemented** as **software** over a **general-purpose computer**.
- It manages **data** in the **digital domain**.
- It interacts with its controlling **5G Core** (click 1.1.1) through **1 backhaul** (click 4.6.1) **remote interface** (click 4.5.3.2) and with **each** of its controlled **Distributed Units** (**DUs**, click 4.7.2) through **1 midhaul** (click 4.6.2) **remote interface**.
- It **possibly** interacts with **other CUs** through the **Xn interface**.
- **Each CU** may control **up to over a hundred DUs**.
- **Each NR node** has **1**, and **only 1**, **CU**.

If the **NR node** is an **en-gNB** (click 4.2), its **CU** does **not** include the **SDAP functional block**, because in this case the **NR node** is **connected** to the **LTE Core**, and **not** to the **5G core**.

The **CU** is sometimes called "**higher-layer processor**".

Source web pages:

- https://www.5gworldpro.com/5g-knowledge/what-is-cu-and-du-in-5g.html#:~:text=The%205G%20BTS%20named%20gNB,RLC%2C%20MAC%20and%20Physical%20layer.
- https://aviatnetworks.com/uncategorized/evolution-transport-5g/
- https://accelleran.com/understanding-the-du-and-cu-in-open-ran-architecture-2/

### 4.7.2 Distributed Unit (DU)

The **Distributed Unit** (**DU**) consists of the **High** and **Low RLC functional blocks** (click 4.5.3), the **High** and **Low MAC functional blocks** (click 4.5.4) and the **High Physical functional block** (click 4.5.7) .Therefore, the **DU** consists of **protocol layer 2** and the **non-time-critical** functions of the **protocol layer 1**. The **DU** conforms to **functional splits 2** and **either 7.1**, or **7.2** or **7.3** (click 4.4).

With **split 7.3**, the **DU** performs the following **essential functions**:

- **Interacting** with its controlling **Central Unit** (**CU**, click 4.7.1).
- **Controlling** its **Radio Units** (**RUs**, click 4.7.3).
- **Time-critical** functions of the **Radio Link Control** (**RLC**) **protocol**, for **reliable time-critical data transmission**.
- **All** the functions of the **Media Access Control protocol**.

- **Channel encoding** and **decoding.**
- **Rate-matching** and **de-matching**.
- **Demodulate** the **signals** (but not modulate them).
- **Perform** the i**DFT** for **each UE**, for **DFT-s-OFDM** (click 11.16) reception. For **specific scenarios** where **OFDMA** is used, this function would be **omitted**.
- **Signal equalization**.
- **Diversity combining** (optionally).
- **Channel estimation**.

With **split 7.2**, the **DU** performs the **following functions**, in **addition** to the ones of **split 7.3**:

- **Modulate** the **signals**.
- **Layer map** the **symbols.**
- **Precoding** (optionally).

With **split 7.1**, the **DU** performs the **following functions**, in **addition** to the ones of **splits 7.3** and **7.2**:

- **RE mapping** and **de-mapping.**
- **Digital beamforming** (optionally).
- **Port reduction**.

The main functions of the **DU** (with **split 7.3**) include:

- **Data Organization and Management**:

  The **DU structures data** into the **right format** for **efficient radio transmission**.

- **Interaction with the Radio Units** (**RUs**, click 4.7.3)

  The **DU directly manages data communication** with its **RUs**, transforming the **structured data** to/from a **baseband signal** in the **digital domain**.

- **Latency Reduction and Efficiency**

  The **DUs** contribute to provide **very-low latency** by managing the **lower layers** of the **protocol stack closer** to the **RU**.

Some characteristics of the **DU** are:

- It is usually **implemented** as **software** over a **general-purpose computer**, **possibly** with **hardware accelerators** (**ASIC**, **FPGA** or **GPU**).
- It manages **data** in the **digital domain**, **baseband signals** in the **frequency domain** and **baseband signals** in the **time domain** with **digital representation**.
- It interacts with its controlling **CU** through **1 midhaul** (click 4.6.2) **remote interface** and with **each** of its controlled **Radio Units** (**RUs**, click 4.7.3) through **1 fronthaul** (click 4.6.3) **remote interface**.

- The **maximum delay** between a **CU** and **each** of its **DUs** is in the order of **10 ms**, which implies a **maximum distance** in the order of **100 km**. For more detail, refer to the **midhaul** (click 4.6.2).
- The **quantity** of **RUs** that can be **controlled/supported** by **each DU** depends on many factors and configuration (space multiplexing, space diversity, channel bandwidth, quantity of antennas, …). To provide an order of magnitude, you may think that **each DU controls/supports** between **8** and **64 RUs**.
- **Each NR node must** have **at least 1 DU**.

Source web pages:

- https://www.5gworldpro.com/5g-knowledge/what-is-cu-and-du-in-5g.html#:~:text=The%205G%20BTS%20named%20gNB,RLC%2C%20MAC%20and%20Physical%20layer.
- https://aviatnetworks.com/uncategorized/evolution-transport-5g/
- https://accelleran.com/understanding-the-du-and-cu-in-open-ran-architecture-2/

### 4.7.3 Radio Unit (RU)

The **Radio Unit** (**RU**) consists of the **Low Physical functional block** (click 4.5.5) and the **Radio Frequency functional block** (click 4.5.8), **all** in **1 encasing**. Therefore, the **RU** consists of the **time-critical** functions of the **protocol layer 1**. The **RU** conforms to **functional split either 7.1**, or **7.2** or **7.3** (click 4.5.5).

With **split 7.1**, the **RU** performs the following **essential functions**:

- Transform **each baseband signal** between the **frequency domain** and the **time domain** with **digital representation** (i.e., **DFT** and **iDFT**).
- **Insert/remove** the **cyclic prefix**.
- **Transform** the **baseband signal** in the **time domain** from **digital representation** into **analog representation**.
- **Up/down convert**, **amplify** and **filter** the **signals**.
- **Analog beamforming** (optionally).
- **Radiate** and **receive** the **electromagnetic beams** to/from the **medium** (air or vacuum).

With **split 7.2**, the **RU** performs the **following functions**, in **addition** to the ones of **split 7.1**:

- **RE mapping** and **de-mapping.**
- **Digital beamforming** (optionally).
- **Port reduction**.

With **split 7.3**, the **RU** performs the **following functions**, in **addition** to the ones of **splits 7.1** and **7.3**:

- **Modulate** the **signals** (but **not** demodulate).
- **Layer mapping** (but **not** layer de-mapping), optionally.
- **Precoding** (but **not** detecting), optionally.

Some characteristics of the **RU** are:

- The **Low Physical functional block** is usually **implemented** as **software** over a **general-purpose computer** with **hardware accelerators** (**ASIC**, **FPGA** or **GPU**) and the **Radio Frequency functional block** is implemented as **hardware devices** (except the **RF digital processing**).
- It manages **baseband signals** in the **time domain** with **both digital** and **analog representation** and **passband signals** in the **time domain**.
- It interacts with its controlling **Distributed Unit** (**DU**, click 4.11.1) through **1 fronthaul** (click 4.6.3) **remote interface** and with the **UEs** camped in its **cell** through the air interface.
- The **maximum delay** between a **DU** and **each** of its **RUs** is in the order of **1 ms**, which implies a **maximum distance** between **5** and **15 km**. For more detail, refer to the **fronthaul** (click 4.6.3).
- The **antennas of each RU** usually **illuminate 1 cell** (click 3.1.1).
- **Each RU** is placed in **1 antenna site** (click 3.1.3). **Each antenna site** may host (and usually does) **several RUs**. In case the **antenna site** belongs to a **tower company**, it will usually host **RUs** from several **operators**.
- **Each NR node must** have **at least 1 RU**.

Source web pages:

- https://www.5gworldpro.com/5g-knowledge/what-is-cu-and-du-in-5g.html#:~:text=The%205G%20BTS%20named%20gNB,RLC%2C%20MAC%20and%20Physical%20layer.
- https://aviatnetworks.com/uncategorized/evolution-transport-5g/
- https://accelleran.com/understanding-the-du-and-cu-in-open-ran-architecture-2/

### 4.7.4 Baseband Unit (BBU)

A **baseband unit** (**BBU**) consists of the following **functional blocks**: **Central Unit** (**CU**, click 4.7.1), **High** and **Low RLC** (click 4.5.2.1), **High** and **Low MAC** (click 4.5.4) and **High physical** (click 4.5.5), **all** in **1 encasing**. Its interface with the Remote Radio Unit (**RRU**, click 4.7.5) therefore conforms to **1** of the **splits 7.x** (click 4.5.7.2). **Each BBU** is connected to **each** of its controlled **RRUs** (click 4.7.5) through **1 fronthaul** (click 4.6.3) **remote interface**.

The **BBU** is a **term** mainly bound to **LTE**. In **5G**, a **BBU** would be considered as **1 CU** plus **1** or **several DUs** (click 4.7.2).

Source web page:

- https://www.techtarget.com/whatis/definition/baseband-unit-BBU

### 4.7.5 Remote Radio Unit (RRU)

A **remote radio unit** (**RRU**) consists of **all** the **required RF chains** (click 4.5.8.1), for **one** or **several cells** (click 3.1.1), **all** in **1 encasing**. The **remote radio unit** is frequently mounted on **top** of the **mast**, very close to the **antenna**(**s**) (click 4.5.8.5) or **antenna array**(**s**) (click 4.5.8.6) connected to them with the corresponding **external RF cables** (click 4.5.8.7).

The following picture shows **three different multi-band RRUs**:




This picture is from Mobile World Congress, Spain.

The following picture shows an **RRU** connected to a **4TR antenna array**, both of them located on top of a mast:



This picture is from the 5TONIC laboratory, at IMDEA Networks and U. Carlos III, Madrid (Spain).

Some **relevant remarks** on this picture are:

- The **RRU** is the **smaller** box on the back.
- The **4TR antenna array** is the **larger** box on the front.
- The **RRU** and the **antenna array** are connected with **8 RF cables** (the thick black cables).
- The **RRU** is connected with **two fiber optic** cables (the thin black cables) to the **distributed unit** (**DU**, click 4.7.2) placed in **another location**.

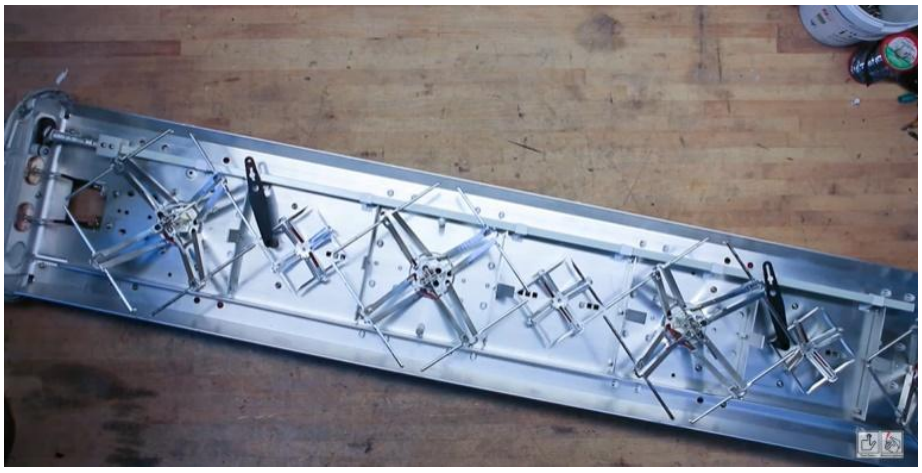
A **remote radio unit** is sometimes called **remote radio head** (**RRH**).

### 4.7.6 Multiband antenna

**One multiband antenna** is composed of **several antenna elements** (click 4.5.8.5) for **each supported band** (i.e., **frequency segment**), **all** in **1 encasing**. It usually requires an

**internal combiner** (click 4.5.8.4). Its **external appearance** is a **whitish rectangular box**, similar to a **single-band standalone antenna**.

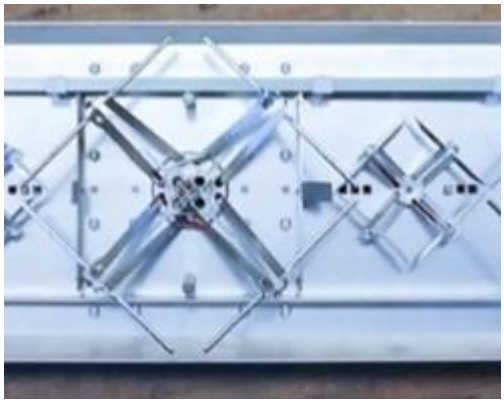
The following **picture** shows the **internal structure** of a **dual band 900 MHz and 1,800 MHz antenna**.



Some **relevant remarks** on this picture are:

- **Each** internal **antenna element** is a **dipole** (click 11.21.8).
- The **antenna** contains **several antenna elements**, in **sets** of **4 dipoles**, where **each dipole** within its **set** is **oriented** in parallel to the **side** of a **square**.
- The **antenna** contains both **900 MHz dipoles** (the **larger** ones) and **1,800 MHz dipoles** (the **smaller** ones).

The following picture shows a closeup of **2 sets** of **4 dipoles**, the **larger** ones are for **900 MHz** and the **smaller** ones are for **1,800 MHz**.



The following is an **excellent** video that shows the **internal components** and **structure** of a **dual band antenna**, although the **verbal description** in the video is somehow **unprecise**:

- https://www.youtube.com/watch?v=kCEX8wOVS2k.

### 4.7.7 Active Antenna Unit (AAU)

An **active antenna unit** consists of **one** (or more) **antenna**(**s**) (click 4.5.8.5) or **antenna array**(**s**) (click 4.5.8.6), with **all** their **required RF chains** (click 4.5.8.1), **all** in **1 encasing**.

The **two first cases** of **RF chain pairs**, together with their **antennas**, that were presented in 4.5.8.1 are also examples of the **internal structure** of **two active antenna units**.

The following picture shows a **64TR active antenna unit** for **band n78**, with **400 MHz** of **bandwidth**, with the usual antenna appearance of a whitish box:




This picture is from Mobile World Congress, Spain.

An **active antenna unit** is sometimes just called "**active antenna**".

## 4.7.8 NR node cost estimates

The **NR node cost** can be split in the **cost** of its **3 subsystems**:

- **Central Unit** (**CU**, click 4.7.1).
- **Distributed Units** (**DUs**, click 4.7.2) controlled by the **CU**.
- **Radio Units** (**RUs**, click 4.7.3) controlled by **all** the **DUs**.

The **cost** of **each** such **subsystem** includes the following **3 cost elements**:

- **Subsystem equipment**: the **cost** of the **telecommunications equipment itself**.
- **Subsystem connectivity**: the **cost** of the **connectivity** between the **Unit**, and the corresponding **upstream equipment**.
- **Subsystem housing**: the **cost** of the **non-telecommunication elements** required to **place** the **Unit**.

### 4.7.8.1 Central Unit cost

The **3 cost elements** of a **CU** are:

- **CU equipment cost**

  It is the **cost** of the **computer servers**, that depends on the **required computing capacity**, which itself depends on the **quantity** and **capacities** of the **DUs** and **RUs** controlled by the **CU**.

- **CU connectivity cost**

  It is the **cost** of the **backhaul** (click 4.6.1) **connectivity** from the **CU** to the **closest point of presence** (**PoP**) of the **transport network**. This **cost** depends **mainly** on the distance, which has a different **range** depending on the **deployment type**.

  **Connectivity** may be provided by the **operator** with its **own infrastructure** (mainly **CapEx**) and/or hiring **service** from a **third party** (mainly **OpEx**).

- **CU housing cost**

  It is the **cost** of **physical housing**, **power** and possibly **climatization** and **security**.

  In **urban** and **suburban** areas, the **physical housing** may be a **street cabinet**, a building **basement** or **rooftop**, or an **outdoor encasing**. The following picture shows an **outdoor encasing** attached to an **antenna post** (that is also used for street lights):



  This picture is from Moskow (Russia).

In **rural** and **remote** areas, the **physical housing** is usually a purpose-built **shelter**. The following picture shows an **NR node shelter**, with its **security fencing** and **diesel power generator** (the big red equipment):



This picture is from Miraflores de la Sierra, Madrid (Spain).

The following is a picture of another view of the **same shelter**, where you can see **2 external air conditioning** units for **climatization**.



For a **cost estimate** of the **CU**, click 4.7.8.4.

### 4.7.8.2 Distributed Unit cost

The **3 cost elements** of a **CU** are:

- **DU equipment cost**

  It is the **cost** of the **computer servers**, with their **accelerators**. The **cost** depends on the **required computing capacity**, which itself depends on the **quantity** and capacities of the **RUs** controlled by the **DU**.

- **DU connectivity cost**

  It is the **cost** of the **midhaul** (click 4.6.2) **connectivity** between the **DU** and its controlling **CU**. This **cost** depends **mainly** on the distance, which has a different **range** depending on the **deployment type**:

  - **Urban** (**short** distance): up to **10 km**.
  - **Suburban** (**medium** distance): **10 km** to **40 km**.
  - **Rural or remote** (**long** distance): **40 km** to **100 km**.

  **Midhaul connectivity** is based on **either fiber** or **radio-link(s)**. **Midhaul connectivity** may be self-provided by the **operator** with its **own infrastructure** (mainly **CapEx**) and/or hiring **service** from a **third party** (mainly **OpEx**).

- **DU housing cost**

  It is the **cost** of **physical housing**, **power** and possibly **climatization** and **security**.

  The usual implementations are quite similar to the **CU housing cost** (click 4.7.8.1).

For a **cost estimate** of the **DU** and its **connectivity** click 4.7.8.4.

### 4.7.8.3 Radio Unit cost

The **3 cost elements** of a **RU** are:

- **RU equipment cost**

  It is the **cost** of the **radio frequency equipment** (click 4.5.8), including the **antennas**. The cost depends on the **band(s)**, on the **radiated power**, on the **radio channel bandwidth(s)**, on the **antenna gain**, on the **form factor**, on **beamforming**, and **very much** on the **MIMO characteristics**. Notice that **each Tx** or **Rx layer** requires (respectively) a **complete Tx** or **Rx RF chain**.

- **RU connectivity cost**

  It is the **cost** of the **fronthaul** (click 4.6.3) **connectivity** between the **RU** and its controlling **DU**. This **cost** depends **mainly** on the **quantity** of **layers**, that determine the **required fronthaul capacity**.

  **Fronthaul connectivity** is **typically** based on **fiber** because of the **very large capacity** required to convey **several digitized baseband signals** (**1** for **each layer**). **Fronthaul connectivity** may be self-provided by the **operator** with its **own infrastructure** (mainly **CapEx**) and/or hiring **service** from a **tower company** (mainly **OpEx**).

- **RU housing cost**

  It is the **cost** of the **site itself** (i.e., the **land** or **site rights**), the **shelter**, the **tower**, the **power supply**, the **climatization** and the **security**. In **rural areas** it **might** also include the **access road**. This is sometimes called "**site construction**" cost.

  **RU housing** may be self-provided by the **operator** with its **own infrastructure** (mainly **CapEx**) and/or hiring **service** from a **tower company** (mainly **OpEx**).

For a **cost estimate** of the **RU** and its **connectivity** click 4.7.8.4.

#### 4.7.8.4 NR node sample cost ranges

The following table provides a **cost range** for **each** of the different **elements** that compose an **NR node**:

| Cost item | Unitary cost [€] | |
|---|---|---|
| | From | To |
| **Central Unit** | 30,000 € | 150,000 € |
| **Distributed Unit** | 15,000 € | 70,000 € |
| **Midhaul connectivity** | 1,000 € | 5,000 € |
| **Radio Unit** | 5,000 € | 25,000 € |
| **Fronthaul connectivity** | 2,000 € | 10,000 € |
| **RU housing** | 25,000 € | 100,000 € |

For a **NR node** that serves a **macro cell** in a **rural area**, the **CU**, the **DU** and the **RU** are **all** at the **same site**. For this case, the **most expensive** element is the **site construction**, that requires:

- **Site** acquisition.
- Access **road** to the site
- **Shelter**, including fencing, cameras and climatization.
- **Either** building a connection to **power network** or installing a **local power** system.
- **Tower**.

The **site construction cost** may **vary largely** depending on **country**, site **location** and many **other factors**. A **rough cost range** for **EU countries** is between **80,000€** and **150,000€** per site. A rough idea of the **cost distribution** of the **site construction** is:

- **Pre-development: 15%** to **25%**

  Site acquisition, zoning, permitting, structural analysis, and other similar costs.
- **Construction elements: 20%** to **30%**

  Steel, concrete, bricks, fences, climatization, cameras, power, and other similar costs.
- **Construction works**: **50%** to **60%**

  Access road, shelter construction, tower foundation, and other similar costs.

## 4.8 Labels in NR node figures

The meaning of the **text labels** in several **figures** contained in this chapter is:

- **Components** labeled with a **super index** are **optional**.
- "**FO**" stands for **optical fiber**, which is one way to **connect** a **DU** with **its RUs**.
- "**BBU**" stands for **Baseband Unit** (click 4.7.4). It is a **subsystem** frequently used in **LTE**, and **not** so much in **5G**.
- "**DFT**" stands for **discrete Fourier transform**.
- "i**DFT**" stands for **inverse discrete Fourier transform**.

- The **suffixes** "***(d)***", "***(f)***", "***(ta)***" and "***(td)***" stand for the **digital domain**, the **frequency domain** and the **time domain** with **analog** or **digital** representation (respectively).
- "***Trp. blck.***" stands for **transport block**.
- "***T/R***" represents in **Tx** (**re**)**transmitted rate-matched block**, and in **Rx** a **correct** or **erroneous encoded block** (i.e., having **one** or **more erroneous bits**).
- "***Baseband signal***" represents the **baseband signal** (**either** in **Tx** or **Rx**). This **baseband signal** contains the **user data** and **also all** the **control data** and **signals**.
- "***Passband signal***" represents the **passband signal** (**either** in **Tx** or **Rx**).
- "***Baseband streams***" represents several **baseband streams** (**either** in **Tx** or **Rx**).
- "***Passband streams***" represents several **passband streams** (**either** in **Tx** or **Rx**).
- "**CU**" stands for **Central Unit** (click 4.7.1). It is one of the possible **subsystems** of a **NR node**.
- "**DU**" stands for **Distributed Unit** (click 4.7.2). It is one of the possible **subsystems** of a **NR node**.
- "**RU**" stands for **Radio Unit** (click 4.7.3). It is one of the possible **subsystems** of a **NR node**.

## 4.9 User Equipment (UE)

A **User Equipment** (**UE**) is a terminal **connected** to the **5G network** through the **air interface**. To **communicate** with the **NR node**, **each UE** has to do basically the **same processing** tasks ((de)coding, (de)modulation, **MIMO**, scrambling, **HARQ**, …) that the **NR node** has to do to **communicate** with the **UE**.

However, the **NR node must** be capable of interacting with **many simultaneous UEs** and with **very different types** of **UEs**. The NR node is also responsible for coordination with **neighboring NR nodes** (e.g., handover or interference management) and for the **configuration** of **each UE** (click 5).

**5G** was specifically designed to provide a **very diverse range** of **services**, and consequently, it has to support **UEs** with **very different capabilities** and **requirements**.

Some **relevant characteristics** of **UEs** are:

- **Battery powered**

  **Many UEs** are **battery powered**, and **battery duration** is an important feature for the users. Consequently, the **design** and **implementation** of **battery powered UEs** has to emphasize **reduced power consumption**. In particular, **IoT UEs** may have **very strict battery limitations**.

- **Small size**

  **Many UEs** are **small sized**. This imposes **limitations** on the **UE design** and **implementation**, such as **antenna types**, **quantity** of **antennas**, **antenna separation**, **RF equipment** or **processing capacity**. In particular, **IoT UEs** may have **very strict size limitations**.

- **Low cost**

  **Smartphones** and **CPEs** targeted to the **low-end market segment** are **UEs** with considerable **cost limitations**. This usually results in **lower quality components** and **low UE capabilities** (click 4.9.1). **IoT UE** most frequently have **extremely demanding cost limitations**, because of the type of **applications** to which they are targeted, and their possible **short life cycle**.

Conversely, **UEs** may have **very diverse communication requirements**, such as:

- **Latency**

  **UEs** targeted to **hard real-time applications** may have **very specific latency requirements**. **Industry robots**, **computer-controlled drones**, **driver-less cars**, **AGVs** and **robotic dogs** are a few examples of **UEs** that may have **low latency requirements** (e.g., a few milliseconds).

- **Reliability**

  **UEs** that involve **risk** to **human lives** and/or **high economic impact** may have **very specific reliability requirements**. **Industry robots**, **computer-controlled drones**, **driver-less cars**, **AGVs** and **robotic dogs** are a few examples of **UEs** that may have **high reliability requirements** (e.g., no more than 0.001% of 20-byte packets can fail to be delivered within 1 millisecond).

- **Throughput**

  **UEs** targeted to **high-performance applications** may have **very specific throughput requirements**. **5G cameras** for **live TV**, **video games** and **virtual reality** are a few examples of **UEs** that may have **high throughput requirements** (e.g., 1 Gbps).

Other requirements may be **low power** consumption, **high communications density** or **positioning accuracy**.

### 4.9.1 UE capability and categories

**Different UEs** have substantially **different capabilities**, in terms of **modulation order** "$Q_m$" (e.g., up to **64QAM**), **carrier aggregation** (e.g., up to **3 carriers**), **MIMO** (e.g., up to **2 Rx layers**), **buffer size** and many other relevant parameters. **Each UE** informs the **NR node** about its **capability** and the **NR node** manages **each specific UE accordingly**.

**Different elements** of the **UE capability** are **grouped** into so-called "**UE Categories**", where **each UE Category** implies a **fixed set** of values for a certain set of relevant **capability** parameters.

**3GPP** has also defined as specific **class** of **UEs** with **low capacity** that is called "**Reduced capability UEs**" (**RedCap UEs**, click 4.9.3.4).

Regarding **transmitted power**, clause 6 of TS 38.101-**1**, **-2** and **-5** define the **power classes** for the **UEs**. **Each power class** determines a **usage scenario** and specifies **different aspects** of transmission, including **transmitted power**, **EIRP** and **other relevant aspects**.

The following tables show the **power classes** defined in **each frequency range**, and the **maximum transmitted power** per **power class** and **band**:

| FR | Power class | | Band(s) | Max Tx power |
|---|---|---|---|---|
| FR1 Terrestrial | 1 | Fixed wireless access (FWA) UE | n14 | 31 dBm |
| | 1.5 | | n41, n77, n78, n79 | 29 dBm |
| | 2 | Vehicular UE | n34, n39, n40, n41, n77, n78, n79, n95, n97, n98, n104 | 26 dBm |
| | 3 | Handheld UE | n1, n2, n3, n5, n7, n8, n12, n13, n14, n18, n20, n24, n25, n26, n28, n30, n34, n38, n39, n40, n41, n47, n48, n50, n51, n53, n65, n66, n70, n71, n74, n77, n78, n79, n80, n81, n82, n83, n84, n85, n86, n89, n91, n92, n93, n94, n95, n97, n98, n99, n100, n101, n104 | 23 dBm |

| FR | Power class | | Band(s) | Max Tx power |
|---|---|---|---|---|
| FR2 | 1 | Fixed wireless access (FWA) UE | n257, n258, n260, n261, n262 | 35 dBm |
| | | | n263 | 25 dBm |
| | 2 | Vehicular UE | n257, n258, n259, n261, n262, n263 | 23 dBm |
| | 3 | Handheld UE | n257, n258, n259, n260, n261, n262 | 23 dBm |
| | | | n263 | 25 dBm |
| | 4 | High power non-handheld UE | n257, n258, n260, n261, n262 | 23 dBm |
| | 5 | Fixed wireless access (FWA) UE | n257, n258, n259 | 23 dBm |
| | 6 | High Speed Train Roof-Mounted UE | n257, n258, n261 | 23 dBm |

| FR | Power class | | Band(s) | Max Tx power |
|---|---|---|---|---|
| FR1 Sat. | 3 | Handheld UE | n255, n256 | 23 dBm |

**Some relevant UE capability elements**

A few **UE capability elements** that are **relevant** for the **UE**'s **throughput** (click 4.9.3) are **informed** by the **UE** to the **NR node** as follows:

- "$v^{(j)}_{layers}$" is the **maximum quantity** of **layers** (click 2.4.3.5) in the **Downlink** or **Uplink component carrier** "$j$".

  For **Downlink**, "$v^{(j)}_{layers}$", is **informed** with `maxNumberMIMO-LayersPDSCH` from `FeatureSetDownlinkPerCC`. Its **data type** is `MIMO-LayersDL`, which is an `ENUMERATED {twoLayers, fourLayers, eightLayers}`.

  For **Uplink**, "$v^{(j)}_{layers}$", is **informed** with `maxNumberMIMO-LayersCB-PUSCH` and `maxNumberMIMO-LayersNonCB-PUSCH`, **both** from `FeatureSetUplinkPerCC`. Their **data type** is `MIMO-LayersUL`, which is an `ENUMERATED {oneLayer, twoLayers, fourLayers}`.

- "$Q_m^{(j)}$" is the **modulation order** (click 11.2.1) in the **Downlink** or **Uplink component carrier** "$j$".

  For **Downlink**, "$Q_m^{(j)}$" is **informed** with `supportedModulationOrderDL` from `FeatureSetDownlinkPerCC`.

  For **Uplink**, "$Q_m^{(j)}$" is **informed** with `supportedModulationOrderUL` from `FeatureSetUplinkPerCC`.

  Their **data type** is `ModulationOrder`, which is an `ENUMERATED {bpsk-halfpi, bpsk, qpsk, qam16, qam64, qam256}`. These **constants** imply, **respectively**, the values of "$Q_m^{(j)}$" equal to **1**, **2**, **4**, **6** and **8**.

  For **Downlink**, the value `bpsk-halfpi` **cannot** be configured.

  For **Uplink**, the value `bpsk` **cannot** be configured.

  For **Downlink**, and **only** for **FR1**, some **UEs** may **also** support "$Q_m^{(j)}$=**10**" (i.e., **1024QAM**). This is **informed** with `pdsch-1024QAM-FR1-r17` from `BandNR`. Its **data type** is `ENUMERATED {supported}`.

- "$sf^{(j)}$" is the **scaling factor** in the **Downlink** or **Uplink component carrier** "$j$". The **scaling factor** is a **characteristic** of **each UE** that **reduces** its **transmission capacity** to the **fraction indicated** by "$sf^{(j)}$".

  For **Downlink**, "$sf^{(j)}$" is informed with `scalingFactor` from `FeatureSetDownlink` and with `scalingFactor-1024QAM-FR1` (specific for **1024QAM**) from `FeatureSetDownlink-v1700`.

  For **Uplink**, "$sf^{(j)}$" is **informed** with `scalingFactor` from `FeatureSetUplink`.

  Their **data type** is `ENUMERATED {f0p4, f0p75, f0p8}`. **These** 3 constants **represent** (respectively) the values **0.8**, **0.75** and **0.4**. If the correspond parameter is **absent**, the **default value** is **1**.

**All** the **parameters** and **data types** in the **bullet points** above do **not correspond** to **UE parameters** (click 5). They are **rather RRC** status **parameters** (click 5.6.1) that are **sent** by the **UE** to the **NR node**. **All** the **parameters** and **data types** listed above are **descendants** of the `UE-NR-Capability` (click 4.9.2) **data type**.

For the **delivery** of the **UE capabilities** from the **UE** to the **NR node**, click 4.9.2.

In respect to other characteristics, some initial **UEs only** supported **NSA access** and **not SA access** (click 10.15). For example, **UEs** with the **5G** modem **Qualcomm X50** (one of the **very first 5G** modems) **only** supported **NSA**, while the ones with **X55** only supported **NSA**, but could be **upgraded** to **SA**. **UEs** with later **Qualcomm modems** all supported **both NSA** and **SA**.

For more detail on **UE radio capabilities**, refer to TS 38.306.

Source web pages:

- https://www.techplayon.com/5g-nr-ue-power-classes/

## 4.9.2 UE capability delivery

The **UE capabilities** can be sent by the **UE** to the **NR node** within the `UECapabilityInformation` **RRC message**. This **RRC message** belongs to the **RRC**

**UL-DCCH-Message class** (click 5.8.5.9). The `UECapabilityInformation` **RRC message** is **sent** by the **UE** as an **answer** to having **received** from the **NR node** the `UECapabilityEnquiry` **RRC message**.

The **first 5 levels** of the **descendant tree** of `UECapabilityInformation` is the following:

```
                                UECapabilityInformation
  rrc-TransactionIdentifier       RRC-TransactionIdentifier
    criticalExtensions              CHOICE
      ueCapabilityInformation         UECapabilityInformation-IEs
        ue-CapabilityRAT-ContainerList UE-CapabilityRAT-ContainerList
                                         SEQUENCE (SIZE
                                         (0..maxRAT-CapabilityContainers))
                                         OF UE-CapabilityRAT-Container
            rat-Type                       RAT-Type
                                             ENUMERATED
            ue-CapabilityRAT-Container       OCTET STRING
          lateNonCriticalExtension       OCTET STRING
          nonCriticalExtension           SEQUENCE{}
      criticalExtensionsFuture   SEQUENCE{}
```

The corresponding **UE capabilities** are conveyed within **each instance** of `UE-CapabilityRAT-Container`. This **data type** is a `SEQUENCE` of `rat-Type` and `ue-CapabilityRAT-Container`:

- `rat-Type` `RAT-Type`

  It indicates the **type** of **radio access** to which the **UE capabilities contained** in `ue-CapabilityRAT-Container` corresponds. Its **data type** is `RAT-Type`, which is an `ENUMERATED {nr, eutra-nr, eutra, utra-fdd-v1610, ...}`. **Each** of its possible values correspond to **1 type of radio access**.

- `ue-CapabilityRAT-Container` `OCTET STRING`

  It contains the **actual value** of the **corresponding UE capabilities**. **Each instance** of **NR UE capabilities** is **specified** as an **instance** of **data type** `UE-NR-Capability`. **Each instance** of `UE-NR-Capability` is **encoded** using **ASN.1 encoding rules** and conveyed as `OCTET STRING`. Notice that `UE-NR-Capability` does **not** have any **parent data type** in TS 38.331.

As indicated in 4.9.1, there are **many diverse NR UE capabilities**. The following **data types** (**all** of them **descendants** of `UE-NR-Capability`) are **also** used to **specify different UE capabilities**:

- `UE-NR-Capability-v1530`
- `UE-NR-Capability-v1540`
- `UE-NR-Capability-v1550`
- `UE-NR-Capability-v1560`
- `UE-NR-Capability-v1570`
- `UE-NR-Capability-v15c0`
- `UE-NR-Capability-v15g0`
- `UE-NR-Capability-v15j0`

- `UE-NR-Capability-v1610`
- `UE-NR-Capability-v1640`
- `UE-NR-Capability-v1650`
- `UE-NR-Capability-v1690`
- `UE-NR-Capability-v16a0`
- `UE-NR-Capability-v16c0`
- `UE-NR-Capability-v1700`
- `UE-NR-Capability-v1740`

Some relevant **UE capability elements** are described in 4.9.1.

**All** the **parameters** and **data types** in the **bullet points** above do **not correspond** to **UE parameters** (click 5). They are **rather RRC** status **parameters** (click 5.6.1) that are **sent** by the **UE** to the **NR node**. **All** the **parameters** and **data types** listed above are **descendants** of the `UE-NR-Capability` (see above) **data type**.

For more detail on **UE capability transfer**, refer to clause 5.6 of TS 38.331.

### 4.9.3 UE throughput and capacity

Regarding the **UE throughput** (click 11.1.2) and **capacity**, it is possible to **distinguish** the following:

- **UE throughput** under **given conditions** (click 4.9.3.1).
- **UE intrinsic capacity** (click 4.9.3.2).
- **UE-in-cell capacity** (click 4.9.3.3). This is the **UE capacity** provided the **NR node** in a given **cell**.

#### 4.9.3.1 UE throughput under given conditions

The **Downlink** or **Uplink UE throughput** "$Th_{UE}$" (click 11.1.2) under **given conditions** is provided by the following **formula**, obtained from the formula of clause 4.1.2 of TS 38.306:

$$Th_{UE} = \sum_{j=1}^{J} \left( v_{Layers}^{(j)} \cdot \frac{N_{PRB}^{BW(j)} \cdot 12}{T_s^{(j)}} \cdot Q_m^{(j)} \cdot R^{(j)} \cdot sf^{(j)} \cdot TDD^{(j)} \cdot \left(1 - OH^{(j)}\right) \cdot 10^{-6} \right)$$

where its terms are:

- "$Th_{UE}$" is the **Downlink** or **Uplink throughput** of the **UE** in Megabits per second (**Mbps**).
- "$\sum_{j=1}^{J}$" is just **adding** the **throughput** of **all** the **UE**'s **component carriers** (click 10.2). **Each term** within the **summatory** is the **throughput** of **component carrier** "$j$". The value of "$J$" is the **quantity** of **either Downlink** or **Uplink** (as corresponds) **aggregated component carriers**. It can take values from **1** to **16**.
- "$v^{(j)}_{layers}$" is the **quantity** of **layers** (click 2.4.3.5) in the **Downlink** or **Uplink component carrier** "$j$".

  For **each Downlink component carrier**, its value can be **2**, **4** or **8**.

For **each Uplink component carrier**, its value can be **1**, **2** or **4**.

This **first term** of the expression within the summatory provides **all** the **layers** in the **Downlink** or **Uplink component carrier** "$j$".

- "$N_{PRB}^{BW(j)}$•12" is the **quantity** of **subcarriers scheduled** to the **UE** in the **Downlink** or **Uplink component carrier** "$j$". The **term** "$N_{PRB}^{BW(j)}$" is the **quantity** of **scheduled RBs**.

  Its value ranges from **1** the **maximum possible**, which is the **channel transmission bandwidth** "$N_{RB}$ (click 2.2.3.6.1) of **carrier** "$j$". It is multiplied by **12** to obtain the **quantity** of **subcarriers**.

Up to now, the expression within the summatory provides **all** the **subcarriers** in the **Downlink** or **Uplink component carrier** "$j$".

- "$T_s^{(j)}$" is the **average symbol interval duration** (click 2.3.2) in the **Downlink** or **Uplink component carrier** "$j$".

  Its value is "10/14•$2^{-\mu(j)}$" ms for **normal cyclic prefix** (click 2.3.2) and "10/12•$2^{-\mu(j)}$" ms for **extended cyclic prefix**. The term "$\mu(j)$" is the numerology for the **Downlink** or **Uplink component carrier** "$j$".

Up to now, the expression within the summatory provides the **rate** of **resource elements** (**REs**, click 1.1.4) in the **Downlink** or **Uplink component carrier** "$j$".

The joint term "$Q_m^{(j)}$•$R^{(j)}$" is the **modulation** and **coding scheme** (click 11.2.1) in the **Downlink** or **Uplink component carrier** "$j$".

- "$Q_m^{(j)}$" is the **modulation order** (click 11.2.1) in the **Downlink** or **Uplink component carrier** "$j$".

  Its value can be **1**, **2**, **4**, **6** or **8**, corresponding (respectively) to [**π/2-BPSK** or **BPSK**], **QPSK**, **16QAM**, **64QAM** or **256QAM**. For **Downlink** in **FR1** some **UEs** may **also** support "$Q_m^{(j)}$=**10**" (i.e., **1024QAM**).

Up to now, the expression within the summatory provides the **maximum physical layer bitrate**, in the **Downlink** or **Uplink component carrier** "$j$".

- "$R^{(j)}$" is the **actual code rate** (click 11.2.3.1) of the **channel code** (**LDPC** in **PDSCH** and **Polar** in **PUSCH**) after **rate matching** in the **Downlink** or **Uplink component carrier** "$j$".

  Its value can be from around **1/30** up to a **maximum** of 948/1,024=**0.926**.

Up to now, the expression within the summatory provides the **theoretical physical layer bitrate** in the **Downlink** or **Uplink component carrier** "$j$".

- "$sf^{(j)}$" is the **scaling factor** in the **Downlink** or **Uplink component carrier** "$j$". The **scaling factor** is a **characteristic** of **each UE** (typically caused by **implementation limitations**) that **reduces** its **transmission capacity** to the **fraction indicated** by "$sf^{(j)}$".

  Its value can be **1**, **0.8**, **0.75** and **0.4**.

Up to now, the expression within the summatory provides the **actual physical layer bitrate**, in the **Downlink** or **Uplink component carrier** "$j$".

- "$TDD^{(j)}$" is the **fraction** of the **total channel time** that is **scheduled** by **TDD** (click 2.3.4) to the **transmission direction** (**either Downlink** or **Uplink**) that is being **analyzed** in this formula for "$Th_{UE}$" for **component carrier** "$j$".

  Its value is from **0** (when **no symbol intervals** are assigned by **TDD scheduling** to the **transmission direction under study**) to close to **1** (when **all symbol intervals** are assigned by **TDD scheduling** to the **transmission direction under study**).

  If the **component carrier** "$j$" is **FDD**, this **factor** is set to **1**.

Up to now, the expression within the summatory provides the actual **physical layer bitrate** in the **transmission direction** (**either Downlink** or **Uplink**) that is being **analyzed** in this formula for "$Th_{UE}$" for **component carrier** "$j$".

- "$OH^{(j)}$" is the **control overhead** in the **Downlink** or **Uplink component carrier** "$j$". **All** the **other terms** of the expression **jointly** provide the **UE capacity** at the **physical layer** when using **all** the **PRBs** are used for **PDSCH/PUSCH**. However, part of the **PREs** have to be used for **PDCCH/PUCCH** and for **physical layer signals** (**DM-RS**, **PT-RS**, …). Also, different **protocols** introduce **headers** that reduce the **throughput**. **All** of this is modeled by the term "$(1-OH^{(j)})$" that **multiplies all** the **other factors** for **each component carrier** "$j$".

  For a **Downlink component carrier**, its value is **0.14** in **FR1** and **0.18** in **FR2**.

  For an **Uplink component carrier**, its value is **0.08** in **FR1** and **0.10** in **FR2**.

Finally, the expression within the summatory provides the actual **physical layer bitrate**, in the **Downlink** or **Uplink component carrier** "$j$".

- "$10^{-6}$" is just a unit conversion to get the result in **Mbps**.

It is important to highlight that **several factors** of the above formula are **strongly interdependent**:

- A higher **modulation** "$Q_m^{(j)}$" implies a **lower coding rate** "$R^{(j)}$" and **vice versa**. Their **joint value** depends on the **signal to interference ratio** (**SNIR**) and other channel conditions.

  For a given channel conditions, the **NR node** has different algorithms to **optimize** their combined value, which is usually called "**modulation** and **coding scheme**" (**MCS**).

- A **higher channel bandwidth** "$BW^{(j)}=N_{PRB}^{BW(j)}\bullet 12\bullet 2^{\mu}$" implies a **lower modulation** and **coding** "$Q_m^{(j)}\bullet R^{(j)}$".

- A **higher channel bandwidth** "$BW^{(j)}$" implies a **worse** "$SNIR^{(j)}$", because the **received power** "$S$" stays the **same** (recall that "$P_t^{(j)}$" is **limited** by **regulation** and **technical conditions**) while **noise** and **interference grow proportionally** to the **bandwidth increase**. A **worse SNIR** implies a **lower modulation** and **coding** "$Q_m^{(j)}\bullet R^{(j)}$".

  The **net result** of **increasing** the **bandwidth** "$BW^{(j)}$" is **favorable** because the **throughput** of **component carrier** "$j$" **grows proportionally** to the **bandwidth** while it **decreases logarithmically** in respect to "$SNIR^{(j)}$". Recall that Shannon's limit is "$Th^{(j)}<W^{(j)}\bullet log_2(SNIR^{(j)})$".

Some relevant remarks on this "$Th_{UE}$" **formula** are:

- The **UE throughput** obviously depends on the "**given conditions**" indicated at the beginning of this section. These **given conditions** include the **UE**'s **location** in respect to the **antenna site**, its **speed**, its **specific capacities**, the **capacities** offered by the **network** and **other conditions**. **All** these aspects determine the **values** of the **factors** contained in the **expression**. The **expression** can also be used as a **theoretical tool**, getting the **expected result** for a certain **configuration** (provided that the **configuration** is **realistic**).
- For a **cell** operating **SUL**, **only one** among the **UL** and the **SUL carrier** (the one with the **higher throughput**) is counted.
- For **Uplink Tx switching** between **carriers**, **only** the **supported MIMO layer combination across carriers** that results in the **highest combined data** rate is counted for the **carriers** in the supported **maximum Uplink throughput**.
- The **UE conveys** to the **NR node** its **maximum capacity** on **layers** "$v^{(j)}_{layers}$", on **modulation order** "$Q_m^{(j)}$" and on **scaling factor** "$sf^{(j)}$", as it is presented in 4.9.1.

For a description of the **UE maximum throughput**, refer to clause 4.1.2 of TS 38.306.

There is an excellent **on-line throughput calculator** at:

- https://5g-tools.com/5g-nr-throughput-calculator/

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_MaxThroughputEstimation.html

### 4.9.3.2 UE intrinsic capacity

The **UE intrinsic capacity** is the **capacity** (i.e., the **maximum throughput**, click 11.1.2) that **each specific UE** (i.e., a concrete **UE** model) can support in the **best possible conditions**

It is **computed** based on the **formula** for the **UE throughput** under **given conditions** (click 4.9.3.1). The **UE intrinsic capacity** is achieved by the **combination** of **parameters** (among **all** the **combinations supported** by the **UE**) that **maximizes** "$Th_{UE}$" using the said **formula**. This implies **computing** the **said formula** over **all** the **combinations** of supported parameters (i.e., **set** of "$J$" **component carriers** and, for **each component carrier** "$j$" in the **set**, the **quantity** of **layers** "$v^{(j)}_{layers}$", **transmission bandwidth** "$N_{PRB}^{BW(j)}$", **modulation order** "$Q_m^{(j)}$"(click 11.2.1) and **scaling factor** "$sf^{(j)}$".

It is **not possible** to compute it by **maximizing each individual factor** because the values of some **factors** are **interrelated** (click 4.9.3.1) and because **not all** the value **combinations** can be **configured**.

Some **relevant comments** on the different factors of the **expression** within the **summatory** are:

- "$N_{PRB}^{BW(j)} \bullet 12$" in many cases will be the **maximum possible**, which is the **channel transmission bandwidth** "$N_{RB}$ (click 2.2.3.6.1) of the **Downlink** or **Uplink component carrier** "$j$".

- "$T_s^{(j)}$" is determined by the **numerology** of and **cyclic prefix** of the corresponding **Downlink** or **Uplink component carrier** "$j$". Consequently, **every UE** has the **same value** for **each** corresponding **component carrier**.
- "$R^{(j)}$" takes its **maximum value** (**0.926**) in **every case**.
- "$TDD^{(j)}$" takes its **maximum value** (**1**) in **every case**.

For more detail on **UE maximum throughput**, refer to clause 4.1.2 of TS 38.306.

#### 4.9.3.3 UE-in-cell capacity

The **UE-in-cell capacity** is **the UE capacity** in a given cell. There are **several variants** of the **UE-in-cell capacity**, as follows:

- **Maximum UE-in-cell capacity**

  It is the **maximum UE throughput** (i.e., its **intrinsic capacity**, click 4.9.3.2) of any **UE** with **sufficient capabilities** to make the most out of the **service** provided by an **unloaded cell**.

  This **metric** does **not** take into account the **actual UE conditions** (i.e., **location**, **speed**, weather, …) because **UE intrinsic capacity** defined as representing the **best possible conditions**.

- **Average UE-in-cell capacity**

  It is the **UE**'s **average throughput** (click 4.9.3) of any **UE** with **sufficient capabilities** to make the most out of the **service** provided by an **unloaded cell**. Notice that it is **not** the same the "**UE**'s **average throughput**" than the "**average UE**'s **throughput**".

  The **UE**'s **throughput** is subject to a **weighted average** over the **different aspects** that determine its **throughput**. It is typically **averaged** over the **UE**'s **location** within the **cell**, but **more aspects** can also be taken into account: average over **weather conditions**, over **time** of **day**, over **speed**, etc.

  This **metric** takes into account the **average UE conditions** (i.e., **location**, **speed**, **weather**, …) but **not** the **cell**'s **load** (i.e., the existence of **other active UEs** in the **cell**).

- **Peak-hour UE-in-cell capacity**

  It is the **UE throughput averaged** over **all** the **UE**'s in a **cell** during the **cell**'s **daily peak-hour**.

  Notice that this **metric** is **dependent** on the **scheduling** (click 4.11.4.1) **algorithm** used in the **cell**. For example, if the **NR node evenly schedules** the **REs** in **port** (click 4.11.5) among the **UEs** (i.e., **irrespective** of their **location**) this would be **suboptimal**, because the **UE throughput decays faster** than **linearly** with the **UE**'s **distance** to the **antenna site**.

  This **metric** takes into account both the **average UE conditions** and the **cell**'s **load**.

For example, imagine a **macrocell** (click 3.2.3) and a **micro-cell** that have the **same maximum UE-in-cell capacity**. In this case, the **microcell** will have **higher average UE-in-cell capacity**, because it is **averaged** over a **smaller cell range** (assuming we are **only averaging** over **UE location**). Regarding the **peak-hour UE-in-cell capacity**, it

could be **higher** in the **macrocell** if its **traffic** in the **peak-hour** is **much lower** than the one of the **microcell**.

#### 4.9.3.4 UE capacity and spectral efficiency

The **initial design objective** for the **5G technology** was to **achieve** a **UE-in-cell Downlink capacity** (click 4.9.3.3) of **20 Gbps**.

The **initial 5G deployments** (**R15**) did **not** support **spatial multiplexing** (i.e., they were **only 1 layer**) and the **top modulation order** "$Q_m$" was **8 bits/symbol** (using **256QAM**). This results in a **spectral efficiency** (click 11.1.2) around **8 bps/Hz**. Achieving the said **20 Gbps** of **UE Downlink capacity** would require **2.33 GHz** of **Downlink channel** bandwidth, which **5G** does **not support**.

When **spatial multiplexing** with **single-user MIMO** was incorporated in **R16**, it enabled **each UE** a **MIMO** capacity **up to 8 Downlink layers** and **4 Uplink layers** (**4T8R**). **Each Downlink layer** may use a **maximum modulation order** "$Q_m$" of **4 bits/symbol** (using **16QAM**). This results in a **spectral efficiency** around **32 bps/Hz**. Achieving the said **20 Gbps** of **UE Downlink capacity** requires a minimum of **666 MHz** of **Downlink channel bandwidth**, which **5G does support** (click 2.2.3.2) in **FR2 bands**.

The combined usage of **Carrier Aggregation** (click 10.2), **Dual Connectivity** (click 10.3), and **MIMO** (click 4.11.4.8) allows **5G UEs** to achieve **very large capacities** in **real implementations**.

### 4.9.4 Reduced capability (RedCap) UEs

A **reduced capability UE** (**RedCap UE**) is a **UE** with the following characteristics:

- Its **maximum bandwidth** is **20 MHz** in **FR1** and **100 MHz** in **FR2**. It does **not** support **any feature/capability** corresponding to a larger bandwidth than the said ones.
- Its **mandatory** supported **data radio bearers** (**DRBs**, click 4.5.2.1) are **8**.
- Its **mandatory** supported **PDCP SN length** is **12 bits**, with **18 bits** being **optional**.
- Its **mandatory** supported **RLC AM SN length** is **12 bits**, with **18 bits** being **optional**.
- For **FR1**, it supports **1 Downlink space-multiplexing layer** if **1 Rx branch** is supported, and **2 Downlink space-multiplexing layer** if **2 Rx branches** are supported.
- For **FR2**, it supports **either 1 or 2 Downlink space-multiplexing layers** and **2 Rx branches**.
- It does **not** support **any feature/capability** corresponding to **more** than **2 Rx branches**, **nor** to **more** than **2 Downlink space-multiplexing layers**, **nor** to **more** than **1 Tx branch**, **nor** to **more** than **1 Uplink space-multiplexing layer**.
- It does **not** support features/capabilities for **carrier aggregation** (click 10.2), **nor** for **multi-radio dual connectivity** (click 10.3), **nor** for **dual-access protocol stack**, **nor** for **conditional PSCell addition and change**, **nor** for **integrated access and backhaul** (**IAB**, click 10.14). It **supports all other** feature groups or components of the feature groups as captured in TR 38.822, unless indicated otherwise.

For more detail on **RedCap UEs**, refer to clause 4.2.21 of TS 38.306.

## 4.10 NR node and UE protocol architecture

The **protocol architecture** of the **gNB**, the **UE** and **related systems** (e.g., the **Core functions**) may be structured in the following **protocol stacks**:

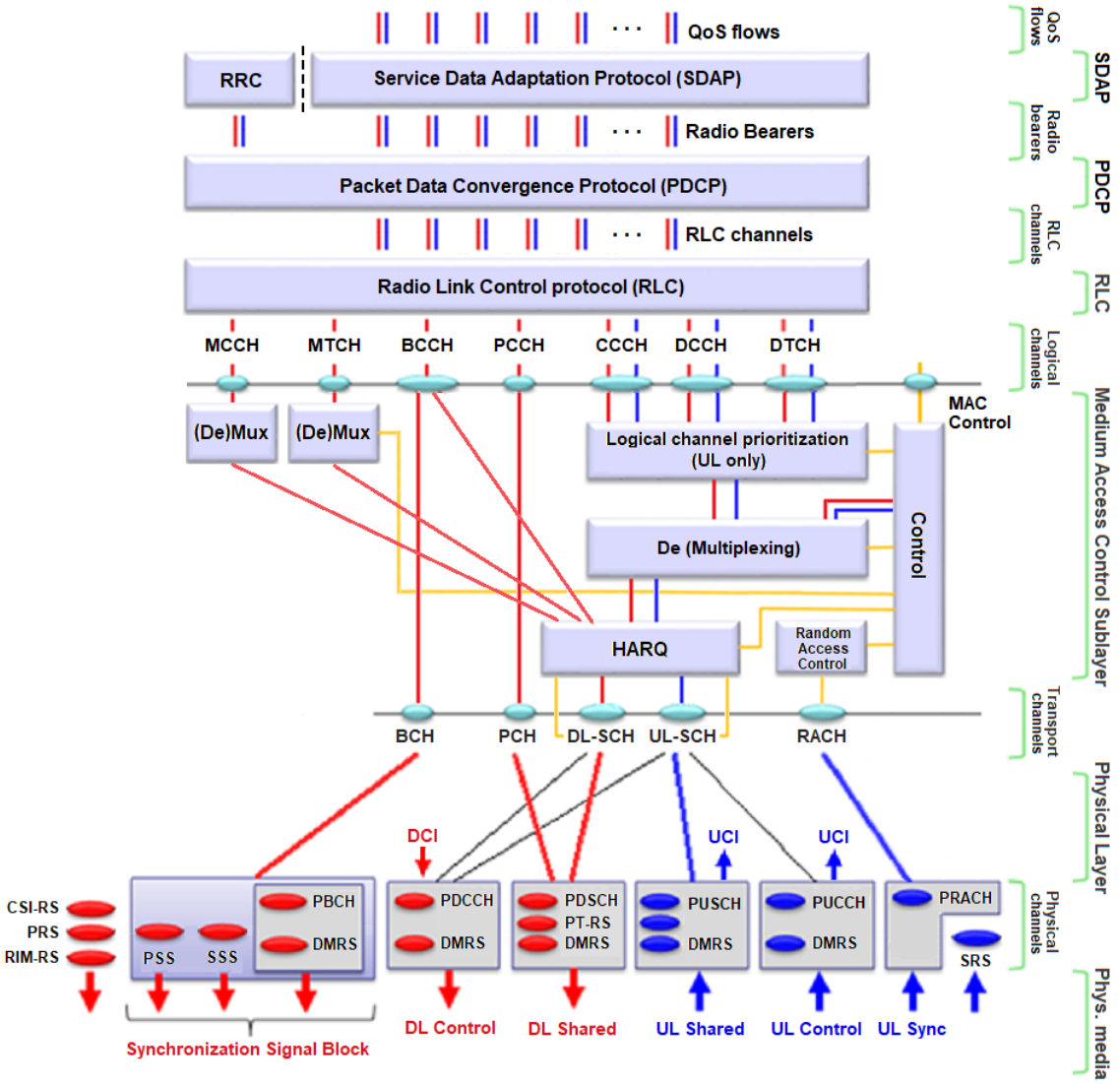
- **Radio access protocol stacks** (click 4.10.1).
- **User-plane protocol stacks** (click 4.10.2).
- **Control-plane protocol stacks** (click 4.10.3).

The **radio access protocol stacks** determine the structure of the **information flow** through them, in terms of **service data units** (**SDUs**), **protocol data units** (**PDUs**), and the corresponding **headers** (click 4.10.4).

For more detail on the **5G protocol architecture**, refer to TS 38.300.

### 4.10.1 NR radio access protocol stacks

The following figure shows the **protocol stack** of the **new radio** (**NR**) **radio access**:



Some **relevant remarks** on this figure are:

- The **radio access protocol stack** is the same at **both ends** (**NR node** and **UE**) of the transmission link, with the **difference** that the **direction** of information flows (**downwards** or **upwards** for **Downlink** or **Uplink**) is **reversed**.
- This figure depicts the **radio access protocol stack** of the **gNB**, because **Downlink** (**red lines**) is **downwards information flow** and **Uplink** (**blue lines**) is **upwards information flow**. In **radio access protocol stack** of a **UE**, the respective **information flows** would be the **opposite**.
- The **RRC protocol** belongs to the **control plane** (click 4.10.3) while the **SDAP protocol** belongs to the **user plane** (click 4.10.2). The **dashed black line** between both represents the **conceptual separation** between the **control plane** and the **user plane**.
- **Black lines** do **not** indicate a **direct mapping** between channels. They **rather** indicate some type of **indirect relation** (for example, **scheduling** or **HARQ response**).
- **Yellow lines** do **not** indicate **information flow**. They **rather** indicate a **control relationship**.
- The **yellow lines** between **HARQ** and **DL-SCH** and **UL-SCH** represent the **HARQ control** for the **retransmission** of **PDSCH codewords** and of **PUSCH codewords**.

For an overview of the **data flow** across the **user-plane protocol stack**, click 4.10.4.

For more detail on the **MAC sublayer**, click 4.5.4.

For more detail on the **binary channels**, click 4.8.

### 4.10.2 User-plane protocol stacks

The **user plane** is the **part** of a **complete protocol architecture** that handles **user data**. The **user plane** and the **control plane** (click 4.10.3) frequently **share** some **protocol entities** at the **lower layers** of the **protocol architecture**.

The **5G user-plane protocol stack** is **access-agnostic** and **packet protocol agnostic**:

- **Radio access agnostic**

  The **5G user-plane protocol stack** is **access-agnostic** because it has been **decoupled** from the **radio access** (**RA**) **protocol stack**. In this way, the **radio access protocol stack** and the **UE-Core protocol stack can evolve independently**. Therefore, the **RA protocol stack** can be **different** in **different access nodes**, and it is **specific** to **each interface** between a **UE** and its **access node**.

- **Packet protocol agnostic**

  The **5G user-plane protocol stack** is **packet protocol agnostic** because it is **decoupled** from the **user-plane packet protocol** used by the **UE**'s **application**. In this way, **each UE**'s **application** may use a different **user-plane packet protocol**, such as **IPv4**, **IPv6**, **both** or **Ethernet**. The **user-plane packet protocol** is called the "**Packet Data Unit** (**PaDU**)" **layer**.

The **different 5G user-plane protocols** and their relationships are now explained.

**Application layer**

The **application layer** is installed in the **UE** and in the **UE**'s **peer system**. The **application layer** therefore interacts **end-to-end**, between the **UE** and its **peer system**. The **UE**'s **peer system** can be for example another **UE**, a **server** or a **middlebox**.

The **application layer** is **not** installed in the **gNB**, **nor** in the **routers/switches** of the **radio access network** (**RAN**) or **transport network**, **nor** in the **Core**'s **user-plane function** (**UPF**), **nor** in the **routers/switches** of the **data network** (**DN**).

In the **UE**, the **application layer** is installed over the **packet data unit** (**PaDU**) **layer**. In the **UE**'s **peer system**, the **application layer** is installed over the **same type** of **PaDU layer** used by the **UE** (unless there is some intermediate protocol conversion).

The **application layer** can include **several protocols**, such as **TCP**, **UDP**, **TLS** and/or **http**.

**Packet data unit** (**PaDU**) **layer**

The **PaDU layer** is installed in the **UE**, in [the **Core**'s **UPF instance** that acts as the **PaDU session anchor** (mobility)] and in the **UE**'s **peer system**. The **PaDU layer** therefore interacts between the **UE** and the [**Core**'s **UPF instance** that acts as the **PaDU session anchor**], between that and the **data network** and between that and the **UE**'s **peer system**.

The **PaDU layer** is **not** installed in the **gNB**, **nor** in the **intermediate routers/switches** of the **radio access network** (**RAN**) or **transport network**, **nor** in the possible **additional Core**'s **UPF instance**.

In the **UE**, the **PaDU layer** is installed over the **SDAP protocol**, that belongs to the **radio access protocol stack** (click 4.10.1). In the [**Core**'s **UPF instance** that acts as the **PaDU session anchor**], the **PaDU layer** is installed over the **GTP-U protocol** on one interface and over the **lower layers** of the interface to the **data network** on the other interface. In the **UE**'s **peer system**, the **PaDU layer** is installed over the **lower layers** of its interface to the **data network**.

The **PaDU Session Type** can be **IPv4**, **IPv6** or **IPv4v6**. Other **PaDU Session Types** can be defined. When the **PaDU Session Type** is **IPv4**, **IPv6** or **IPv4v6**, the **Packet Data Units** are (respectively) **IPv4 packet**, **IPv6 packets** or **both** of them. When the **PaDU Session Type** is **Ethernet**, the **Packet Data Units** are **Ethernet frames**.

**GTP-U protocol**

The **GTP-U protocol** is the **user-plane** of the **GPRS tunneling protocol**. This protocol supports **tunneling** user data over the **N3 interface** (i.e., between the **gNB** and the **UPF**) and over the **N9 interface** (i.e., between **2 UPF instances**, refer to TS 29.281). **GTP-U** shall **encapsulate all end user PaDUs**. **GTP-U** provides encapsulation on a **per PaDU Session** level. **GTP-U** carries the marking associated with a **QoS Flow** (refer to clause 5.7). **GTP-U** is also used over **N4** interface (as defined in TS 29.244).

The **GTP-U protocol** is installed in the **gNB** and in the **UPF instance**(**s**). The **GTP-U protocol** therefore interacts between the **gNB** and the [**Core**'s **UPF instance** that acts as the **PaDU session anchor**], and between that and the possible **additional Core**'s **UPF instance** (if it exists).

The **GTP-U protocol** is **not** installed in the **UE**, **nor** in the **intermediate routers/switches** of the **radio access network** (**RAN**) or **transport network**, **nor** in the **UE**'s **peer system**.

In the **gNB** and in the **UPF instance(s)**, the **GTP-U protocol** is installed over **UDP/IP**. These **UDP/IP protocol entities** are **exclusively** for the **internal** communication between subsystems of the **5G network**, including the **Radio Access Network**, the **Transport Network** and the **functions** of the **Core**. These **UDP/IP protocol entities** have **no relationship**, and do **not interact** with, the possible **UDP/IP protocol entities** used in the **UE**'s **user-plane**.

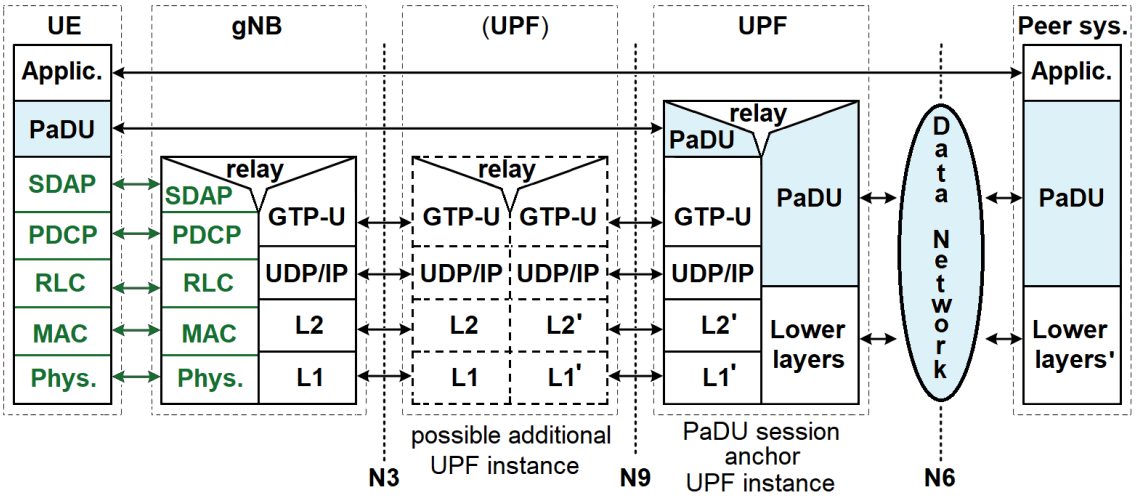
**Intermediate routers/switches**

There can be **intermediate routers/switches** between the **gNB** and the **intermediate UPF instance**, between the **intermediate UPF instance** and the [**UPF instance** that acts as the **PaDU session anchor**] and between the **later UPF instance** and the **Data Network**.

These **routers/switches** belong to the **radio access network** (**RAN**) or to the **transport network**.

**Example of user-plane protocol stacks**

The following figure shows an example of the different **user-plane protocol stacks** that have just been described:




Some **relevant comments** on this figure are.

- This example corresponds to the communication between a **UE** and a **peer system** that is accessible through a **Data Network** (e.g., the internet or a private corporate network). Other service cases (e.g., **LAN-like service** between **UE**'s) have a different configuration, but the **protocol stacks** are basically the same.
- The **protocols** of the **radio access network** have been highlighted with **green** text.
- The **PaDU layer**, including its usage in the **Data Network**, has been highlighted with **light blue** filling.
- **N3**, **N6** and **N9** are the names of the corresponding **interfaces**.

- If there is **no intermediate UPF** (depicted with dashed lines), the interface between the **gNB** and the **UPF** is **N3**.
- The **L1/L2 layers** at **both interfaces** of the **possible additional UPF instance** can be **different**.
- The **lower layers** at **both interfaces** of the **Data Network** can be **different**.
- The possible **intermediate routers/switches** are **not shown** in the figure in order to **focus** on the **specific 5G protocols**.

### 4.10.3 Control-plane protocol stacks

The **control plane** is the **part** of a **complete protocol architecture** that handles **control data**. The **control plane** and the **user plane** (click 4.10.2) frequently **share** some **protocol entities** at the **lower layers** of the **protocol architecture**.

The **5G control-plane protocol stack** is **access-agnostic** because it has been **decoupled** from the **radio access** (**RA**) **protocol stack**. In this way, the **radio access protocol stack** and the **UE-Core protocol stack can evolve independently**. Therefore, the **RA protocol stack** can be **different** in **different access nodes**, and it is **specific** to **each interface** between a **UE** and its **access node**.

The **different 5G control-plane protocols** and their relationships are now explained.

**Non-access stratum** (**NAS**) **protocol**

The **NAS protocol** is installed in the **UE** and in the **Core**'s **access and mobility management function** (**AMF**). The **NAS protocol** therefore interacts **end-to-end**, between the **UE** and the corresponding **AMF instance**.

The **NAS protocol** is **not** installed in the **gNB**, **nor** in the **routers/switches** of the **radio access network** (**RAN**) or **transport network**.

In the **UE**, the **NAS protocol** is installed over the **radio resource control** (**RRC**, click 5.8) **protocol**, that belongs to the **radio access protocol stack** (click 4.10.1). In the **AMF**, the **NAS protocol** is installed over the **NGAP protocol**.

For more detail on **NAS**, click 10.15.

**Next generation application protocol** (**NGAP**)

**NGAP** is installed in the **UE** , in **both interfaces** of the **gNB** and in the **Core**'s **access and mobility management function** (**AMF**) **instance**. **NGAP** therefore interacts between **the** UE and the **gNB** and between the **gNB** and the **Core**'s **AMF instance**.

**NGAP** is **not** installed in the **routers/switches** of the **radio access network** (**RAN**) or **transport network**.

For more detail on **NGAP**, refer to TS 38.413.

**Radio Resource Control** (**RRC**) **protocol**

The **RRC protocol** is the **main signaling protocol** to **configure** and **manage** the **radio access** of **each UE**. The **RRC protocol** belongs to the **radio access protocol stack** (click 4.10.1).

The **RRC protocol** is installed in the **UE** and in the **gNB**. The **RRC protocol** therefore interacts between the **UE** and the **GNB**.

The **RRC protocol** is **not** installed in any **Core**'s **funtion**, **nor** in the **routers/switches** of the **radio access network** (**RAN**) or **transport network**.

**Both** at the **UE** and at the **gNB**, the **RRC protocol** is installed over the **packet data convergence protocol** (**PDCP**, click 4.5.2).

For more detail on **RRC**, click and refer to TS 38.331.

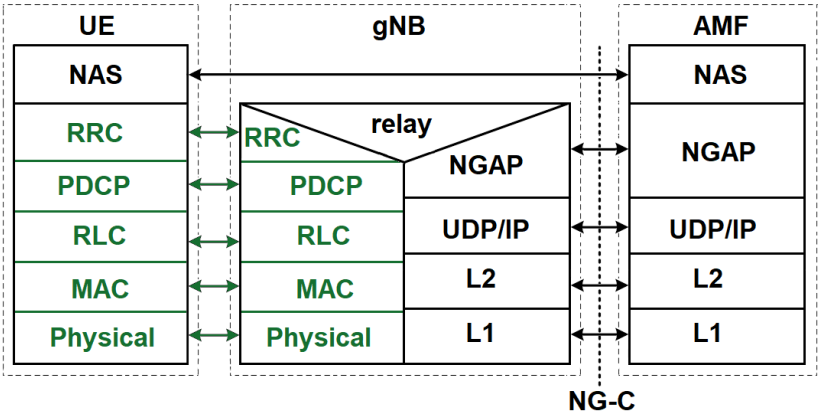
**Intermediate routers/switches**

There can be **intermediate routers/switches** between the **gNB** and the **AMF instance**.

These **routers/switches** belong to the **radio access network** (**RAN**), to the **transport network** or to the **Core**.

**Example of control-plane protocol stacks**

The following figure shows an example of the different **control-plane protocol stacks** that have just been described:

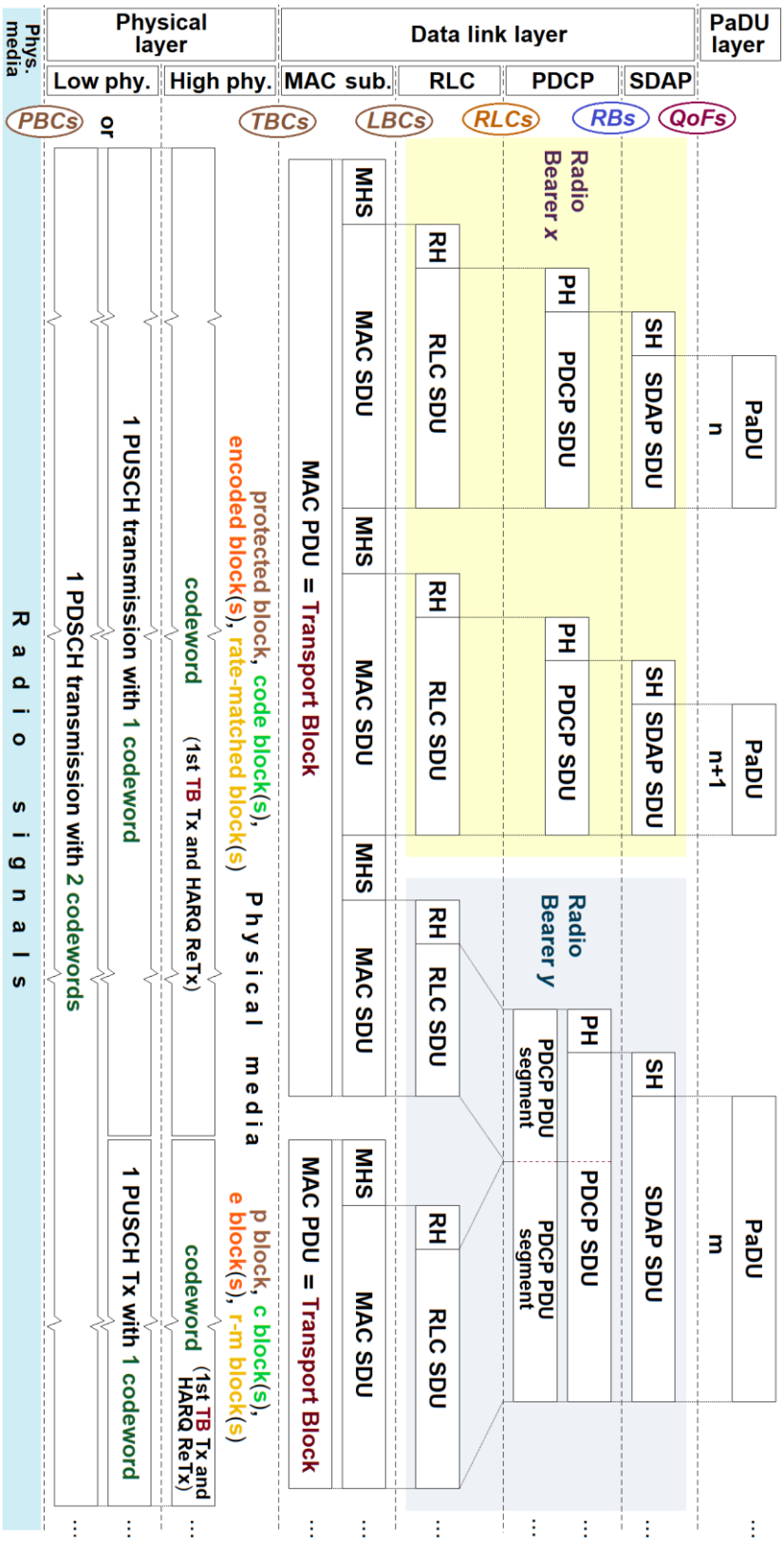



Some **relevant comments** on this figure are.

- This example corresponds to the **NAS** communication between a **UE** and a **Core**'s **AMF instance**.
- The **protocols** of the **radio access network** have been highlighted with **green** text.
- **NG-C** is the **Next Generation control-plane interface**.
- There can be **intermediate routers/switches** between the **gNB** and the **AMF instance**. These **routers/switches** belong to the **radio access network** (**RAN**) or to the **transport network**. These **routers/switches** are **not shown** in the figure in order to **focus** on the **specific 5G protocols**.
- The possible **intermediate routers/switches** are **not shown** in the figure in order to **focus** on the **specific 5G protocols**.

### 4.10.4 Data flow across the user-plane protocol stack

The following figure shows the **data flow** across the different **layers** and **sublayers** of the **user-plane protocol stack**, **either Downlink** or **Uplink** (see **PDSCH/PUSCH** at the bottom of the figure):



Some **relevant remarks** on this figure are:

- The **Packet Data Unit** (**PaDU**) of the **top layer** can be configured to be an **IPv4 packet**, an **IPv6 packet**, **both** of them, an **Ethernet frame** or a **protocol data unit** from some **other protocol**.
- The **PaDU layer** does not belong to the **radio access protocol stack** (click 4.10.1).
- The **3 color ovals** labeled "**QoFs**", "**RBs**" and "**RLCs**" represent (respectively) the "**quality of service flows**", the "**radio bearers**" and the "**RLC channels**".
- The **3 brown ovals** labeled "**LBCs**", "**TBCs**" and "**PBCs**" represent (respectively) the "**logical binary channels**", the "**logical binary channels**" and the "**logical binary channels**".
- **Transmission** is **right** to **left**, while reception is **left** to **right**.
- **Each PDCP SDU contains either** a **SDAP PDU** or an **RRC PDU**, although this figure **only** shows the **SDAP PDU**.
- The **PDCP PDUs** may be **segmented**, as it is shown in the **radio bearer** "***y***".
- **Several RLC PDUs** from the **same** and/or from **different radio bearers** may be **multiplexed** into the **same MAC PDU**, as it is **shown** in the figure.
- The text "**SH**", "**PH**", "**RH**" and "**MHS**" stand (respectively) for "**SDAP header**", "**PDCP header**", "**RLC header**" and "**MAC header** or **subheader**".
- In the "**High physical**" **sublayer**, **each transport block** (click 4.12.1.1) is processed into **1 protected block** (click 4.12.3) which is processed into **1 or more code block(s)** (click 4.12.4). **Each code block** is processed into **1 encoded block** (click 4.12.5) which is processed into **1 rate-matched block** (click 4.12.6). For an overview of this processing, click 4.12.2.
- In the **first transmission** of a **transport block**, **all** its corresponding **rate-matched blocks** are mapped (click 4.12.7) into **1 codeword** (click 4.12.1.2). However, for successive **HARQ retransmissions** (click 4.5.6.6), **each codeword** may contain **only** a **subset** of the said corresponding **rate-matched blocks**.
- **Each codeword** is several times **larger** than the **transport block** that it contains, mainly due to the **FEC redundancy** introduced by the **LDPC coding** (click 4.12.5). This is represented in the figure with **crocked lines** in the **borders** of the **codewords** and of the **PDSCH/PUSH transmissions**.
- The figure shows how the **codewords** can be mapped to **PDSCH transmissions or** to **PUSCH transmissions**.
- In the **example** shown in the figure, **1 PDSCH transmission** contains **2 codewords**, but it is also possible that it contains **only 1 codeword**.
- **1 PUSCH transmission** contains **1** and **only 1 codeword** (as shown in the figure).
- The **Low MAC sublayer** contains the **HARQ protocol** (click 4.5.5.8) and the **random access procedure** (click 9).
- For the **PUSCH-PUCCH multiplexing** of data, click 4.12.8.

For more detail on the **MAC sublayer**, click 4.5.4.

For more detail on the **binary channels**, click 4.8.

For the **NR node functional blocks**, the **NR node physical functions**, the **payload processing** of the **physical binary channels**, the **protocol architecture** of the **New Radio** and the **data flow** through the **protocol architecture**, click 4.3.

### 4.10.5 Non-access stratum (NAS)

The **Non-Access Stratum** (**NAS**) is the set of **protocols** and **messages** between the **User Equipment** (**UE**) and the **Core** functions **AMF** and **SMF** that are **not directly** related to the **radio access**. **AMF** is the **access and mobility management function**. **SMF** is the **session management function**.

**NR** provides **reliable in-sequence** delivery of **NAS messages** over **signaling radio bearers** (**SRBs**, click 4.5.2.1) in **RRC**, **except** at **handover** where **losses** or **duplication** can occur when the **packet data convergence protocol** (**PDCP**) is **re-established**. Within **RRC**, **NAS messages** are sent in **transparent containers**. **Piggybacking** of **NAS messages** in **RRC** can occur in the following scenarios:

- At **bearer establishment/modification/release** in the **Downlink**.
- For **transferring** the **initial NAS message** during **connection setup** and **connection resume** in the **Uplink**.

**Multiple NAS messages** can be sent in a **single Downlink RRC message** during **PDU Session Resource establishment** or **modification**. In this case, the **order** of the **NAS messages** contained **in** the **RRC message** shall be the same as that in the **corresponding NG-AP message** so to **ensure** the **in-sequence** delivery of **NAS messages**.

Notice that in addition to the **integrity protection** and **ciphering** performed by **NAS**, **NAS messages** can **also** be **integrity protected** and **ciphered** by **PDCP**.

## 4.11 NR binary channels and signals

**5G** has **three main sub-layers** of **binary channels**:

- **Logical binary channels** (click 4.11.1)

  They perform different operations (packet **mux/demux**, **HARQ** (click 4.5.5), **Uplink logical channel prioritization** and **random access** control) over **user** data and **control** data.

  They are placed **between** the **radio link control** (**RLC**, click 4.5.2.1) **sublayer** and the **medium access control** (**MAC**, click 4.5.4) **sublayer**. Specifically, between the **Low RLC** (click 4.5.3.2) **functional block** and the **High MAC** (click 4.5.4.1) **functional block**.

- **Transport binary channels** (click 4.11.2)

  They **multiplex/demultiplex** data from/to **logical channels** to/from **physical channels**. The **mux/demux** is done at the **message/packet level**.

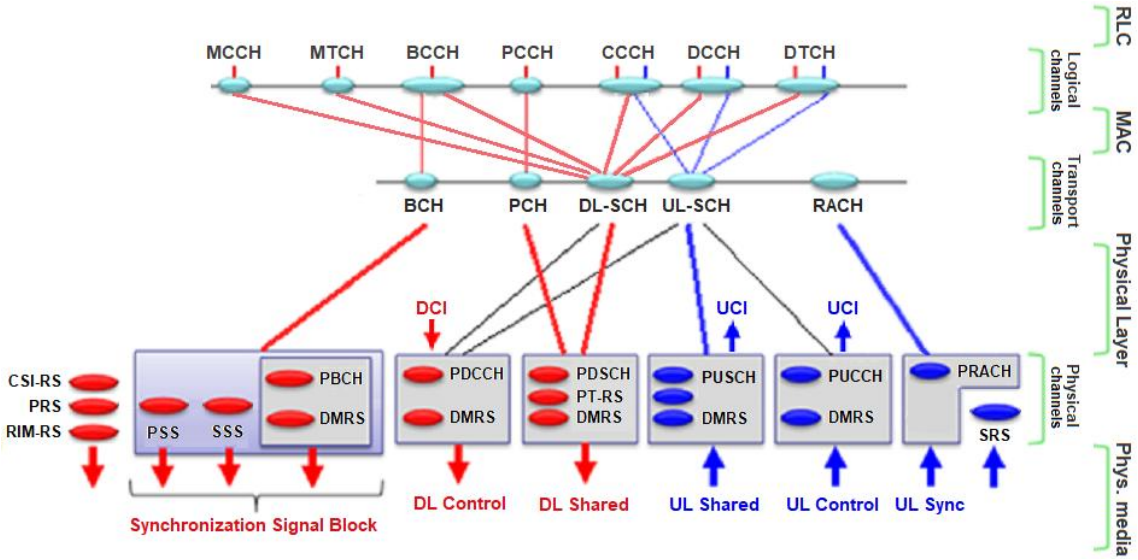
  They are placed **between** the **MAC sublayer** (click 4.5.4) and the **physical layer** (click 4.5.5) Specifically, **between** the **Low MAC** (click 4.5.4.2) **functional block** and the **High physical functional block**.

- **Physical binary channels** (click 4.11.3)

  They are the **lowest level** binary channels and consist of the actual **transmission** in terms of **subcarriers** and **symbols**. They are used to **separate** the **transmission** of different types of data (**signal synchronization**, **low level control**, **scheduling**, **HARQ** (**N**)**ACK**s, **higher layer control**, **user data**, …) over the **air interface**.

  They are placed **between** the **physical layer** (click 4.5.5) and the **physical media**. Specifically, **between** the **Low physical functional block** and the **physical media**.

The **mapping between binary channels** follows the **hierarchy** shown in the following figure:




Some **relevant remarks** on this figure are:

- The **binary channels**, and **binary channel hierarchy**, are the same at **both ends** (**NR node** and **UE**) of the transmission link, just the **direction** of information flows (**downwards** or **upwards** for **Downlink** or **Uplink**) is reversed.
- This figure depicts the **binary channels** at the **NR node**: **Downlink** (**red lines**) is **downwards information flow** and **Uplink** (**blue lines**) is **upwards information flow**.
- **Black lines** do **not** indicate a **direct mapping** between channels. They **rather** indicate some type of **indirect relation** (for example, **scheduling** or **HARQ response**).
- **PDCCH** and **PUCCH** are the two **only** physical channels that **do not** carry data from **any transport** channel. However, they carry **Downlink control information** (**DCI**) and **Uplink control information** (**UCI**) for the physical layer, as indicated in the figure.
- **RACH** is the **only transport** channel that does not carry data from any **logical** channel.

Notice that the figure includes, **in addition** to the **physical channels**, the **physical signals**:

- **DL Primary Synchronization Signal** (**PSS**, click 6.19)

- **DL Secondary Synchronization Signal** (**SSS**, click 6.19)
- **DL Channel-State Information Reference Signal** (**CSI-RS**, click 6.20)
- **DL Positioning Reference Signal** (**PRS**, click 6.21)
- **DL Remote Interference Management Reference Signal** (**RIM-RS**, click 6.22)
- **UL Sounding Reference Signal** (**SRS**).

**DL/UL Demodulation Reference Signals** (**DM-RS**) and **DL/UL Phase Tracking Reference Signal** (**PT-RS**) are also physical signals, but they are considered **part of** their corresponding **physical channel**.

For an **extended figure** that shows the **complete RAN protocol architecture**, click 4.10.1.

For an overview of the **data flow** across the **user-plane protocol stack**, click 4.10.4.

For more detail on the **MAC sublayer**, click 4.5.4.

**LTE** has the same **three main sub-layers** of **binary channels** as **5G**, but the **individual binary** channels **differ**.

Recall from 14.1 that **Sidelink** channels are **not** covered in this book.

Do not confuse these **binary channels** with the **analog radio channels** (click 2.2.3) that contain the **binary channels** and **signals**.

For more detail on **logical** and **transport channels**, refer to clause 4.5 of TS 38.321.

Source web pages:

- https://www.electronics-notes.com/articles/connectivity/5g-mobile-wireless-cellular/data-channels-physical-transport-logical.php

### 4.11.1 Logical binary channels

They are placed **between** the **radio link control** (**RLC**, click 4.5.2.1) **sublayer** and the **medium access control** (**MAC**, click 4.5.4) **sublayer**. Specifically, between the **Low RLC** (click 4.5.3.2) **functional block** and the **High MAC** (click 4.5.4.1) **functional block**.

There are **2 different logical channels**. **Two** are used **only Downlink**, and the **other 3** are the **only** channels used **both Downlink** and **Uplink**

Some of these channels will be familiar names from the **4G LTE** system as the names have been carried over.

Recall from 14.1 that **Sidelink** channels are **not** covered in this book.

For more detail on **logical channels**, refer to clauses 4.5.3 and 4.5.4 of TS 38.321 plus clauses 6.2.2 and 6.2.3 of TS 38.300.

#### 4.11.1.1 BCCH (Broadcast Control Channel) - Downlink

It is used for **broadcasting control** information to the **UEs** within the cell.

It is **broadcast only**. It is **common** to **all** the **UEs**. It is used **Downlink**.

The **system information** transmitted through **BCCH** is structured into **different Information blocks**:

- **Master Information Block** (**MIB**)

  There is **only 1 MIB**, which is **broadcast** over the **BCH transport channel** and then **broadcast** over the **PBCH physical channel**.

- **System Information Blocks** (**SIBs**)

  There are **several system information blocks** (**SIBs**) and positioning system information blocks (**posSIBs**). These are **broadcast** over the (**transport**) **DL-SCH** channel and then **broadcast** over the (**physical**) **PDSCH** channel.

For the **NR RRC messages** associated to **BCCH** when mapped to **BCH**, click 5.8.5.1.

For the **NR RRC messages** associated to **BCCH** when mapped to **DL-SCH**, click 5.8.5.2.

#### 4.11.1.2 Paging Control Channel (PCCH) - Downlink

It is used to **page** the **UEs** whose location at **cell level** is **not known** to the **network**. As a result, the **paging message** needs to be **transmitted** in **multiple cells**.

It is **broadcast-only**. It is **common** to **all** the **UEs**. It is used **Downlink**.

The **PCCH logical channel** is **broadcast** over the **PCH transport channel** and then **broadcast** over the **PDSCH physical channel**.

For the **NR RRC messages** associated to **PCCH**, click 5.8.5.6.

#### 4.11.1.3 Common Control Channel (CCCH) – Downlink and Uplink

It is used for transmitting **control information** to and from the **UEs**. It is used for **initial access**, for those **UEs** that do **not** have a **radio resource control** (**RRC**) connection.

It is **unicast only**. It is **dedicated** to **each UE**. It is used **both Downlink** and **Uplink**.

The **CCCH logical channel** is mapped **Downlink** to the **DL-SCH transport channel** and to the **PDSCH physical channel**.

The **CCCH logical channel** is mapped **Uplink** to the **UL-SCH transport channel** and to the **PUSCH physical channel**.

For the **Downlink NR RRC messages** associated to **CCCH**, click 5.8.5.3.

For the **Uplink NR RRC messages** associated to **CCCH**, click 5.8.5.7 and 5.8.5.8.

#### 4.11.1.4 Dedicated control channel (DL-DCCH) - Downlink and Uplink

It carriers **dedicated control information** between the network and **each specific UE**. It is used by the **UE** and the network **after** a **radio resource control** (**RRC**) connection has been established.

It is **unicast-only**. It is **dedicated** to **each UE**. It is used **both Downlink** and **Uplink**.

The **DCCH logical channel** is mapped **Downlink** to the **DL-SCH transport channel** and to the **PDSCH physical channel**.

The **DCCH logical channel** is mapped **Uplink** to the **UL-SCH transport channel** and to the **PUSCH physical channel**.

For the **Downlink NR RRC messages** associated to **DCCH**, click 5.8.5.4.

For the **Uplink NR RRC messages** associated to **DCCH**, click 5.8.5.9.

#### 4.11.1.5 Dedicated traffic channel (DTCH) - Downlink and Uplink

It is used for **carrying Downlink** and **Uplink user information** between the **network** and **each specific UE**.

It is **unicast-only**. It is **dedicated** to **each UE**. It is used **both Downlink** and **Uplink**.

The **DTCH logical channel** is mapped **Downlink** to the **DL-SCH transport channel** and to the **PDSCH physical channel**.

The **DTCH logical channel** is mapped **Uplink** to the **UL-SCH transport channel** and to the **PUSCH physical channel**.

#### 4.11.1.6 Multicast and broadcast service (MBS) control channel (MCCH) - Downlink

It is used to **carry control information** related to **MBS sessions**, such as:

- **Scheduling information**

  **When** and **where MBS** data will be transmitted. **Frequency-time resources** for **MTCH** (**Multicast Traffic Channel**) and **session configuration information**.

- **Session configuration information**

  **Identifiers** for **MBS sessions**. **Parameters required** to **receive** and **decode MBS content**.

- **Security and Access Parameters**

  **Keys** or **references**, when **access control** is used.

It is **broadcast** or **multicast**. It is **common** to **all** the **UEs** (**broadcast**) or to a given **group** of **UEs** (**multicast**). It is used **Downlink**.

The **MCCH logical channel** is **broadcast/multicast** over the **DL-SCH transport channel** and **broadcast/multicast** over the **PDSCH physical channel**.

For the **NR RRC messages** associated to **MCCH**, click 5.8.5.5.

#### 4.11.1.7 Multicast and broadcast service (MBS) traffic channel (MTCH) - Downlink

It is used to **carry multicast** or **broadcast user data**.

It is **broadcast** or **multicast**. It is **common** to **all** the **UEs** (**broadcast**) or to a given **group** of **UEs** (**multicast**). It is used **Downlink**.

The **MTCH logical channel** is **broadcast/multicast** over the **DL-SCH transport channel** and **broadcast/multicast** over the **PDSCH physical channel**.

### 4.11.2 Transport binary channels

They **multiplex/demultiplex** data from/to **logical channels** to/from **physical channels**. The **mux/demux** is done at the **message/packet level**.

They are placed **between** the **MAC sublayer** (click 4.5.4) and the **physical layer** (click 4.5.5) Specifically, **between** the **Low MAC** (click 4.5.4.2) **functional block** and the **High physical functional block**.

There are **3 Downlink transport binary channels** and **2 Uplink transport binary channels**.

- **BCH (Broadcast Channel) – Downlink**

  It is used **only** for **broadcasting** the **MIB**, received from the (**logical**) **BCCH** channel, over the (**physical**) **PBCH** channel. It is **broadcast-only**.

- **PCH (Paging Channel) – Downlink**

  It is used for **broadcasting paging** information received from the (**logical**) **PCCH** channel over the (**physical**) **PDSCH** channel. The **PCH** supports discontinuous reception, **DRX**, to enable the **UE** to **save battery power** by waking up at a specific time to receive the **PCH**. It is **broadcast-only**.

  In order that the **PCH** is received by **all UEs** in the **cell**, the **PCH must** be **broadcast** over the **entire cell** as a **single message**. When the **NR node** uses **Tx beam sweeping** (click 6.1.2.3), the previous process is done using **several** different **PCH instances**.

- **DL-SCH (Downlink Shared Channel) - Downlink**

  It is the main **transport channel** used for transmitting **Downlink control** and **user data**. It carries information **from/to all** the **logical** channels carrying it **to/from** the (**physical**) **PDSCH** channel. The **DL-SCH** is also used for transmitting some parts of the **BCCH system information**, specifically the **SIB**. **Each UE** has a **DL-SCH** for **each** cell it is connected to.

  It is **broadcast**, **multicast** and **unicast**. It supports **all** the key **5G NR features**, including dynamic rate adaptation, HARQ, channel aware scheduling, and space-division multiplexing.

- **UL-SCH (Uplink Shared Channel) - Uplink**

  It is the **Uplink counterpart** to the **DL-SCH** that is, the **Uplink transport channel** used for transmission of **Uplink data**.

- **RACH (Random Access Channel) - Uplink**

  It carries the **RA preamble sequence index** (**within** the **RA preamble**, **encoded** as the **RA preamble sequence**) which is for the **random access procedure** (click 9).

Recall from 14.1 that **Sidelink** channels are **not** covered in this book.

For more detail on **transport channels**, refer to clauses 4.5.2 and 4.5.4 of TS 38.321 plus clause 5.5 and 6.2.3 of TS 38.300.

### 4.11.3 Physical binary channels and signals

They are the **lowest level** binary channels and consist of the actual **transmission** in terms of **subcarriers** and **symbols**. They are used to **separate** the **transmission** of different types of data (**signal synchronization**, **low level control**, **scheduling**, **HARQ** (**N**)**ACK**s, **higher layer control**, **user data**, …) over the **air interface**.

They are placed **between** the **physical layer** (click 4.5.5) and the **physical media**. Specifically, **between** the **Low physical functional block** and the **physical media**.

Most **physical binary channels** have **transport** and/or **logical ch**annels mapped onto them to carry higher sublayer data. Additionally, the physical channels carry payload data

or control data about data transmission characteristics like modulation, reference signal multiplexing, transmit power, RF resources, etc.

The **5G physical binary channels** resemble those of **LTE**, **removing PHICH** and **PCIICH**. Also, the behavior of **PDCCH** in **5G** is administered by **layer 3** procedures.

In **5G**, there are **6 physical channels**: **3 Downlink** and **3 Uplink**.

The **3 Downlink physical channels are**:

- **Physical Broadcast Channel** (**PBCH**):

  It is a **broadcast-only** channel. It carries information from the (**transport**) **Broadcast Channel** (**BCH**) and from the (**logical**) **broadcast control channel** (**BCCH**). The **PBCH** is sent, together with the **Primary** and **Secondary Synchronization Signals** (**PSS**/**SSS**), within the **SSB**.

- **Physical Downlink Control Channel** (**PDCCH**)

  It is a **broadcast-only** channel. It carriers the **Downlink Control Information** (**DCI**). It does **not** carry information from any **transport channel** (**nor** from any **logical channel**).

- **Physical Downlink Shared Channel** (**PDSCH**)

  It is a **broadcast**, **multicast** and **unicast** channel. It carries information from the (**transport**) **Downlink Shared Channel** (**DL-SCH**) and from the (**transport**) **Paging Channel** (**PCH**). It also carries information from **all** the **logical channels**.

The **3 Uplink physical channels are**:

- **Physical Random Access Channel** (**PRACH**)

  It is a **unicast-only** channel. It carries information from the (**transport**) **Random Access Channel** (**RACH**) and from **no logical channel**.

- **Physical Uplink Control Channel** (**PUCCH**)

  It is a **unicast-only** channel. It carries the **Uplink Control Information** (**UCI**). It does **not** carry information from any **transport channel** (**nor** from any **logical channel**).

- **Physical Uplink Shared Channel** (**PUSCH**)

  It is a **unicast-only** channel. It carries information from the (**transport**) **Uplink Shared Channel** (**UL-SCH**) and from the (**logical**) **common control channel** (**CCCH**), from the (**logical**) **dedicated control channel** (**DCCH**) and from the (**logical**) **dedicated traffic channel** (**DTCH**). It **may** also carry **Uplink Control Information** (**UCI**).

There are **7 channel-specific reference signals**, **4 Downlink** and **3 Uplink**.

The **4 Downlink channel-specific reference signals** are:

- **PBCH** Demodulation Reference Signal (**PBCH DM-RS**).
- **PDCCH** Demodulation Reference Signal (**PDCCH DM-RS**).
- **PDSCH** Demodulation Reference Signal (**PDSCH DM-RS**).
- **PDSCH** Phase-Tracking Reference Signal (**PDSCH PT-RS**).

The **3 Uplink channel-specific reference signals** are:

- **PUCCH** Demodulation Reference Signal (**PUCCH DM-RS**).
- **PUSCH** Demodulation Reference Signal (**PUSCH DM-RS**).
- **PUSCH** Phase-Tracking Reference Signal (**PUSCH PT-RS**).

Notice that **all** physical **channels**, except **PRACH**, contain (in addition to the **channel payload**) **channel-specific Demodulation Reference Signals** (**DM-RS**, click 4.11.3.2.1).The **2 shared physical channels** (**PDSCH** and **PUSCH**) contain **also Phase-Tracking Reference Signals** (**PT-RS**, click 4.11.3.2.2).

There are **6 standalone physical signals**: **5 Downlink** and **1 Uplink**.

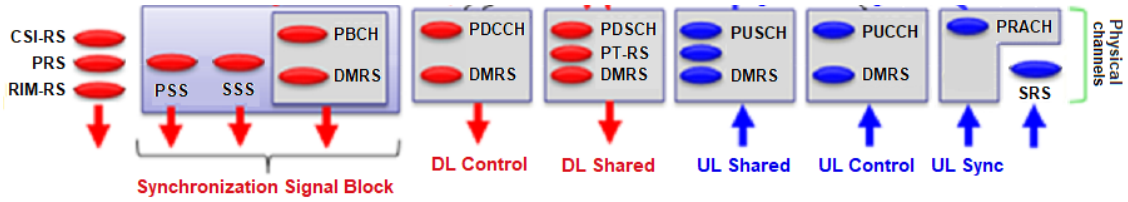
The **5 Downlink standalone physical signals** are:

- Primary Synchronization Signal (**PSS**).
- Secondary Synchronization Signal (**SSS**).
- Channel-State Information Reference Signal (**CSI-RS**).
- Positioning Reference Signal (**PRS**).
- Remote Interference Management Reference Signal (**RIM-RS**).

The **1 Uplink standalone physical signal** is:

- Sounding Reference Signal (**SRS**).

The following figure summarizes the **physical channels** and **signals**:




Click 1.1.4 for a **clarifying figure** that shows an **approximate** bidimensional (in **frequency** and **time**) representation of a **resource grid** (click 2.2.5.6) of the **Downlink currently active** (click 2.2.7.4) **bandwidth part** (click 2.2.7), for the duration of **1 frame** (click 2.3.1), indicating with colors how the **different RBs** (click 2.2.5.7) have been **scheduled** (click 4.11.4) to the **different physical binary channels** (click 4.11.3).

Recall from 14.1 that **Sidelink** channels are **not** covered in this book.

### 4.11.3.1 Summary of characteristics of the 6 physical channels

The following table summarizes some relevant characteristics of the **6 Physical Channels**:

| | **Physical Channels** | | | | | |
|---|---|---|---|---|---|---|
| | **PBCH** | **PDCCH** | **PDSCH** | **PUCCH** | **PUSCH** | **PRACH** |
| **Transm. direction** | Downlink | Downlink | Downlink | Uplink | Uplink | Uplink |
| **Main purposes** | Master Information Block (MIB) | Scheduling of Shared Channels | Data and Control | (N)ACK, Scheduling Requests, CSI | Data and Control | Uplink sync. and C-RNTI. |

| | Physical Channels | | | | | |
|---|---|---|---|---|---|---|
| | **PBCH** | **PDCCH** | **PDSCH** | **PUCCH** | **PUSCH** | **PRACH** |
| **Carried Information** | MIB, PBCH SI | Phys. Downlink Control Information (DCI) | User data and higher layer control (e.g. SIB1, other SIBs, RRC, …) | Phys. Uplink Control Information (UCI) | User data, higher layer control and non-SR UCI | RA seq. index, SSB index and RA-RNTI |
| **Subcarrier spacings in FR1 [kHz]** | 15, 30, ~~60~~ | 15, 30, 60 | 15, 30, 60 | 15, 30, 60 | 15, 30, 60 | ***1.25***, ***5***, 15, 30, ~~60~~ |
| **Subcarrier spacings in FR2 [kHz]** | ~~60~~, 120, 240, 480, 960 | 60, 120, ~~240~~, 480, 960 | 60, 120, ~~240~~, 480, 960 | 60, 120, ~~240~~, 480, 960 | 60, 120, ~~240~~, 480, 960 | 60, 120, ~~240~~, 480, 960 |
| **Quantity of Resource Blocks (or of subcarriers)** | 12 RBs (sent with PSS/SSS as SSB) | CORESETs (COR#0 is 24, 48 or 96 RBs) | Scheduled by DCI F1 or F4 messages | Depends on `format` and on UCI length | Scheduled by DCI F0 messages | Same quantity of SCs as the sequence length |
| **CP duration [$T_c$] and [µs]** | normal-short: $144\kappa \bullet 2^{-\mu}$; normal-long: $160\kappa \bullet 2^{-\mu}$<br>normal-short: $4.69 \bullet 2^{-\mu}$; normal-long: $5.21 \bullet 2^{-\mu}$; | | | | | Various formats |
| | extended: $512\kappa \bullet 2^{-\mu}$ [$T_c$] ; extended: $16{,}7 \bullet 2^{-\mu}$ [µs] | | | | | Specific CP |
| **Symbol duration [$T_c$] and [µs]** | $2{,}048\kappa \bullet 2^{-\mu}$ [$T_c$]<br>$66.67 \bullet 2^{-\mu}$ [µs] | | | | | Specific |
| **Frequency and symbol location** | Variable | | | | | |
| **Sending Period** | 5, 10, **20**, (40, 80, 160) ms | When needed | Scheduled | When needed | **Data**: scheduled<br>**UCI**: when needed | Triggered |
| **Modulation(s)** | QPSK | QPSK | QPSK, QAM of 16, 64, 256. Also QAM of 1,024 in FR1. | `f0`: no modulat.<br>`f1`: QPSK or BPSK<br>`f2`: QPSK<br>`f3` / `f4`: QPSK or π/2-BPSK | **Data**: π/2-BPSK, QPSK, QAM of 16, 64 and 256<br>**UCI**: QPSK | Zadoff-Chu sequence (no modulation) |
| **Coding(s) {*N*=UCI message bitlength [bits]}** | CRC plus polar code | CRC plus polar code | CRC plus LDPC | CRC plus polar code | **Data**: CRC plus LDPC<br>**UCI**: CRC plus: polar code when {$N \geq 12$} or block code when {$N < 12$} | None |
| **HARQ** | No | No | Yes | No | Yes | No |
| **Freq. hopping** | No | | | Yes | No | |
| **Transport Channel(s) carried** | BCH | None | DL-SCH, PCH | None | UL-SCH | RACH |
| **Logical Channel(s) carried** | BCCH | None | All | None | CCH, DCCH, DTCH | None |
| **Contains DM-RS** | No | Yes | Yes | Yes (except `format0`) | Yes | No |
| **Contains PT-RS** | No | No | Yes | No | Yes | No |
| **Antenna Port(s)** | 4000 | 2xxx | 1xxx | 2xxx | 1xxx | 4xxx |
| **Clause of TS 38.211** | 7.3.3 | 7.3.2 | 7.3.1 | 6.3.2 | 6.3.1 | 6.3.3 |
| **Click section** | 6.2 | 6.7 | 6.13 | 7.5 | 7.10 | 7.1 |

Some **relevant remarks** on this table are:

- **Row "Carried information"**

  For **PUSCH**, the text "Non-SR UCI" means **UCI messages** that d**o not** contain **Scheduling Requests**.

For **PRACH**, the **carried information** is:

- The **index** of the **RA preamble sequence** (click 7.3.1).
- The value of the **ssb-time-domain-index** of the **beam** that has been **selected** (click 6.1.2.3) by the **UE**. This is **only** in case that the **NR node** performs **Tx beam sweeping** (click 6.1.2.3).
- The **RA-RNTI** (click 9.1.7), except for the **2-step CFRA procedure**.
- The text "Non-SR UCI" means **UCI messages** that d**o not** contain **Scheduling Requests**.

- **Row "Quantity of Resource Blocks (or of subcarriers)"**

  The text "COR#0" stands for **CORESET#0**. The texts "F0", "F1" and "F4" stand (respectively) for "Formats 0_x", "Formats 1_y" and "Formats 4_z" of DCI messages.

- **Rows "CP duration"** and **"Symbol duration"**

  The symbol "$\kappa$" is "$T_c/T_s$=64".

- **Row "Sending period"**

  The sending period of **20 ms** is highlighted in **bold** font, because it is the most usual configuration. The sending periods of 40, 80 and 160 ms are listed between parenthesis because they only apply to special low-power cells. For more detail, click 6.1.2.1.3.

- **Rows "Subcarrier spacing…"**

  The **values** in ~~slashed grey~~ font **cannot** be used: the reason to list these values in this way is to **highlight** that they **cannot** be used. The **subcarrier spacing values** in ***bold italic*** font can **only** be used for **PRACH**: the reason to list these values in this way is to **highlight** that they can **only** be used for **PRACH**.

- **Row "Modulation(s)"**

  In the cell for **PUCCH Modulation(s)**:

  - The terms "`f0`", "`f1`" and "`f2`" stand for the **PUCCH** (respectively) "`format0`", "`format1`" and "`format2`". The term "`f3`/`f4`" stands for **PUCCH** "`format3`" and "`format4`".
  - Format **F1** uses **BPSK** when sending **1 bit** and **QPSK** when sending **2 bits**.
  - The case of **π/2-BPSK** only applies if **transform-precoding** (click 4.12.12) is enabled.

- **Row "Quantity of Resource Blocks (or of subcarriers)"**

  The text "DCI F0" and "DCI F1/4" refer, respectively, to the corresponding **DCI** formats (i.e., "**Formats 0_x**" and "**Formats 1_y and 4_z**").

- **Row "Modulation(s)"** for **PUSCH** and **PDSCH**

  **PDSCH** and **PUSCH** are used for transferring **user data** and **higher layer control**. This is why these **2 shared channels** require the **greatest range** of **modulation schemes**. A **high modulation order** "$Q_m$" (click 11.2.1) is required to achieve a **high throughput** while in **good coverage**, while a **low modulation**

**order** is required to **maximize cell range** (click 3.10.1.1) or for **bad channel conditions**.

**Both PDSCH** and **PUSCH** support **QPSK**, **16QAM**, **64QAM** and **256QAM**.

**PDSCH** additionally supports **BPSK**, which is used to **maximize Downlink coverage** and **1024QAM** (**only** in **FR1**) which is used to **maximize Downlink throughput**.

**PUSCH** additionally supports **π/2-BPSK**, which is used to **maximize Uplink coverage**. Notice that **π/2-BPSK** is **only** permitted when the **PUSCH** is using the **DFT-s-OFDM** (click 11.16) waveform, which is also designed to **boost Uplink coverage**. **π/2-BPSK** offers lower a **lower Peak to Average Power Ratio** (**PAPR**) than **BPSK**. **BPSK** suffers from **zero crossings** every time the bitstream changes from **0** to **1**, or from **1** to **0**. These **zero crossings** are caused by the **180-deg phase change** when moving from one modulation constellation to the other. **Zero crossings** cause the **signal envelope** to pass through zero and consequently results in a **high PAPR**.

- **Row** "**Modulation**(**s**)" for **PUCCH**

  **PUCCH** uses different modulation schemes depending on the **PUCCH** `format`. In general, lower order modulation schemes are used by **PUCCH** to help ensure reliable reception across the whole coverage area:

  - **PUCCH** `format0` does **not** use a **modulation scheme**.
  - **PUCCH** `format1` can use either **BPSK** or **QPSK** depending on the quantity of information to be transferred. **BPSK** is used when **PUCCH** `format1` transfers a single bit of information, whereas **QPSK** is used when **PUCCH** `format1` transfers two bits of information.
  - **PUCCH** `format2` always uses **QPSK**.
  - **PUCCH** `format3` and `format4` can use either **π/2-BPSK** or **QPSK** depending on the configuration from the NR node.

- **Row** "**Modulation**(**s**)" for **PRACH**

  **PRACH** does not use a **modulation** scheme. For more detail on **Zadoff-Chu sequences**, click 11.20.4.

- **Row** "**Modulation**(**s**)" for **PBCH** and **PDCCH**

  **PBCH** and **PDCCH** **both** use **QPSK** to help ensure reliable reception across the whole coverage area. **PBCH** also uses a **low code rate** to further help reliable reception.

- **Row** "**Coding**(**s**)…"

  In the cell for **PUSCH**, the symbol "$N$" is the same as "$N_{bit}^{UCI}$" (click 7.8.2.1), which is the **bitlength** "$N$" of the **UCI message**. The cell indicates **2 cases** for "{$N \geq 12$}" and for "{$N < 12$}". The case of "{$N < 12$}" is divided into **3 subcases**:

  - "$N=1$": **repetition coding** (click 7.8.2.3.2 and refer to clause 5.3.3.1 of TS 38.212).
  - "$N=2$": **simplex coding** (click 7.8.2.3.3 and refer to clause 5.3.3.2 of TS 38.212).

- "3≤*N*≤11": **(32, k) Reed-Muller coding** (refer to clause 5.3.3.3 of TS 38.212).

### 4.11.3.2 Summary of characteristics of the 7 channel-specific reference signals

The following table summarizes some relevant characteristics of the **7 channel-specific reference signals** (i.e., **physical signals associated** to **physical channels**):

| | **Channel-specific Reference Signals** | | | | | | |
|---|---|---|---|---|---|---|---|
| | **PBCH DM-RS** | **PDCCH DM-RS** | **PDSCH DM-RS** | **PDSCH PT-RS** | **PUCCH DM-RS** | **PUSCH DM-RS** | **PUSCH PT-RS** |
| **Transm. direction** | Downlink | | | | Uplink | | |
| **Carried Information** | None | | | | | | |
| **Subcarrier spacing** | The **same** as the corresponding **physical channel** | | | | | | |
| **Symbol interval location** | Every symbol interval | Channel dependent | | | | | |
| **Subcarrier location** | 1 every 4 subcarriers | Channel dependent | | | | | |
| **CP and symbol duration** | The same as the corresponding physical channel | | | | | | |
| **Complex symbols** | Same complex symbols as the ones of QPSK | | | | `f1`, `f4`: specific `f1`, `f3`: π/2-BPSK | Same complex symbols as the ones of QPSK | |
| **Generating Sequence** | ***c***() pseudo random sequence | | | | `f1`, `f4`: **ZC** `f2`, `f3`: ***c***() | ***c***() pseudo random sequence | |
| **Antenna port(s)** | 4,000 to 4,999 | 2,000 | 1,000 to 1,999 | | 2,000 | 0000 | |
| **Clause of TS 38.211** | 7.4.1.4 | 7.4.1.3 | 7.4.1.1 | 7.4.1.2 | 6.4.1.3 | 6.4.1.1 | 6.4.1.2 |
| **Click section** | 6.6 | 6.11 | 6.14.5 and 6.17.1 | 6.17.2 | 7.6.9 | 7.11.5 and 7.14 | 7.11.5 and 7.14 |

Some **relevant remarks** on this table are:

- The **pseudo random sequence** "***c***()" is the **first** "***n***" bits of a **gold-sequence**. For more detail on "***c***()", click 11.20.1.
- **Row** "**Symbol location**": the **symbol interval location** of the corresponding **DM-RS** or **PT-RS** is indicated in respect to the **scheduled symbol intervals** for the corresponding **physical channel**.
- **Row** "**Subcarrier location**": the **subcarrier location** of the corresponding **DM-RS** or **PT-RS** is indicated in respect to the **scheduled subcarrier** for the corresponding **physical channel**.
- **Row** "**Antenna port(s)**" (click 4.11.4.8.3): the **antenna port range** used for **PBCH** and for its **DM-RS** are the **same**. The **antenna port range** used for **PDSCH** and for its **DM-RS** and its **PT-RS** are the **same**.
- **Column** "**PUCCH DM-RS**": a) **each term** "`fn`" denotes the **PUCCH resource** `format` "`n`" (click 7.6.6); b) the term "specific" denotes the usage of **specific complex symbols** that **may**, or **may not**, **match** any digital modulation; "**ZC**" denotes a **Zadoff-Chu** sequence (click 11.20.4).
- **Row** "**Modulation**": There is **no** "**Modulation**" **row**, because **channel-specific reference signals** do not use modulation schemes. **Channel-specific reference**

**signals** are rather **sequences** of **complex numbers** and those sequences are mapped directly onto their allocated **REs** before generating the waveform for transmission across the air-interface.

#### 4.11.3.2.1 DM-RS

**Demodulation Reference Signals** (**DM-RS**) are **known signals** transmitted in **REs interleaved** with **payload REs** to allow the **receiver** an **estimation** of the **instantaneous channel state** in order to **correctly demodulate** the **received data symbols**. **DM-RS** are **sequences** of **complex symbols** (i.e., they are **not** symbols from a **digital modulation**). However, the **complex symbols** used for **5G DM-RS** are the **same** as the ones from the **QPSK constellation**

**DM-RS sequences** are **deterministic** and have **low autocorrelation**, allowing the **receiver** to separate them from **noise** and **interference**. **Each DM-RS sequence** is generated with a **pseudo-random sequence**.

**DM-RS** are **specifically designed** for **channel estimation** during **data demodulation**.

**Each physical channel**, **except PRACH**, has its **specific DM-RS**:

- **PBCH DM-RS** (click 6.6)
- **PDCCH DM-RS** (click 6.11)
- **PDSCH DM-RS** (click 6.14.5 and 6.17.1)
- **PUCCH DM-RS** (click 7.6.9)
- **PUSCH DM-RS** (click 7.11.5 and 7.14)

**PRACH** does **not** have **DM-RS** because it does **not** use any **modulation**. **PRACH** is the **only physical channel** that conveys a **complex Zadoff-Chu sequence**. **All** of the **other physical channels** convey a **sequence** of **symbols** from a **digital modulation**.

#### 4.11.3.2.2 PT-RS

**Phase-tracking reference signal** (**PT-RS**) are **known signals** transmitted in **REs interleaved** with **payload REs**. They are used track the **phase** of the **local oscillator** at the **receiver** and at the **transmitter**. This enables the **suppression** of **phase noise** and **common phase error**, which is particularly **important** at **high frequencies** (e.g., millimeter wave).

Due to the properties of **phase noise**, **PT-RS** can have **low density** in the **frequency domain** but **high density** in the **time domain**. **PT-RS** are present **both Downlink** (associated with **PDSCH**) and **Uplink** (associated with **PUSCH**).

When transmitted, **PT-RS** is **always** associated with **1 DM-RS antenna port** (click 4.11.5) and it is confined to the **scheduled REs** of **PDSCH/PUSCH**. **Time** and **frequency densities** of **PT-RS** are **adapted** to **signal-to-noise-ratio** (**SNR**) and **scheduled bandwidth**.

**Each** of the **2 shared physical channels**, **PDSCH** and **PUSCH**, has its **specific PT-RS**. For more detail on **PDSCH PT-RS**, click 6.17.2.

The **other 4 physical channels** do **not** have **PT-RS**.

### 4.11.3.3 Summary of characteristics of the 6 standalone physical signals

The following table summarizes some relevant characteristics of the **6 Standalone Physical Signals** (i.e., signals **not** associated with any physical channel):

| | **Standalone Physical Signals** | | | | | |
|---|---|---|---|---|---|---|
| | **PSS** | **SSS** | **CSI-RS** | **RIM-RS** | **PRS** | **SRS** |
| **Transm. direction** | Downlink | Downlink | Downlink | Downlink | Downlink | Uplink |
| **Main purposes** | Synchro and PCI | Power measures and PCI | Downlink channel characteriz. | Remote interference management | Positioning | Uplink channel characteriz. |
| **Carried Information** | PCI's identity | PCI's group | None | | | |
| **Subcarrier spacings [kHz]** | Same as PBCH (4.11.3.1) | | 15, 30, 60 or 120 | 15, 30 or 60 | | 15, 30, 60, 120 or 240 |
| **Quantity of subcarriers or Resource Blocks** | 127 SCs (sent with PBCH as SSB) | | Configurable with RRC | Configurable between neighbor gNBs | Configurable with RRC (e.g., 24 to 272 RBs) | Configurable with RRC (e.g., 24 to 272 RBs) |
| **CP duration [$T_c$] and [µs]** | Same as carrier | | | | | |
| **Symbol duration [$T_c$] and [µs]** | Same as carrier | | | | | |
| **Sent in symbol interval(s)** | First of SSB | Third of SSB | Configurable with RRC | Configurable between neighbor gNBs | Configurable with RRC | Configurable with RRC |
| **Sending Period** | Same as PBCH (4.11.3.1) | | | | | |
| **Instances in burst** | Band tables | | | | | |
| **Modulation** | BPSK | | QPSK | | | |
| **Generating Sequence** | Specific M-sequence | Specific Gold sequence | ***c***() pseudo random sequence | ***c***() pseudo random sequence | ***c***() pseudo random sequence | ***c***() pseudo random sequence |
| **Antenna Port(s)** | Same as PBCH | | 3xxx | Any Downlink antenna port | 5000 | 1xxx |
| **Clause of TS 38.211** | 7.4.2.2 | 7.4.2.3 | 7.4.1.5 | 7.4.1.6 | 7.4.1.7 | 6.4.1.4 |
| **Click section** | 6.1 and 6.19 | | 6.20 | 6.22 | 6.21 | 7.15.9 |

Some **relevant remarks** on this table are:

- **Row "Quantity of subcarriers…"**

  The **subcarriers** or **RBs** for **PSS**, **SSS** and **SRS** are **exact**. However, the **RBs** for the **other signals** correspond to **frequent configurations**, but **other values** may be used.

- For more detail on the **pseudo random binary sequence** "***c***()" click 11.20.1.
- For more detail on **Gold binary sequences**, click 11.20.5.
- For more detail on **M-sequences** (**binary**) click 11.20.6.
- For more detail on **antenna ports**, click 4.11.5.

Recall from 14.1 that **Sidelink** channels are **not** covered in this book.

Source web pages:

- https://www.electronics-notes.com/articles/connectivity/5g-mobile-wireless-cellular/data-channels-physical-transport-logical.php
- https://www.nrexplained.com/modulation

## 4.11.4 Scheduling, transmission/reception occasions and transmissions/receptions

### 4.11.4.1 Scheduling

**Scheduling** is the **process** of the **deciding** for **each transmission occasion** (click 4.11.4.2), for **each transmission**, for **each reception occasion**, and for **each reception**, its **time-domain resources**, its **frequency-domain resources**, its **space-domain resources** and its **characteristics**. This process is carried out for **each physical channel** (click 4.11.3.1) and for **each standalone physical signal** (click 4.11.3.3).

- The **time-domain resources** are the **symbol intervals** or **time instants**.
- The **frequency-domain resources** are the **subcarriers** or the **reference frequencies**.
- The **space-domain resources** are the **beams**, the **layers**, the **antenna ports** and/or the **precoding information**.
- The **characteristics** are the **modulation and coding**, the **redundancy version** and **others**.

As already indicated above, these **resources** and **characteristics** are **scheduled** for **each transmission occasion** (click 4.11.4.2), for **each transmission**, for **each reception occasion**, and for **each reception.** This leads to the following **specific scheduling decisions**:

- **Transmission scheduling.**
  - **Scheduling** of the **transmission occasions**.
  - **Scheduling** of the actual **transmissions**.
- **Reception scheduling.**
  - **Scheduling** of the **reception occasions**.
  - **Scheduling** of the actual **receptions**.

**Neighbor NR nodes usually coordinate** (click 4.11.4.7) their **scheduling decisions** in order to **minimize co-channel interference** at the **UEs** and **improve** their **performance**.

The result of **scheduling decisions** results in the actual **usage** of the **REs** from the **resource grid** (click 2.2.5.6) of the **currently active** (click 2.2.7.4) **bandwidth part** for the **different binary channels and signals**.

Click 1.1.4 for a **clarifying figure** that shows an **approximate** bidimensional (in **frequency** and **time**) representation of a **resource grid** (click 2.2.5.6) of the **Downlink currently active** (click 2.2.7.4) **bandwidth part** (click 2.2.7), for the duration of **1 frame** (click 2.3.1), indicating with colors how the **different RBs** (click 2.2.5.7) have been **scheduled** (click 4.11.4) to the **different physical binary channels** (click 4.11.3).

In **5G**, the **frequency time-domain resources** (**RBs**) and the **time-domain resources** (**symbol intervals**) are **assigned independently**. However, in **LTE**, they are **jointly assigned**. The **resource assignment unit** in **LTE** is the **LTE Resource Block** (**RB**), which is **all** the data **symbols** of **1 subframe** over **12** contiguous **subcarriers** (refer to clause 4.4.4.1 of TS38.211). In **LTE**, within each **subframe**, each active **UE** is assigned **1**, **and only 1**, **range** of **contiguous** (in frequency) **LTE Resource Blocks**. This is a "**linear range**" within the range of data Resource Blocks in the carrier.

For more detail on **scheduling**, refer to clause 10 of TS 38.300.

### 4.11.4.2 Transmission/reception occasions and transmissions/receptions

**Transmission occasion and reception occasion**

**Each UE transmission occasion** or **each UE reception occasion** is a **set** of **resources** and **characteristics** that were **jointly allocated** and where **1 actual UE transmission** or **1 actual UE reception may** (respectively) take place.

**Transmission and reception**

**Each actual transmission** or **each actual reception** is a **set** of **REs** that were **jointly allocated** with some **transmission characteristics** and that **actually contain** "**1 information unit**" (see below). An **actual transmission** or an **actual reception** are usually just called a **transmission** or a **reception** (respectively).

**Information unit**

The **concrete content** of "**1 information unit**" depends on the **physical binary channel**:

- **1 PBCH transmission/reception**

  It contains the **PBCH system information** (click 5.3.2) and the **MIB** (click 5.3.3), with the corresponding **PBCH DM-RS** (click 6.3).

- **1 PDCCH transmission/reception**

  It contains **1 Downlink Control Information** (**DCI**, click 6.9) **message**, with the corresponding **PDCCH DM-RS** (click 6.8.9).

- **1 PDSCH transmission/reception**

  It contains **1** or **2 codeword**(**s**) (click 4.12.1.2), with the corresponding **PDSCH DM-RS** (click 6.14.5) and **PDSCH PT-RS**.

- **1 PRACH transmission/reception**

  It contains **1 random access preamble** (click 7.3).

- **1 PUCCH transmission/reception**

  It contains **1 Uplink Control Information** (**UCI**, click 7.7) **instance** with the corresponding **PUCCH DM-RS** (click 7.6.5.1), except in **PUCCH resource** **format0**, that has **no DM-RS**.

- **1 PUSCH transmission/reception**

  It contains **1 codeword** (click 4.12.1.2) with the corresponding **PUSCH DM-RS** (click 7.11.5) and **PUSCH PT-RS**.

The **transmitter transmits** in **some** (or **all**) of the **transmission occasions**. However, the **receiver should** be **ready** to **receive** in **all** the **reception occasions**, but it will **only receive** in the ones that correspond to an actual **transmission**, which results in an **actual reception**. In this way, **transmissions** are **not lost** because of **scheduling incoherence**.

**Unused transmission occasions** and **reception occasions** are **undesirable**, because they imply **wasted capacity**. However, they are **required** in **some procedures** to provide **flexibility** and **low delay**.

#### 4.11.4.3 Built-in, semi-persistent and dynamic scheduling

The **NR node** can configure the **UE scheduling** with the 3 possible scheduling types:

- **Built-in scheduling**
- **Semi-persistent scheduling** (**configured grants**)
- **Dynamic scheduling** (**DCI assignments**)

**Built-in scheduling** decisions are **built-in** in the **UEs**, and its usage is usually restricted to **initial access** and **link-recovery** procedures.

**Semi-persistent** and **dynamic scheduling** are decided by **each NR node** (click 4.11.4.8), which **signals** them to **each** of its **UEs**.

**Semi-persistent scheduling**

**Each semi-persistent scheduling** (**SPS**) decision results in **1 configured grant** (**CG**) that is **configured**, using the **RRC** (click 5) **protocol**, to the corresponding **UE**(**s**) and for a **given duration**.

**Each configured grant** consists of a **set** of **periodical reception occasions** (for **PDSCH**) or of **periodical transmission occasions** (for **PUSCH**).

**Each configured grant** can be **activated** or **deactivated** with a **DCI message** (click 6.9) and a specific **RNTI**. Section 6.9.10.2 contains a **useful table** of **RNTIs** with their **DCI Formats**, **usage** and **mapping** to **channels**. **Deactivation** can also occur due to a **timeout** of the **configured grant** or using **RRC Deactivation** can be confirmed with a **MAC PDU** that contains the **configured grant confirmation MAC-CE**.

If the **UE** receives a **DCI message** with a **dynamic assignment** (see below) while it has a **configured grant**, the **UE complies** with the **dynamic assignment**, and it does **not affect** the **configured grant**.

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_ConfiguredScheduling.html#Configure_CS_for_Downlink

**Dynamic scheduling**

**Each dynamic scheduling** (**DS**) decision results in **1 assignment** of **either PDSCH** or **PUSCH physical resource elements** (**PREs**, click 2.2.5.3) in **port** and their **specific characteristics**. **Each** such **assignment** of **resources** results (respectively) in **either 1 PDSCH reception** or in **1 PUSCH transmission**.

The **NR node scheduling algorithm** allocates the **available PDSCH PREs** and the **available PUSCH PREs** among the **UEs** and sends **each UE** its corresponding (respectively) **PDSCH assignment**(**s**) and/or **PUSCH assignments**.

**Each PDSCH assignment** or **each PUSCH assignment** is **signaled** to the corresponding **UE** (or **group** of **UEs**) with **1 DCI message** (click 6.9). **Each DCI message** can contain **either** a **few PDSCH assignments** or a **few PUSCH assignments**. **Each PDSCH assignment** or **each PUSCH assignment** contains (respectively which **Rx** or **Tx antenna port**(**s**) to use, and for **all such antenna port**(**s**), **1 combination** of **physical resource block**(**s**) and **symbol interval**(**s**) to use.

**Each DCI message** contains **specific fields** (click 6.9.1) to **signal** the **time-domain resource assignment**(**s**), the **frequency-domain resource assignment**(**s**) the **antenna**

**ports** and **precoding**, the **modulation**(**s**) and **coding**(**s**), the **redundancy version**(**s**) and **other characteristics**.

For a **useful table** of **DCI message Formats** with their **usage** and **RNTIs**, click 6.9.

#### 4.11.4.4 Channel-specific scheduling

The **concrete procedures** for **5G scheduling** depend on **each specific channel**, **each standalone signal** or **each information element**, as is described in:

- **PBCH scheduling** (click 6.3)
- **PDCCH scheduling** (click 6.8)
- **PDSCH scheduling** (click 6.14)
- **PRACH scheduling** (click 7.6)
- **PUCCH scheduling** (click 7.5)
- **PUSCH scheduling** (click 7.11)
- **System Information scheduling** (click 5.2)

The **dynamic scheduling** for the "**shared**" **physical binary channels** (i.e., **PDSCH** and **PUSCH**) is **signaled** by the **NR node** to the **UEs** by sending **DCI messages** (click 6.9) through **PDCCH**.

**Each DCI message** with **Format 0_0** (click 6.9.2) or **0_2** (click 6.9.4) **signals** to **1 UE** the indication for **1 PUSCH transmission. Each DCI message** with **Format 0_1** (click 6.9.3) **signals to 1 UE the indication for 1 to 8 PUSCH transmission(s).**

**Each DCI message** with **Format 1_0** (click 6.9.5) or **1_2** (click 6.9.7) **signals** to **1 UE** the indication for **1 PDSCH reception**. **Each DCI message** with **Format 1_1** (click 6.9.6) **signals** to **1 UE** the indication for **1** to **8 PDSCH reception**(**s**). **Each DCI message** with **Format 4_0** (click 6.9.9.1), **4_1** (click 6.9.9.2) or **4_2** (click 6.9.9.3) **signals** to **1 group** of **UEs** the indication for **1 PDSCH reception**.

#### 4.11.4.5 Multislot scheduling

**Multislot scheduling** consists of **1 DCI message** that **schedules 1 given PUSCH transmission** or **1 given PUCCH transmission** to be "**repeated** " in **multiple slots**. This allows the **NR node** to **combine** the **PUSCH signals** or the **PUCCH signals** from the **multiple slots** resulting in an **improved reception** at the **NR node**.

**Multislot scheduling** is a **diversity technique** (click 11.8) to improve the **Uplink coverage** of an **NR Node**.

**Multislot** scheduling has been used in previous **OFDM** wireless communication systems such as **LTE**.

There are **2 types** of **multislot scheduling**:

- **Slot repetition**

  The **PUSCH transmission** or **PUCCH transmission** is **literally repeated** in the **successive slots**. The **received transmissions** are **added** in the **time domain** in the **NR node**, resulting in an **improved Signal** to **Noise** plus **Interference Ratio** (**SNIR**).

- **Slot aggregation**

  The **PUSCH transmission** or **PUCCH transmission** is **encoded** with a **larger size**, over the **different slots**, resulting in a **decoding gain** at the **NR node**.

#### 4.11.4.6 Mini-slot scheduling

**Mini-slot scheduling** consists of **scheduling** (click 4.11.4.1) transmission resources on a **shorter scale** than a **slot** (click 2.3.2), and of the capacity of **preempting** an **already scheduled slot-based transmission**. This allows for **extremely fast transmission** of data that requires **very low latency**.

**Mini-slot scheduling** is very useful to provide the **small data transmission** (**SDT**, click 10.9) service and to support the **URLLC** and **mMTC** use cases (click 10.10).

A **mini-slot** occupies **2**, **4** or **7 symbol intervals**. It enables **non-slot based scheduling**, and it is the **minimum scheduling unit** used in **5G**. A **mini-slot** can be positioned **asynchronously** with respect to the **beginning** of a **slot**.

**Mini-slot scheduling** can be **combined** with **dynamic TDD slots**, for **fast Downlink-Uplink switching**.

A **mini-slot** contains **DM-RS** at position(s) relative to the **start** of the **mini-slot**.

Note that **some UEs** targeting certain use cases may **not** support **all mini-slot lengths**, **nor all starting positions** within a slot.

Since a **slot** is defined as a **fixed quantity** of **symbol intervals**, a larger subcarrier spacing leads to a shorter slot duration. In principle this could be used to support low-latency transmission, but as the cyclic prefix also shrinks when increasing the subcarrier spacing, it is not a feasible approach in all deployments. In order to allow for low-latency transmission in such cases, **5G** incorporates **mini-slot transmission**.

**Mini-slot scheduling** combines the two following mechanisms:

- **Intra-slot scheduling**

  The **NR node can assign** a **transmission** to take place in the **same slot** where the **assignment** is **sent**.

- **Preemptive scheduling**

  The **NR node can assign** a **transmission** that takes place in **REs** that have been **previously scheduled** to **other UE(s)**.

**Intra-slot scheduling**

The **NR node can assign** a **transmission** to take place in the **same slot** where the **assignment** is **sent**.

In **conventional scheduling**, a **transmission** takes place in a **slot** that **follows** the **slot** where the **assignment** was **sent**.

**Intra-slot scheduling** is performed with the **same DCI messages** and **fields**. The most relevant involved **field** is "***Time domain resource assignment***" (click 6.9.1). This **field** is used for the **PDSCH** and **PUSCH time scheduling** (click 6.14.4 and 7.11.4). When **intra-slot scheduling** is used, there may be a **different interpretation** of the **said field**.

**Preemptive scheduling**

The **NR node can assign** a **transmission** that takes place in the **REs** that has been **previously scheduled** to **other UE(s)**. Notice that this is **not** possible in **conventional scheduling**.

When the **REs** of a **given UE** have been **preempted**, the **NR node notifies** the **UE** the preemption, via **MAC signaling**.

The **Downlink REs** whose **assigned REs** have been **preempted** to a **given UE** are **set** to "**erased**" by **that UE** in its **FEC decoding**.

Source web pages:

- https://howltestuffworks.blogspot.com/2019/11/5g-nr-time-domain-slots-and-slot-formats.html
- https://medium.com/@jessica.chchuang/5g-urllc-mini-slot-in-5gnr-b830a534da8c
- https://www.sharetechnote.com/html/5G/5G_MiniSlot.html
- https://www.telecomhall.net/t/what-is-the-5g-mini-slot-concept/12147

### 4.11.4.7 NR node coordination of the scheduling in neighbor cells

This section describes the **NR node coordination** of the **scheduling** in **neighbor cells** (click 3.1.2). For the **coordination**, within **each NR node**, of its **scheduling**, its **beamforming** and its **MIMO**, click 4.11.4.8.

**Co-channel interference** between **same-channel neighbor cells** (click 3.1.2) can be largely **mitigated** with **scheduling**: the **UEs** which are in **border areas** are **scheduled Resource Blocks** on distant frequencies **among them**. More robust modulation and coding can also be used.

This requires the **coordination** of the **scheduling** in **each pair** of **same-channel neighbor cells**. **Each cell** has **few same-channel neighbor cells**, and consequently, the coordination **algorithms** might be **reasonably simple**.

The **2 cells** in **each pair** of **same-channel neighbor cells** might belong to the **same NR node** (e.g., **2 consecutive sectors**, click 3.10.6.3) or to **2 neighbor NR nodes** (e.g., **2 consecutive NR nodes** (click 3.10.5)).

### 4.11.4.8 NR node coordination between scheduling, beamforming and MIMO

This section describes the **coordination**, within **each NR node**, of its **scheduling**, its **beamforming** and its **MIMO**. For the **NR node coordination** of the **scheduling** in **neighbor cells**, click 4.11.4.7.

#### 4.11.4.8.1 Coordination procedure

The **5G scheduling** (click 4.11.4) is **very much coordinated** with **both beamforming** (click 2.4.1) and **MIMO** (click 2.4.3). This is required so that **UEs** with **overlapping beams** are **not scheduled** at the **same time** unless they are **properly separated** in **frequency**. In order to avoid **interferences**, the **NR node scheduling** has to **maximize** the **UE** separation, **each** represented as a **3 dimensional vector**, composed of **frequency**, **time** and **space**.

The **main steps** in the **coordination** between **scheduling**, **beamforming** and **MIMO** are:

1. **Channel Estimation** (click 4.5.7.6)

   **Downlink**, estimation from **CSI-RS** (click 2.4.5):

   - The **NR node** transmits **quasi-static** or **periodic beams** with **channel state information reference signals** (**CSI-RS**).
   - **Each and every UE measures** the **best beam(s)** and **reports back** detailed **channel information**, such as **RSRP**, **PMI**, **RI**, **LI**, **CQI**, **CRI** and **SSBRI**.

   **Uplink**, estimation from **SRS**:

   - **Each UE transmits sounding reference signals** (**SRS**) to allow the NR node to estimate the **Uplink spatial signatures**.

2. **Selection** of **beams** and **UEs**
   - The **scheduler** selects **UEs** that have **high channel quality indicator** (**CQI**) on specific **beams**.
   - The **scheduler** defines **UE groups** for **space-division multiplexing** such that their **preferred beams** are **angularly separated** (i.e., they are **spatially orthogonal**).

3. **Deliver** the **UE assignments** over **PDCCH**
   - The **NR node** delivers **each PDSCH/PUSCH assignment** that has been **scheduled** in the **previous point** to **each** corresponding **UE** (or **group** of **UEs**).
   - **Each assignment** is delivered as a **DCI message** sent through **PDCCH**.
   - **Each DCI message defines** (explicitly or explicitly) the **assigned frequency-time resources** (i.e., the **Resource Elements**) and the **space resources** (**beam** and/or **MIMO layers**).

4. **Transmission and reception**

   **Downlink**
   - The **NR node** uses selected **precoders** (based on **CSI**) to form **beams** and/or **space-division layers**.
   - The **data streams** are then **transmitted** over those **beams**, possibly reusing time-frequency **REs** across **UEs**.
   - **Each** destination **UE** uses the information in its **assigning DCI message** (see **step 3** above) to **decode** the **received** transmission with the **correct parameters** for its receive-side **spatial processing** (e.g., **MMSE combining**).

   **Uplink**
   - **Each UE** configures its **Uplink transmission** (**REs**, **precoding, etc**) following the indications from its **assigning DCI message** (see **step 3** above).
   - The **data stream(s)** are then **transmitted** over the corresponding **REs**, **beams** and **layer(s)**.

- The **NR node** uses **space-division filters** (receive beams) and **detection** (2.4.2) to **separate** the **streams** from **different UEs** on the **same PRBs**.

#### 4.11.4.8.2 Coordination principles

The main **principles** that guide the **coordination** between **scheduling**, **beamforming** and **MIMO** are:

- **Channel Knowledge**

  **Scheduling** decisions **depend** heavily on **CSI** (**Channel State Information**, click 2.4.5). **Accurate CSI** is **required** for:

  - **Beam selection** (click 6.1.2.3) and **precoding** (click 2.4.2)
  - **Interference management**
  - **Spatial separation** of **UEs**

- **Beam-Based Scheduling**
  - The **scheduler** must **assign** to **each UE** the **time resources** (i.e., **symbol intervals**) that correspond to the **time range** where the **beam** (**analog beamforming**) is **illuminating** the location of the **UE**.
  - In **TDD bands** with **beamformed reference signals**, a **UE** is scheduled only on **beams** where its **SNIR** is above a certain threshold.

- **Multi-User MIMO** (**MU-MIMO**)
  - The **scheduler must** select **UE groups** that are **spatially separable** (i.e., their **channels** are **sufficiently orthogonal**).
  - **Space-division multiplexing** is possible when the **NR node** can form **beams** (**precoders**) that do not interfere with each other.

- **Precoder Selection:**
  - The **NR node** selects its **precoding matrices** based on **UE-reported CSI** or **codebook indices** (e.g., **precoding matric indicator**).
  - The **scheduler** ensures that **UEs multiplexed** in the **same time-frequency resources** get **compatible precoders** (i.e., **minimize** mutual **interference**).

For **massive MIMO**, the **quantity** of **spatial layers** per **UE** and the **number** of **simultaneously scheduled UEs** are **limited** by:

- The **quantity** of **antennas** or of **RF chains** (whichever is **lower**).
- The **accuracy** of **CSI**.
- The available **computational capacity** for **precoding**.

**Beamforming** and **MIMO** are **particularly frequent** in **FR2-1 bands** and in **FR2 bands**.

#### 4.11.4.8.3 Coordination for control data

Same as for **data** transmission, the transmission of **physical layer control information** is subject to **coordination** between **scheduling**, **MIMO** and **beamforming**, although in a **simpler** way.

**PDCCH transmission**

The **PDCCH transmissions** are **not** subject to **space-division multiplexing** (i.e., they are **single layer**).

The **PDCCH transmissions** are usually **beamformed**, but this **beamforming** is typically **simpler** than for **PDSCH/PUSCH**. A **single beam** (or a **predefined set** of **beams**) is used to ensure that **all intended UEs** in the **coverage area** can **correctly decode PDCCH**.

If **increased reliability** is required, **PDCCH** can be transmitted **over multiple beams**, but this is **not space-division multiplexing** (click 11.6.3) and it is rather **space diversity** (click 11.8.1). For the **differences** between both **techniques**, click 11.10.

**PUCCH transmission**

The **PUCCH transmissions** are **not** subject to **space-division multiplexing** (i.e., they are **single layer**).

The **PUCCH transmissions** are usually **not beamformed**.

The **PUCCH transmissions** take place through a **single antenna port** (i.e.,

### 4.11.5 Antenna ports

An **antenna port** is a **spatial path** from **transmitter to receiver**, through which **signals** are **transmitted**, and that is **independent** of the **spatial paths** corresponding to **other antenna ports**. The **formal definition** of a **5G antenna port** is done in clause 4.4.1 of TS 38.211:

> *"An antenna port is defined such that the channel over which a symbol on the antenna port is conveyed can be inferred from the channel over which another symbol on the same antenna port is conveyed."*

**Two antenna ports** are **quasi co-located** when they are **sufficiently close** (in relation to the **wavelength**), such that their **large-scale channel properties** are **similar**.

The **formal definition** of **quasi co-located 5G antenna ports** is done in clause 4.4.1 of TS 38.211:

> *"Two antenna ports are said to be quasi co-located if the large-scale properties of the channel over which a symbol on one antenna port is conveyed can be inferred from the channel over which a symbol on the other antenna port is conveyed. The large-scale properties include one or more of delay spread, Doppler spread, Doppler shift, average gain, average delay, and spatial Rx parameters."*

An **antenna port** identifies a **logical antenna**, which **cannot** be related to **any specific physical antenna** in the **array**. It represents a **logical transmission resource** that incorporates **beamforming** (click 2.4.1) and **spatial processing** of the signal.

**Each antenna port** is associated with a specific **set** of **reference signals**, which allows the **receiver** to perform **channel estimation** for that **port**. For the **UE**, the **antenna port** is a **logical abstraction** used for **channel estimation** and **demodulation**.

A **single antenna port** may use **multiple array antennas** (e.g., in **beamforming** scenarios). The **mapping** between **antenna ports** and **physical antennas** is determined by the **beamforming** and **precoding strategies** of the **NR node**.

**Each layer** (click 2.4.3.5) always requires **at least 1 dedicated antenna port**. However, **each antenna port** does **not** always support a **layer**. **Antenna ports** may be used for purposes **other** than supporting a **layer**.

**5G** defines several **antenna ports both** for **Downlink transmission** (click 4.11.5.1) and for **Uplink transmission** (click 4.11.5.2).

Source web page:

- https://www.sharetechnote.com/html/5G/5G_AntennaPorts.html

### 4.11.5.1 Antenna ports for Downlink

The following **antenna ports**, and their usage, are defined for the **Downlink transmission direction**:

- **Antenna port** from **4,000** to **4,999** for **SSB** (click 6.1), including the **PBCH DM-RS** (click 6.3).
- **Antenna ports** from **2,000** to **2,999** for **PDCCH** (click 6.9). The **PDCCH DM-RS** (click 6.11) are sent on **port 2,000**.
- **Antenna ports** from **1,000** to **1,999** for **PDSCH** (click 6.16), including the **PDSCH demodulation reference signals** (**PDSCH DM-RS**) and the **PDSCH phase-tracking reference signals** (**PDSCH PT-RS**). For these reference signals, click 4.11.3.2.
- **Antenna ports** from **3,000** to **3,999** for **channel state information reference signals** (**CSI-RS**, click 6.20).
- **Antenna ports** from **5,000** to **5,999** for **positioning reference signals** (**PRS**, click 6.21).
- **Any Downlink antenna port** for **remote interference management reference signals** (**RIM-RS**) and for **Non-Zero Power Channel State Information Reference Signals** (**NZP CSI-RS**). At any given moment, **only 1 Downlink antenna port** may be used. However, this **antenna port** may be **cyclically changed**.

The **UE** shall **not assume** that **two antenna ports** are **quasi co-located** (click 4.11.5) with respect to any **QCL type** unless it is **explicitly indicated**.

For **PDSCH DM-RS**, the **channel** over which a **PDSCH symbol** on **one antenna port** is conveyed can be **inferred** from the **channel** over which a **DM-RS symbol** on the **same antenna port** is conveyed **only** if the **two symbols** are within the **same resource** as the scheduled **PDSCH**, in the **same slot**, and in the same **PRG** as described in clause 5.1.2.3 of TS 38.214.

For **PDCCH DM-RS**, the **channel** over which a **PDCCH symbol** on **one antenna port** is conveyed can be **inferred** from the **channel** over which a **DM-RS symbol** on the same **antenna port** is conveyed **only** if the **two symbols** are **within resources** for which the **UE** may assume the **same precoding** being used as described in clause 7.3.2.2.

For **PBCH DM-RS**, the **channel** over which a **PBCH symbol** on **one antenna port** is conveyed can be **inferred** from the **channel** over which a **DM-RS symbol** on the same **antenna port** is conveyed **only** if the **two symbols** are within a **SSB** transmitted within the **same slot**, and with the **same block index** according to clause 7.4.3.1.

For more detail on **antenna ports** for the **Downlink transmission direction**, refer to clause 7.2 and 7.4.1 of TS 38.211.

### 4.11.5.2 Antenna ports for Uplink

The following **antenna ports** are defined for the **Uplink transmission direction**:

- **Antenna ports** from **0** to **999** for **PUSCH demodulation reference signals** (**PUSCH DM-RS**, click 4.11.3.2.1) and for **PUSCH phase-tracking reference signals** (**PUSCH PT-RS**, click 4.11.3.2.2).
- **Antenna ports** from **1,000** to **1,999** for **PUSCH** (click 7.10) and **SRS** (click 4.11.3.3).
- **Antenna ports** from **2,000** to **2,999** for **PUCCH** (click 7.5). **PUCCH DM-RS** (click 4.11.3.2) are sent on **port 2,000**.
- **Antenna port 4,000** for **PRACH** (click 7.1).

If **PUSCH** repetition **Type B** as described in clause 6.1 of TS 38.214 is applied to a **physical channel**, the **UE** transmission shall be such that the **channel** over which a **symbol** on the **antenna port** used for **Uplink** transmission is conveyed can be inferred from the **channel** over which another symbol on the same **antenna port** is conveyed if the **two symbols** correspond to the same actual repetition of a **PUSCH** transmission with repetition **Type B**.

If **intra-slot frequency hopping** is **not enabled** for a **physical channel** and **PUSCH** repetition **Type B** is **not** applied to the **physical channel**, the **UE** transmission shall be such that the **channel** over which a **symbol** on the **antenna port** used for **Uplink** transmission is conveyed can be inferred from the **channel** over which **another symbol** on the **same antenna port** is conveyed if the **two symbols** correspond to the **same slot**.

If **intra-slot frequency hopping** is **enabled** for a **physical channel**, the **UE** transmission shall be such that the **channel** over which a **symbol** on the **antenna port** used for **Uplink** transmission is conveyed can be inferred from the **channel** over which **another symbol** on the **same antenna port** is conveyed only if the **two symbols** correspond to the **same frequency hop**, regardless of whether the **frequency hop distance** is **zero** or **not**.

If **DM-RS bundling** is applied to **PUSCH** and/or **PUCCH** repetitions and/or transport-block processing over **multiple slots** as described in clause 6.1.7 of TS 38.214, the **UE** transmission shall be such that the **channel** over which a symbol on the antenna port used for **Uplink** transmission is conveyed can be inferred from the **channel** over which **another symbol** on the **same antenna port** is conveyed if the **two symbols** are transmitted within the **same actual time-domain window**.

For more detail on **antenna ports** for the **Uplink transmission direction**, refer to clauses 6.2 and 7.4.1 of TS 38.211.

## 4.12 PDSCH and PUSCH payload processing

The **payload processing** for **PDSCH** and for **PUSCH** is **almost** the **same**. This is why both are **presented together** in this section. Whenever there is a **difference** between the **PDSCH** and the **PUSCH processing**, it will be **explicitly indicated**.

For an **overview** of the **PDSCH** and **PUSCH Tx payload processing**, click 4.12.2.

For an **overview** of the **PDSCH** and **PUSCH Rx payload processing**, click 4.12.16.

For an overview of the **data flow** across the **user-plane protocol stack**, click 4.10.4.

For a **mapping** between the **terms** (e.g., **protected block**) and the **symbols** (e.g., "$N_{blk}^{CBS(q)}$") of **PDSCH** and **PUSCH processing** used in **this book** and the corresponding **terms** and **symbols** (e.g., ***A***, ***C***, …) used in the **TS documents** click 4.12.17.

## 4.12.1 PDSCDH and PUSCH Transport blocks and codewords

**Transport blocks** (click 4.12.1.1) are the **data units** exchanged between the **Low MAC sublayer** and the **physical layer**.

**Codewords** (click 4.12.1.2) are the **data units** scheduled by **each DCI message** (click 6.9) for their **transmission** through **PDSCH** or through **PUSCH**.

### 4.12.1.1 PDSCH and PUSCH transport blocks

A **transport block** is the **data unit exchanged** between the **Low MAC sublayer** and the **physical layer** through the **transport binary channels** (click 4.11.2).

In **Tx**, **each transport block** is delivered **downwards** from **Low MAC** to **physical**, while in **Rx**, **each transport block** is delivered **upwards** from **physical** to **Low MAC**.

**Each transport block** is **either** sent from the **NR node** to **1 UE** (or **1 group** of **UEs**) through **PDSCH** or sent from **1 UE** to the **NR node** through **PUSCH**.

**Each transport block** contains **1 MAC PDU** (click 4.5.5.1). The **MAC PDU** can contain **user data** (e.g., **voice**, **video** or **file transfer**) and/or **control information** (e.g., **MAC signaling** or **RRC messages**).

**Each PDSCH transmission** can contain **1 or 2 transport block(s)** (or part of them, in **HARQ retransmissions**, click 4.5.6.6). **Each PUSCH transmission** contains **1 transport block** (or part of it, in **HARQ retransmissions**).

For **PDSCH Tx** and **Rx**, **each** instance of **1 or 2 transport block(s)** is **exchanged downwards/upwards** between the **Low MAC** and **physical**, **either** through the **Downlink transport shared channel** (**DL-SCH**, click 4.11.2) or through the **transport paging channel** (**PCH**, click 4.11.2). Delivering **2 transport blocks** in **1 PDSCH transmission** is sometimes called "**Dual transport block transmission**".

For **PUSCH Tx** and **Rx**, the **transport block** is **exchanged downwards/upwards** between the **Low MAC** and **physical**, through the **Uplink transport shared channel** (**UL-SCH**, click 4.11.2).

The variable "**q=0**" identifies the **first transport block** and "**q=1**" identifies the **second PDSCH transport block** (in case it exists, and **only** for **PDSCH**) of **each transmission**. The value of "**q**" is **inherited** by **all** the **blocks** (and their associated variables) that result from the **Tx** processing of **transport block** "**q**".

In case that **2 PDSCH transport blocks** are **sent** in **1 PDSCH transmission**, **both** belong to the **same PDSCH HARQ process** (click 4.5.6.1).

**Each transport block** is **represented** by the **bit sequence** "$tb^{(q)}(i_{tb})$", that has **bitlength** "$N_{bit}^{TB(q)}$". The variable "$i_{tb}$" is its **bit index**, that ranges from "**0**" to "$N_{bit}^{TB(q)}$-1". The **most significant bit** of **each transport block** "**q**" is mapped to the **lowest order** information bit "$tb^{(q)}(0)$", as defined in clause 6.1.1 of TS38.321.

**Each transport block** transmission **can** have **different** values of the **DCI fields** "***Modulation and coding scheme***" (click 11.2), "***New data indicator***" (click 4.5.6.3) and "***Redundancy version***" (click 4.5.6.6). For the case of **2 PDSCH transport blocks**, they

are **jointly scheduled** with **1 DCI message** with **Format 1_1** or **4_2**, **each** such **DCI message** has **2 sets** of the said **3 fields**.

The **maximum bitlength** of a **transport block** is **1,277,992 bits** (~**1.22 Mib**) which is the same as **159,749 bytes** (~**156.0 KiB**). The **actual bitlength** "$N_{bit}^{TB(q)}$" of a given **transport block** "**q**" is on most cases **considerably smaller** than the **maximum bitlength**. The **transport block bitlength** "$N_{bit}^{TB(q)}$" is sometimes called the **transport block size**, "**TBS**").

For the **UE** determination of the **PDSCH transport block bitlength** "$N_{bit}^{TB(q)}$", click 6.16.2 and refer to clause 5.1.3.2 of TS 38.214.

For the **UE** determination of the **PUSCH transport block bitlength** "$N_{bit}^{TB(q)}$" click 7.13 and refer to clause 6.1.4.2 of TS 38.214.

For the **transport block** within the **PDSCH/PUSCH data flow** across the **user-plane protocol stack**, click 4.10.4.

### 4.12.1.2 PDSCH and PUSCH codewords

A **codeword** is the **data unit** scheduled by **each DCI message** (click 6.9) for their **transmission** through **PDSCH** or through **PUSCH**.

**Each codeword** is **either** sent from the **NR node** to **1 UE** (or **1 group** of **UEs**) through **PDSCH** or sent from **1 UE** to the **NR node** through **PUSCH**.

For the **first transmission** of a **transport block** (click 4.12.1.1), **each codeword** contains **1 complete transport block**. For successive **HARQ retransmissions** (click 4.5.6.6), **each codeword** contains **either 1 complete transport block** or **part** of it:

**Each PDSCH transmission** contains:

- **Unicast** and **multicast**: **1** or **2 codeword**(**s**) (click 4.12.1.2) in **1** to **8 layers**.
- **Broadcast**: **1 codeword** in **1 layer**.

**Each PUSCH transmission** contains **only 1 unicast PUSCH codeword** in **1** to **4 layers**.

The variable "**q**=**0**" (click 4.12.1.1) identifies the **first codeword** and "**q**=**1**" identifies the **second PDSCH codeword** (in case it exists) of **each transmission**. The value of "**q**" is **inherited** by **all** the **blocks** (and their associated variables) that result from the **Rx** processing of **codeword** "**q**".

**Each codeword** is **represented** by the **bit sequence** "$cw^{(q)}(i_{cw})$", that has **bitlength** "$N_{bit}^{CW(q)}$". The variable "$i_{cw}$" is its **bit index**, that ranges from "**0**" to "$N_{bit}^{CW(q)}$-1". The **most significant bit** of **each codeword** "**q**" is mapped to the **lowest order** information bit "$cw^{(q)}(0)$".

**Each codeword** transmission **can** have **different** values of the **DCI fields** "***Modulation and coding scheme***" (click 11.2), "***New data indicator***" (click 4.5.6.3) and "***Redundancy version***" (click 4.5.6.6). For the case of **2 PDSCH codewords**, they are **jointly scheduled** with **1 DCI message** with **Format 1_1** or **4_2**, **each** such **DCI message** has **2 sets** of the said **3 fields**.

For more detail on the **DCI scheduling** of **codewords** in **each PDSCH/PUSCH transmission**, click 6.9.

For the **codeword** within the **PDSCH/PUSCH data flow** across the **user-plane protocol stack**, click 4.10.4.

### 4.12.2 PDSCH and PUSCH Tx payload processing overview

The **physical layer processes** the **data** that is finally sent within the **PDSCH payload REs** and within the **PUSCH payload REs**.

The **PDSCH Tx** processing by the **physical layer starts** in **1** or **2** **DL-SCH transport block(s)** (click 4.12.1.1) and **ends producing** (respectively) an **instance** of **1** or **2** **PDSCH codeword(s)** (click 4.12.1.2). An **instance** of **1** or **2** **PDSCH codeword(s)** can also be produced by **HARQ retransmission** (click 4.5.6.6). In this later case, **each** such **codeword** contains **either** a **complete transport block** or **part** of it. **Each instance** of **1** or **2** **PDSCH codeword(s)** is sent in **1 PDSCH transmission**, regardless of it having arisen from a **first transmission** or from a **HARQ retransmission**.

The **PUSCH Tx** processing by the **physical layer starts** with **1** **UL-SCH transport block** and **ends producing** **1** **PUSCH codeword**. Under some conditions (click 4.12.8) the **PUSCH codeword** can be **multiplexed** (click 4.12.8) with a **PUCCH codeword** (click 7.8.2.5) into a **multiplexed codeword**. Be aware that **1** **PUSCH codeword** can also be produced by **HARQ retransmission** (click 4.5.6.6). In this later case, **each** such **codeword** contains **either** a **complete message** or **part** of it. **Each PUSCH codeword** (or **multiplexed codeword**) is sent in **1 PUCH transmission**, regardless of it having arisen from a **first transmission** or from a **HARQ retransmission**.

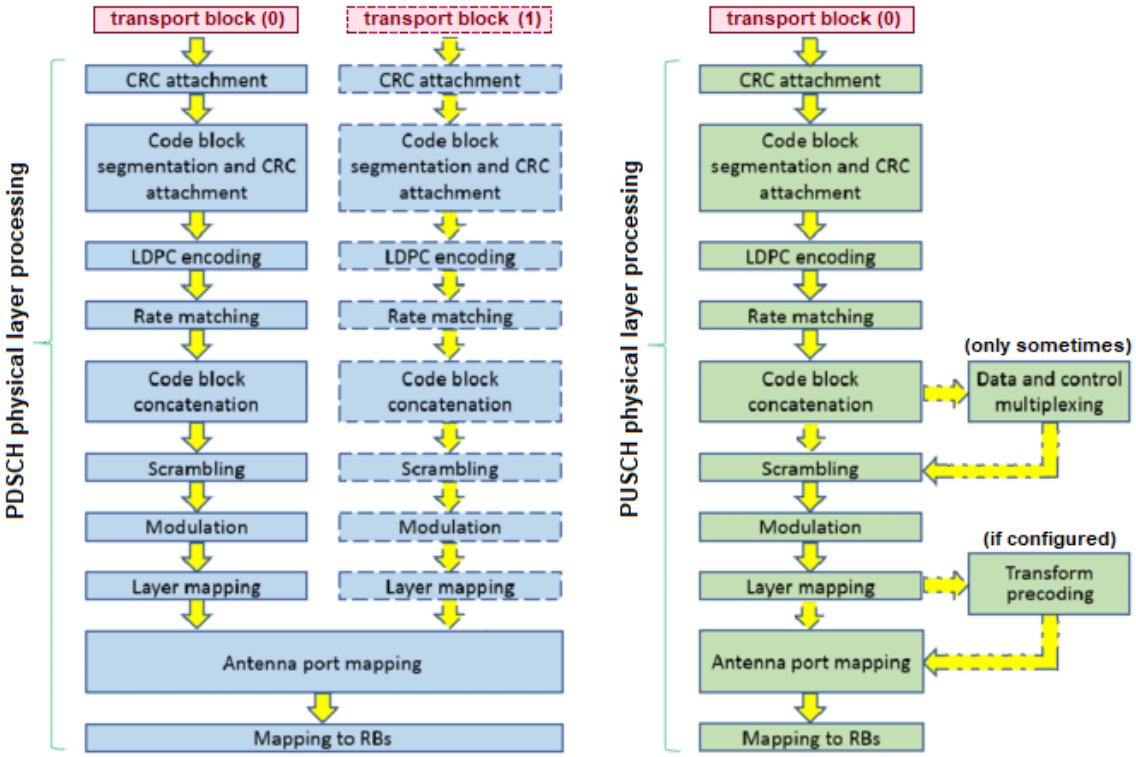
**Each instance** of **1** or **2** **codeword(s)** is **further processed** to obtain **1** or **2** **modulated sequence(s)** (i.e., **sequence(s)** of **symbols**). The **1** or **2** **modulated sequence(s)** are **further processed** to result in **as many sets** of **PREs** in **port** as the **quantity** "$N_{port}$" of used **antenna port numbers**. **All** these **sets** of **PREs** in **port** correspond to **1 PDSCH** or **PUSCH transmission**.

**Each PDSCH** or **PUSCH transmission** is **scheduled** (respectively) with **1 PDSCH** or **PUSCH DCI assignment** or with a **PDSCH** or **PUSCH semi-persistent grant**. Recall that **1 DCI message** (click 6.9) with **Format 0_1** or **1_1** can schedule (respectively) up to **8 PUSCH transmissions** or **8 PDSCH transmissions**.

Along the **initial stages**, **each** of the variables "$i_{xyz}$" (e.g., "$i_{tb}$" or "$i_{pb}$") is the **bit index** that identifies **each bit** of the corresponding **bit sequence** "$xyz(i_{xyz})$" (e.g., "$tb^{(q)}(i_{tb})$" or "$pb^{(q)}(i_{pb})$"). **Each** such variable "$i_{xyz}$" ranges from **0** to the **bitlength** "$N_{bit}^{XYZ}$" (e.g., "$N_{bit}^{TB(q)}$" or "$N_{bit}^{PB(q)}$") **minus 1** of the corresponding **bit sequence** "$xyz(i_{xyz})$".

Along the **final stages**, **each** of the variables "$j_{qrs}$" (e.g., "$j_{ms}$" or "$j_{lys}$") is the **symbol index** that identifies **each symbol** of the corresponding **symbol sequence** "$qrs(j_{qrs})$" (e.g., "$ms^{(q)}(j_{ms})$" or "$lys^{(lay)}(j_{lys})$"). **Each** such variable "$j_{qrs}$" ranges from **0** to the **symbol length** "$N_{symb}^{QRS(q)}$" (e.g., "$N_{symb}^{MS(q)}$" or "$N_{symb}^{layer}$") **minus 1** of the corresponding **symbol sequence** "$qrs(j_{qrs})$".

The following figure illustrates the **PDSCH processing** and the **PUSCH processing**:




**All** the above **processing** can be performed in the following **stages**:

1. **Append** a "$CRC^{PB(q)}$" to **each** of the **1** or **2 transport block**(**s**) resulting in **1** or **2 protected block**(**s**).

   The variable "$tb^{(q)}(i_{tb})$" represents the **bit sequence** of the **transport block** "**q**". The variable "$i_{tb}$" is the **bit index** within **each transport block**.

   The identifier "$N_{bit}^{TB(q)}$" is the **bitlength** of **transport block** "**q**". The **bit index** "$i_{tb}$" of **each transport block** ranges from **0** to "$N_{bit}^{TB(q)}$-1".

   The variable "$pb^{(q)}(i_{pb})$" represents the **bit sequence** of the **protected block** "**q**", that was produced from the **transport block** "**q**". The variable "$i_{pb}$" is the **bit index** within **each protected block**.

   The identifier "$N_{bit}^{PB(q)}$" is the **bitlength** of the **protected block** "**q**". The **bit index** "$i_{pb}$" of **each protected block** ranges from **0** to "$N_{bit}^{PB(q)}$-1".

   The variable "$q \in \{0,1\}$" is the identifier of the **1** or **2 transport block**(**s**) in **each PDSCH transmission**. The variable "**q**=0" is the identifier of the **transport block** in **each PUSCH transmission**. The variable "**q**" is the **inherited** identifier of **all** the **blocks** and **codewords** produced from **transport block** "**q**".

   For more detail, click 4.12.3.

2. **Transform each** of the **1** or **2 protected block**(**s**) into **1** or **2 sequences** of "$N_{blk}^{CBS(q)}$" **code block**(**s**), where **each code block** includes its **specific CRC**.

   The variable "$cb_{cn}^{(q)}(i_{cb})$" represents the **bit sequence** of the **code block** in the **sequential position** "*cn*" within the **code block sequence** "**q**", that was produced from **transport block** "**q**". The variable "*cn*" identifies the **sequential position**

of **each** **code block** within its **code block sequence**. The variable "$i_{cb}$" is the **bit index** within **each** **code block**.

The identifier "$N_{blk}^{CBS(q)}$" is the **quantity** of **code blocks** in **code block sequence** "**q**". The variable "$cn$" of **code block sequence** "**q**" ranges from **0** to "$N_{blk}^{CBS(q)}$-1". The identifier "$N_{bit}^{CB(q)}$" is the **bitlength** of **each** and **every** **code block** from **code block sequence** "**q**". The **bit index** "$i_{cb}$" of **each** **code block** ranges from **0** to "$N_{bit}^{CB(q)}$-1". Notice that **all** the **code blocks** "$cb_{cn}^{(q)}(i_{cb})$" from a given **code block sequence** have the **same bitlength** "$N_{bit}^{CB(q)}$".

For more detail, click 4.12.4.

3. **LDPC encode** **each** **code block** into **1** **encoded block**, resulting in **1** or **2** **sequences** of "$N_{blk}^{CBS(q)}$" **encoded block**(**s**), that are **stored** in the **HARQ Tx buffer**.

   The variable "$eb_{cn}^{(q)}(i_{eb})$" represents the **bit sequence** of the **encoded block** in the **sequential position** "$cn$" within the **complete** **encoded block sequence** "**q**", that was produced from **transport block** "**q**". The variable "$cn$" identifies the **sequential position** of **each** **code block** within its **complete** **encoded block sequence**.

   The variable "$eb_{tn}^{(q)}(i_{eb})$" represents the **bit sequence** of the **encoded block** in the **sequential position** "$tn$" within the **transmitted** **encoded block sequence** "**q**", that was produced from **transport block** "**q**". The variable "$tn$" identifies the **sequential position** of **each** **encoded block** within its **transmitted** **encoded block sequence**.

   The variable "$i_{eb}$" is the **bit index** within **each** **encoded block**.

   The identifier "$N_{blk}^{CBS(q)}$" is the **quantity** of **encoded blocks** in the **complete** **encoded block sequence** "**q**", which is the same as the **quantity** of **code blocks** in **code block sequence** "**q**".

   The identifier "$N_{blk}^{EBS(q)}$" is the **quantity** of **encoded blocks** in the **transmitted** **encoded block sequence** "**q**". For a **new transmission**, "$N_{blk}^{EBS(q)}=N_{blk}^{CBS(q)}$". For a **retransmission**, "$N_{blk}^{EBS(q)}\leq N_{blk}^{CBS(q)}$".

   The variable "$tn$" of the **transmitted** **encoded block sequence** "**q**" ranges from **0** to "$N_{blk}^{EBS(q)}$-1". The identifier "$N_{bit}^{EB(q)}$" is the **bitlength** of **each** and **every** **encoded block** from **encoded block sequence** "**q**". The **bit index** "$i_{eb}$" of **each** **encoded block** ranges from **0** to "$N_{bit}^{EB(q)}$-1". Notice that **all** the **encoded blocks** from a given **encoded block sequence** (either complete or transmitted) "**q**" have the **same bitlength** "$N_{bit}^{CB(q)}$".

   The value of "$N_{bit}^{EB(q)}$", is **substantially larger** than the value of "$N_{bit}^{CB(q)}$".

   For more detail, click 4.12.5.

4. **Rate-match** **each** **transmitted** **encoded block** into **1** **rate-matched block**, resulting in **1** or **2** **sequences** of "$N_{blk}^{EBS(q)}$" **rate-matched blocks**.

   The variable "$rb_{tn}^{(q)}(i_{rm})$" represents the **bit sequence** of the **rate-matched block** in the **sequential position** "$tn$" within the **rate-matched block sequence** "**q**", that was produced from **transport block** "**q**". The variable "$tn$" identifies the **sequential position** of **each** **rate-matched block** within its **rate-matched block sequence**. The variable "$i_{rm}$" is the **bit index** within **each** **rate-matched block**.

The identifier "$N_{blk}^{EBS(q)}$" is the **quantity** of **rate-matched blocks** in **rate-matched block sequence** "**q**". The variable "$N_{bit}^{RBtn(q)}$" is the **bitlength** of **rate-matched block** in the **sequential position** "$tn$" within the **rate-matched block sequence** "**q**". The **bit index** "$i_{rm}$" of **each rate-matched block** ranges from **0** to "$N_{bit}^{RBtn(q)}$-1".

**HARQ retransmissions** (click 4.5.6.6) **may** take place in **this stage**. In case of **retransmissions**, if there are **2 stored encoded block sequences** "**q=0**" and "**q=1**", this may result in **only 1 rate-matched block sequence** "**q=0**" or "**q=1**".

For more detail, click 4.12.6.

5. **Concatenate sequentially every rate-matched block** in **each** of the **1** or **2 sequence(s)** of **rate-matched block(s)** into **1** or **2 codeword(s)**.

   The variable "$cw^{(q)}(i_{cw})$" represents the **bit sequence** of the **codeword** "**q**", that was produced from the **transport block** "**q**". The variable "$i_{cw}$" is the **bit index** within **each codeword**.

   The identifier "$N_{bit}^{CW(q)}$" is the **bitlength** of the **codeword** "**q**", where "$N_{bit}^{CW(q)}=\Sigma_{tn}N_{bit}^{RBtn(q)}$". The **bit index** "$i_{cw}$" of **each codeword** ranges from **0** to "$N_{bit}^{CW(q)}$-1".

   For more detail, click 4.12.7.

6. **PUSCH only** and **optional: multiplex 1 PUSCH codeword** and/or **1 PUCCH codeword** into **1 multiplexed codeword** for its **delivery** in **1 PUSCH transmission**.

   The variable "$mcw^{(0)}(i_{mcw})$" represents the **bit sequence** of the **multiplexed codeword** "**0**", that was produced from the **PUSCH transport block** "**0**". The variable "$i_{mcw}$" is the **bit index** within the **multiplexed codeword** "**0**".

   The identifier "$N_{bit}^{MCW(0)}$" is the **bitlength** of the **multiplexed codeword** "**0**", where "$N_{bit}^{MCW(0)}>N_{bit}^{CW(0)}$". The **bit index** "$i_{mcw}$" of the **multiplexed codeword** ranges from **0** to "$N_{bit}^{MCW(q)}$-1".

   For more detail, click 4.12.8.

7. **Scramble each codeword** or **multiplexed codeword** into **1 scrambled codeword**.

   For **PDSCH** it is **1** or **2 codeword(s)**. For **PUSCH** it is **either 1 codeword** or **1 multiplexed codeword**.

   The variable "$scw^{(q)}(i_{scw})$" represents the **bit sequence** of the **scrambled codeword** "**q**", that was produced from the **transport block** "**q**". The variable "$i_{scw}$" is the **bit index** within **each scrambled codeword**.

   The identifier "$N_{bit}^{SCW(q)}$" is the **bitlength** of the **scrambled codeword** "**q**". The **bit index** "$i_{scw}$" of the **scrambled codeword** ranges from **0** to "$N_{bit}^{MCW(q)}$-1".

   The **bitlength** "$N_{bit}^{SCW(q)}$" is the **same** as the **bitlength** of its corresponding **codeword** "$N_{bit}^{CW(q)}$", or **multiplexed codeword** "$N_{bit}^{MCW(0)}$".

   For more detail, click 4.12.9.

8. **Modulate each** of the **1** or **2 scrambled codeword**(**s**) into **1** or **2 modulated sequence**(**s**) of **complex-valued symbols**.

   The variable "$ms^{(q)}(j_{ms})$" represents the **symbol sequence** of the **modulated sequence** "**q**", that was produced from the **transport block** "**q**". The variable "$j_{ms}$" is the **symbol index** within **each modulated sequence**.

   The identifier "$N_{symb}^{MS(q)}$" is the **symbol length** of the **modulated sequence** "**q**". The **symbol index** "$j_{ms}$" of the **modulated sequence** ranges from **0** to "$N_{symb}^{MS(q)}$-1".

   For more detail, click 4.12.10.

9. **Map** the **1** or **2 modulated sequence**(**s**) into "$N_{layer}$" **layer sequence**(**s**).

   The variable "$lys^{(lay)}(j_{lys})$" represents the **symbol sequence** of the **layer sequence** corresponding to the variable "$lay$". The variable "$j_{lys}$" is the **symbol index** within **each layer sequence**.

   The identifier "$N_{layer}$" is the **quantity** of **layers**. The variable "$lay$" ranges from **0** to "$N_{layer}$-1". The identifier "$N_{symb}^{layer}$" is the **symbol length** of **every layer sequence**. The **symbol index** "$j_{lys}$" of the **layer sequence** ranges from **0** to "$N_{symb}^{layer}$-1".

   For more detail, click 4.12.11.

10. **PUSCH only** and **optional**: **transform-precode** the **only layer sequence**.

    A **Discrete Fourier Transform** is applied to the **only symbol sequence** "$lys^{(0)}(j_{lys})$" of the **only layer** "**0**".

    The result is **1 antenna port sequence** "$aps^{(ap0)}(j_{aps})$" of **complex valued symbols**. the **symbol index** "$j_{aps}$" of **every sequence** ranges from **0** to "$N_{symb}^{layer}$".

    **PUSCH transform-precoding** (click 4.12.12) and **PUSCH precoding** (click 4.12.13) **cannot** be **both enabled**, but **both** can be **disabled**.

    For more detail, click 4.12.12.

11. **Precode** the "$N_{layer}$" **layer sequence**(**s**) into "$N_{port}$" **precoded antenna port sequence**(**s**) (**optional**).

    The variable "$aps^{(ap)}(j_{aps})$" represents the **symbol sequence** of the $aps^{(ap)}(j_{aps})$ corresponding to the variable "$ap$". The variable "$j_{aps}$" is the **symbol index** within **each precoded antenna port sequence**.

    The identifier "$N_{port}$" is the **quantity** of **antenna ports**. The variable "$ap$" ranges from "$ap_0$" to "$ap_{Nport-1}$", **each** of which is an **antenna port number**. The **symbol index** "$j_{aps}$" of the **precoded antenna port sequence** ranges from **0** to "$N_{symb}^{layer}$-1".

    Notice that the **quantity** of **symbols** "$N_{symb}^{layer}$" in **each** and **every symbol sequence** remains the **same** after **precoding**.

    **PUSCH precoding** and **PUSCH transform-precoding** (click 4.12.12) **cannot** be **both enabled**, but **both** can be **disabled**.

    For more detail, click 4.12.13.

12. **Map each symbol** of **each** of the "$N_{port}$" **antenna port sequences** "$aps^{(ap)}(j_{aps})$" to **1 payload virtual resource elements** (**VREs**) in **port** "$(vsc, si, ap)_{\mu}$ of the

**same antenna port number** "*ap*". **All** the **payload VREs** in **port** are **scheduled** for **1 PDSCH transmission** or for **1 PUSCH transmission**.

There are "$N_{layer}$" **sets** of **payload VREs** in **port**. **Each** and **every set** of **payload VREs** in **port** contains "$N_{symb}^{layer}$" **payload symbols**.

The variable "*vsc*" is the **virtual subcarrier index**, "*sc*" is the **subcarrier index**, "*si*" is the **symbol interval index** and "*ap*" is the **antenna port number**.

For more detail, click 4.12.14.

13. **Map each payload Virtual Resource Element (VRE)** in **port** "(*vsc*, *si*, *ap*)" to **1 payload Physical Resource element (PRE)** in **port** "(*sc*, *si*, *ap*)" of the **same antenna port number** "*ap*".

    There are "$N_{layer}$" **sets** of **payload PREs** in **port**. **Each** and **every set** of **payload PREs** in **port** contains "$N_{symb}^{layer}$" **payload symbols**.

    For more detail, click 4.12.15.

In case of **2 PDSCH transport blocks**, they **can** have a **different bitlength** and/or a **different CRC bitlength** and/or a **different maximum code block bitlength** and/or a **different quantity** of **code blocks** and/or a **different code block bitlength** and/or a **different LDPC base graph** and/or a **different code rate** and/or a **different quantity** of **encoded blocks** and/or a **different encoded block bitlength** and/or a **different modulation**.

In case of **HARQ retransmitted data** and **2 retransmitted sequences** of **encoded blocks**, they **can** have a **different bitlength** and/or a **different LDPC base graph** and/or a **different code rate** and/or a **different quantity** of **encoded blocks** and/or a **different encoded block bitlength** and/or a **different modulation**.

For a **mapping** between the **terms** (e.g., **protected block**) and the **symbols** (e.g., "$N_{blk}^{CBS(q)}$") of **PDSCH** and **PUSCH processing** used in **this book** and the corresponding **terms** and **symbols** (e.g., ***A***, ***C***, …) used in the **TS documents** click 4.12.17.

For the **NR node functional blocks**, the **NR node physical functions**, the **payload processing** of the **physical binary channels**, the **protocol architecture** of the **New Radio** and the **data flow** through the **protocol architecture**, click 4.3.

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_PDSCH.html

### 4.12.3 PDSCH and PUSCH transport block CRC appending

This consists of **appending** a "$CRC^{PB(q)}$" to **each** of the **1 or 2 transport block(s)** to produce **1** or **2 protected** a "$CRC^{PB(q)}$" to **each** of the **1 or 2 transport block(s)** resulting in **1** or **2 protected block(s)**.

The variable "$tb^{(q)}(i_{tb})$" represents the **bit sequence** of the **transport block** "**q**". The variable "$i_{tb}$" is the **bit index** within **each transport block**.

The identifier "$N_{bit}^{TB(q)}$" is the **bitlength** of **transport block** "**q**". The **bit index** "$i_{tb}$" of **each transport block** ranges from **0** to "$N_{bit}^{TB(q)}$-1".

The variable "$pb^{(q)}(i_{pb})$" represents the **bit sequence** of the **protected block** "**q**", that was produced from the **transport block** "**q**". The variable "$i_{pb}$" is the **bit index** within **each protected block**.

The identifier "$N_{bit}^{PB(q)}$" is the **bitlength** of the **protected block** "**q**". The **bit index** "$i_{pb}$" of **each protected block** ranges from **0** to "$N_{bit}^{PB(q)}$-1".

The variable "**q**$\in\{0,1\}$" is the identifier of the **1** or **2 transport block**(**s**) in **each PDSCH transmission**. The variable "**q**=0" is the identifier of the **transport block** in **each PUSCH transmission**. The variable "**q**" is the **inherited** identifier of **all** the **blocks** and **codewords** produced from **transport block** "**q**".

**All** the **bits** of **each transport block** "**q**" are used to calculate the "$\mathbf{CRC}^{\mathbf{PB(q)}}$" that is included at the **end** of its corresponding **protected block**.

The **bitlength** of **protected block** "**q**" is the **bitlength** "$N_{bit}^{TB(q)}$" of **transport block** "**q**" plus the **bitlength** "$N_{bit}^{CRC\text{-}PB(q)}$" of its **CRC**:

$$N_{bit}^{PB(q)} = N_{bit}^{TB(q)} + N_{bit}^{CRC\text{-}PB(q)}$$

The **bitlength** "$N_{bit}^{CRC\text{-}PB(q)}$" of "$\mathbf{CRC}^{\mathbf{PB(q)}}$" and its **generator polynomial** depends on the **bitlength** "$N_{bit}^{TB(q)}$" of the **transport block** "**q**", as follows:

- ***If*** $N_{bit}^{TB(q)} \leq$ **3,824** bits (478 bytes), ***then***,

  $N_{bit}^{CRC\text{-}PB(q)} =$ **16 bits**, and the **generator polynomial** is $g_{CRC16}(D)$ (refer to clause 5.1 of TS 38.212).

- ***If*** $N_{bit}^{TB(q)} >$ **3,824** bits (478 bytes), ***then***,

  $N_{bit}^{CRC\text{-}PB(q)} =$ **24 bits**, and the **generator polynomial** is $g_{CRC24}(D)$ (refer to clause 5.1 of TS 38.212).

For a **mapping** between the **terms** (e.g., **protected block**) and the **symbols** (e.g., "$N_{blk}^{CBS(q)}$") of **PDSCH** and **PUSCH processing** used in **this book** and the corresponding **terms** and **symbols** (e.g., ***A***, ***C***, …) used in the **TS documents** click 4.12.17.

The procedure for **appending** a **CRC** to **each transport block** is **identical** for **PUSCH** and for **PDSCH**, with the subtle difference that for **PUSCH** it is always **only 1 transport block** and **only 1 codeword**.

For more detail on **appending** a **CRC** to **each transport block**, refer to clauses 5.1 (general), 6.2.1 (for **PUSCH**) and 7.2.1 (for **PDSCH**) of TS 38.212.

### 4.12.4 PDSCH and PUSCH code block segmentation and CRC appending

This consists of **transforming each** of the **1** or **2 protected block**(**s**) into **1** or **2 sequences** of "$N_{blk}^{CBS(q)}$" **code block**(**s**), where **each code block** includes its **specific CRC**.

The variable "$cb_{cn}^{(q)}(i_{cb})$" represents the **bit sequence** of the **code block** in the **sequential position** "***cn***" within the **code block sequence** "**q**", that was produced from **transport block** "**q**". The variable "***cn***" identifies the **sequential position** of **each code block** within its **code block sequence**. The variable "$i_{cb}$" is the **bit index** within **each code block**.

The identifier "$N_{blk}^{CBS(q)}$" is the **quantity** of **code blocks** in **code block sequence** "**q**". The variable "***cn***" of **code block sequence** "**q**" ranges from **0** to "$N_{blk}^{CBS(q)}$-1". The identifier "$N_{bit}^{CB(q)}$" is the **bitlength** of **each** and **every code block** from **code block sequence** "**q**". The **bit index** "$i_{cb}$" of **each code block** ranges from **0** to "$N_{bit}^{CB(q)}$-1". Notice that **all** the **code blocks** "$cb_{cn}^{(q)}(i_{cb})$" from a given **code block sequence** have the **same bitlength** "$N_{bit}^{CB(q)}$".

The **maximum code block bitlength** "$N_{bit}^{CB(q)}{}_{max}$" (click 4.12.4.1) and the **bitlength** "$N_{bit}^{PB(q)}$" of the **protected block** "**q**" determine the **quantity** (click 4.12.4.2) of **code blocks** and their **bitlength** (click 4.12.4.3). The **quantity** of **code blocks**, their **bitlength** and the **contents** of the **protected block** "**q**" determine the **contents** (click 4.12.4.4) of the **code blocks**.

The **code block sequence** may be interpreted as being organized in **code block groups** (**CBGs**, click 4.12.4.5).

The procedure for the **segmentation** of a **protected block** into **code blocks** is **identical** for **PUSCH** and for **PDSCH** (except for the remark in the last paragraph of clause 6.2.3 of TS 38.212).

For more detail on **code block segmentation**, refer to clauses 5.1 and 5.2.2 (general), 6.2.3 (for **PUSCH**) and 7.2.3 (for **PDSCH**) of TS 38.212.

### 4.12.4.1 Maximum code block bitlength

The **maximum code block bitlength** "$N_{bit}^{CB(q)}{}_{max}$" of **code block sequence** "**q**" is **directly determined** by the **LDPC base graph** used for "**q**":

- If **LDPC base graph 1** is used for "**q**":

  $N_{bit}^{CB(q)}{}_{max} = \mathbf{8{,}448}$ bits

- If **LDPC base graph 2** is used for "**q**":

  $N_{bit}^{CB(q)}{}_{max} = \mathbf{3{,}840}$ bits

***If*** the **bitlength** "$N_{bit}^{PB(q)}$" of the **protected block** (click 4.12.3) "**q**" is **less than or equal to** the corresponding **maximum code block bitlength** "$N_{bit}^{CB(q)}{}_{max}$", ***then***, **no segmentation** is performed. ***Else***, **segmentation** is performed.

### 4.12.4.2 Quantity of code blocks

**Quantity of code blocks when no segmentation**

The **quantity** of **code blocks** "$N_{blk}^{CBS(q)}$" in the **code block sequence** "**q**", is **1**:

$$N_{blk}^{CBS(q)} = \mathbf{1}$$

$$cn = \mathbf{0}$$

**Quantity of code blocks when segmentation**

The **quantity** of **code blocks** "$N_{blk}^{CBS(q)}$" in the **code block sequence** "**q**", produced from **protected block** "**q**", is:

$$N_{blk}^{CBS(q)} = \mathbf{ceiling}(N_{bit}^{PB(q)} / (N_{bit}^{CB(q)}{}_{max} - 24))$$

$$cn \in \{0..N_{blk}^{CBS(q)}-1\}$$

where **24** is the **bitlength** "$N_{bit}^{CRC\text{-}CB}$" of the **specific CRC** "$CRC_n^{CB(q)}$" that is **included** within **every code block** that is produced by **segmentation**.

The way the **different variables** ("$N_{bit}^{TB(q)}$", "$N_{blk}^{CBS(q)}$", **LDPC base graph**, …) are **selected** and/or **computed guarantees** that "$N_{bit}^{PB(q)}/N_{blk}^{CBS(q)}$" is an **integer**, which is **required** by some processing operations.

For a **mapping** between the **terms** (e.g., **protected block**) and the **symbols** (e.g., "$N_{blk}^{CBS(q)}$") of **PDSCH** and **PUSCH processing** used in **this book** and the corresponding **terms** and **symbols** (e.g., ***A***, ***C***, …) used in the **TS documents** click 4.12.17.

### 4.12.4.3 Bitlength of each and every code block

**All** the **code blocks** in **each code block sequence** "**q**" have the **same bitlength** "$N_{bit}^{CB(q)}$", that complies with:

$$N_{bit}^{PB(q)} \leq N_{bit}^{CB(q)} \leq N_{bit}^{CB(q)}{}_{max}$$

The **bitlength** "$N_{bit}^{CB(q)}$" depends on the **LDPC base graph** "**q**" that is used to **FEC-encode** the **code blocks** in **sequence** "**q**", as follows:

- $N_{bit}^{CB(q)} = 22 \cdot LS_{minc}^{(q)}$ for **LDPC base graph 1**.
- $N_{bit}^{CB(q)} = 10 \cdot LS_{minc}^{(q)}$ for **LDPC base graph 2**.

The variable "$LS_{minc}^{(q)}$" is the **minimum value** of the **lifting size** "$LS$" that **complies** with the corresponding expression:

- ***If* no segmentation**

  $$LS_{minc}^{(q)} \bullet K_b^{(q)} \geq N_{bit}^{PB(q)}$$

- ***If* segmentation**

  $$LS_{minc}^{(q)} \bullet K_b^{(q)} \geq N_{bit}^{PB(q)} / N_{blk}^{CBS(q)} + 24$$

The **possible values** of the **lifting size** "$LS$" are provided in table 5.3.2-1 of TS 38.212:

| *Set of lifting sizes* ($LS$) | *Set index* ($i_{LS}$) |
|---|---|
| {2, 4, 8, 16, 32, 64, 128, 256} | 0 |
| {3, 6, 12, 24, 48, 96, 192, 384} | 1 |
| {5, 10, 20, 40, 80, 160, 320} | 2 |
| {7, 14, 28, 56, 112, 224} | 3 |
| {9, 18, 36, 72, 144, 288} | 4 |
| {11, 22, 44, 88, 176, 352} | 5 |
| {13, 26, 52, 104, 208} | 6 |
| {15, 30, 60, 120, 240} | 7 |

Notice that "$N_{bit}^{PB(q)} / N_{blk}^{CBS(q)} + 24$" is the **minimum possible code block bitlength**, and the **actual code block bitlength** "$N_{bit}^{CB(q)}$" is most frequently **larger** than it.

The value of "$K_b^{(q)}$" is an **integer** determined as follows:

- For **LDPC base graph 1**

  $$K_b^{(q)} = 22$$

- For **LDPC base graph 2**

  `If` $N_{bit}^{PB(q)} > 640$, then $K_b^{(q)} = 10$

  `elseif` $N_{bit}^{PB(q)} > 560$, then $K_b^{(q)} = 9$

  `elseif` $N_{bit}^{PB(q)} > 192$, then $K_b^{(q)} = 8$

  `else` $K_b^{(q)} = 6$

For a **mapping** between the **terms** (e.g., **protected block**) and the **symbols** (e.g., "$N_{blk}^{CBS(q)}$") of **PDSCH** and **PUSCH processing** used in **this book** and the corresponding **terms** and **symbols** (e.g., $A$, $C$, …) used in the **TS documents** click 4.12.17.

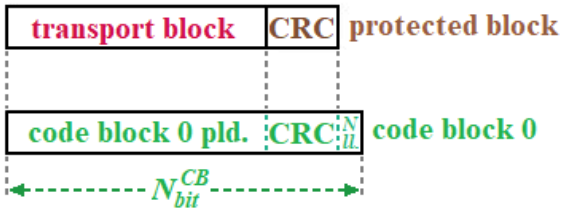
### 4.12.4.4 Contents of the code blocks

**Contents of the code blocks when no segmentation:**

The **bit sequence** "$cb_0^{(q)}(i_{cb})$" of the **only** **code block** that belongs to **code block sequence** "**q**", has the following content:

1. Its **first** "$N_{bit}^{TB(q)}$" bits are its **payload** and are the same as the **transport block** "**q**".
2. **Followed** by the **16** or **24** bits of the **CRC** "$CRC^{PB(q)}$" of the **protected block**.
3. **Followed** by the **required quantity** of **padding** "***Null***" **bits** up to the **code block bitlength** "$N_{bit}^{CB(q)}$". This special "***Null***" **value** will be used when producing the **rate-matched blocks** (click 4.12.6.3).

The following figure shows **1 protected block** that has been **mapped** into **1 code block**:




Expressing the above explanation in a formal way:

$cb_0^{(q)}(i) = pb^{(q)}(i) = tb^{(q)}(i)$ for $i \in \{0..(N_{bit}^{TB(q)}-1)\}$

$cb_0^{(q)}(i) = pb^{(q)}(i)$ for $i \in \{N_{bit}^{TB(q)}..(N_{bit}^{TB(q)}+N_{bit}^{CRC\text{-}PB(q)}-1)\}$

$cb_0^{(q)}(i = Null$ for $i \in \{N_{bit}^{PB(q)}..(N_{bit}^{CB(q)}-1)\}$

Notice that:

$N_{bit}^{PB(q)} = N_{bit}^{TB(q)} + N_{bit}^{CRC\text{-}PB(q)}$

$CRC^{CB0(q)} = CRC^{PB(q)}$

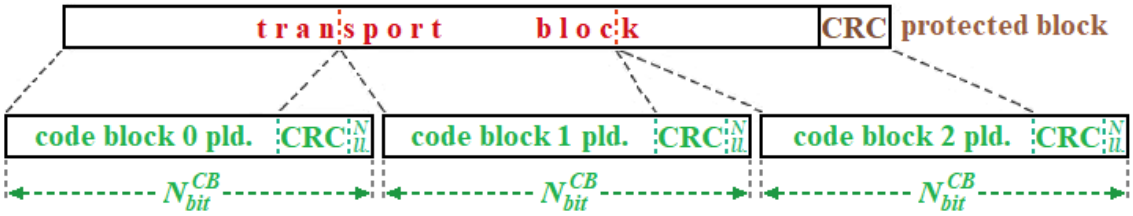
$N_{bit}^{CRC\text{-}CB(q)} = N_{bit}^{CRC\text{-}PB(q)} \in \{16,24\}$

**Contents of the code blocks when segmentation:**

The **bit sequence** "$cb_{cn}^{(q)}(i_{cb})$" of **code block** with **sequence** **index** "$n$", that belongs to **code block sequence** "**q**", has the following content:

1. Its **first** "$N_{bit}^{PB(q)}/N_{blk}^{CBS(q)}$" bits are its **payload**, and are taken, **sequentially**, from **protected block** "**q**".
2. **Followed** by its **24 CRC bits** "$crc_{cn}^{CB(q)}(i_{crccbcn})$", which are computed over the "$N_{bit}^{PB(q)}/N_{blk}^{CBS(q)}$" **payload** bits, using the generator polynomial "$g_{CRC24}(D)$" (refer to clause 5.1 of TS 38.212). These bits are
3. **Followed** by the **required quantity** of **padding** "***Null***" **bits** up to the **bitlength** "$N_{bit}^{CB(q)}$" of **every** **code block** that belongs to **sequence** "**q**". This special "***Null***" **value** will be used when producing the **rate-matched blocks** (click 4.12.6.3).

The following figure shows **1 protected block** that has been **segmented** into **3 code blocks**:




Expressing the above explanation in a formal way (pseudo code):

$i_{pb} = 0$ ;
**for** $cn = 0$ **to** $N_{blk}^{CBS(q)}-1$
  **for** $i_{cb} = 0$ **to** $N_{bit}^{PB(q)}/N_{blk}^{CBS(q)}-1$
    $cb_{cn}^{(q)}(i_{cb}) = pb(i_{pb})$
    $i_{pb} = i_{pb} + 1$
  **end for**
  **for** $i_{crccbcn} = 0$ **to** 23
    $cb_{cn}^{(q)}(N_{bit}^{PB(q)}/N_{blk}^{CBS(q)}+i_{crccbcn}) = crc_{cn}^{CB(q)}(i_{crccbcn})$
  **end for**
  **for** $i_{cb} = N_{bit}^{PB(q)}/N_{blk}^{CBS(q)}+24$ **to** $N_{bit}^{CB(q)}-1$
    $cb_{cn}^{(q)}(i_{cb}) = Null$
  **end for**
**end for**

Some **relevant comments** on this **pseudo-code** are:

- **Every code block** has the **same payload bitlength** "$N_{bit}^{PB(q)}/N_{blk}^{CBS(q)}$" because "$N_{bit}^{PB(q)}/N_{blk}^{CBS(q)}$" is **always** an **integer**. **Likewise**, **every code block** has the **same quantity** of ***Null*** bits.
- The **code block payload**:

  $cb_{cn}^{(q)}(0)$, $cb_{cn}^{(q)}(1)$, …, $cb_{cn}^{(q)}(N_{bit}^{PB(q)}/N_{blk}^{CBS(q)}-1)$

  of **each code block** is used to calculate its **specific CRC** bits:

  $crc_{cn}^{CB(q)}(0)$, …, $crc_{cn}^{CB(q)}(23)$,

  using the **generator polynomial** "$g_{CRC24}(D)$" (refer to clause 5.1 of TS 38.212).

For a **mapping** between the **terms** (e.g., **protected block**) and the **symbols** (e.g., "$N_{blk}^{CBS(q)}$") of **PDSCH** and **PUSCH processing** used in **this book** and the corresponding **terms** and **symbols** (e.g., ***A***, ***C***, …) used in the **TS documents** click 4.12.17.

### 4.12.4.5 Code block groups (CBG)

If **CBG mode** (click 4.5.6.4) is configured, the **code blocks** in **each sequence** "**q**" are **considered** as being **grouped** in **code block groups** (**CBGs**). However, there is **no modification** to **any bit value**. Consequently, the **CBGs only** exist **implicitly**, because they are **jointly considered** as such, **both** by the **HARQ transmitter** and by the **HARQ**

**receiver**, for the purpose of **HARQ acknowledgement** (click 4.5.6.7) and of **HARQ retransmission** (click 4.5.6.6).

For more detail on **code block groups**, refer to clauses 5.1.7.1 (for **PDSCH**) and 6.1.5.1 (for **PUSCH**) of TS 38.214.

For the **maximum quantity** of **CBGs** for **each transport block**, click 4.5.6.4.

### 4.12.5 PDSCH and PUSCH code block LDPC encoding

This consists of **LDPC encoding** **each code block** into **1 encoded block**, resulting in **1** or **2 sequences** of "$N_{blk}^{CBS(q)}$" **encoded block(s)**, that are **stored** in the **HARQ Tx buffer**.

The variable "$eb_{cn}^{(q)}(i_{eb})$" represents the **bit sequence** of the **encoded block** in the **sequential position** "$cn$" within the **complete encoded block sequence** "**q**", that was produced from **transport block** "**q**". The variable "$cn$" identifies the **sequential position** of **each code block** within its **complete encoded block sequence**.

The variable "$eb_{tn}^{(q)}(i_{eb})$" represents the **bit sequence** of the **encoded block** in the **sequential position** "$tn$" within the **transmitted encoded block sequence** "**q**", that was produced from **transport block** "**q**". The variable "$tn$" identifies the **sequential position** of **each encoded block** within its **transmitted encoded block sequence**.

The variable "$i_{eb}$" is the **bit index** within **each encoded block**.

The identifier "$N_{blk}^{CBS(q)}$" is the **quantity** of **encoded blocks** in the **complete encoded block sequence** "**q**", which is the same as the **quantity** of **code blocks** in **code block sequence** "**q**".

The identifier "$N_{blk}^{EBS(q)}$" is the **quantity** of **encoded blocks** in the **transmitted encoded block sequence** "**q**". For a **new transmission**, "$N_{blk}^{EBS(q)}=N_{blk}^{CBS(q)}$". For a **retransmission**, "$N_{blk}^{EBS(q)}\leq N_{blk}^{CBS(q)}$".

The variable "$tn$" of the **transmitted encoded block sequence** "**q**" ranges from **0** to "$N_{blk}^{EBS(q)}$-1". The identifier "$N_{bit}^{EB(q)}$" is the **bitlength** of **each** and **every encoded block** from **encoded block sequence** "**q**". The **bit index** "$i_{eb}$" of **each encoded block** ranges from **0** to "$N_{bit}^{EB(q)}$-1". Notice that **all** the **encoded blocks** from a given **encoded block sequence** (either complete or transmitted) "**q**" have the **same bitlength** "$N_{bit}^{CB(q)}$".

The value of "$N_{bit}^{EB(q)}$", is **substantially larger** than the value of "$N_{bit}^{CB(q)}$".

The **bitlength** "$N_{bit}^{EB(q)}$ of **every encoded block** "$eb_{cn}^{(q)}(i_{eb})$ that belongs to the **encoded block sequence** "**q**" is:

- $N_{bit}^{EB(q)}=66LS_{minc}^{(q)}$, when **LDPC base graph 1** is used for "**q**".
- $N_{bit}^{EB(q)}=50LS_{minc}^{(q)}$, when **LDPC base graph 2** is used for "**q**".

where using **either LDPC base graph 1** or **2** for **each** value of "**q**" was determined in 6.14.6, and the value of $LS_{minc}^{(q)}$, was determined in 4.12.4.

The following table presents a summary of some values up to now:

| | Code block $N_{bit}^{CB(q)}$ [bits] | Encoded block $N_{bit}^{EB(q)}$ [bits] | Added [bits] | Code rate |
|---|---|---|---|---|
| **Base graph 1** | $22LS_{minc}^{(q)}$ | $66LS_{minc}^{(q)}$ | $44LS_{minc}^{(q)}$ | 1/3 |
| **Base graph 2** | $10LS_{minc}^{(q)}$ | $50LS_{minc}^{(q)}$ | $40LS_{minc}^{(q)}$ | 1/5 |

The **LDPC channel coding** of **each code block** in **this stage** has the following **4 steps**:

1. **Operating** with the ***Null*** bits
2. **Determine** the **lifting size set index** "$i_{LS}$"
3. **Generate** the **complete LDPC code bits**
4. **Copy** the **complete LDPC code bits** at the **end** of the **encoded block**

**Each** of these **steps** is now explained in more **detail**.

1. **Operating** with the ***Null*** bits

   This consists of **a**) **copying** the **last** "$(N_{bit}^{CB(q)}-2LS_{minc}^{(q)})$" **bits** from "$cb_{cn}^{(q)}(i_{cb})$" at the **beginning** of "$eb_{cn}^{(q)}(i_{eb})$" and **b**) obtaining "$cb_{cn}'^{(q)}(i_{cb})$" by **replacing each *Null*** bit in "$cb_{cn}^{(q)}(i_{cb})$" by a "**0**". Notice that the ***Null*** bits in "$cb_{cn}^{(q)}(i_{cb})$" may **only** be located at its **final part**, **starting** at its **bit index** "$2LS_{minc}^{(q)}$".

   Expressing the above explanation in a **formal way** (pseudo-code):

   ```
   for i_cb=2LS_minc^(q) to N_bit^CB(q)-1
      eb_cn^(q)(i_cb-2LS_minc^(q)) = cb_cn^(q)(i_cb)
      if cb_cn^(q)(i_cb) = Null
         cb_cn'^(q)(i_cb) = 0
      end if
   end for
   for i_cb=0 to 2LS_minc^(q)-1
      cb_cn'^(q)(i_cb) = cb_cn^(q)(i_cb)
   end for
   ```

   Notice that the **first** "$2LS_{minc}^{(q)}$" **bits** of **each code block** "$cb_{cn}^{(q)}(i_{cb})$" have **not** been **copied** into the **encoded block** "$eb_{cn}^{(q)}(i_{eb})$". However, their **information** does **not disappear**, and it is **present** (in an **encoded** way) within the **trailing** "$N_{bit}^{EB(q)}-(N_{bit}^{CB(q)}-2LS_{minc}^{(q)})$" **LDPC code bits** of the **encoded block** "$eb_{cn}^{(q)}(i_{eb})$".

   Recall that the ***Null*** bits were **introduced** in the **stage** of **segmenting** into **code blocks** (click 4.12.4).

2. **Determine** the **lifting size set index** "$i_{LS}^{(q)}$"

   This consists of finding the **row** that contains "$LS_{minc}^{(q)}$" in the following table (table 5.3.2-1 of TS 38.212), and obtain the resulting value of the **lifting size set index** "$i_{LS}$":

| *Set of lifting sizes* ($LS$) | *Set index* ($i_{LS}$) |
|---|---|
| {2, 4, 8, 16, 32, 64, 128, 256} | 0 |
| {3, 6, 12, 24, 48, 96, 192, 384} | 1 |
| {5, 10, 20, 40, 80, 160, 320} | 2 |
| {7, 14, 28, 56, 112, 224} | 3 |
| {9, 18, 36, 72, 144, 288} | 4 |
| {11, 22, 44, 88, 176, 352} | 5 |
| {13, 26, 52, 104, 208} | 6 |
| {15, 30, 60, 120, 240} | 7 |

3. **Generate** the **LDPC code bits**

This consists of **finding** the **quantity** "$N_{bit}^{EB(q)}$-($N_{bit}^{CB(q)}$-2$LS_{minc}^{(q)}$)" of **LDPC code bits**:

$$\boldsymbol{l}_{cn}^{(q)}=[\,\boldsymbol{l}_{cn}^{(q)}(0),\boldsymbol{l}_{cn}^{(q)}(1),\boldsymbol{l}_{cn}^{(q)}(2),\ldots,\boldsymbol{l}_{cn}^{(q)}(N_{bit}^{EB(q)}-(N_{bit}^{CB(q)}-2LS_{minc}^{(q)})-1)]^{T}$$

that comply with $\mathbf{H}^{(q)}\times\begin{bmatrix}cb'^{(q)}_{cn}\\ \boldsymbol{l}^{(q)}_{cn}\end{bmatrix}=[\mathbf{0}]$

where:

- The matrix "$\mathbf{H}^{(q)}$" is obtained from the matrix "$\mathbf{H_{BG}}^{(q)}$" (see below).
- $cb_{cn}{}'^{(q)}=[cb_{cn}{}'^{(q)}(0),cb_{cn}{}'^{(q)}(1),cb_{cn}{}'^{(q)}(2),\ldots,cb_{cn}{}'^{(q)}(N_{bit}^{CB(q)}-1)]^{T}$
- [**0**] is a **column vector** with **all** its elements equal to **0**.
- The operations are performed in **2-element finite arithmetic** "Galois Field(2)".

Notice that the **quantity** "$N_{bit}^{EB(q)}$-$N_{bit}^{CB(q)}$+2$LS_{minc}^{(q)}$" of **LDPC code bits exceeds** in "2$LS_{minc}$" **bits** the quantity "$N_{bit}^{EB(q)}$-$N_{bit}^{CB(q)}$", which is the **difference** between the **bitlength** of an **encoded block** and of a **code block**. The reason (as was pointed out at the **end** of the **previous step**) is that the **first** "2$LS_{minc}^{(q)}$" **bits** of **each code block** "$cb_{cn}^{(q)}(i_{cb})$" have **not** been **copied** into the **encoded block** "$eb_{cn}^{(q)}(i_{eb})$".

The matrix $\mathbf{H_{BG}}^{(q)}$ depends on the **base graph** of **encoded block sequence** "**q**". For **LDPC base graph 1**, the matrix $\mathbf{H_{BG}}^{(q)}$ has **46 rows** (with row indices ***ro***=0 to 45) and **68 columns** (with column indices "***co***=0 to 67"). For **LDPC base graph 2**, the matrix $\mathbf{H_{BG}}^{(q)}$ has **42 rows** (with row indices "***ro***=0 to 41") and **52 columns** (with column indices ***co***=0 to 51).

The matrix $\mathbf{H}^{(q)}$ is obtained by **replacing each element** (having **row index** "***ro***" and **column index** "***co***") in "$\mathbf{H_{BG}}^{(q)}$" with a "$LS_{minc}^{(q)}\times LS_{minc}^{(q)}$" **matrix**, according to the following:

- If the values of "***ro***" and "***co***" are **not** included in table 5.3.2-2 (for **LDPC base graph 1**) or table 5.3.2-3 (for **LDPC base graph 2**), both of TS 38.212, the **element** is **replaced** by an **all zero** matrix [**0**]. **Both tables** are **parameterized** with "$i_{LS}^{(q)}$", whose value has been obtained in the **previous step**.
- Otherwise, the **element** is **replaced** by a **circular permutation matrix** "$\mathbf{I}(P_{ro,co})$".

  The matrix "$\mathbf{I}(P_{ro,co})$" is obtained by **circularly shifting** "$P_{ro,co}$" times to the **right** the **identity matrix** "**I**" of dimensions "$LS_{minc}^{(q)}\times LS_{minc}^{(q)}$".

  The value of "$P_{ro,co}$" is "$V_{ro,co}$ ***mod*** $LS_{minc}^{(q)}$".

  The value of "$V_{ro,co}$",is obtained from the corresponding table 5.3.2-2 or 5.3.2-3 of TS 38.212, as follows:

  - The values of "***i***" and "***j***" in the table are the **row** "***ro***" and **column** "***co***" indices of the **element** from "$\mathbf{H_{BG}}^{(q)}$" that is being replaced (see above).
  - **Both tables** are **parameterized** with "$i_{LS}$", whose value "$i_{LS}^{(q)}$" was obtained in the **previous step**.

4. **Copy all** the **LDPC code bits** at the **end** of the **encoded block**

   This consists of **copying all** the "$N_{bit}^{EB(q)}$-($N_{bit}^{CB(q)}$-$2LS_{minc}^{(q)}$)" **LDPC code bits** "$l_{cn}^{(q)}(i)$" into the **same quantity** of the **last bits** of the **encoded block** "$eb_{cn}^{(q)}(i_{eb})$".

   This is done with the **pseudo-code**:

   ```
   for i =0 to N_bit^EB(q)-(N_bit^CB(q)-2LS_minc^(q))-1
      eb_cn^(q)(i +(N_bit^CB(q)-2LS_minc^(q))) = l_cn^(q)(i)
   end for
   ```

   Notice that:

   - The **first** "$N_{bit}^{CB(q)}$-$2LS_{minc}^{(q)}$" **bits** of the **encoded block** "$eb_{cn}^{(q)}(i_{eb})$" are the **last** bits from the **code block** "$cb_{cn}^{(q)}(i_{cb})$".
   - The **first** "$N_{bit}^{CB(q)}$-2·$LS_{minc}^{(q)}$" **bits** of the **encoded block** "$eb_{cn}^{(q)}(i_{eb})$" **may** contain the special value ***Null*** (copied from "$cb_{cn}^{(q)}(i_{cb})$"). This special "***Null***" value is **suppressed** in the **bit selection step** (click 4.12.6.3) within the **rate-matching stage** (click 4.12.6).

The resulting **1** or **2 sequence(s)** of **encoded block(s)** is/are **stored** in a **Tx buffer** for its/their possible use in a **HARQ retransmission** (click 4.5.6.6).

For a **mapping** between the **terms** (e.g., **protected block**) and the **symbols** (e.g., "$N_{blk}^{CBS(q)}$") of **PDSCH** and **PUSCH processing** used in **this book** and the corresponding **terms** and **symbols** (e.g., ***A***, ***C***, …) used in the **TS documents** click 4.12.174.12.10.

The procedure for **LDPC channel coding** is **identical** for **PUSCH** and for **PDSCH**.

For more detail on **LDPC channel coding**, refer to clause 5.3.2 (general), 6.2.4 (for **PUSCH**) and 7.2.4 (for **PDSCH**) of TS 38.212.

For the **general LDPC channel coding techniques**, click 11.2.3.3.

## 4.12.6 PDSCH and PUSCH LDPC rate-matching

This consists of **LDPC rate-matching** **each** **transmitted** **encoded block** into **1 rate-matched block**, resulting in **1** or **2 sequences** of "$N_{blk}^{EBS(q)}$" **rate-matched blocks**.

The variable "$rb_{tn}^{(q)}(i_{rm})$" represents the **bit sequence** of the **rate-matched block** in the **sequential position** "***tn***" within the **rate-matched block sequence** "**q**", that was produced from **transport block** "**q**". The variable "***tn***" identifies the **sequential position** of **each** **rate-matched block** within its **rate-matched block sequence**. The variable "$i_{rm}$" is the **bit index** within **each** **rate-matched block**.

The identifier "$N_{blk}^{EBS(q)}$" is the **quantity** of **rate-matched blocks** in **rate-matched block sequence** "**q**". The variable "$N_{bit}^{RBtn(q)}$" is the **bitlength** of **rate-matched block** in the **sequential position** "***tn***" within the **rate-matched block sequence** "**q**". The **bit index** "$i_{rm}$" of **each** **rate-matched block** ranges from **0** to "$N_{bit}^{RBtn(q)}$-1".

**HARQ retransmissions** (click 4.5.6.6) **may** take place in **this stage**. In case of **retransmissions**, if there are **2** **stored** **encoded block sequences** "**q=0**" and "**q=1**", this may result in **only** **1 rate-matched block sequence** "**q=0**" or "**q=1**".

The **quantity** of **rate-matched block(s)** "$N_{blk}^{RBS(q)}$" produced in this **stage** depends on the **HARQ mode** (click 4.5.6.4):

- In **HARQ transport block mode**, "$N_{blk}^{RBS(q)} = N_{blk}^{EBS(q)} = N_{blk}^{CBS(q)}$" for **each** of the corresponding **1** or **2 unacknowledged transport block(s)**.
- In **HARQ CBG mode**, **only** the **rate-matched block(s)** that correspond to **unacknowledged CBGs** are **produced** and **transmitted**.

For more detail, refer to clauses 5.1.7.2 and 6.1.5.2 of TS 38.214.

The **rate-matching** procedure consists of **deciding how many bits**, and **which ones**, are **taken** from **each encoded block** into its **corresponding rate-matched block**.

The **rate-matching procedure** has the **3** following **steps**:

1. **Bitlength determination** (click 4.12.6.2)

   This consists of determining the **bitlength** "$N_{bit}^{RBtn(q)}$"of **each** and **every rate-matched block** "$rb_{tn}^{(q)}(i_{rm})$" in the **1** or **2 rate-matched block sequence(s)**.

2. **Bit selection** (click 4.12.6.3)

   This consists of determining the **bit index** "$i_{eb}$=$k_0$" for **each** of the "$N_{blk}^{RBS(q)}$" **rate-matched block(s)** "$rb_{tn}^{(q)}(i_{rm})$". This **bit index** "$i_{eb}$=$k_0$" is the **first bit** in the corresponding **encoded block** "$eb_m^{(q)}(i_{eb})$" starting from which a **bit sequence** of "$N_{bit}^{RBtn(q)}$" **non-*Null* bits** is **circularly extracted**. This **bit sequence** of "$N_{bit}^{RBtn(q)}$" **non-*Null* bits** is the **rate-matched block**.

   The **bit index** "$i_{eb}$=$k_0$" depends on the **redundancy version** "$rv_{id}$" of the corresponding **transport block**.

3. **Bit interleaving** (click 4.12.6.4)

   This consists of **assigning round-robin** the **bits** of **each rate-matched block** among the **transmitted symbols** (for time **diversity**, click 11.8), which results in **bit interleaving**.

For a **mapping** between the **terms** (e.g., **protected block**) and the **symbols** (e.g., "$N_{blk}^{CBS(q)}$") of **PDSCH** and **PUSCH processing** used in **this book** and the corresponding **terms** and **symbols** (e.g., ***A***, ***C***, …) used in the **TS documents** click 4.12.174.12.10.

The procedure for **rate-matching** is **identical** for **PUSCH** and for **PDSCH**, but **some** of the **variables** are **different**, as presented in 4.12.6.1.

The **LDPC rate-matching procedure** described in this book is the **option** that does **not use layered bit-rate matching**. (i.e., $I_{LBRM}$=0). A detailed explanation on **rate-matching** may be found in "*An overview of channel coding for 5G NR cellular communications*", Jung Hyun Bae, Ahmed Abotabl, Hsien-Ping Lin, Kee-Bong Song and Jungwon Lee, Industrial Technology Advances, vol 8, 2019.

For more detail on **LDPC rate-matching**, refer to clauses 5.4.2 (general), 6.2.5 (for **PUSCH**) and 7.2.5 (for **PDSCH**) of TS 38.212.

### 4.12.6.1 PDSCH and PUSCH rate-matched block variables

Depending on whether the **rate-matching** is for **PDSCH** or for **PUSCH**, the variable "$I_{LBRM}$" (**layer-based rate-matching**) and other variables from the procedures described in 4.12.6.2, 4.12.6.1 and 4.12.6.3, take different values, as indicated below.

**Rate matching in PDSCH**

$I_{LBRM} = 1$

For more detail on **rate-matching** for **PDSCH**, refer to clause 7.2.5 from TS 38.212

**Rate matching in PUSCH**

In **PUSCH**, there are the following **3 cases**:

- ***If*** `rateMatching` from `PUSCH-ServingCellConfig` (click 7.15.8) is **set** to `limitedBufferRM`, ***then***,

  $I_{LBRM} = 1$

- ***If*** [[[`rateMatching` from `PUSCH-ServingCellConfig` is **not** set to `limitedBufferRM`] and [`numberOfSlotsTBoMS-r17` (from `PUSCH-Allocation-r16`) is **absent** from the **resource allocation table**]] or [[**it** is **present**] and [**its** value is **1** in the **row** indicated by the **DCI message field** "***Time domain resource assignment***"]]], ***then***,

  $I_{LBRM} = 0$

- ***If*** [[`numberOfSlotsTBoMS-r17` from `PUSCH-Allocation-r16` is **present** in the **resource allocation table**] and [its value is **larger than 1** in the **row** indicated by the **DCI message field** "***Time domain resource assignment***" in **DCI**]], ***then***,
  - ***If*** `rateMatching` from `PUSCH-ServingCellConfig` (click 7.15.8) is **set** to `limitedBufferRM`, ***then*** "$I_{LBRM} = 1$", ***else***, "$I_{LBRM} = 0$".
  - "$N_{bit}^{CW}$" is **total quantity** of **coded bits** available for transmission of the **codeword** in the **slot**.
  - ***If*** the **slot** is the **first slot** within the "$N_s$" **slots** allocated for the **transmission** of the **transport block** over **multiple slots**,

    ***then***,

    "$k_0$" is set by imputing the values of "$rv_{id}$" and **LDPC base graph** of the corresponding **transport block** in table 5.4.2.1-2 of TS 38.312.

    ***else***,

    $k_0 = (k'_0 + H + \tau)\ mod\ N_{cb}$

    where "$k'_0$" denotes the **index** of **starting coded bit**, "$H$" is the **total quantity** of **coded bits** available for transmission of the **transport block**, and "$\tau$" denotes the **quantity** of **skipped filler bits** (if any), the **three** of them referred to the **previous slot** within the "$N_s$" **slots** according to clause 5.4.2.1 of TS 38.312 and assuming **no UCI multiplexing**.

For more detail on **rate-matching** for **PUSCH**, refer to clause 6.2.5 of TS 38.212

### 4.12.6.2 PDSCH and PUSCH rate-matched block bitlength determination

This consists of determining the **bitlength** "$N_{bit}^{RBtn(q)}$" of **each** and **every** **rate-matched block** "$rb_{tn}^{(q)}(i_{rm})$" in the **1** or **2 rate-matched block sequence(s)**.

The **bitlength** "$N_{bit}^{RBtn(q)}$" of **each** **rate-matched block** is the **assigned transmission bitlength** "$N_{bit}^{Tx(q)}$" (click 6.14.6) **divided** by the **quantity** of **rate-matched block(s)** "$N_{blk}^{RBS(q)}$" (see below) **rounded up** or **down** to a **multiple** of "$N_{layer} \cdot Q_m^{(q)}$" (click 6.14.6

and 4.12.10). The variable "**q**∈{**0**, **1**}" is the **identifier** of **each** of the **1 or 2 transport block**(**s**) (click 4.12.1.1).

Within **each sequence** "**q**", its **initial rate-matched block**(**s**) are **rounded down**, while its **last ones** are **rounded up**, such that the **added bitlength** of **all** its **rate-matched block**(**s**) is **exactly** the **assigned transmission bitlength** "$N_{bit}^{Tx(q)}$". In this way, **all** the **available bits** in "$N_{bit}^{Tx(q)}$" are **used**, and the **maximum possible code rate** for "$N_{bit}^{Tx(q)}$" is **achieved**.

The value "$N_{layer} \cdot Q_m^{(q)}$" is the **quantity** of **bits** conveyed by **each symbol**, across **all** the **layers**. This allows to **exactly spread** (click 4.12.6.4) the **bits** of **each rate-matched block** among the **transmitted symbols**.

Notice that up to this **stage**, **all** the **code blocks** and **all** the **encoded blocks** in the **same code/encoded block sequence** had the **same bitlength**, denoted as "$N_{bit}^{CB(q)}$" and "$N_{bit}^{EB(q)}$" (respectively). However, the **rate-matched block**(**s**) in the **same rate-matched block sequence** "**q**" **may** have **different bitlengths** "$N_{bit}^{RBtn(q)}$".

In a more formal way, the **bitlength** $N_{bit}^{RBtn(q)}$ of the **scheduled rate-matched block** $N_{bit}^{RBtn(q)}$ with **transmitted block sequence index** "*tn*" is **computed** with the following **pseudo-code**:

```
for idx = 1 to N_blk^EBS
    if idx ≤ N_blk^RBS(q)' - ((N_bit^Tx(q)/(N_layer·Q_m^(q))) mod N_blk^RBS(q)')-1
        then
            N_bit^RBtn(idx)(q) = N_layer·Q_m^(q) floor((N_bit^Tx(q)/N_blk^RBS(q)')/(N_layer·Q_m^(q)))
        else
            N_bit^RBtn(idx)(q) = N_layer·Q_m^(q) ceiling((N_bit^Tx(q)/N_blk^RBS(q)')/(N_layer·Q_m^(q)))
    endif
end for
```

where "*tn*(*idx*)" returns the **ordered** (low to high) **value** in the "*idx*" position within "*tn*". Recall from 4.12.5 that the variable "*tn*" contains the "$N_{blk}^{EBS}$" **block indexes** that should be **retransmitted**.

For more detail on **bitlength determination**, refer to clause 5.4.2.1 of TS 38.212.

### 4.12.6.3 PDSCH and PUSCH rate-matched block bit selection

This consists of:

1. Determining the value of **bit index** "$k_0$" within **each** of the "$N_{blk}^{EBS(q)}$" **transmitted encoded block** "$eb_{tn}^{(q)}(i_{eb})$".
2. Starting with the **bit** that has **index** "$i_{eb}=k_0$", **circularly extract** "$N_{bit}^{RBtn(q)}$" **non-*Null*** bits from "$eb_{tn}^{(q)}(i_{eb})$", which produces the **rate-matched block** "$rb_{tn}^{(q)}(i_{rm})$". This **bit index** "$i_{rb}=k_0$" is the **first bit** in the corresponding **encoded block** "$eb_m^{(q)}(i_{eb})$" starting from which a **bit sequence** of "$N_{bit}^{RBtn(q)}$" **non-*Null* bits** is **circularly extracted**. This **bit sequence** of "$N_{bit}^{RBcn(q)}$" **non-*Null* bits** is the **rate-matched block**.

The variable "**q**∈{**0**, **1**}" is the **identifier** of **each** of the **1 or 2 transport block**(**s**) (click 4.12.1).

The value of "$k_0$" is obtained from table 5.4.2.1-2 of TS 38.212:

| $rv_{id}$ | $k_0$ | |
|---|---|---|
| | Base graph 1 | Base graph 2 |
| **0** | 0 | 0 |
| **1** | $17LS_{minc}^{(q)}$ | $13LS_{minc}^{(q)}$ |
| **2** | $33LS_{minc}^{(q)}$ | $25LS_{minc}^{(q)}$ |
| **3** | $56LS_{minc}^{(q)}$ | $43LS_{minc}^{(q)}$ |

where $rv_{id}$ is the **redundancy version** of **this transmission**.

The **redundancy version** "$rv_{id}$" is **0** in the **first transmission** (**new data**) of a **transport block** and is **usually circularly incremented** in **each successive HARQ retransmission** (**retransmitted data**). However, the **NR node** may use **other sequences** for the **successive values** of "$rv_{id}$", depending on the use case.

Recall from 4.12.6 that the **rate-matching** described in this **book** is the **option** that does **not use layered bit-rate matching**. (i.e., "$I_{LBRM}$=0"), hence the simplified table above.

In a more formal way, the **bit selection** for **each rate-matched block** "$rb_{tn}^{(q)}(i_{rb})$" is done with the following **pseudo-code**:

$i_{eb} = 0$

$i_{rb} = 0$

**while** $i_{rb} < N_{bit}^{RB}$

  **if** $eb_{tn}^{(q)}((k_0 + i_{eb}) \bmod N_{bit}^{EB(q)}) \neq Null$

    $rb_{tn}^{(q)}(i_{rb}) = eb_{tn}^{(q)}((k_0 + i_{eb}) \bmod N_{bit}^{EB(q)})$

    $i_{rb} = i_{rb} + 1$

  **end if**

  $i_{eb} = i_{eb} + 1$

**end while**

Notice that the **pseudo-code** is **independent** of the value of "$tn$" and "**q**".

For more detail on **bit selection**, refer to clause 5.4.2.1 of TS 38.212.

### 4.12.6.4 PDSCH and PUSCH rate-matched block bit interleaving

This consists of **assigning** the **bits** of **each rate-matched block** in a **round-robin** way among the **symbols** of the **modulated sequence** (for time **diversity**, click 11.8), which results in a **bit interleaving**. This **bit interleaving** is done with the following **pseudo-code**:

**for** $j_{ms} = 0$ **to** $N_{bit}^{RB(q)}/Q_m^{(q)} - 1$

  **for** $i_Q = 0$ **to** $Q_m^{(q)} - 1$

    $rb_{tn}^{(q)}{}'(j_{ms} Q_m^{(q)} + i_Q) = rb_{tn}^{(q)}(i_Q N_{bit}^{RB(q)}/Q_m^{(q)} + j_{ms})$

  **end for**

**end for**

The variable "$q \in \{0, 1\}$" is the **identifier** of **each** of the **1 or 2 transport block(s)** (click 4.12.1).

Notice that "$N_{bit}^{RB(q)}/Q_m^{(q)}$" is the **quantity** of **modulation symbols** over which the **rate-matched block** will be **transmitted**, and that "$N_{bit}^{RB(q)}/Q_m^{(q)}$" is an **integer** (see the pseudo-code for "$N_{bit}^{RB(q)}$" slightly above).

To keep the notation "$rb_{tn}^{(q)}$" for the **bit sequence** of the **rate-matched block(s)**:

$$rb_{tn}^{(q)}(i_{rb}) = rb_{tn}^{(q)\prime}(i_{rb})$$

**Each** of the resulting **sequences** "$rb_{tn}^{(q)}(i_{rb})$" is a **rate-matched block**.

For a **mapping** between the **terms** (e.g., **protected block**) and the **symbols** (e.g., "$N_{blk}^{CBS(q)}$") of **PDSCH** and **PUSCH processing** used in **this book** and the corresponding **terms** and **symbols** (e.g., ***A***, ***C***, …) used in the **TS documents** click 4.12.174.12.10.

For more detail on **bit interleaving**, refer to clause 5.4.2.2 of TS 38.212.

## 4.12.7 PDSCH and PUSCH rate-matched block concatenation

This consists of **concatenating sequentially every rate-matched block** in **each** of the **1** or **2 sequence(s)** of **rate-matched block(s)** into **1** or **2 codeword(s)**.

The variable "$cw^{(q)}(i_{cw})$" represents the **bit sequence** of the **codeword** "**q**", that was produced from the **transport block** "**q**". The variable "$i_{cw}$" is the **bit index** within **each codeword**.

The identifier "$N_{bit}^{CW(q)}$" is the **bitlength** of the **codeword** "**q**", where "$N_{bit}^{CW(q)}=\Sigma_{tn}N_{bit}^{RBtn(q)}$". The **bit index** "$i_{cw}$" of **each codeword** ranges from **0** to "$N_{bit}^{CW(q)}-1$".

This **concatenation** may be formally expressed as:

$$cw^{(q)}(0), \ldots, cw^{(q)}(N_{bit}^{CW(q)}-1) = r_0^{(q)}(0), r_0^{(q)}(1), \ldots, r_0^{(q)}(N_{bit}^{RB0(q)}-1),$$
$$r_1^{(q)}(0), r_1^{(q)}(1), r_1^{(q)}(2), \ldots, r_1^{(q)}(N_{bit}^{RB1(q)}-1),$$
$$\ldots,$$
$$r_{Nblk(q)-1}(0), r_{Nblk(q)-1}(1),\ldots, r_{Nblk(q)-1}(N_{bit}^{RB(Nblk(q)-1)(q)}-1)$$

where the **super index** or **subindex** "$_{Nblk(q)}$" stands for "$N_{blk}^{EBS(q)}$".

The **codeword bitlength** "$N_{bit}^{CW(q)}$" is the **addition** of the **bitlengths** "$N_{bit}^{RBcn(q)}$" of **all** the **rate-matched blocks** that belong to **sequence** "**q**".

Since "$N_{bit}^{RBcn(q)}/Q_m^{(q)}$" is always an **integer** (click 4.12.6.4), it follows that "$N_{bit}^{CW(q)}/Q_m^{(q)}$" is also an **integer**.

For a **mapping** between the **terms** (e.g., **protected block**) and the **symbols** (e.g., "$N_{blk}^{CBS(q)}$") of **PDSCH** and **PUSCH processing** used in **this book** and the corresponding **terms** and **symbols** (e.g., ***A***, ***C***, …) used in the **TS documents** click 4.12.174.12.10.

The procedure for **rate-matched block concatenation** is **identical** for **PUSCH** and for **PDSCH**, with the subtle difference that for **PUSCH** it is always **only 1 transport block** and **only 1 codeword**.

For more detail on **rate-matched block concatenation**, refer to clauses 5.5 (general), 6.2.6 (for **PUSCH**) and 7.2.6 (for **PDSCH**) of TS 38.212.

### 4.12.8 PUSCH codeword and PUCCH codeword (UCI) multiplexing

This stage can **only** be applied to **PUSCH** and is **not** applied in **PDSCH**.

It consists of **multiplexing 1 PUSCH codeword** and/or **1 PUCCH codeword** into **1 multiplexed codeword** for its **delivery** in **1 PUSCH transmission**.

The variable "$mcw^{(0)}(i_{mcw})$" represents the **bit sequence** of the **multiplexed codeword** "**0**", that was produced from the **PUSCH transport block** "**0**". The variable "$i_{mcw}$" is the **bit index** within the **multiplexed codeword** "**0**".

The identifier "$N_{bit}^{MCW(0)}$" is the **bitlength** of the **multiplexed codeword** "**0**", where "$N_{bit}^{MCW(0)} > N_{bit}^{CW(0)}$". The **bit index** "$i_{mcw}$" of the **multiplexed codeword** ranges from **0** to "$N_{bit}^{MCW(q)}-1$".

It is possible to **multiplex** [a **PUSCH codeword** (click 4.12.1.2)] with [a **PUCCH codeword** (click 7.8.2.5) that does **not** contain **scheduling request** (**SR**)] when they **coincide** in **time**, and **either**:

- The [**UL-SCH** (click 4.11.2) **transport block** (click 4.12.1.1) that resulted in the **PUSCH codeword**] and the [**UCI message** (click 7.7) that resulted in the **PUCCH codeword**] have the **same priority** (**high/low**).
- The **UCI message** that resulted in the **PUCCH codeword** is a **HARQ-ACK**.

For the **specific case** of a **PUCCH codeword** arising from an **aperiodic channel state information** (**A-CSI**) **report**, it can be sent through **PUSCH** even if there is **no available PUSCH codeword**.

***If*** the **PUCCH codeword** carries **HARQ-ACK** with **1** or **2 bits**, ***then***, it is **multiplexed** with the **PUSCH codeword** by **puncturing PUSCH**. ***Else***, it is **multiplexed** by **rate-matching PUSCH**.

This **PUCCH/PUSCH codeword multiplexing** procedure is quite complicated and has several cases. It contains up to **7 stages** and is described in **eleven** and a half **pages** of **pseudo code** in clause 6.2.7 of TS 38.212. However, the conceptual content is low. For this reason, this procedure is **not described** in this book, and you may refer to the said clause.

The **multiplexed codeword** is passed on to the **next stage** (the **codeword scrambling stage**, click 4.12.9).

For the **processing** of **complete UCI messages** through **PUCCH** , click 7.8.

For more detail on **PUCCH/PUSCH codeword multiplexing**, refer to clauses 6.2.7, 6.3.2.6 and 6.3.2.7 of TS 38.212.

### 4.12.9 PDSCH and PUSCH codeword scrambling

This consists of **scrambling each codeword** or **multiplexed codeword** into **1 scrambled codeword**.

For **PDSCH** it is **1** or **2 codeword**(**s**). For **PUSCH** it is **either 1 codeword** or **1 multiplexed codeword**.

The variable "$scw^{(q)}(i_{scw})$" represents the **bit sequence** of the **scrambled codeword** "**q**", that was produced from the **transport block** "**q**". The variable "$i_{scw}$" is the **bit index** within **each scrambled codeword**.

The identifier "$N_{bit}^{SCW(q)}$" is the **bitlength** of the **scrambled codeword** "**q**". The **bit index** "$i_{scw}$" of the **scrambled codeword** ranges from **0** to "$N_{bit}^{MCW(q)}$-1".

The **bitlength** "$N_{bit}^{SCW(q)}$" is the **same** as the **bitlength** of its corresponding **codeword** "$N_{bit}^{CW(q)}$", or **multiplexed codeword** "$N_{bit}^{MCW(0)}$".

**PDSCH codeword scrambling** (click 4.12.9.1) and **PUSCH codeword scrambling** (click 4.12.9.2) are different.

### 4.12.9.1 Scramble each PDSCH codeword

The **scrambling expression** is:

$$scw^{(q)}(i) = cw^{(q)}(i) \oplus c^{(q)}(i)$$

where "⊕" is an **exclusive-or**.

The **scrambling sequence** "$c(i)$" (click 11.20.1) is a **pseudo-random binary sequency**.

The **scrambling sequence** is **initialized** (click 11.20.1) with a **different seed** "$c_{init}^{(q)}$" for **each codeword**, resulting in the **same scrambling sequence**, but with a **different circular shift**. The **scrambling sequence seed** "$c_{init}^{(q)}$" for **codeword** "**q**" is:

$$c_{init}^{(q)} = n_{RNTI} \cdot 2^{15} + q \cdot 2^{14} + n_{ID}$$

where:

- "$n_{RNTI}$" is the **RNTI** used to mask the **CRC** of the **DCI message** that **assigned this PDSCH** transmission.
- "$n_{ID}$" takes a value from **0** to **1,023**, and there are the following **3 cases**:
  - ***If*** [[[the **PUSCH transmission** is **not** scheduled with **DCI Format 1_0** (click 6.9.5) in a **common search space** (**CSS**, click 6.8.5)] ***and*** [the **RNTI** is **either** of **C-RNTI**, **MCS-C-RNTI** or **CS-RNTI**]]

    ***or*** [[the **PUSCH transmission** is **not** scheduled with **DCI Format 1_0** (click 6.9.5) in a **common MBS frequency resource**] ***and*** [the **RNTI** is **either** of **G-RNTI**, **G-CS-RNTI**, or **MCCH-RNTI**]]]

    ***and*** [`dataScramblingIdentityPDSCH` from `PDSCH-Config` (click 6.18.6) is **present**], ***then***:

    "$n_{ID}$" is `dataScramblingIdentityPDSCH`
  - ***If*** [the **PUSCH transmission** is **not** scheduled with **DCI Format 1_0** (click 6.9.5) in a **common search space** (**CSS**, click 6.8.5)] and [the **PUSCH transmission** is scheduled using a **CORESET** with `coresetPoolIndex-r16` from `ControlResourceSet` (click 6.12.6) equal to `1`] ***and*** [the **RNTI** is **either** of **C-RNTI**, **MCS-C-RNTI** or **CS-RNTI**] ***and*** [`dataScramblingIdentityPDSCH`, `dataScramblingIdentityPDSCH2-r16` (both from `PDSCH-Config`) and `coresetPoolIndex-r16` from `ControlResourceSet` are configured together within the **same instance** of `PDSCH-Config`)] ***and*** [`dataScramblingIdentityPDSCH`], ***then***:

    "$n_{ID}$" is `dataScramblingIdentityPDSCH2-r16`

- ***Else***:

  "$n_{ID}$" is the value "$N_{ID}^{cell}$". This is the **Physical Cell Identity** (**PCI**, click 3.5.3.2) of the cell.

- The multiplication by "$2^z$" is equivalent to a **left shift** by "**z**" bits.

For more detail on **PDSCH scrambling**, refer to clauses 5.2.1 (scrambling sequence) and 7.3.1.1 of TS 38.211.

### 4.12.9.2 Scramble each PUSCH codeword

The **pseudocode** for the **scrambling** of **each PUSCH codeword** is:

```
i_cw = 0
while i_cw < N_bit^CW(0)
   if cw^(0)(i_cw) = x                    -- UCI placeholder bit
      then
         scw^(0)(i_cw) = 1
      else
         if cw^(0)(i_cw) = y              -- UCI placeholder bit
            then
               scw^(0)(i_cw) = scw^(0)(i_cw -1)
         else
               scw^(0)(i_cw) = cw^(0)(i_cw) ⊕ c^(0)(i_cw)
      end if
   end if
   i_cw = i_cw + 1:
end while
```

The **pseudocode** for the **scrambling** of **each PUSCH multiplexed codeword** is exactly the same, replacing "$cw(i_{cw})$" by "$mcw(i_{mcw})$" and "$N_{bit}^{CW}$" by "$N_{bit}^{MCW}$".

The **tags** "***x***" and "***y***" are defined as part of the **channel coding** for **small block lengths** (click 7.8.2.3.2 and 7.8.2.3.3, and refer to clauses 5.3.3.1 and 5.3.3.2 of TS 38.212).

The **scrambling sequence** "$c(i_{cw})$" (click 11.20.1) is a **pseudo-random binary sequency**.

The **scrambling sequence** is **initialized** (click 11.20.1) with the **seed** "$c_{init}$", that depends on the **contents** of the **UL-SCH transport block** as follows:

- ***If*** it is **MsgB** (click 9.3.7):

$$c_{init} = n_{RNTI} \cdot 2^{16} + n_{RAPID} \cdot 2^{10} + n_{ID}$$

- ***else***:

$$c_{init} = n_{RNTI} \cdot 2^{15} + n_{ID}$$

where:

- "$n_{RNTI}$" is the **RNTI** used to mask the **CRC** of the **DCI message** that **assigned this PUSCH** transmission. Consequently, for the transmission of **MsgB**, "$n_{RNTI}$" is the **RA-RNTI** (click 6.9.10.2).
- "$n_{RAPID}$" is the is the **index** of the **random-access preamble** (click 7.3.1) transmitted as **msgA** (click 9.3.6 and refer to clause 5.1.3A of TS 38.321).
- "$n_{ID}$" takes a value from **0** to **1,023**, and there are the following **3 cases**:
  - ***If*** [the **PUSCH transmission** is **not** scheduled with **DCI Format 0_0** (click 6.9.2) in a **common search space** (**CSS**, click 6.8.5)] and [the **RNTI** is **either** of **C-RNTI**, **MCS-C-RNTI**, **SP-CSI-RNTI** or **CS-RNTI**] and [`dataScramblingIdentityPUSCH` from `PUSCH-Config` (click 7.15.5) is **present**], ***then***:

    "$n_{ID}$" is `dataScramblingIdentityPUSCH`
  - ***If*** [the **PUSCH transmission** is triggered by a **Type-2 random access procedure** (click 9.1 and refer to clause 8.1A of TS 38.213)] and [`msgA-DataScramblingIndex-r16` from `MsgA-PUSCH-Config-r16` (click 9.5.7) is **present**], ***then***:

    "$n_{ID}$" is `msgA-DataScramblingIndex-r16`
  - ***Else***:

    "$n_{ID}$" is the value "$N_{ID}^{cell}$". This is the **Physical Cell Identity** (**PCI**, click 3.5.3.2) of the cell.
- The multiplication by "$2^z$" is equivalent to a **left shift** by "**z**" bits.

For a **mapping** between the **terms** (e.g., **protected block**) and the **symbols** (e.g., "$N_{blk}^{CBS(q)}$") of **PDSCH** and **PUSCH processing** used in **this book** and the corresponding **terms** and **symbols** (e.g., ***A***, ***C***, …) used in the **TS documents** click 4.12.174.12.10.

For more detail on **PUSCH scrambling**, refer to clauses 5.2.1 (scrambling sequence) and 6.3.1.1 of TS 38.211.

## 4.12.10 PDSCH and PUSCH modulation of each scrambled codeword

This consists of **modulating each** of the **1** or **2 scrambled codeword**(**s**) into **1** or **2 modulated sequence**(**s**) of **complex-valued symbols**.

The variable "$ms^{(q)}(j_{ms})$" represents the **symbol sequence** of the **modulated sequence** "**q**", that was produced from the **transport block** "**q**". The variable "$j_{ms}$" is the **symbol index** within **each modulated sequence**.

The identifier "$N_{symb}^{MS(q)}$" is the **symbol length** of the **modulated sequence** "**q**". The **symbol index** "$j_{ms}$" of the **modulated sequence** ranges from **0** to "$N_{symb}^{MS(q)}$-1".

The **symbol length** "$N_{symb}^{MS(q)}$" of a **modulated scrambled codeword** is its **bitlength** "$N_{bit}^{CW(q)}$" divided by its **modulation order** "$Q_m^{(q)}$" (click 11.2.1):

$$N_{symb}^{MS(q)} = N_{bit}^{CW(q)} / Q_m^{(q)}$$

Recall from 4.12.7 that "$N_{bit}^{CW} / Q_m^{(q)}$" is an **integer**.

If there are **2 scrambled codewords**, **each** may be **modulated** with its **own specific modulation**, in spite of the fact that the **2 scrambled codewords** are delivered in the **REs** of **one PDSCH transmission**.

The following table presents the **modulation techniques** that can be used:

| Modulation | Modulation order ($Q_m$) | PDSCH | PUSCH | |
|---|---|---|---|---|
| | | | Trans. precod. disabled | Trans. precod. enabled |
| **π/-BPSK** | 1 | No | | Yes |
| **QPSK** | 2 | Yes | | |
| **16QAM** | 4 | Yes | | |
| **64QAM** | 6 | Yes | | |
| **256QAM** | 8 | Yes | | |
| **1024QAM** | 10 | Only in **FR1** | No | |

In **LTE**, its usual **modulation** and **coding scheme** results in around a **10%** of **HARQ** (click 4.5.6) **retransmitted packets**, because it is chosen to **optimize throughput** at the **cost** of **increasing delay**. This is why **5G** supports **several modulation** and **coding configurations** that can **reduce** the **HARQ retransmission percentage**, thus **reducing delay**.

For a **mapping** between the **terms** (e.g., **protected block**) and the **symbols** (e.g., "$N_{blk}^{CBS(q)}$") of **PDSCH** and **PUSCH processing** used in **this book** and the corresponding **terms** and **symbols** (e.g., $A$, $C$, …) used in the **TS documents** click 4.12.174.12.10.

For more detail on **codeword modulation**, refer to clauses 5.1 (general), 6.3.1.2 (for **PUSCH**) and 7.3.1.2 (for **PDSCH**) of TS 38.211.

For an overview of **modulation techniques**, click 11.2.1.

## 4.12.11 PDSCH and PUSCH symbol to layer mapping

This consists of **mapping** the **1** or **2 modulated sequence**(**s**) into "$N_{layer}$" **layer sequence**(**s**).

The variable "$lys^{(lay)}(j_{lys})$" represents the **symbol sequence** of the **layer sequence** corresponding to the variable "$lay$". The variable "$j_{lys}$" is the **symbol index** within **each layer sequence**.

The identifier "$N_{layer}$" is the **quantity** of **layers**. The variable "$lay$" ranges from **0** to "$N_{layer}$-1". The identifier "$N_{symb}^{layer}$" is the **symbol length** of **every layer sequence**. The **symbol index** "$j_{lys}$" of the **layer sequence** ranges from **0** to "$N_{symb}^{layer}$-1".

The **quantity** of **layers** is **restricted** by the **quantity** of **modulated sequence**(**s**). For **1 modulated sequence** "**q**=**1**"there may be **1** to **4 layers**, and for **2 modulated sequences** "**q**∈{**0**, **1**}" there may be **5** to **8 layers**.

Notice that the **total quantity** of **symbols** remains the **same**:

$$N_{layer} \bullet N_{symb}^{layer} = N_{symb}^{MS(0)} + N_{symb}^{MS(1)}$$

**Each** of the **1** or **2 modulated sequence**(**s**) "$ms^{(q)}(j_{ms})$" is **mapped** to **half** the available **layers**, and **its symbols** are allocated **round-robin** among **its** assigned **layers**. If there are **2 modulated sequences** and the **quantity** of **layers** "$N_{layer}$" is **odd**, then, the **second**

**modulated sequence** (i.e., **q=1**) gets **one more layer** than the **other modulated sequence**. This **mapping** is formally expressed in table 7.3.1.3-1 of TS 38.211: $j_{lys}$

| Quantity of layers | Quantity of modulated sequences | Symbol to layer mapping $j_{lys}$ = 0, 1,..., $N_{symb}^{layer}$-1 | $N_{symb}^{MS(q)}$, $N_{symb}^{layer}$ |
|---|---|---|---|
| 1 | 1 | $lys^{(0)}(j_{lys}) = ms^{(0)}(j_{lys})$ | $N_{symb}^{MS(0)} = N_{symb}^{layer}$ |
| 2 | 1 | $lys^{(0)}(j_{lys}) = ms^{(0)}(2j_{lys})$<br>$lys^{(1)}(j_{lys}) = ms^{(0)}(2j_{lys}+2)$ | $N_{symb}^{MS(0)} = 2N_{symb}^{layer}$ |
| 3 | 1 | $lys^{(0)}(j_{lys}) = ms^{(0)}(3j_{lys})$<br>$lys^{(1)}(j_{lys}) = ms^{(0)}(3j_{lys}+2)$<br>$lys^{(2)}(j_{lys}) = ms^{(0)}(3j_{lys}+2)$ | $N_{symb}^{MS(0)} = 3N_{symb}^{layer}$ |
| 4 | 1 | $lys^{(0)}(j_{lys}) = ms^{(0)}(3j_{lys})$<br>$lys^{(1)}(j_{lys}) = ms^{(0)}(4j_{lys}+2)$<br>$lys^{(2)}(j_{lys}) = ms^{(0)}(4j_{lys}+2)$<br>$lys^{(3)}(j_{lys}) = ms^{(0)}(4j_{lys}+3)$ | $N_{symb}^{MS(0)} = 4N_{symb}^{layer}$ |
| 5 | 2 | $lys^{(0)}(j_{lys}) = ms^{(0)}(2j_{lys})$<br>$lys^{(1)}(j_{lys}) = ms^{(0)}(2j_{lys}+1)$<br>$lys^{(2)}(j_{lys}) = ms^{(1)}(3j_{lys}+2)$<br>$lys^{(3)}(j_{lys}) = ms^{(1)}(3j_{lys}+1)$<br>$lys^{(4)}(j_{lys}) = ms^{(1)}(3j_{lys}+2)$ | $N_{symb}^{MS(0)} = 2N_{symb}^{layer}$<br>$N_{symb}^{MS(1)} = 3N_{symb}^{layer}$ |
| 6 | 2 | $lys^{(0)}(j_{lys}) = ms^{(0)}(3j_{lys})$<br>$lys^{(1)}(j_{lys}) = ms^{(0)}(3j_{lys}+1)$<br>$lys^{(2)}(j_{lys}) = ms^{(0)}(3j_{lys}+2)$<br>$lys^{(3)}(j_{lys}) = ms^{(1)}(3j_{lys})$<br>$lys^{(4)}(j_{lys}) = ms^{(1)}(3j_{lys}+1)$<br>$lys_{(5)}(j_{lys}) = ms^{(1)}(3j_{lys}+2)$ | $N_{symb}^{MS(0)} = 3N_{symb}^{layer}$<br>$N_{symb}^{MS(1)} = 3N_{symb}^{layer}$ |
| 7 | 2 | $lys^{(0)}(j_{lys}) = ms^{(0)}(3j_{lys})$<br>$lys^{(1)}(j_{lys}) = ms^{(0)}(3j_{lys}+1)$<br>$lys^{(2)}(j_{lys}) = ms^{(0)}(3j_{lys}+2)$<br>$lys^{(3)}(j_{lys}) = ms^{(1)}(4j_{lys})$<br>$lys^{(4)}(j_{lys}) = ms^{(1)}(4j_{lys}+1)$<br>$lys^{(5)}(j_{lys}) = ms^{(1)}(4j_{lys}+2)$<br>$lys^{(6)}(j_{lys}) = ms^{(1)}(4j_{lys}+3)$ | $N_{symb}^{MS(0)} = 3N_{symb}^{layer}$<br>$N_{symb}^{MS(1)} = 4N_{symb}^{layer}$ |
| 8 | 2 | $lys^{(0)}(j_{lys}) = ms^{(0)}(4j_{lys})$<br>$lys^{(1)}(j_{lys}) = ms^{(0)}(4j_{lys}+1)$<br>$lys^{(2)}(j_{lys}) = ms^{(0)}(4j_{lys}+2)$<br>$lys^{(3)}(j_{lys}) = ms^{(0)}(4j_{lys}+3)$<br>$lys^{(4)}(j_{lys}) = ms^{(1)}(4j_{lys})$<br>$lys^{(5)}(j_{lys}) = ms^{(1)}(4j_{lys}+1)$<br>$lys^{(6)}(j_{lys}) = ms^{(1)}(4j_{lys}+2)$<br>$lys^{(7)}(j_{lys}) = ms^{(1)}(4j_{lys}+3)$ | $N_{symb}^{MS(0)} = 4N_{symb}^{layer}$<br>$N_{symb}^{MS(1)} = 4N_{symb}^{layer}$ |

For a **mapping** between the **terms** (e.g., **protected block**) and the **symbols** (e.g., "$N_{blk}^{CBS(q)}$") of **PDSCH** and **PUSCH processing** used in **this book** and the corresponding **terms** and **symbols** (e.g., ***A***, ***C***, …) used in the **TS documents** click 4.12.174.12.10.

For more detail on **maping** to **layers**, refer to clauses 6.3.1.3 (for **PUSCH**) and 7.3.1.3 (for **PDSCH**) of TS 38.211.

Source web pages:

- https://www.nrexplained.com/pdschmod

### 4.12.12 PUSCH transform precoding (optional)

This **PUSCH step** is **only** performed if the **optional PUSCH transform precoding** is **enabled**. **Transform precoding** consists of applying a **Discrete Fourier Transform** that produces a **single-carrier like signal**. This is equivalent to using the **single-carrier frequency-division multiple access** (**SC-FDMA**, click 11.16)

The result is **1 antenna port sequence** "$aps^{(ap0)}(j_{aps})$" of **complex valued symbols**. the **symbol index** "$j_{aps}$" of **every sequence** ranges from **0** to "$N_{symb}^{layer}$".

**PUSCH transform precoding** and **PUSCH precoding** (click 4.12.13) **cannot** be **both enabled**, but **both** can be **disabled**.

For more detail on **PUSCH transform precoding**, refer to clause 6.3.1.4 of TS 38.211.

### 4.12.13 PDSCH and PUSCH precoding (optional)

This converts the "$N_{layer}$" **layer sequence**(**s**) into "$N_{port}$" **precoded antenna port sequence**(**s**) (**optional**). **Each antenna port sequence** is a **stream**.

The variable "$aps^{(ap)}(j_{aps})$" represents the **symbol sequence** of the $aps^{(ap)}(j_{aps})$ corresponding to the variable "$ap$". The variable "$j_{aps}$" is the **symbol index** within **each precoded antenna port sequence**.

The identifier "$N_{port}$" is the **quantity** of **antenna ports**. The variable "$ap$" ranges from "$ap_0$" to "$ap_{Nport-1}$", **each** of which is an **antenna port number**. The **symbol index** "$j_{aps}$" of the **precoded antenna port sequence** ranges from **0** to "$N_{symb}^{layer}$-1".

Notice that the **quantity** of **symbols** "$N_{symb}^{layer}$" in **each** and **every symbol sequence** remains the **same** after **precoding**.

For an **overview** of **precoding**, click 2.4.2.

For a **mapping** between the **terms** (e.g., **protected block**) and the **symbols** (e.g., "$N_{blk}^{CBS(q)}$") of **PDSCH** and **PUSCH processing** used in **this book** and the corresponding **terms** and **symbols** (e.g., ***A***, ***C***, …) used in the **TS documents** click 4.12.17.

**PDSCH precoding**

The **specification** of **PDSCH processing** from clause 7.3.1 of TS 38.211, does **not** include **precoding**. The corresponding clause is 7.3.1.4, and it specifies **port mapping instead** of **precoding**.

The reason is that TS 38.211 specifies the functionality of the **UE** and **not** of the **NR node**. Consequently, clause 7.3.1 does **not** specify the **PDSCH Tx processing**. It rather specifies the **PDSCH Rx processing**. This is why clause 7.3.1.4 specifies **port mapping instead** of **precoding**.

In case of **space-division MIMO**, the **NR node certainly** has to perform a **precoding stage**. This is **not specified**, but the reader has a **reference** about the corresponding **procedure** with the **PUSCH precoding** described **above**.

**PUSCH precoding**

For **each value** of "$j_{aps}$=$j_{lys}$" within their **range** (from **0** to "$N_{symb}^{layer}$", the **vertical matrix** of **precoded antenna port symbols** "$[aps^{(ap)}(j_{aps})]^{T}$" is obtained by **multiplying** the **precoding matrix** "$W$" by the **vertical matrix** of **layer symbols** $[lys^{(lay)}(j_{lys})]^{T}$:

$$[aps^{(ap0)}(j_{aps}), \ldots, aps^{(ap(Nport-1))}(j_{aps})]^{T} = W \times [lys^{(0)}(j_{lys}), \ldots, lys^{(Nlayer-1)}(j_{lys})]^{T}$$

where the **list** of **antenna port numbers** "$ap_0$" to "$ap_{Nport-1}$" shall be determined according to the procedure in TS 38.214.

The **dimension** of the **precoding mattrix** "$W$" is "$N_{port} \times N_{layer}$".

For **non-codebook PUSCH transmission** or "$N_{port}$=**1**", the **precoding matrix** "$W$" is the **identity matrix**.

For **codebook-based PUSCH transmission** and "$N_{port}$>**1**", the **precoding matrix** "$W$" is given by tables 6.1.3.5-**1** to -**7** of TS 38.211.

For more detail on **PUSCH precoding**, refer to clause 6.3.1.5 of TS 38.211.

**PUSCH precoding** and **PUSCH transform precoding** (click 4.12.12) **cannot** be **both enabled**, but **both** can be **disabled**.

## 4.12.14 PDSCH and PUSCH symbol sequence to VREs mapping

This consists of **mapping each antenna port sequence** "$aps^{(apn)}(j_{aps})$" of **complex valued symbols** of **each** of the "$N_{layer}$" **Tx antenna ports** "$ap$" to **1 set** of **payload virtual resource elements** (**VREs**) in **port** "$(vsc, si, ap)_{\mu}$ of the **same antenna port**. **All** the **payload VREs** in **port** are **scheduled** for **1 PDSCH transmission** or for **1 PUSCH transmission**.

The **quantity** of **VREs** in **every set** of **VREs** in **port** "$(vsc, si, ap)$" is "$N_{symb}^{layer}$".

The variable "$vsc$" is the **virtual subcarrier index**, "$sc$" is the **subcarrier index**, "$si$" is the **symbol interval index** and "$ap$" is the **antenna port number**.

The **first symbol** (index "$j_{aps}$=0") in **each antenna port sequence** is **mapped** to the **payload VRE** with the **subcarrier** of **lowest frequency** and the **earliest symbol interval**. **Subsequent symbols** are **mapped** to **VREs** by **successively increasing first** the **subcarrier index** and after the **symbol interval index**.

The **mapping** is expressed in a more formal way as follows:

1. The **first symbol** "$aps^{(apn)}(0)$" in the **antenna port sequence** "$aps^{(apn)}(j_{aps})$" is **mapped** to the **scheduled payload VRE** in **port** "$(vsc, si, ap)_{\mu}$", where "$vsc$" is the **lowest frequency subcarrier** among the **scheduled payload VREs** of **symbol interval** "$si$", "$si$" is the **first symbol interval** in the **scheduled payload VREs**, and the **antenna port number** "$ap$" is the one with **index** "$apn$" in the **antenna port list** "$apl()$" (expressing this formally: "$ap$=$apl(apn)$".
2. As the value of the **symbol index** "$j_{aps}$" of the **antenna port sequence index increases**, the value of the **virtual subcarrier index** "$vsc$" is also **increased**.
3. When "$vsc$" **cannot** be further **increased**, the value of the **symbol interval index** "$si$" is **increased** and "$vsc$" returns to the **lowest frequency subcarrier** among the **scheduled payload VREs** of **symbol interval** "$si$". The procedure goes to **step 2**.
4. When the "$si$" **cannot** be further **increased**, the **procedure finishes**.

Recall that in the case of **DFA-Type 0** or **UFA-Type 0 frequency allocation**, the **VREs** may be located in **non-contiguous virtual resource groups** (**VRGs**, click 6.14.3.2), but this does **not affect** the mapping **procedure just described**.

For **PDSCH**, the **payload VREs** are **all** the **scheduled VREs** whose **PRBs** are declared as available for **PDSCH** according to clause 5.1.4 of TS 38.214 and that:

- Are **not** used for **transmission** of the associated **DM-RS** or **DM-RS** intended for other **co-scheduled UEs** as described in clause 7.4.1.1.2 of TS 38.211.
- Are **not** used for **non-zero-power CSI-RS**, which is according to clause 7.4.1.5 of TS 38.211, and **not** configured by `TRSResourceSet`, if the corresponding **PRBs** are for a **PDSCH** scheduled by a **PDCCH** with the **CRC scrambled** by **C-RNTI**, **MCS-C-RNTI**, **CS-RNTI**, **G-RNTI** for **multicast**, **G-CS-RNTI**, or a **PDSCH** with **SPS**, **except** if the **non-zero-power CSI-RS** is a **CSI-RS** configured by `CSI-RS-Resource-Mobility` within `MeasObjectNR` or **except** if the **non-zero-power CSI-RS** is an **aperiodic nonzero-power CSI-RS resource**.
- Are **not** used for **PT-RS** according to clause 7.4.1.2.
- Are **not** declared as "**not available for PDSCH**" according to clause 5.1.4 of TS 38.214.

For **PUSCH**, the **payload VREs** are **all** the **scheduled VREs** that are **not** used for **transmission** of the associated **DM-RS**, **PT-RS**, or **DM-RS** intended for other **co-scheduled UEs** as described in clause 6.4.1.1.3 of TS 38.211.

**Mapping example**

The following example shows the **mapping** of the **antenna port sequence** "$aps^{(3)}(j_{aps})$" to its **payload VREs** in **port** "$(vsc, si, 1005)_{\mu}$" (click 4.11.4.8.3), where the **antenna port** with **antenna port index** "**apn**=3" in the **antenna port list** "*apl*(3)" is "1005". The **antenna port sequence** in this example has **24 symbols**, "$aps^{(3)}(0)$" to "$aps^{(3)}(23)$", which

is mapped onto the **assigned PDSCH payload virtual subcarriers** "*vsc*" **179** to **190** (**1 VRB**) and the **assigned PDSCH payload symbol intervals** "*si*" **3** and **4**:

| *apn=3 [port index]* | | *si [symbol interval index]* | |
|---|---|---|---|
| | | **3** | **4** |
| *vsc [virtual subcarrier index]* | **191** | $aps^{(3)}(11)$ (191, 3, 1005) | $aps^{(3)}(23)$ (191, 4, 1005) |
| | **190** | $aps^{(3)}(10)$ (190, 3, 1005) | $aps^{(3)}(22)$ (190, 4, 1005) |
| | **189** | $aps^{(3)}(9)$ (189, 3, 1005) | $aps^{(3)}(21)$ (189, 4, 1005) |
| | **188** | $aps^{(3)}(8)$ (188, 3, 1005) | $aps^{(3)}(20)$ (188, 4, 1005) |
| | **187** | $aps^{(3)}(7)$ (187, 3, 1005) | $aps^{(3)}(19)$ (187, 4, 1005) |
| | **186** | $aps^{(3)}(6)$ (186, 3, 1005) | $aps^{(3)}(18)$ (186, 4, 1005) |
| | **185** | $aps^{(3)}(5)$ (185, 3, 1005) | $aps^{(3)}(17)$ (185, 4, 1005) |
| | **184** | $aps^{(3)}(4)$ (184, 3, 1005) | $aps^{(3)}(16)$ (184, 4, 1005) |
| | **183** | $aps^{(3)}(3)$ (183, 3, 1005) | $aps^{(3)}(15)$ (183, 4, 1005) |
| | **182** | $aps^{(3)}(2)$ (182, 3, 1005) | $aps^{(3)}(14)$ (182, 4, 1005) |
| | **181** | $aps^{(3)}(1)$ (181, 3, 1005) | $aps^{(3)}(13)$ (181, 4, 1005) |
| | **180** | $aps^{(3)}(0)$ (180, 3, 1005) | $aps^{(3)}(12)$ (180, 4, 1005) |

Notice how the **first** symbol "$aps^{(3)}(0)$" is mapped to the **assigned PDSCH payload VRE** in **port** "(179, 3, 1,000)", the **second** symbol "$aps^{(3)}(1)$" is mapped to the **assigned PDSCH payload VRE** in **port** "(180, 3, 1005)". This continues, until the **symbol** "$aps^{(3)}(12)$" is mapped to the **assigned PDSCH payload VRE** in **port** "(179, 4, 1005)", and so on, to finalize with **symbol** "$aps^{(3)}(23)$" mapped to the **assigned PDSCH payload VRE** in **port** "(190, 4, 1005)". The **antenna port number** is always "1005".

Notice that the mapping can **only** be done to **PDSCH payload VREs**, this is, to **VREs** that are **mapped** to a **PDSCH payload PRE** (click 6.14.5). If a **VRE** in **port** "$(vsc, si, ap)_{\mu}$" is **not mapped** to a **PDSCH payload PRE**, it is **excluded** from the mapping of **symbols** to **VREs**, and it therefore **does not appear** in the mapping sequence.

Continuing with the example above, imagine that **VREs** in **port** "(179, 3, 1005)", "(185, 3, 1005)", "(181, 4, 1005)" and "(184, 4, 1005)" are mapped to **4 PDSCH PREs** in **port** that are **not payload PREs** in **port**. In this case, the **antenna port sequence**

"$aps^{(3)}(j_{aps})$" of **antenna port** "1005" would have **20 symbols**, "$aps^{(3)}(0)$" to "$aps^{(3)}(19)$", and their **mapping** onto the **20 assigned PDSCH payload VREs** in **port** would be:

| *apn=3 [port index]* | | *si [symbol interval index]* | |
|---|---|---|---|
| | | 3 | 4 |
| *vsc [virtual subcarrier index]* | 191 | $aps^{(3)}(9)$ (191, 3, 1005) | $aps^{(3)}(19)$ (191, 4, 1005) |
| | 190 | $aps^{(3)}(8)$ (190, 3, 1005) | $aps^{(3)}(18)$ (190, 4, 1005) |
| | 189 | $aps^{(3)}(7)$ (189, 3, 1005) | $aps^{(3)}(17)$ (189, 4, 1005) |
| | 188 | $aps^{(3)}(6)$ (188, 3, 1005) | $aps^{(3)}(16)$ (188, 4, 1005) |
| | 187 | $aps^{(3)}(5)$ (187, 3, 1005) | $aps^{(3)}(15)$ (187, 4, 1005) |
| | 186 | ***not PD*** (186, 3, 1005) | $aps^{(3)}(14)$ (186, 4, 1005) |
| | 185 | $aps^{(3)}(4)$ (185, 3, 1005) | ***not PD*** (185, 4, 1005) |
| | 184 | $aps^{(3)}(3)$ (184, 3, 1005) | $aps^{(3)}(13)$ (184, 4, 1005) |
| | 183 | $aps^{(3)}(2)$ (183, 3, 1005) | $aps^{(3)}(12)$ (183, 4, 1005) |
| | 182 | $aps^{(3)}(1)$ (182, 3, 1005) | ***not PD*** (182, 4, 1005) |
| | 181 | $aps^{(3)}(0)$ (181, 3, 1005) | $aps^{(3)}(11)$ (181, 4, 1005) |
| | 180 | ***not PD*** (180, 3, 1005) | $aps^{(3)}(10)$ (180, 4, 1005) |

For a **mapping** between the **terms** (e.g., **protected block**) and the **symbols** (e.g., "$N_{blk}^{CBS(q)}$") of **PDSCH** and **PUSCH processing** used in **this book** and the corresponding **terms** and **symbols** (e.g., ***A***, ***C***, …) used in the **TS documents** click 4.12.17.

For more detail on **mapping antenna port symbols** to the **assigned PDSCH payload VREs**, refer to clauses 7.3.1.5 (for **PDSCH**) and 6.3.1.6 (for **PUSCH**) of TS 38.211.

## 4.12.15 PDSCH and PUSCH payload VRE to payload PRE mapping

This consists of **mapping each payload Virtual Resource Element** (**VRE**) in **port** "(*vsc*, *si*, *ap*)" to **1 payload Physical resource element** (**PRE**) in **port** "(*sc*, *si*, *ap*)" of the **same port**.

There are "$N_{layer}$" **sets** of **payload VRE sequences** in **port**, and the **same quantity** of **sets** of **payload PREs** in **port**. **Each set** of **payload REs** in **port** contains "$N_{symb}^{layer}$" **symbols**.

**Each payload Virtual Resource Block** (**VRB**) that has been **assigned** to a given **UE** with **one DCI message** is mapped to a **payload Physical Resource Block** (**PRB**), where the **PRBs** will actually be **received** (**PDSCH**) or **transmitted** (**PUSCH**).

The **mapping** of **payload VRBs** to **payload PRBs** does **not change** the **RE port index value** "*ap*", **nor** the **RE symbol interval index value** "*si*": **each VRE** in **port** "($sc_a$, *si*, *ap*)" is **always** mapped to a **PRE** in **port** "($sc_b$, *si*, *ap*)", where "$sc_a$" and "$sc_b$"

are **spaced away** an **integer quantity** of **resource blocks** (i.e., a **multiple** of **12 subcarriers**).

**PDSCH payload VRB** to **payload PRB mapping** (click 4.12.15.1) and **PUSCH payload VRB** to **payload PRB mapping** (click 4.12.15.2) are **different**.

### 4.12.15.1 PDSCH payload VRB to payload PRB mapping

There are **two PDSCH payload VRB** to **PRB payload mapping** schemes: **non-interleaved mapping** and **interleaved mapping**. **Which one** to use depends firstly on whether **DFA-Type 0** or **DFA-Type 1 frequency resource allocation** is used. The **frequency resource allocation DFA-Type** to use is indicated by configuration.

- **DFA-Type 0** (**bitmap**, click 6.14.3.2) **frequency resource allocation**

  The **mapping scheme** is always **non-interleaved**.

- **DFA-Type 1** (**start-and-length**, click 6.14.3.3) **frequency resource allocation**

  The **mapping scheme** may be **non-interleaved** or **interleaved**. The **mapping scheme** is indicated in **DCI 1_x** field "***VRB-to-PRB mapping***": value **0** indicates **non-interleaved**, and value **1** indicates **interleaved**. In case no mapping scheme has been configured to the **UE**, the **UE** uses **non-interleaved** mapping.

**Non-interleaved VRB-to-PRB mapping scheme**

In the **non-interleaved mapping**, **each VRB** is mapped to the **same PRB**.

Consequently, **each VRE** in **port** "(***vsc***, ***si***, ***ap***)" is mapped to **1 PRE** in **port** "(***sc***, ***si***, ***ap***)" having the **same subcarrier index**, the **same symbol interval index** and the **same antenna port identifier**.

The **PRE** subcarrier **index** "***sc***" refers to the subcarrier index of the **active BWP**, except for **PDSCH** transmissions **assigned** with a **DCI 1_0 message** in a **common search space** (e.g., during **cell search**) in which case it corresponds to the **CORESET#0 subcarrier index**.

**Interleaved VRB-to-PRB mapping scheme**

In the **interleaved VRB-to-PRB mapping**, the **PRBs** that have the **same RB indexes** as the **PDSCH assigned VRBs** are **bundled** in groups of **consecutive PRBs** called **Physical Resource Block Bundles** (**PRBBs**). There are **three cases** for the **bundle size**, as follows:

**a)** **PDSCH assignment** with a **DCI 1_0 message** with the **CRC** masked by **SI-RNTI** in **CSS-Type 0 common** search space in **CORESET#0**:

   **Each PRB Bundle** is composed of **2** consecutive **PRBs**, starting from the **lowest frequency PRB** of **CORESET#0**. Recall that that the number of **PRBs** of **CORESET#0** is always **even**.

**b)** **PDSCH assignment** with a **DCI 1_0 message** in any **common** search space **other** than "**CSS-Type 0** in **CORESET#0**":

   **Each Physical Resource Block Bundle** is composed of **2** consecutive **PRBs**, **aligned** to the **Common Resource Block** (**CRB**) **range** (click 2.2.4.3), starting from the **lowest frequency assigned PDSCH PRB**. In case **CORESET#0** is defined, the **PRBBs** correspond to the **PRBs** of **CORESET#0**: otherwise, the **PRBBs** correspond to the **PRBs** of the **BWP**. The **alignment** of **PRBBs** to the

**CRB range** causes, as an exception, that the **first and/or last PRBB may** have only **1 PRB**, as follows:

- The **first PRBB** only has **1 PRB** if the **CRB index** of the **first assigned PDSCH PRB** is **odd**.
- The **last PRBB** only has **1 PRB** if the **CRB index** of the **last assigned PDSCH PRB** is **odd**.

Notice that this case applies to both: a) **all common** search spaces **not** in **CORESET#0**; and b) **all common** search spaces other than **CSS-Type 0** in **CORESET#0**.

When two **PDCCH** candidates from two **linked common search spaces** as indicated by parameter `searchSpaceLinkingId-17` from `SearchSpaceExt-v1700` are detected, and the two linked common search spaces are associated with different control resource sets, the control resource set with the lowest number among the two linked control resource sets is used.

**c)** For **all other cases**:

Each **Physical Resource Block Bundle** is composed of "$L_i$" consecutive **PRBs**, **aligned** to the **Common Resource Block (CRB) range**, starting from the **lowest frequency PRB**. The **alignment** of **PRBBs** to the **CRB range** causes, as an exception, that the **first and/or last PRBB may** have less than "$L_i$" consecutive **PRBs**, as follows:

- The **quantity** of **PRBs** of the **first PRBB** is:

$$L_i - (PRB_{BWP,i}^{start} \bmod L_i)$$

- The **quantity** of **PRBs** of the **last PRBB** is:

$$L_i - (PRB_{BWP,i}^{end} \bmod L_i)$$

The values in these expressions are as follows:

- "$L_i$" is the quantity of **PRBs** of each **PRBB** in **BWP *i***. For **DCI Formats 1_0** and **1_1** in a **UE specific search space**, "$L_i$" is set by parameter `vrb-ToPRB-Interleaver`. For **DCI Format 1_2**, "$L_i$" is set by parameter `vrb-ToPRB-InterleaverDCI-1-2`.
- "$PRB_{BWP,i}^{start}$" is the **CRB index** of the **first assigned PDSCH PRB** in **BWP *i***.
- "$PRB_{BWP,i}^{end}$" is the **CRB index** of the **last assigned PDSCH PRB** in **BWP *i***.

The difference between the **PRBBs** in case **a**) and case **b**) is that in case **a**) the **PRBBs** are **not** aligned to the **CRB range**. The difference between the **PRBBs** in case **b**) and in case **c**) is that in case **b**) the **PRBB size** is **2 PRBs**, while in case **c**) it is ***Li* PRBs**.

The resulting **PRBBs** are **indexed** from **0** (lowest frequency) to "$N_{bundle}$-1", where "$N_{bundle}$" is the quantity of **PRBBs**.

The **assigned PDSCH VRBs** are also **organized** in **bundles**, in the **same quantity** of **VRB bundles** "$N_{bundle}$", and are **indexed** from **0** to "$N_{bundle}$-1".

The **highest index VRBB** is **mapped** to the **highest index PRBB**. For the other bundles, the **VRBB** with **index** "***j***" is mapped to **PRBB** with **index** "***f(j)***", where the function "***f***()" is defined as:

For **even** ***j***: $f(j) = 0, 1, 2, \ldots, C\text{-}1$
For **odd** ***j***: $f(j) = C, C+1, C+2, \ldots, N_{bundle}-2$
where $C = floor(N_{bundle} / 2)$

Some **particular cases** for the above rules follow.

- The **UE** is not expected to be configured with $L_i$=2 simultaneously with a **Precoding Resource Block Group** (**PRG**) size of **4** as defined in TS 38.214.
- The **UE** may assume that the **same precoding** in the **frequency domain** is used within each **PRBB** and the **bundle size** is determined by clause 5.1.2.3 in TS 38.214. The **UE** shall not make any assumption that the same precoding is used for different bundles of common resource blocks.
- For **PDSCH** transmissions **assigned** with a **DCI Format 4_1** or **4_2**, and using **G-RNTI** or **G-CS-RNTI**, the quantities "$PRB_{BWP,i}^{start}$" and "$PRB_{BWP,i}^{end}$" in the rules above are replaced by "$PRB_{MBS,i}^{start}$" and "$PRB_{MBS,i}^{end}$", respectively, and "$L_i$" is the bundle size for the common **MBS frequency resource** provided by parameter `vrb-ToPRB-Interleaver` from `PDSCH-Config` (click 6.18.6), within parameter `pdsch-ConfigMulticast-r17` from `CFR-ConfigMulticast-r17`.
- For **PDSCH** transmissions **assigned** with a **DCI Format 4_0**, and using **G-RNTI** or **MCCH-RNTI**, the quantities "$PRB_{BWP,i}^{start}$" and "$PRB_{BWP,i}^{end}$" in the rules above replaced (respectively) by "$PRB_{MBS,i}^{start}$" and "$PRB_{MBS,i}^{end}$", and "$L_i$=2".

When two **PDCCH** candidates from two **linked common search spaces** as indicated by parameter `searchSpaceLinkingId-17` from `SearchSpaceExt-v1700` are detected, and the two linked common search spaces are associated with different control resource sets, the control resource set with the **lowest quantity** among the two linked control resource sets **is used**.

For a **mapping** between the **terms** (e.g., **protected block**) and the **symbols** (e.g., "$N_{blk}^{CBS(q)}$") of **PDSCH** and **PUSCH processing** used in **this book** and the corresponding **terms** and **symbols** (e.g., ***A***, ***C***, …) used in the **TS documents** click 4.12.17.

For more detail on the **PDSCH mapping** from the **scheduled Virtual Resource Blocks** (**VRBs**) to **scheduled Physical Resource Blocks** (**PRBs**), refer to clause 7.3.1.6 of TS 38.211.

### 4.12.15.2 PUSCH payload VRB to payload PRB mapping

There is **only 1 PUSCH payload VRB** to **PRB payload mapping** scheme: **non-interleaved mapping**.

For **PUSCH frequency-domain resource allocation UFA-Type 0** and **UFA-Type 1** (click 7.11.3), **each VRB index** is **mapped** to the **same PRB index**, except for:

- **PUSCH scheduled** by an **Uplink grant** within a **random access response** (**RAR**, click 9.3.3).
- **PUSCH scheduled** by a **DCI format 0_0 message** with the **last 16 bits** of its **CRC masked** by **TC-RNTI** in the **active Uplink bandwidth part** "***i***" starting at

"$N_{BWP,i}^{start}$", including all resource blocks of the **initial Uplink bandwidth part** starting at "$N_{BWP,0}^{start}$", and having the **same subcarrier spacing** and **cyclic prefix** as the **initial uplink bandwidth part**,

In **either** of these **2 cases**, **each VRB** with **index** "***n***" is **mapped** to **PRB** with **index**:

$$n + N_{BWP,0}^{start} - N_{BWP,i}^{start}$$

For **PUSCH resource allocation UFA-Type 2** click 7.11.3), **each VRB index** is **mapped** to the **same PRB index**.

For a **mapping** between the **terms** (e.g., **protected block**) and the **symbols** (e.g., "$N_{blk}^{CBS(q)}$") of **PDSCH** and **PUSCH processing** used in **this book** and the corresponding **terms** and **symbols** (e.g., ***A***, ***C***, …) used in the **TS documents** click 4.12.17.

For more detail on **mapping** the **assigned Virtual Resource Blocks** (**VRBs**) to **assigned Physical Resource Blocks** (**PRBs**), refer to clause 7.3.1.6 of TS 38.211.

## 4.12.16 PDSCH and PUSCH Rx payload processing overview

**Data** are **received** by the **NR node** (**PUSCH**) or by **each UE** (**PDSCH**) as **signals** arriving to **each** of its **1** or **more Rx antennas**. In case of **space-division MIMO**, the **NR node** or **each UE** can obtain the **transmitted PDSCH** or **PUSCH layers** after processing its **multiple Rx signals** with a **detector** (click 2.4.2). The result is "$N_{layer}$" **sets** of **scheduled payload Physical Resource Elements** (**PREs**) in **port** "(*sc*, *si*, *po*)".

The **physical layer processes** these "$N_{layer}$" **sets** and the **resulting 1** or **2 ordered transport block**(**s**) are handed **upwards**. For **PDSCH**, they are handed **either** through the **transport Downlink shared channel** (**DL-SCH**) or through the **transport paging channel** (**PCH**). For **PUSCH**, they are handed through the **transport Uplink shared channel** (**UL-SCH**), and there may also be a **received UCI message**.

In order to **process** the **received** "$N_{layer}$" **sets** of **PREs** in **port** "(*sc*, *si*, *po*)", the **NR node** has **all** the required **state information**. However, the **UE must** have **previously** received a **DCI message 1_x** or **4_x** that indicates:

- The **location** of its **assigned PDSCH PREs**. For more detail, click 6.14.5.
- The **modulation** and **coding** used for **each** of the **1** or **2 sequence**(**s**) of **rate-matched block**(**s**). For more detail, click 6.16.1.
- The **redundancy version** "$rv_{id}$" for **each**, or **both**, of the **1** or **2 sequence**(**s**) of **rate-matched block**(**s**). For more detail, click 6.16.1.
- Whether there is **1** or **2 sequence**(**s**) of **rate-matched block**(**s**) and the **bitlength** "$N_{bit}^{TB(q)}$" of **each** of the **1** or **2 ordered transport block**(**s**). For more detail, click 6.16.2.
- The **LDPC encoding** (**base graph** and **code rate**) used for **each** of the **1** or **2 sequence**(**s**) of **rate-matched block**(**s**). For more detail, click 6.16.4.

The **processing** of the **received** "$N_{layer}$" **sets** of **PREs** in **port** "(*sc*, *si*, *po*)", is the **PUSCH Rx payload processing**, and it is the **reciprocal** to the **transmission processing** (click 4.12.2).

The **scheduled payload Physical Resource Elements** (**PREs**) are **processed** to obtain the **1 or 2 transport block**(**s**) (click 4.12.1) and/or an **UCI message** in **up to** the following **10 stages**:

1. **De-map each payload Physical Resource Element (VRE)** in **port** "(*sc*, *si*, *ap*)" to **1 payload Virtual Resource Element (PRE)** in **port** "(*vsc*, *si*, *ap*)" of the **same port**.

   The variable "*vsc*" is the **virtual subcarrier index**, "*sc*" is the **subcarrier index**, "*si*" is the **symbol interval index** and "*p*" is the **antenna port index**.

   There are "$N_{layer}$" **sets** of **VRE**s in **port**, and the **same quantity** of **sets** of **VRE**s in **port**. **Each** and **every set** of **RE**s in **port** contains "$N_{symb}^{layer}$" **symbols**.

   The variable "$N_{layer}$" is the **quantity** of **layers**.

   For more detail, click 4.12.15.

2. **De-map each** of the "$N_{layer}$" **sets** of **Rx payload VREs** in **port** "$(vsc, si, ap)_{\mu}$" to **1 symbol** of the **antenna port sequence** "$aps^{(apn)}(j_{aps})$" of the **same antenna port**.

   **All** the **payload VREs** in **port** are **received** in **1 PDSCH reception** or in **1 PUSCH reception**.

   **Each** and **every antenna port sequence** "$aps^{(apn)}(j_{aps})$" has the **same symbol length** "$N_{symb}^{layer}$". The **symbol index** "$j_{aps}$" ranges from 0 to "-1".

   For more detail, click 4.12.14.

3. For **PDSCH**, **port-layer de-mapping**. For **PUSCH**, **optional inverse DFT** or **optional diversity combining**.

   For **PDSCH**, **de-map each antenna port number** "$ap_n$" to **1 layer** "*lay*". The **set** "$\{ap_0, .., ap_{(Nlayer-1)}\}$" of **antenna port numbers** shall be determined according to the procedure in TS 38.212.

   For more detail on **PDSCH**, **port-layer de-mapping**, refer to clause 7.3.1.4 of TS 38.211.

   For **PUSCH**:

   - Perform an **inverse Fourier Transform**, **only** in case that **transform precoding** (click 4.12.12) was applied in **PUSCH transmission**.
   - Perform **detection** (click 2.4.2), **only** in case that **precoding** (click 4.12.13) was applied in **PUSCH transmission**.

   The result of this **stage** is "$N_{layer}$" **layer sequence(s)** "$lys^{(lay)}(j_{lay})$" of **complex valued symbols**.

   The **symbol sequence** "$lys^{(layer)}(j_{lys})$" of **every layer sequence** has the **same symbol length** "$N_{symb}^{layer}$". The **symbol index** "$j_{lys}$" **ranges** from **0** to "$N_{symb}^{layer}$-1".

   The **layer index** "*lay*" ranges from **0** to "$N_{layer}$-1". The variable "$N_{layer}$" is the **quantity** of **layers**.

4. **De-map** the "$N_{layer}$" **layer sequence(s)** into **1** or **2 modulated sequence(s)**.

   The **symbol sequence** "$ms^{(q)}(j_{ms})$" of **each modulated sequence** "**q**" has **symbol length** "$N_{symb}^{MS(q)}$". The **symbol index** "$j_{ms}$" **ranges** from **0** to the "$N_{symb}^{MS(q)}$-1".

   For more detail, click 4.12.11.

5. **Demodulate each** of the **1** or **2 modulated sequence(s)** of **complex-valued symbols** into **1** or **2 scrambled codeword(s)**.

The **bit sequence** "$scw^{(q)}(i_{scw})$" of **each scrambled codeword** has **bitlength** "$N_{bit}^{SCW(q)}$". The **bit index** "$i_{scw}$" **ranges** from **0** to "$N_{bit}^{SCW(q)}$-1".

For more detail, click 4.12.10.

6. **Descramble each** of the **1** or **2 scrambled codeword(s)** into **1** or **2 codeword(s)** or into **1 multiplexed codeword**.

   For **PDSCH** it is **1** or **2 codeword(s)**. For **PUSCH** it is **either 1 codeword** or **1 multiplexed codeword**.

   The **bit sequence** "$cw^{(q)}(i_{cw})$" of **each codeword** "**q**" has **bitlength** "$N_{bit}^{CW(q)}$", where "$N_{bit}^{CW(q)}=\Sigma_{tn}N_{bit}^{RBtn(q)}$". The **bit index** "$i_{cw}$" **ranges** from **0** to "$N_{bit}^{CW(q)}$-1".

   The **bit sequence** "$mcw(i_{mcw})$" of **each multiplexed codeword** has **bitlength** "$N_{bit}^{MCW(0)}$". The **bit index** "$i_{mcw}$" **ranges** from **0** to "$N_{bit}^{MCW(0)}$-1".

   For more detail, click 4.12.9.

7. **PUSCH only** and **optional**: **demultiplex 1 multiplexed codeword** into **1 PUSCH codeword** and/or **1 PUCCH codeword** (**UCI message**).

   For more detail, click 4.12.8.

8. **Segment each** of the **1** or **2 codeword(s)** into **1** or **2 sequence(s)** of "$N_{blk}^{RBS(q)}$" **received rate-matched block(s)**.

   The **bit sequence** "$rb_{rn}^{(q)}(i_{rb})$" of **each rate-matched block** that belongs to the **received rate-matched block sequence** "**q**" has its **specific bitlength** "$N_{bit}^{RBtn(q)}$". The **bit index** "$i_{rb}$" **ranges** from **0** to "$N_{bit}^{RBS(q)}$-1".

   The variable "***rn***" is the **block index** (within the **complete rate-matched block sequence** "**q**") of the **received rate-matched blocks**. The variable "***rn***" takes "$N_{blk}^{EBS(q)}$" values from **0** to "$N_{blk}^{CBS(q)}$-1".

   For more detail, click 4.12.7.

9. **De-rate-match each rate-matched block** resulting in **1** or **2 sequence(s)** of "$N_{blk}^{RBS(q)}$" **encoded block(s)**.

   The **bit sequence** of **each encoded block** "$rb_{tn}^{(q)}(i_{rb})$" has **bitlength** "$N_{bit}^{RBtm(q)}$". The **bit index** "$i_{rb}$" **ranges** from **0** to "$N_{blk}^{RBS(q)}$-1".

   The variable "***rn***" is the **block index** (within the **complete encoded block sequence** "**q**") of the **received encoded blocks**. The variable "***rn***" takes "$N_{blk}^{EBS(q)}$" values from **0** to "$N_{blk}^{CBS(q)}$-1".

   For more detail, click 4.12.6.

10. **LDPC decode each received encoded block** into **1 code block**. **Every code block** "$cb_{cn}^{(q)}(i_{cw})$" has the **same bitlength** "$N_{bit}^{CB(q)}$". **Every encoded block** "$eb_{rn}^{(q)}(i_{eb})$" has the **same bitlength** "$N_{bit}^{EB(q)}$". The following procedure is then applied:

    - **Each encoded block** is **decoded**, **jointly** with **every** of its previous **transmissions** (if any) that are stored in the **Rx wrong buffer**$^{(q)}$. If it is **correctly decoded** (including the **code block CRC**) it is **stored** in its corresponding **sequence** position in the **Rx correct buffer**$^{(q)}$.

- In **TB mode** (click 4.5.6.4):
  - If **any** wrong **code block** is **received**, a **NACK** is sent for the corresponding **transport block**.
  - If **all** the **correct code block(s)** of a **protected block** have been **received**, they are **concatenated** and the **CRC** of the resulting **protected block** is verified. **If** the **CRC** is wrong, a **NACK** is sent for the corresponding **transport block**. **Otherwise**, **both** the **correct Rx buffer**$^{(q)}$ and the **wrong Rx buffer**$^{(q)}$ are **cleared**, and **if** this **PDSCH**, an **ACK** is sent.
- In **CBG mode** (click 4.5.6.4):
  - If **any** wrong **code block** is **received**, a **NACK** is sent for the corresponding **CBG**.
  - If **all** the **correct code block(s)** of a **CBG** have been **received**, and this is **PDSCH**, an **ACK** is sent for the corresponding **CBG**.
  - If **all** the **correct code block(s)** of a **protected block** have been **received**, they are **concatenated** and the **CRC** of the resulting **protected block** is verified. **If** the **CRC** is **wrong**, the **protected block** is **discarded**. **Both** the **correct Rx buffer**$^{(q)}$ and the **wrong Rx buffer**$^{(q)}$ are **cleared**.
- If **transport block** "**0**" has been **received**, it is **delivered upwards**.
- If **transport block** "**1**" has been **received**, and **either transport block** "**0**" has been **received** or the **maximum quantity** of **retransmissions** "$N_{Rtx,max}^{HARQ}$" for **TB** "**0**" has been **reached**, then the **transport block** "**1**" is **delivered upwards**.
- For **PDSCH**, the **transport channel** used for the **upwards delivery** of a **transport block** depends on the **RNTI** (click 6.9.10) used to **mask** the associated **DCI message**. If it is a **C-RNTI** or **SI-RNTI**, then the **transport Downlink shared channel** (**DL-SCH**) is used. If it is a **P-RNTI**, then the **transport paging channel** (**PCH**) is used.

  For **PUSCH**, the **transport channel** used for the **upwards delivery** of a **transport block** is always the **transport Uplink shared channel** (**UL-SCH**).

Notice that **neither code block(s) nor code block group(s)** carry a **sequence number**. However, they are **correctly acknowledged** and **ordered** based on their **sequential position** within the **allocated VREs**, **both** in **new data** and in **successively retransmitted data**.

For more detail on **LDPC decoding**, click 4.12.5, 4.12.4 and 4.5.6.

For **PUSCH** it is always **only 1 transport block** and **only 1 codeword**.

For the **PDSCH** and **PUSCH Tx payload processing stages**, click 4.12.2.

For the **NR node functional blocks**, the **NR node physical functions**, the **payload processing** of the **physical binary channels**, the **protocol architecture** of the **New Radio** and the **data flow** through the **protocol architecture**, click 4.3.

### 4.12.17 Mapping to PDSCH/PUSCH terms and symbols in TS documents

The **terms** and **symbols** (e.g., $a_i$, $b_i$, $c_i$, $A$, $B$, $C$, …) used in the **TS documents** do **not help** to **understand** the explanations about the **PDSCH** and **PUSCH payload processing**. This is why in **this book** the **terms** and **symbols** used in **TS documents** have been **replaced** by **more illustrative terms** and **symbols**. The same has been done for the **terms** and **symbols** used **UCI message processing** (click 7.8.3).

The following table presents a **mapping** between the **PDSCH/PUSCH processing terms** and **symbols** in **this book** and in the **TS documents**, to facilitate the **cross-checking** of information between **this book** and the **TS documents**:

| | Terminology mapping | |
|---|---|---|
| **Element** | **This book** | **TS 38.211 and 38.212** |
| **Term** | **transport block "q"** | **transport block** |
| **Transport block index** | **q** | **q** |
| **Bit sequences** | $tb^{(q)}(i_{tb})$ | $a_i$ |
| **Bit sequence index** | $i_{tb}$ | $_i$ (subindex) |
| **Bit seq. length [bits]** | $N_{bit}^{TB(q)}$ | $A$ |
| **Term** | **protected block "q"** | **no term** |
| **Bit sequences** | $pb^{(q)}(i_{pb})$ | $b_i$ |
| **Bit sequence index** | $i_{tb}$ | $_i$ (subindex) |
| **Bit seq. length [bits]** | $N_{bit}^{PB(q)}$ | $L$ |
| **Term** | **CRC$^{PB(q)}$ of protected block "q"** | **CRC of transport block** |
| **Bit sequences** | $crc_{PB}^{(q)}(i_{crcpb})$ | $p_i$ |
| **Bit sequence index** | $i_{tb}$ | $_i$ (subindex) |
| **Bit seq. length [bits]** | $N_{bit}^{CRC\text{-}PB(q)}$ | $L$ |
| **Term** | **code block sequence "q"** | **sequence of code blocks** |
| **Quantity of blocks** | $N_{blk}^{CBS(q)}$ | $C$ |
| **Complete block index** | $cn$ | $r$ |
| **Term** | **code block of seq. "q"** | **code block** |
| **Bit sequences** | $cb_{cn}^{(q)}(i_{cb})$ | $c_{ri}$ |
| **Bit sequence index** | $i_{cb}$ | $_i$ (subindex) |
| **Max. bit seq. length [bits]** | $N_{bit}^{CB(q)}{}_{max}$ | $K_{cb}$ |
| **Bit seq. length [bits]** | $N_{bit}^{CB(q)}$ | $K_r$ |
| **Term** | **CRC$^{CB(q)}$ of code block of sequence "q"** | **CRC of code block** |
| **Bit sequence** | $crc_{CBcn}^{(q)}(i_{crccbcn})$ | $p_{ri}$ |
| **Bit seq. length [bits]** | $N_{bit}^{CRC\text{-}CB(q)}$ | $L$ |
| **Term** | **lifting size** | **lifting size** |
| **Lifting size [bits]** | $LS$ | $Z$ |
| **Min. compliant lifting size [bits]** | $LS_{minc}$ | $Z_c$ |
| **Term** | **encoded block sequence "q"** | **sequence of encoded code blocks** |
| **Quantity of stored blocks** | $N_{blk}^{CBS(q)}$ | $C$ |
| **Quantity of transmitted blocks** | $N_{blk}^{EB(q)}$ | $C$ |
| **Complete block index** | $cn$ | $r$ |
| **Possibly partial block list** | $tn$ | $r$ |

| | Terminology mapping | |
|---|---|---|
| **Element** | **This book** | **TS 38.211 and 38.212** |
| **Term** | **encoded block** of **sequence "q"** | **encoded code block** |
| **Stored bit sequences** | $eb_{cn}^{(q)}(i_{eb})$ | $d_{ri}$ |
| **Transmitted bit sequences** | $eb_{tn}^{(q)}(i_{eb})$ | $d_{ri}$ |
| **Bit sequence index** | $i_{eb}$ | $_i$ (subindex) |
| **Bit seq. length [bits]** | $N_{bit}^{EB(q)}$ | $N_r$ |
| **Term** | **rated-matched block sequence "q"** | **sequence** of **rate-matched code blocks** |
| **Quantity of blocks** | $N_{blk}^{EB(q)}$ | $C$ |
| **Transmitted block index** | $tn$ | $r$ |
| **Received block index** | $rn$ | $r$ |
| **Term** | **rate-matched block** of **sequence "q"** | **rate-matched code block** |
| **Bit sequence** | $rb_{tn}^{(q)}(i_{rb})$ | $f_{ri}$ |
| **Bit sequence index** | $i_{rb}$ | $_i$ (subindex) |
| **Bit seq. length [bits]** | $N_{bit}^{RBtn(q)}$ | $E_r$ |
| **Term** | **codeword "q"** | **codeword** of **TB "q"** |
| **Bit sequence** | $cw^{(q)}(i_{cw})$ | $g_i$ (in TS 38.212)<br>$b_i^{(q)}$ (in TS 38.211) |
| **Bit sequence index** | $i_{cw}$ | $_i$ (subindex) |
| **Bit seq. length [bits]** | $N_{bit}^{CW(q)}$ | $G$ (in TS 38.212)<br>$M_{bit}^{(q)}$ (in TS 38.211) |
| **Term** | **multiplexed codeword "0"** | **codeword** of **TB "0"** |
| **Bit sequence** | $mcw^{(0)}(i_{mcw})$ | $g_i$ (in TS 38.212)<br>$b_i^{(0)}$ (in TS 38.211) |
| **Bit sequence index** | $i_{mcw}$ | $_i$ (subindex) |
| **Bit seq. length [bits]** | $N_{bit}^{MCW(0)}$ | $G$ (in TS 38.212)<br>$M_{bit}^{(0)}$ (in TS 38.211) |
| **Term** | **scrambled codeword "q"** | **scrambled codeword** of **TB "q"** |
| **Bit sequence** | $scw^{(q)}(i_{cw})$ | $b_{\sim i}$ |
| **Bit sequence index** | $i_{cw}$ | $_i$ (subindex) |
| **Bit seq. length [bits]** | $N_{bit}^{SCW(q)}$ | $M_{bit}^{(q)}$ |
| **Term** | **modulated sequence "q"** | **modulated sequence** |
| **Symbol sequence** | $ms^{(q)}(j_{ms})$ | $d^{(q)}{}_i$ |
| **Symbol sequence index** | $j_{ms}$ | $_i$ (subindex) |
| **Sym. seq. length [symbs]** | $N_{symb}^{MS(q)}$ | $M_{symb}^{(q)}$ |
| **Term** | **Layers** | **Layers** |
| **Quantity of layers** | $N_{layer}$ | $v$ |
| **Term** | **layer sequence "*lay*"** | **layer mapped sequence** |
| **Layer identifier** | $lay$ | $v$ |
| **Symbol sequence** | $lys^{(lay)}(j_{lys})$ | $x_i$ |
| **Symbol sequence index** | $j_{lys}$ | $_i$ (subindex) |
| **Sym. seq. length [symbs]** | $N_{symb}^{layer}$ | $M_{symb}^{layer}$ |

| | Terminology mapping | |
|---|---|---|
| **Element** | **This book** | **TS 38.211 and 38.212** |
| **Term** | **antenna port sequence "*aps*"** | **port mapped sequence** |
| **Quantity of ports** | $N_{port}$ | $\rho$ |
| **Antenna port identifier** | $ap$, $ap_0$,.., $ap_{Nport-1}$ | $p_0$, .., $p_{\rho-1}$ |
| **Symbol sequence** | $aps^{(ap)}(j_{aps})$ | $y_i$ or $z_i$ |
| **Symbol sequence index** | $j_{aps}$ | $_i$ (subindex) |
| **Sym. seq. length [symbs]** | $N_{symb}^{layer}$ | $M_{symb}^{po}$ |
| **VRE in port** | ($vsc$, $si$, $ap$) | $(k', l)_p$ |
| **PRE in port** | ($sc$, $si$, $ap$) | $(k, l)_p$ |
| **Virtual subcarrier index** | $vsc$ | $k'$ |
| **Subcarrier index** | $sc$ | $k$ |
| **Symbol interval index** | $si$ | $l$ |

Some **relevant remarks** on this table are:

- The term "***i***" is the **bit index** within **each** and **every bit sequence**, **both** in **this book** and in **TS documents**. In **this book** is expressed as a **variable** "bit(***i***)", while in the **TS document** is expressed as a **subindex** "$bit_i$".
- The term "***j***" is the **symbol index** within **each** and **every symbol sequence**, in **this book**, while in **TS documents** it is "***i***". In **this book** is expressed as a **variable** "symbol(***j***)", while in the **TS document** is expressed as a **subindex** "$symbol_i$".
- The text "**TB**" stands for "**transport block**".
- The text "**symbs**" stands for **symbols**.

# 5. UE configuration parameters and the RRC protocol

The **UE** requires a **very large** quantity (up to **thousands**, click 5.7.8) of **parameter instances** (click 5.7.2.3) to **configure** its **physical layer**. The **UE** needs **all these parameter instances** to **correctly interoperate** with the **RAN NR node**(**s**) the control the **serving cell**(**s**) where the **UE camps**. These **parameters** to **configure** the **UE**'s **physical layer** will be usually called "**UE parameters**" within this book. The **UE parameters** are **specified** in TS 38.331.

In addition to the **UE parameters**, TS 38.331 also specifies the **RRC messages** (click 5.8.5), the **parameters** that are sent by the **UE** to the **NR node** (click 4.1) and the **UE variables** (click 5.6.2).

The **network provides** the **UE** with **all** these **parameter instances**. In case at any given moment the **UE** is **missing** some of these parameters, it applies its **specified** or **default configuration** (click 5.7.3.7).

The **UE parameters** can be **classified** based on **different criteria** (click 5.1).

The **UE parameters** can be organized into **six UE parameter groups** (see the table header below). **Each** of the **six UE parameter groups** has its own **specific characteristics** (click 5.2).

In addition to the **UE parameters** (click 5.6), TS 38.331 also contains the specification of **UE parameters** that are **sent** from the **UE** to the **NR node** (click 5.6.1) and the specification some **UE variables** (click 5.6.2).

The **UE** gets its **UE parameters** in **several stages** (click 5.4).

This **chapter** does **not contain** the specific explanations for the **UE parameters** (and their **data types**) used to configure **each UE physical layer element**. The said explanations are contained in a **specific section** for **each physical layer element**. For a **listing** of the said sections, click 5.4.

The configuration **parameters** for the **mapping** and **scheduling** of **other SIBs and posSIBs** are presented in 5.5.

The **higher-layer UE parameters** (see table below) contain **almost all** the **UE parameters**. They are defined as part of the **RRC protocol specification** in TS 38.331. The said **TS** document is **quite cryptic**, and you may therefore find useful the **explanation** about it from 5.6.

The **specification** of almost **all UE parameters** is done within the definition of the **RRC protocol**. Also, the **NR RRC messages** of the **RRC protocol** are used (among other functions) to deliver **most** of the **UE parameters**. For a description of the **RRC protocol**, click 5.8.

**Most** of the **UE parameters** (as well as other **5G** system elements) are **specified** using **ASN.1** (click 5.9).

For some **conventions** on **fonts** and **colors** related to **UE parameters** and other **data elements**, click 1.3.

The following **table summarizes** the characteristics of the **UE parameters** sent by the **network** to configure the **UE**'s **physical layer**. The **second row** of **the table body** lists the **six UE parameter groups**. The **first four rows** of the **table body** contain **criteria** to

**classify** the **UE parameters**. **Each** of the **remaining rows** of the **table body** contains the **summary** of a **given characteristic** for the corresponding **UE parameters**.

| | **UE parameters (sent by the network)** | | | | | | |
|---|---|---|---|---|---|---|---|
| | **PCI par. Group** (5.3) | **PBCH-SI par. group** (5.3.2) | **MIB par. group** (5.3.3) | **SIB1 par. group** (5.3.4) | **Other SIBs and posSIBs par. group** (5.3.5) | **Non-SI par. gr.** (5.3.6) | |
| | | | | | | **Non-SI common pars.** | **Dedicated pars.** |
| **Production layer** | Parameters produced by physical layer | | Parameters produced by higher layers (defined as part of the RRC protocol) | | | | |
| **SI or Non-SI** | SI parameters | | | | | Non-SI parameters | |
| **Sub-types of SI parameters** | Minimum System Information (MSI) parameters | | | | Other System Information pars. | N/A | |
| **Configuration type** | Common parameters | | | | | | Dedicated parameters |
| **Scope** | All UEs in the cell where it is sent | | | | All UEs in the cell or in an SI area | All UEs in expl. cell | Spec. UE in expl. cell |
| **Production timing** | * | Every SSB | *Every 80 ms | Every 160 ms | N/A | | |
| **Delivery timing** | Every SSB | | | *Every 20 ms | *Periodic or on-demand (SIB1) | N/A | |
| **Data represent.** | Bitfield | | Specified as ASN.1 data types | | | | |
| **Delivery procedure(s)** | Cell search | | | Cell select. and RRC reconfig. | After receiving SIB1 | RRC connection establishment, resume and re-establishment | |
| **Delivered as** | Seq. var. | Pld. & seq. var. | Payload REs | | | | |
| **Delivery mode** | Broadcast in every network cell type (except NSSB-Cells) | | | Initially broadcast. Later, also unicast | Broadcast and/or unicast | Unicast in any network or UE cell type | |
| **Scheduling** | In i.a.: cell search. After i.a.: semi-persistent with common RRC parameters. | | | DCI 1_0 SI-RN | DCI 1_0 SI-RN | DCI Format 1_x message | |
| **Sent in physical channel or signal** | PSS/SSS | PBCH | PDSCH | | | | |

Some **relevant remarks** on this table are:

- Row **2** (in the heading): the text "**par. group**" and "**pars.**" in some table cells stand (respectively) for "**parameter group**" and "**parameters**".
- Rows **1** to **4** (in the table body) contain the **4 classification criteria** (click 5.1) of the **UE parameters**. **Each** of the **next 9** rows contains **1 characteristic** of the **UE parameter groups**.
- Row "**SI or Non-SI**": the text "**SI**" and "**Non-SI**" stand (respectively) for "**System Information**" and "**Non System Information**".
- Notice that **some common parameters** are included in the **SI parameters** while **others** are included in the **Non-SI parameters**. However, the **dedicated parameters** are **only** included in the **Non-SI parameters**.
- Row "**Scope**": the text "**expl.**" stands for "**explicit**". It indicates that the **serving cell** to which **each** of the different **sets** of **parameters** applies is **explicitly** indicated **within each parameter set**.

- Row "**Production timing**": the asterisk (*) indicates:
  - For **PCI**, its **production timing is whenever** its value is changed for the cell, which is a very long period (months/years).
  - For **MIB**, its **production timing is periodical**. The period is **80 m**s, unless the **SSB burst period** is larger, in which case the **period** is the same as the **SSB burst period**.
- Row "**Delivery timing**": the asterisk (*) indicates:
  - For **SIB1**, its usual **broadcast period** is **20** ms, but there are some cases in which it can have a **different** value (click 5.3.4).
  - For **other SIBs** and **posSIBs**, **both** their **broadcast period** and **which** ones can be delivered **on-demand** is configured with specific **SIB1 parameters** within **SIB1**.
- Row "**Data represent.**": the text "**Seqs.**" stands for "**sequences**". It indicates that the information is conveyed by mapping it to a **sequence**, **chosen** from a given **set** of **sequences**.

  Notice that although **MIB**, **SIB1** and **other SIBs** and **posSIB**s are **first** received by the **UE before** the **RRC connection establishment**, they are **specified** in **ASN.1** as an **NR RRC message** (click 5.8.5) or as an **RRC data type**.
- Row "**Delivery procedure(s)**": for more detail on **cell search**, click 8.2. The text "**Cell select.**" stands for "**cell selection**" (click 8.1). For more detail on **RRC connection establishment**, click 8.11.
- Row "**Scheduling**": the text "**i.a.**" stands for "**initial access**" (click 8). The text "**Bcst**" and "**Ucst**" stand (respectively) for **Broadcast** and **Multicast**. The text "**DCI 1_0 SI-RN**" stands for **DCI Format 1_0 message** with its **CRC** masked with **SI-RNTI**. The text "**Per**" and "**On-dem**" stand (respectively) for "**Periodical delivery**" and "**On-demand delivery**".

Do not confuse the **delivery mode** of the parameters with the **nature** of the **parameters itself** (click 5.3.7): even if **common** parameters are on some cases **delivered** via **unicast dedicated signaling**, it does not change the **fact** that they are **common** parameters, and they are **not dedicated** parameters.

In addition to the **UE**'s **physical layer configuration parameters** delivered by the **network** to the **UE**, the **network** and the **UE constantly exchange physical layer control information**. The **network** delivers to the **UE Downlink control information** (**DCI**) **messages** and the **UE** delivers to the network **Uplink control information** (**UCI**) **messages**.

For more detail on **SI parameters**, refer to clause 5.2 of TS 38.331 and clause 7.3.1 of TS 38.300.

For more detail on **Non-SI parameters**, refer to TS 38.331.

Source web pages:

- http://howltestuffworks.blogspot.com/2019/10/5g-nr-system-information.html

## 5.1 Classifications of UE parameters

The **UE parameters** sent by the network may be **classified** according to different **criteria**. The **table** from 5 (see slightly above) summarizes their **classification** under the following four **criteria**:

- Criterion of **production protocol layer** (click 5.1.1).
- Criterion of belonging to **System Information or not** (click 5.1.2).
- Criterion of **configuration type** (**common or dedicated**) (click 5.1.3).
- For **System Information** parameters, criterion of belonging to **Minimum System Information** (**MSI**) or to **other System Information** (click 5.1.4).

### 5.1.1 Criterion of production protocol layer

Depending on the **protocol layer where they are produced**, we can identify the following **two classes** of **UE parameters**:

- **Physical layer** (**L1**) **parameters**

  This **class** contains the **parameters produced** by the **physical layer itself**.

  It contains the **parameters** within the **PCI** (click 5.3) and the **PBCH SI** (click 5.3.2) parameter groups.

- **Higher-layers parameters**

  This **class** contains the **parameters produced** by **layers** above the **physical layer**.

  It contains the **parameters** within the `MIB` (click 5.3.3), `SIB1` (click 5.3.4), **other SIBs** and **posSIBs** (click 5.3.5) and **non-SI** (click 5.3.6) parameter groups.

  Do not mistake the **production layer** with the layer to which the **parameters apply**. Even if these parameters are **produced** by **higher layers**, they apply **only** to the configuration of the **UE**'s **physical layer**.

  In the **TS** documents, they repeatedly use the text "**higher layer parameter**" to refer to higher layer parameters. In this book, we will identify **higher layer parameters** by writing their **names** in `blue Courier font` and will **not repeat** the text "higher layer parameter".

  **Almost all** the **higher-layers** parameters are defined as part of the **RRC protocol**, in TS 38.331. For this reason, in this book we will sometimes call them "**RRC UE parameters**".

  For more detail on **higher-layers parameters**, click 5.6.

For the **font** and **color combinations** that we use in this book when **writing** the **names** of different data elements, click 1.3.

### 5.1.2 Criterion of belonging to System Information or not

Depending on whether they provide (or **not**) the **very basic** configuration **before RRC set up** has taken place, we can identify the following **two** classes of **UE parameters**:

- **System Information parameters**

  This **class** contains the **parameters** that provide the **very basic** configuration **before RRC set up** has taken place. **System information** parameters consist of **a**

**small subset** of configuration parameters that the **UE** needs to receive for its **primary cell** t

It contains the **parameters** within the **PCI** (click 5.3), **PBCH SI** (click 5.3.2), `MIB` (click 5.3.3), `SIB1` (click 5.3.4) and **other SIBs** and **posSIBs** (click 5.3.5) parameter groups.

- **Non System Information parameters**

  This **class** contains the **parameters delivered after RRC set up** has taken place. **Non system information** parameters consist of **almost all** the configuration parameters the the **UE** needs to configure its **physical layer** in respect to **all** the **cells** and **all** the **RAN nodes** to which it is **connected** at any given time.

  It contains the **parameters** within the **non-SI** (click 5.3.6) parameter group. This is, it contains the **non-SI common parameters** and the **dedicated parameters**.

### 5.1.3 Criterion of configuration type (common or dedicated)

Depending on the **UEs to which they apply**, we can identify the following **two** classes of **UE parameters**:

- **Common or cell-specific parameters**

  This **class** contains the **parameters** that apply to **all** the **UEs** in **each serving cell**.

  It contains the **parameters** within the **PCI** (click 5.3), **PBCH SI** (click 5.3.2), `MIB` (click 5.3.3), `SIB1` (click 5.3.4) and **other SIBs** and **posSIBs** (click 5.3.5) parameter groups, **plus** the **non-SI common parameters** (from the **non-SI** (click 5.3.6) parameter group).

  The **two terms** "**common parameters**" and "**cell-specific parameters**" are used in different contexts to denote the same configuration parameters. The term "**common**" is used because these parameters are **common** to **all** the **UEs** in the serving cell. The term "**cell-specific**" is used because these parameters are **specific** for the corresponding **serving cell**. Throughout this book will mainly use the term "**common**".

- **Dedicated or UE-specific parameters**

  This **class** contains the **parameters** that apply to **one individual UE**.

  It contains the **dedicated parameters** (from the **non-SI** (click 5.3.6) parameter group).

  **Each and every** (set of) **dedicated parameter**(s) **must either supersede**, or be **coherent with**, its corresponding (set of) **common parameter**(s)

  The **two terms** "**dedicated parameters**" and "**UE-specific parameters**" are used in different contexts to denote the same configuration parameters. The term "**dedicated**" is used because these parameters are **dedicated** to **1 UEs**. The term "**UE-specific**" is used because these parameters are **specific** for the corresponding **UE**. Throughout this book will mainly use the term "**dedicated**".

**Parameters that are common or dedicated (context dependent)**

Some **parameters** may be **either common or dedicated** depending on **context**.

For example, **parameters** that have the **data type** `SCS-SpecificCarrier` (click 2.2.6.12) are **common** when they are included within an instance of

**UplinkConfigCommon** (click 2.2.6.9) through **FrequencyInfoUL** (click 2.2.6.11). However, the said parameters are **dedicated** when they are included within and instance of **UplinkConfig** (click 2.2.8.4). When the **parameters** of a **data type** are **either common or dedicated**, the **parameters** of **each** and **every** of its **descendants** are **also either common or dedicated**.

**Parameters that are common and dedicated**

Some **parameters** contain **both common** parameters **and dedicated** parameters.

For example, parameters that have the data type **BWP-Downlink** (click 2.2.8.5) contain the **common** parameter **bwp-Common** **and** the **dedicated** parameter **bwp-Dedicated**.

### 5.1.4 Criterion of belonging to Minimum System Information (MSI) or to other System Information

This criterion applies **only** to **parameters** belonging to **System Information** (click 5.1.2).

Depending on being **always received** (during **initial access**) by the **UE** or **not**, we can identify the following **two** classes of **System Information UE parameters**:

- **Minimum System Information parameters**

  This **class** contains the **system information parameters** that are **always received** by the **UE** for its **initial access**.

  It contains the **parameters** within the **PCI** (click 5.3), **PBCH SI** (click 5.3.2), **MIB** (click 5.3.3) and **SIB1** (click 5.3.4) **parameter groups**.

- **Other System Information parameters**

  This **class** contains the **system information parameters** that are **not always received** by the **UE** for its **initial access**.

  It contains the **parameters** within the **other SIBs** and **posSIBs** (click 5.3.5) parameter group.

## 5.2 UE parameter groups

The different **UE parameter groups sent** by the **network** for the configuration of the **UE**'s **physical layer** are:

- **PCI** (Physical Cell Identity) parameter group, click 5.3
- **PBCH System Information** parameter group, click 5.3.2.
- **MIB** (master information base) parameter group, click 5.3.3.
- **SIB1** (system information base 1) parameter group, click 5.3.4.
- **Other SIBs** and **posSIBs** parameter group, click 5.3.5.
- **Non-SI** parameter group, click 5.3.6.

  This group includes **both** the **Non-SI common parameters** and the **dedicated parameters**.

The different **UE parameter groups**, their **classification** and their **characteristics** are **summarized** in the table at 5. The **specific description** of **each UE parameter group** is

done in the **corresponding sections** indicated in the **bullet list above**. The **different classifications** that can be done over the **UE parameter groups** are presented in 5.1.

The **configuration type** and the **delivery mode** of the **UE parameter groups** are **not independent** and they are rather **somehow interrelated** (click 5.3.7).

In **LTE**, `MIB`, `SIB1` and `SIB2` are **mandatory** for the **UE** to access a cell and consequently, **all the System Information** (**SI**) is periodically **broadcast**. This approach of **periodically broadcasting all SI** over the entire cell area is **energy inefficient**, for example, when there are no **UEs** in the cell. Moreover, if this approach were used in **5G**, it would lead to excessive signaling overhead in cases where **several beams** are used to broadcast system information to be able to reach the entire coverage area of the cell.

For more detail on **system information parameters**, refer to clause 7.3.2 of TS 38.300 and to clause 5.2 of TS 38.331.

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_Mib_Sib.html

## 5.3 Stages in UE physical layer configuration

We can distinguish **three stages** in the **UE physical layer configuration**:

1. **Initial access stage** (click 8)

   This stage **starts** when the **UE** is **powered up** (or it **enters coverage** after a **long time** during which it was **out** of **coverage**).

   Along this stage, the **UE** gets by **broadcast** through **PBCH** and **PDSCH** the **SI parameters** (**common**) for its **primary cell**.

   These parameters allow the **UE** to perform the **RRC protocol setup**.

   The mandatory **SI parameters** are **the physical cell identity** (**PCI**), the **PBCH System Information**, `MIB` and `SIB1`, **all** of them for its **primary cell**. The **UE may** also possibly get some **other SIBs** and **posSIBs** for its **primary cell**. The **UE** gets **first PCI**, **PBCH SI** and `MIB`, **all** of them **within SSB** (click 6.1). Their parameters allow the **UE** to receive `SIB1` through **PDSCH** (click 6.13).

   This stage **finishes** when the **UE completes** the **RRC protocol setup**.

2. **Complete configuration stage**

   This stage **takes place** when the **UE completes** its **RRC protocol setup** (click 5.8.3.1), at the end of its **initial access procedure** (click 8).

   In this stage, the **UE** gets by **unicast** through **PDSCH** its **Non-SI parameters** (**both common** and **dedicated**). These parameters can be delivered in an `RRCSetup` (click 5.8.5.11) **message**.

   These parameters contain the **complete configuration** for its **primary cell**. If **carrier aggregation** is being used, they **also** contain the **complete configuration** for the **UE**'s **master cell group**. If **dual connectivity** is being used, they **also** contain the **complete configuration** for the **UE**'s **secondary cell group**. All these parameters allow the **UE** to **transmit and receive** in **all** the **bandwidth parts** of **all** its **configured cells** (from the **master cell group**, and if it exists, from the **secondary cell group**).

This stage **finishes** when the **UE** has **obtained** its **complete physical layer configuration**.

3. **Dynamic reconfiguration stage**

   This stage **starts after** the **UE** has **obtained its complete physical layer configuration**.

   Along this stage, the **UE**'s **physical layer** is **dynamically reconfigured** as required by its **changing conditions** (**mobility**, **channels**, **channel state**, **traffic**, …).

   This stage **finishes** when the **UE** is **powered off** or **changes cell** while in the **RRC_INACTIVE state** or in the **RRC_IDLE** (click 5.8.2) **state**. The **UE** will then have to run an **initial access procedure**.

**Dynamic reconfiguration stage**

Upon **change** of **relevant system information** for a **SCell**, the network **releases** and **adds** the corresponding **SCell**. For a **PSCell**, the required **SI** can **only** be **changed** with the procedures of `ReconfigurationWithSync` (click 3.9.8) (e.g., **handover**). For more detail, refer to clause 5.2.1 of TS 38.331.

Change of **system information** (other than for **ETWS**/**CMAS**, refer to clause 16.4 of TS 38.300) only occurs at **specific radio frames**, i.e. the concept of a **modification period** is used. **System information** may be transmitted several times with the same content within a modification period, as defined by its scheduling. The modification period is configured by **system information**.

When the network changes (some of the) **system information**, it first notifies the **UEs** about this change, i.e., this may be done throughout a **modification period**. In the **next modification period**, the network transmits the **updated system information**. Upon receiving a change notification, the **UE** acquires the **new system information** from the **start** of the **next modification** period. The **UE** applies the **previously acquired system information** until the **UE** acquires the **new system information**.

For more detail on **system information modification**, refer to clause 7.3.3 of TS 38.300.

### 5.3.1 PCI parameter group

The **PCI parameter group** serves the **UE** to know what is the **physical cell identity** (**PCI**, click 3.5.3.2) of **each** and **every** of its **serving cells**.

Notice that there are **several RRC parameters** that contain **values** of **Physical Cell Identities**, that are used for different **configuration** and **control** purposes. **Each** of **these parameters** has the **data type** `PhysCellId` (click 3.5.3.2). Do **not confuse** the **values** of **these parameters** with the **actual PCI** associated to **each** and **every cell**, although they are obviously very related.

For a **summary** of its **characteristics** compared to the ones of **other parameter groups**, refer to the table in 5.

The **main** characteristics of **PCI parameters** are:

- **Contained parameters**

  The **PCI parameter group only** contains the **Physical Cell Identity** (**PCI**). The **physical cell identity uniquely identifies** (for **physical layer purposes**) each **cell** among **all** its **neighbor cells**. It is a **quasi-static** identifier with **1,008 different**

**values**. For more detail on **PCI structure** and **semantics** and also on **other cell identifiers**, click 3.5.3.

- **Classification**

  **PCI** is **classified** (click 5.1) as: produced by the **physical layer**, belonging to **SI** (and within **SI**, it belongs to **Minimum System Information**) and being **common**.

- **Scope**

  **PCI** is **common** to **all** the **UEs** in **the cell** where it is **broadcast**. It refers **only** to the **cell** where it is **broadcast**.

- **Production timing**

  **PCI** is **semi-static** and is **only changed** in long periods (**months**, **years**).

- **Delivery timing**

  **PCI delivery timing** is the **same** as the one of **SSB** (click 6.1). The **SSB** is **broadcast** in **periodical bursts**, with one (or more) **SSB instances** in each **burst**. **One SSB burst** is **usually** sent every **20 ms**, but other values can be configured. For more detail on **SSB scheduling**, click 6.1.2.

  **PCI cannot** be requested **on-demand**.

- **Data representation**

  **PCI** is **represented** as a bitfield.

- **Delivery procedure(s)**

  **PCI** for a **cell** is received by the **UE** during **cell search** (click 8.2), within **cell selection** (click 8.1).

- **Delivered as**

  **PCI** is **delivered** as **sequence variations** of **PSS** and **SSS**. For more detail on the **delivery processing** of **PCI**, click 6.19.1 and 6.19.3.

- **Delivery mode**

  **PCI** is **broadcast** in **every network cell type** (except for **NSSB-Cells**, click 3.2.1).

  **PCI** may be **received** in any **UE cell type** (click 3.2.2).

- **Scheduling**

  **PCI** is sent within **SSB**, so its **scheduling** is the one of **SSB**. The **SSB** is **broadcast** in **periodical bursts**, with one (or more) **SSB instances** in each **burst**. **One SSB burst** is usually sent every **20 ms**, but other values can be configured (click 6.1.2.2.3).

- **Sent in physical channel or signal**

  **PCI** is **jointly** sent between **PSS** and **SSS**, within **SSB**.

### 5.3.2 PBCH system information parameter group

The **PBCH system information parameter group** serves the **UE** for **beam selection** and **frame synchronization**.

For a **summary** of its **characteristics** compared to the ones of **other parameter groups**, refer to the table in 5.

In this book, we will identify the **PBCH system information** parameters by writing their **names** in **brown Calibri font**.

The **main** characteristics of the **PBCH system information parameters** are:

- **Contained parameters**

  **PBCH SI** contains **very few** parameters (within **only 10** or **11 bits**). For more detail on the **parameters within PBCH SI**, click 6.4.1.

- **Classification**

  **PBCH SI** parameters are **classified** (click 5.1) as: produced by the **physical layer**, belonging to **SI** (and within **SI**, belonging to Minimum System Information) and being **common**.

- **Scope**

  **PBCH SI** parameters are **common** to **all** the **UEs** in the **cell** where it is **broadcast**. Its **parameters** refer **only** to the **cell** where it is **broadcast**.

- **Production timing**

  **PBCH SI** parameters are **produced** in **every SSB instance**. Its contents therefore **change** in **every transmitted SSB instance**, because **PBCH system information** contains beam selection information and other **time-sensitive** information. The **SSB** is **broadcast** in **periodical bursts**, with one (or more) **SSB instances** in each **burst**. **One SSB burst** is usually sent every **20 ms**, but other values can be configured (click 6.1.2.2.3).

- **Delivery timing**

  **PBCH SI** parameters' **delivery timing** is the **same** as the one of **SSB** (click 6.1). The **SSB** is **broadcast** in **periodical bursts**, with one (or more) **SSB instances** in each **burst**. **One SSB burst** is **usually** sent every **20 ms**, but other values can be configured. For more detail on **SSB scheduling**, click 6.1.2.

  **PBCH SI** parameters **cannot** be requested **on-demand**.

- **Data representation**

  **PBCH SI** parameters are represented as **bitfields**.

- **Delivery procedure(s)**

  **PBCH SI** parameters are received by the **UE** during **cell search** (click 8.2), within **cell selection** (click 8.1).

- **Delivered as**

  **PBCH SI** parameters are **jointly** delivered as **PBCH payload REs** and as **sequence variations** of the **PBCH DM-RS**. For more detail on the **delivery processing** of **PBCH SI** parameters, click 6.5.

- **Delivery mode**

  **PBCH SI** parameters are **broadcast** in **every network cell type** (except for **NSSB-Cells**, click 3.2.1).

  **PBCH SI** parameters may be **received** in any **UE cell type** (click 3.2.2).

- **Scheduling**

  **PBCH SI** is sent within **SSB**, so its **scheduling** is the one of **SSB**. The **SSB** is **broadcast** in **periodical bursts**, with one (or more) **SSB instances** in each **burst**. **One SSB burst** is usually sent every **20 ms**, but other values can be configured (click 6.1.2.2.3).

- **Sent in physical channel or signal**

  **PBCH SI** parameters are **sent** through **PBCH**, within **SSB**.

### 5.3.3 MIB parameter group

The **MIB** (master information block) **parameter group** serves the **UE** to determine **cell barred status** and provide it with **essential physical layer** configuration. These parameters are **first received** by the **UE** during its **initial access** (click 8) in its **primary cell** (click 3.2.2). These parameters allow the **UE** to **receive Downlink**, in order to get **SIB1** (click 5.3.4).

For a **summary** of its **characteristics** compared to the ones of **other parameter groups**, refer to the table in 5.

The **main** characteristics of the **MIB parameters** are:

- **Contained parameters**

  **MIB** contains **very few** parameters (within only **23 bits**). For more detail on the **parameters within MIB**, click 8.12.3.

- **Classification**

  **MIB** parameters are **classified** (click 5.1) as: produced by the **higher layers**, belonging to **SI** (and within **SI**, belonging to Minimum System Information) and being **common**.

- **Scope**

  **MIB** parameters are **common** to **all** the **UEs** in the **cell** where it is **broadcast**. Its **parameters** refer **only** to the **cell** where it is **broadcast**.

- **Production timing**

  **MIB** parameters are produced (sent over **BCH**) **periodically**. The production **period** is **80 ms**, unless the **SSB burst period** is larger, in which case it is the **same** as the **SSB burst period** (click 6.1.2.2.3).

  For more detail on **system information modification**, click 5.5.3.5.

- **Delivery timing**

  **MIB** parameters' **delivery timing** is the **same** as the one of **SSB** (click 6.1). The **SSB** is **broadcast** in **periodical bursts**, with one (or more) **SSB instances** in each **burst**. **One SSB burst** is **usually** sent every **20 ms**, but other values can be configured. For more detail on **SSB scheduling**, click 6.1.2.

  **MIB** parameters **cannot** be requested **on-demand**.

  Since **MIB transmission timing** is **more often** than its **production timing**, the **same MIB value** is **repeated** in **several** consecutive **SSBs**. It **cannot** be requested **on-demand**.

For more detail on **system information modification**, click 5.5.3.5.

- **Data representation**

  `MIB` parameters are **specified** in **ASN.1** as an **RRC** `MIB` **message** (i.e., the `MIB` **data type**, click 8.12.3), within an **RRC** `BCCH-BCH-Message` **class** (click 5.8.5.1). This **RRC message class** has a **specific bitwise representation** (click 6.4.2).

- **Delivery procedure(s)**

  `MIB` parameters are received by the **UE** during **cell search** (click 8.2), within **cell selection** (click 8.1).

- **Delivered as**

  `MIB` parameters are delivered as **PBCH payload REs**. For more detail on the **transmission processing** of `MIB`, click 6.5.

- **Delivery mode**

  `MIB` parameters are **broadcast** in **every network cell type** (except for **NSSB-Cells**, click 3.2.1). It is **broadcast** through the **logical broadcast control channel** (**BCCH**), through the **transport broadcast channel** (**BCH**), and through the **physical PBCH**.

  `MIB` may be **received** in any **UE cell type** (click 3.2.2).

- **Scheduling**

  `MIB` is sent within **SSB**, so its **scheduling** is the one of **SSB**. The **SSB** is **broadcast** in **periodical bursts**, with one (or more) **SSB instances** in each **burst**. **One SSB burst** is usually sent every **20 ms**, but other values can be configured (click 6.1.2.2.3).

- **Sent in physical channel or signal**

  `MIB` parameters are **sent** through **PBCH**, within **SSB**.

For more detail on `MIB` **characteristics**, refer to clause 5.2.1 of TS 38.331.

### 5.3.4 `SIB1` parameter group

The `SIB1` (**system information block 1**) serves the **UE** to perform its low-level physical configuration (e.g., **cell selection**, **power control**, …), to configure its **Downlink** and **Uplink initial bandwidth parts** and to **run** the **random access procedure** (click 9). `SIB1` is sometimes called "**remaining minimum system information** (**RMSI**)".

For a **summary** of its **characteristics** compared to the ones of **other parameter groups**, refer to the table in 5.

The **main** characteristics of the `SIB1` **parameters** are:

- **Contained parameters**

  `SIB1` contains over a hundred **common** parameter **instances** for the **UE**'s **primary cell** during **initial access**. For more detail on the **parameters within** `SIB1`, click 5.7.4.

- **Classification**

  `SIB1` parameters are **classified** (click 5.1) as: produced by the **higher layers**, belonging to **SI** (and within **SI**, belonging to **Minimum System Information**) and being **common**.

- **Scope**

  SIB1 parameters are **common** to **all** the **UEs** in the **cell** where it is **sent**. Its **parameters** refer **only** to the **cell** where it is **sent**.

- **Production timing**

  SIB1 is produced (sent over **DL-SCH**) **periodically**, every **160 ms**.

  For more detail on **system information modification**, click 5.5.3.5.

- **Delivery timing**

  SIB1 **delivery timing** depends on its **delivery mode**:

  - When SIB1 is **delivered** by **broadcast**:

    The delivery is **periodical**. The **default** transmission ("repetition") period is **20 ms**. The **actual repetition** period is up to the network, where we can identify the following **two** cases:

    - For **SSB+CORESET** multiplexing **Pattern 1**

      The SIB1 **repetition period** is **20 ms**.

    - For **SSB+CORESET** multiplexing **Patterns 2** or **3**

      The SIB1 **repetition period** is the same as the **SSB burst period** (clause 13 of TS 38.213).

    Since SIB1 **broadcast timing** is **more often** than its **production timing**, the **same** SIB1 **value** in **repeated** in **several** consecutive transmissions.

  - When SIB1 is **delivered** in **unicast**:

    The SIB1 **transmission timing** is **controlled** by the **network**, based on different **procedures** (e.g., **handover**).

  For more detail on **system information modification**, click 5.5.3.5.

- **Data representation**

  SIB1 parameters are **specified** in **ASN.1** as an **RRC** SIB1 **message** (i.e., the SIB1 **data type**, click 5.7.4). The **physical layer** imposes to a SIB1 **message** a **maximum size** of **2,976 bits** (refer to clause 5.2.1 of TS 38.331).

- **Delivery procedure(s)**

  The SIB1 **message** is enclosed within an **RRC message class** that depends on its **delivery mode**, as follows:

  - When SIB1 is **delivered** by **broadcast**:

    SIB1 parameters are received by the **UE** during **cell selection** (click 8.1).

    The SIB1 message is **enclosed** within an **RRC** BCCH-DL-SCH-Message **class** (click 5.8.5.2) that is **broadcast** through **PDSCH**.

  - When SIB1 is **delivered** in **unicast**:

    SIB1 parameters are received by the **UE** during **RRC reconfiguration**.

The SIB1 message is **enclosed** in the optional parameter dedicatedSIB1-Delivery (as an OCTET STRING) of an RRCReconfiguration **message**. This message is itself enclosed within an **RRC** DL-DCCH-Message **class** (click 5.8.5.4). This is **unicast** through **PDSCH**.

Notice that although SIB1 is provided to the **UE** **before** the **RRC connection establishment**, it is **represented** as an **RRC** SIB1 **message** (click 5.7.4).

The **two** corresponding **ancestor trees** (click 5.9.8) of SIB1 **up to** the **top** are:

```
                                BCCH-DL-SCH-Message
message                           BCCH-DL-SCH-MessageType
  c1                                CHOICE
    systemInformationBlockType1   SIB1 (click 5.7.4)

                                DL-DCCH-Message (click 5.8.5.4)
message                           DL-DCCH-MessageType (click 5.8.5.4)
  c1                                CHOICE
    rrcReconfiguration              RRCReconfiguration (0,1) (click 5.8.5.4)
      rrcReconfiguration              RRCReconfiguration-IEs (click 5.8.5.4)
        nonCriticalExension             RRCReconfiguration-v1530-IEs (0,1)
          dedicatedSIB1-Delivery          OCTET STRING
                                          (CONTAINING SIB1) (0,1)
```

Notice that SIB1 is also included as an OCTET STRING in a **Sidelink** message, but this is **not** covered in this book.

- **Delivered as**

  SIB1 parameters are delivered as **PDSCH payload REs**. For more detail on the **delivery processing** of SIB1, click 4.12.

- **Delivery mode**

  SIB1 **delivery mode** by the network depends on the **network cell type**:

  - **SIB1-Cells** (click 3.2.1)

    SIB1 is **broadcast**.

    SIB1 **may also** be **unicast**, via **dedicated signaling**, to **UEs** in **RRC_CONNECTED** (click 5.8.2).

  - **NSIB1-Cells** (click 3.2.1)

    SIB1 is **only unicast**, via **dedicated signaling** to **UEs** in **RRC_CONNECTED**.

  The possibility of SIB1 **unicast** delivery allows to **avoid** the **periodical broadcast** of SIB1 in **NSIB1-Cells**, thus making **SSB** the **only always-on signal** in a **cell** (except for **NSSB-Cells**, that do **not** even send **SSB**.

  SIB1 is **received** by the **UE** in different ways, depending on the **UE cell type**:

  - **PCell** (click 3.2.2)

    SIB1 **is received** by the **UE** by **broadcast** during **cell selection**.

  - **PSCell** and **SCells**

    SIB1 **is received** by **unicast dedicated signaling** once the **UE** is **RRC_CONNECTED**. Nevertheless, the **UE** shall acquire the MIB of the **PSCell** to get **SFN timing** of the **secondary cell group** (which may be different from the **master cell group**).

Notice that a given cell can be a **network**'s **SIB1-Cell** and a **UE**'s **SCell**, because the **UE**'s **primary cell** is a **different one**. Therefore, the **network broadcasts** `SIB1` in this cell, but is **also unicasts** `SIB1` to this **UE**, because this cell is **not** the **primary cell** of this **UE**.

Recall from 5.3.7 that even if `SIB1` is on some cases **delivered** via **unicast dedicated signaling**, it does not change the **fact** that `SIB1` **only** contains **common parameters** (`SIB1` **does not contain any dedicated parameter**).

For more detail on `SIB1` **delivery**, refer to clause 5.2.1 of TS 38.331.

- **Scheduling**

  The **scheduling** for the delivery of `SIB1` is done with a **DCI Format 1_0 message** (click 6.9.5.2) with its **CRC** masked with **SI-RNTI**.

- **Sent in physical channel or signal**

  `SIB1` is **sent** through **PDSCH**.

For more detail on `SIB1`, refer to clause 5.2.1 of TS 38.331.

### 5.3.5 Other SIBs and posSIBs parameter group

The **other SIBs parameters** serve the **UE** to perform **cell re-selection**, **ETWS**, **CMAS**, **GPS timing** and other specific procedures and functions.

The **posSIBs parameters** serve the **UE** to determine its **geographical position**.

The "**other SIBs** and **posSIBs**" are sometimes called "**other System Information**".

For a **summary** of its **characteristics** compared to the ones of **other parameter groups**, refer to the table in 5.

The **main** characteristics of the **other SIBs** and **posSIBs parameters** are:

- **Contained parameters**

  The **other SIBs** and **posSIBs** parameter group contains the parameters within **all** the **other SIBs** (i.e., all **system information blocks other** than `SIB1`), and within **all** the **positioning information blocks** (**posSIBs**). For more detail on **parameters within other SIBs**, click 5.7.6. For more detail on the **parameters within posSIBs**, click 5.7.7 .

- **Classification**

  The **other SIBs** and **posSIBs parameters** are **classified** (click 5.1) as: produced by the **higher layers**, belonging to **SI** (and within **SI**, they are the **other System Information**) and being **common**.

- **Scope**

  **Other SIBs** and **posSIBs** parameters are **common** to **all** the **UEs** in **either** a **cell** or a **system information area** (**SI area**).

  Each **other SIB** or **posSIB** can be configured to be either **cell-specific** or **area-specific**, with `areaScope` from `SIB-TypeInfo` (click 5.5.3.3), within `SIB1` (click 5.3.4). The **cell-specific SIB** is applicable **only** within a **cell** that **provides the SIB**. The **area-specific SIB** is applicable within an **SI area** (click 3.5.4).

  For more detail on **system information areas**, refer to clause 5.2.1 of TS 38.331.

- **Production timing**

  The parameters **within each individual other SIB** or **posSIB** is produced with a **specific timing**, according to its **contents**.

  For more detail on **system information modification**, click 5.5.3.5.

- **Delivery timing**

  The parameters **within each individual other SIB** or **posSIB** may be **either** delivered **periodically or** requested **on-demand**. Parameter `si-SchedulingInfo` from `SIB1` (click 5.3.4) indicates if **each** of them is delivered **periodically** or requested **on-demand**.

  - When **periodical timing**:

    They are **periodically broadcast**, with a **period indicated** in `si-SchedulingInfo` from `SIB1` (click 5.3.4).

    **All** the **other SIBs** or **posSIBs** that are **mapped** (click 5.5.1) to the **same** `SystemInformation` (click 5.7.5) **message** are transmitted with the **same periodicity**.

  - When **on-demand timing** (which is unusual):

    It depends on the **RRC** state, as follows:

    - For **UEs** in **RRC_IDLE** or in **RRC_INACTIVE** (click 5.8.2) while small data transmission (**SDT**, click 10.9) is **not ongoing**:

      - **UE** request for other **SIBs** and/or **posSIBs** is performed within a **random access procedure** (click 9). For more detail on a **System Information request** with a **random access procedure**, refer to the row "**On-demand system information**" from the table in 9.4.2.

        Depending on whether the requested **other SIBs** or **posSIBs are associated**, or **not**, to a **subset** of the **PRACH resources**, we have **2 cases**:

        - They **are associated**, then:

          **Msg1** (click 9.3.2) is used to **trigger** the **request process**. The **minimum granularity** of the **request** is **one SI message** (i.e., a set of **SIBs**). One **RA procedure** (click 9) can be used to request **multiple SI messages**.

        - They are **not associated**, then:

          **Msg3** (click 9.3.4) is used to **trigger** the **request process**.

    - For **UEs** in **RRC_CONNECTED** (click 5.8.2):

      A **UE request** for the parameter within **other SIBs** and/or **posSIBs** may be sent to the network, if so configured by the network (click 5.5.2), in a **dedicated** manner (i.e., via **UL-DCCH**, click 5.8.5.9). The **granularity** of the request is **one SIB**. The **NR node** may respond with an `RRCReconfiguration` (**DL-DCCH**, click 5.8.5.4) **message including** the requested **SIB(s)**.

  For more detail on **system information modification**, click 5.5.3.5.

For more detail on **other SIBs** and **posSIBs characteristics**, refer to clause 5.2.1 of TS 38.331.

- **Data representation**

  **Other SIBs** and **posSIBs** parameters are **specified** in **ASN.1** as an **RRC** `SystemInformation` **message** (i.e., the `SystemInformation` **data type**, click 5.7.5).

- **Delivery procedure(s)**

  **Other SIBs** and **posSIBs** parameters received by the **UE** after the reception of `SIB1`.

  The `SystemInformation` **message** is enclosed within an **RRC message class** that depends on its **delivery mode** (see 2 bullets further below):

  - When **Other SIBs** and **posSIBs** are **delivered** by **broadcast**:

    `SystemInformation` is **enclosed** within an **RRC message** `BCCH-DL-SCH-Message` **class** (click 5.8.5.2) that is **broadcast** through **PDSCH**.

  - When **Other SIBs** and **posSIBs** are **delivered** in **unicast**:

    `SystemInformation` is **enclosed** in the optional parameter `dedicatedSystemInformationDelivery` (as an `OCTET STRING`) of an `RRCReconfiguration` **message** (click 5.8.5.4). This message is itself enclosed within an **RRC** `DL-DCCH-Message` **class** (click 5.8.5.4). This is **unicast** through **PDSCH**.

  The **2** corresponding **ancestor trees** (click 5.9.8) of `SystemInformation` **up to** the **top** are:

```
                          BCCH-DL-SCH-Message (click 5.8.5.2)
message                     BCCH-DL-SCH-MessageType (click 5.8.5.2)
  c1                          CHOICE
    systemInformation           SystemInformation (click 5.7.5)

                          DL-DCCH-Message (click 5.8.5.4)
message                     DL-DCCH-MessageType (click 5.8.5.4)
  c1                          CHOICE
    rrcReconfiguration          RRCReconfiguration (0,1) (click 5.8.5.4)
      rrcReconfiguration          RRCReconfiguration-IEs (click 5.8.5.4)
        nonCriticalExension         RRCReconfiguration-v1530-IEs (0,1)
          dedicatedSystemInformationDelivery
                                        OCTET STRING (CONTAINING
                                        SystemInformation) (0,1) (click 5.7.5)
```

  Notice that `SystemInformation` is **also** included as an `OCTET STRING` in both:

  - Parameter `sCellSIB20-r17` from `SCellConfig` (click 3.9.4). However, this is a **particular case** to deliver **only SIB20** for a **secondary cell**, and it is **not** a general way to convey **other SIBs** and **posSIBs**.
  - A **Sidelink** message. However, recall that **Sidelink** is **not** covered in this book.

  A **single** **RRC** `SystemInformation` (click 5.7.5) message may contain **either several other SIBs or several posSIBs** (but **not** a combination of **both** types). For more detail on the **mapping** of **other SIBs** and **posSIBs** to an **RRC** `SystemInformation` message, click 5.5.1.

- **Delivered as**

  **Other SIBs** and **posSIBs** parameters are delivered as **PDSCH payload REs**. For more detail on the **delivery processing** of **Other SIBs** and **posSIBs**, click 4.12.

- **Delivery mode**

  When their **delivery timing** is **periodical** (see below), the **other SIBs** and **posSIBs** parameters are **always broadcast**.

  When their **delivery timing** is **on-demand** (see below), it is a decision by the **network** to deliver them **either** by **broadcast or** by **unicast** via **dedicated signaling** (to **UEs** in **RRC_CONNECTED**, click 5.8.2). For example, the **network** would provide the **system information** via **unicast** if the **UE** has an **active bandwidth part** with **no common search space** configured to monitor **system information**, on **paging**, or upon **request** from the **UE**. For **PSCell** and **SCells**, the **network** provides the **required SI** by dedicated signaling. Nevertheless, the **UE** shall acquire `MIB` of the **PSCell** to get **SFN timing** of the **secondary cell group** (which may be different from the **master cell group**). For more detail on this, refer to clause 5.2.1 of TS 38.331.

  Recall from 5.3.7 that even if **other SIBs** and **posSIBs** are on some cases **delivered** via **unicast dedicated signaling**, it does not change the **fact** that **other SIBs** and **posSIBs** **only** contain **common parameters** (**other SIBs** and **posSIBs do not contain any dedicated parameter**).

- **Scheduling**

  The **scheduling** for the **delivery** of **other SIBs** and **posSIBs** is done with a **DCI Format 1_0 message** with its **CRC** masked by **SI-RNTI** (click 6.9.5.2).

- **Sent in physical channel or signal**

  **Other SIBs** and **posSIBs** parameters are sent through **PDSCH**.

### 5.3.6 Non-SI parameter group

The **Non-SI parameter group** serves the **UE** to perform an (almost) **complete** configuration of its **physical layer**, in respect to **all** the **cells** (click 3.1.1) to which the **UE** may be connected, within **each** of the (up to **2**) **cell groups** (click 3.2.2) of the **UE**.

For a **summary** of its **characteristics** compared to the ones of **other parameter groups**, refer to the table in 5.

The **main** characteristics of the **non-SI parameters** are:

- **Contained parameters**

  The **Non-SI parameter group contains all** the **RRC UE parameters not** belonging to **System Information parameters**. Even though the **Non-SI parameter group** is **disjoint** from the **SI parameters**, it contains a **superset** of the **configuration information** contained by the **SI parameters**. For example, the information within **data type** `DownlinkConfigCommon` (click 2.2.6.8), that belongs to **Non-SI parameters**, is a superset of the information within **data type** `DownlinkConfigCommonSIB` (click 2.2.6.4) that belongs to **SI parameters** (specifically to `SIB1` **parameters**).

The **non-SI parameter group** is divided in two subgroups: the **non-SI common parameters** and the **dedicated parameters** (see table from 5). The **main difference** between both subgroups is that the **former** are **all common** parameters while the **latter** are **all dedicated** parameters. Be aware that the parameters belonging to each subgroup are **not** always **clearly separated** in the **ASN.1** specification. For example, the **non-SI common** parameter `sCellConfigCommon` and the **dedicated** parameter `sCellConfigDedicated` are **both** from the **same data type** `SCellConfig` (click 3.9.4)

For more detail on the **structure** of parameters **within** the **Non-SI parameter group**, click 5.7.8.

- **Classification**

  **Non-SI parameters** are **classified** (click 5.1) as: produced by the **higher layers**, and belonging to **Non-SI**.

- **Scope**

  The **non-SI parameters** contain **several sets** of **parameter instances**.

  The **cell** to which **each set** of **parameter instances** applies is **explicitly** indicated in an **instance** of `physCellId` from `ServingCellConfigCommon` (click 3.9.6) **contained** in the **set**.

  **Each set** of **non-SI common parameters** is **common** to **all** the **UE** in the **cell** that is **explicitly indicated** within the **parameter set**.

  **Each set** of **dedicated parameters** is **specific** to the **UE** and to the **cell** that is **explicitly indicated** within the **parameter set**.

- **Production timing**

  **Non-SI parameters** are **produced according** to **network**'s **consideration**.

- **Delivery timing**

  **Non-SI parameters** are **delivered according** to **network**'s **consideration.**

  The are typically **delivered** during **initial access** as part of the first **RRC connection establishment**. They are later delivered as the **channel** and **UE** conditions **change**.

- **Data representation**

  **Most** of the **non-SI parameters** are **represented** as an **instance** of `CellGroupConfig` (click 3.7.3), of `MAC-CellGroupConfig` (click 3.7.4) and of `PhysicalCellGroupConfig` (click 3.7.5), within the **cell group** parameters (click 3.7).

- **Delivery procedure**(**s**)

  **Non-SI parameters** are received by the **UE during RRC connection establishment** (click 5.8.3.1), **during RRC connection resume** and **during RRC reconfiguration**.

  Almost **all Non-SI parameters** belong to the **cell group parameters** click 3.7.

  For the **ancestor trees** (click 5.9.8) of the **cell group parameters up to** the above **3 procedures**, click 3.7.1.

**Non-SI parameters** are also exchanged in **RRC inter-node messages** (click 5.8.6). For the **ancestor tree** of the **cell group parameters up to** the `HandoverCommand` **RRC inter-node message**, click 3.7.2.

Be aware that the **ASN.1** statement "`OCTET STRING (CONTAINING RRCReconfiguration)`" may be found in **7 RRC data types**.

- **Delivered as**

  **Other SIBs** and **posSIBs** parameters are delivered as **PDSCH payload REs**. For more detail on the **delivery processing** of **Other SIBs** and **posSIBs**, click 4.12.

- **Delivery mode**

  **Non-SI parameters** are always delivered in **unicast**.

  Do not confuse the **delivery mode** of **non-SI parameters** with the **nature** of **non-SI parameters** themselves (click 5.3.7): even if **non-SI parameters** are **only delivered** via **unicast dedicated signaling**, it does not change the **fact** that **non-SI parameters** contain **common parameters** (in **addition** to **dedicated parameters**).

- **Scheduling**

  The delivery of **Non-SI parameters** is **scheduled** with a **DCI Format 1_x message** (click 6.9).

- **Sent in physical channel or signal**

  **Non-SI parameters** are **sent** through **PDSCH** in every **cell type**.

### 5.3.7 Relationship between configuration type and delivery mode

It is worth pointing out the following **relationships** between the **configuration type** (**common** or **dedicated**, click 5.1.3) and the **delivery mode** (**broadcast** and/or **unicast**, click 5.2) of the different parameter groups. These **relationships** are reflected in the **table** from 5.

**Common** parameters in **PCI**, **PBCH SI** and `MIB` are **always broadcast**. Notice that **PCI**, **PBCH SI** and `MIB` **only** contain **common** parameters.

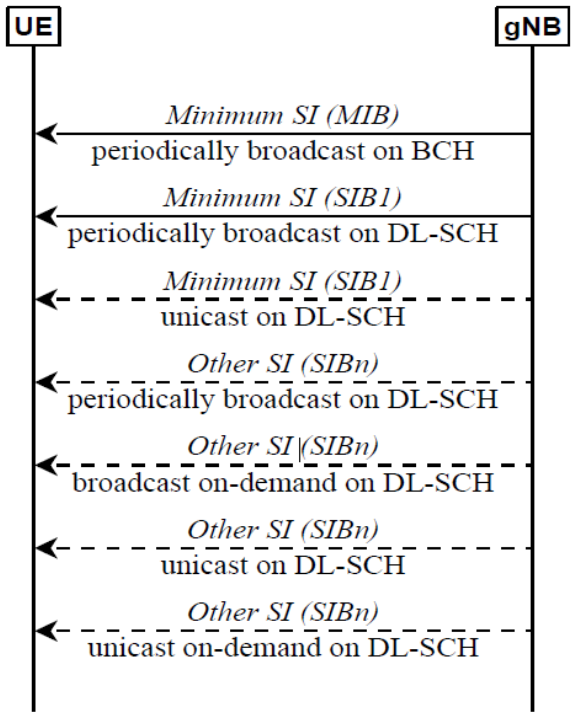
**Common** parameters in **system information bases** and **non-SI common parameters** may be **both broadcast** and **unicast**. Notice that **system information bases only** contain **common** parameters.

**Dedicated** parameters are **always unicast**.

**Broadcast delivery** is **only** used for **common parameters**. In other words, if a **parameter** is **broadcast**, it **must** be a **common** parameter.

**Unicast delivery** is **always** used for **dedicated parameters** and **may be** used **both** for **system information base parameters** (which are **all common**) and for **Non-SI common** parameters. In other words, if a **parameter** is **unicast**, it may be a **dedicated** parameter, **system information base parameter** (which are **all common**) or a **Non-SI common** parameter.

Regarding the **parameter groups** belonging to **System information**, the following figure summarizes their **delivery alternatives**:




## 5.4 UE parameters for some physical layer elements

The **parameters** that configure some relevant **physical layer elements** are described in the following sections:

- **Carrier** parameters, click 2.2.5.8.
- **Bandwidth part** parameters, click 2.2.8.
- **Subcarrier spacing** parameters, click 2.2.10.5.
- **TDD** parameters, click 2.3.5.
- **Network**, **node** and **cell identity** parameters, click 3.6.
- **Cell group** parameters, click 3.7.
- **Cell** parameters, click 3.9
- **Other SIBs** and **posSIBs mapping** and **scheduling** parameters, click 5.5.
- **PBCH** parameters, click 6.6.
- **PDCCH** parameters, click 6.12.
- **PDSCH** parameters, click 6.18.
- **PRACH** parameters, click 7.4.
- **PUCCH** parameters, click 7.8.2.
- **PUSCH** parameters, click 7.15.
- **Initial access** parameters, click 8.11.
- **Random access** parameters, click 9.5.

The **contents** or the **structure** of **parameters within** some relevant **parameter groups** is presented in the following sections:

- Parameters **within PBCH System Information**, click 6.4.1.
- Parameters **within** `MIB`, click 8.12.3.
- Parameters **within** `SIB1`, click 5.7.4.
- Parameters **within other SIBs** and **posSIBs**: `SystemInformation`, click 5.7.5.
- **Structure** of parameters **within** the **Non-SI parameter group**, click 5.7.8.
- **RRC message classes** and **RRC NR messages**, click 5.8.5.
- **RRC inter-node messages**, click 5.8.6.

## 5.5 Other SIBs and posSIBs mapping and scheduling parameters

The **mapping** of **each** of the **other SIBs and posSIBs** (click 5.3.5) to a **specific RRC** `SystemInformation` (click 5.7.5) **message** can be **configured** in different ways. An **overview** is presented in 5.5.1, and the **most relevant related data types** within `SIB1` are described in 5.5.3.

The **scheduling** of **each** such **specific** `SystemInformation` **message can also** be **configured** in different ways. The scheduling is **usually** by **periodical broadcast** (with **configurable delivery timing**), but it can also by **on-demand**, with **unicast delivery**. An **overview** is presented in 5.5.2, and the **most relevant related data types** within `SIB1` are described in 5.5.3.

The **UE**'s **other SIBs and posSIBs mapping** and **scheduling** is configured with:

- **Common parameters** that are **broadcast** (or **unicast**) within `SIB1`, as described in section 5.5.3.

**Other SIBs and posSIBs mapping** and **scheduling** does **not** have any parameter within `MIB` (click 5.3.3), **nor** within the **Non-SI parameter group** (click 5.3.6).

For an overall explanation on **UE configuration parameters** click 5. For a brief overview of **ASN.1**, click 5.9.

### 5.5.1 Overview of other SIBs and posSIBs mapping parameters

The mapping of **other SIBs** to **RRC** `SystemInformation` (click 5.7.5) messages is configured with `schedulingInfoList` from `SI-SchedulingInfo` (click 5.5.3.1) and with `schedulingInfoList2-r17` from `SI-SchedulingInfo-v1700`. The **first data type** is a **son** of `SIB1`. The **second data type** is a **descendant** of `SIB1`, through a **series** of parameters and data types.

The mapping of **posSIBs** to **RRC** `SystemInformation` (click 5.7.5) messages is configured with `posSchedulingInfoList-r16` from `PosSI-SchedulingInfo-r16`, and with `schedulingInfoList2-r17` from `SI-SchedulingInfo-v1700`. The **first data type** is a **son** of `SIB1`. The **second parameter** and **data type** are the **same** as the one used for the **mapping** of **other SIBs** (see previous paragraph).

**SIBs** and **posSIBs must** be mapped to **separate** **RRC** `SystemInformation` messages, **even** if configured using **the same** `schedulingInfoList2-r17`.

Only **SIBs** or **posSIBs** having the **same broadcast periodicity** (click 5.5.2) can be mapped to the **same RRC** `SystemInformation` message. **Each SIB** is contained in **only one RRC** `SystemInformation` message (i.e., it **cannot** be **split** among several messages). In the case of **posSIB**, a **posSIB** carrying **GNSS Generic Assistance Data** for **different GNSS/SBAS** is contained in **different RRC** `SystemInformation` messages. Each **SIB** and **posSIB**, including a **posSIB** carrying **GNSS Generic Assistance Data** for **one GNSS/SBAS**, is contained **at most once** in that **RRC** `SystemInformation` message.

Note that the **physical layer** imposes a **limit** to the **maximum size** a **SIB** or **posSIB** can have. The **maximum size** of a **RRC** `SIB1` message or of a **RRC** `SystemInformation` message is **2,976 bits** (refer to clause 5.2.1 of TS 38.331).

For more detail on **configuring** the **mapping other SIBs** and **posSIBs** to **RRC messages**, refer to clause 5.2.1 of TS 38.331.

## 5.5.2 Overview of other SIBs and posSIBs scheduling parameters

**Each NR RRC** `SystemInformation` (click 5.7.5) **message** corresponds to a given **format** that has been previously configured. **Each** of these **formats** indicates what **other SIBs or** what **posSIBs** does each `SystemInformation` **message** contain. to the formats that have been configured (click 5.5.1). The configuration also indicates if **each** `SystemInformation` **message** is **periodically broadcast** or **unicast on-demand**.

For the case of **periodical broadcast**, the **period** corresponding to **each format** is configured with `si-Periodicity` from `SchedulingInfo` (click 5.5.3.2).

The `SystemInformation` **messages** cannot be transmitted at any moment. Every `SystemInformation` **message** belonging to a given **format must** be transmitted within an **SI-window** corresponding to that **format**. An **SI window** is a **fixed duration time interval** that **occurs periodically**. The **SI window duration** is the same for **all** the `SystemInformation` **formats**, and it is configured with `si-WindowLength` from `SI-SchedulingInfo` (click 5.5.3.1).

**Each** `SystemInformation` (click 5.7.5) **message format** is associated with an **SI-window**. The **SI-windows** of different `SystemInformation` **message formats do not overlap**. Within a given **SI-window** it is **only** possible to transmit `SystemInformation` **messages** belonging to the **format** corresponding to that **SI-window**. A `SystemInformation` **message** may be transmitted **several times** within the **SI-window**.

Notice that the network always configures `si-WindowLength` to be shorter than or equal to `si-Periodicity` from any of the `SchedulingInfo` (click 5.5.3.2) within `schedulingInfoList` from `SI-SchedulingInfo` (click 5.5.3.1). Notice that `si-WindowLength` is expressed in **slots**, while the **different** `si-Periodicity` are expressed in **frames**.

For more detail on **configuring** the **delivery timing mapping other SIBs** and **posSIBs** to **RRC messages**, refer to clause 5.2.1 of TS 38.331.

### 5.5.3 Other SIBs and posSIBs mapping and scheduling parameters within `SIB1`

For **other SIBs** and **posSIBs scheduling parameters** within `SIB1` (click 5.3.4), the **most relevant related data types** are:

- `SI-SchedulingInfo` (click 5.5.3.1): common parameters
- `SchedulingInfo` (click 5.5.3.2): common parameters
- `SI-RequestConfig` (click 5.5.3.4): common parameters
- `PosSI-SchedulingInfo-r16`: common parameters
- `SI-SchedulingInfo-v1700`: common parameters
- `SI-SchedulingInfo2-r17`: common parameters
- `SI-SchedulingInfo-v1740`: common parameters
- `SIB-Mapping` (click 5.5.3.2): common parameters
- `SIB-TypeInfo` (click 5.5.3.3): common parameters
- `SIB-Mapping-v1700`: common parameters
- `SIB-TypeInfo-v1700`: common parameters
- `PosSIB-MappingInfo-r16`: common parameters
- `PosSIB-Type-r16`: common parameters

The following **data type**, **son** of `SI-RequestConfig` (click 5.5.3.4), is also relevant:

- `SI-RequestResources` (click 5.5.3.5): common parameters

The **ancestor tree** (click 5.9.8) of these **most relevant related data types** up to `SIB1` is:

```
                                    SIB1 (click 5.7.4)
si-SchedulingInfo                     SI-SchedulingInfo (0,1) (click 5.5.3.1)
  schedulingInfoList                    SchedulingInfo (1..maxSI-Message) (5.5.3.2)
    sib-MappingInfo                       SIB-Mapping (5.5.3.2)
                                            SIB-TypeInfo (1..maxSIB) (5.5.3.2)
  si-RequestConfig                      SI-RequestConfig (0,1) (click 5.5.3.4)
  si-RequestConfigSUL                   SI-RequestConfig (0,1) (click 5.5.3.4)
nonCriticalExtension                  SIB1-v1610-IEs (0,1)
  posSI-SchedulingInfo-r16              PosSI-SchedulingInfo-r16 (0,1)
    posSchedulingInfoList-r16             PosSchedulingInfo-r16 (1..maxSI-Message)
      posSIB-MappingInfo-r16                PosSIB-MappingInfo-r16
                                              PosSIB-Type-r16 (1..maxSIB)
    posSI-RequestConfig-r16               SI-RequestConfig (0,1) (click 5.5.3.4)
    posSI-RequestConfigSUL-r16            SI-RequestConfig (0,1) (click 5.5.3.4)
    posSI-RequestConfigRedCap-r17         SI-RequestConfig (0,1) (click 5.5.3.4)
  nonCriticalExtension                  SIB1-v1630-IEs (0,1)
    nonCriticalExtension                  SIB1-v1700-IEs (0,1)
      si-SchedulingInfo-v1700               SI-SchedulingInfo-v1700 (0,1)
        schedulingInfoList2-r17               SchedulingInfo2-r17
                                              (1..maxSI-Message))
          sib-MappingInfo-r17                   SIB-Mapping-v1700
                                                  SIB-TypeInfo-v1700 (1..maxSIB)
        dummy                                 SI-RequestConfig (0,1) (click 5.5.3.4)
      nonCriticalExtension                  SIB1-v1740-IEs (0,1)
        si-SchedulingInfo-v1740               SI-SchedulingInfo-v1740 (0,1)
          si-RequestConfigRedCap-r17            SI-RequestConfig (0,1) (click 5.5.3.4)
```

Some **relevant remarks** on this **ancestor tree** are:

- For the **ancestor trees** of `SIB1` **up to** the **top**, click 5.3.4.
- The constant `maxSI-Message` is `32`, which is the **maximum quantity** of `SystemInformation` (click 5.7.5) **message formats** that can be configured to **each UE**.
- The constant `maxSIB` is `32`, which is the **maximum quantity** of **other SIBs** or of **posSIBs** that can be contained in a single `SystemInformation` **message**.

### 5.5.3.1 `SI-SchedulingInfo` data type

It is **only broadcast** (or **unicast**) within `SIB1` (click 5.3.4).

For its **ancestor tree up to** `SIB1`, click 5.5.3.

Its **only parent data type** is `SIB1` (click 5.3.4).

**All** its parameters are **common** (click 5.1.3).

**Each** of its **instances** contains the **scheduling configuration** for **other SIBs** in the **cell** where `SIB1` is sent. It is **restricted** to **other SIBs** defined in **R15** and **R16** (i.e., up to `SIB14-r16`).

#### 5.5.3.1.1 SI-SchedulingInfo parameters

Its **five parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- **schedulingInfoList** **SEQUENCE (SIZE (1..maxSI-Message)) OF** **SchedulingInfo** (click 5.5.3.2): mandatory, common

  It does **not** have a **field description** in TS 38.331, but there is some information about it in the **running text**.

  It is then **assumed** that it contains the **scheduling configuration** for **other SIBs** (click 5.5.2). Each **nameless parameter** with **data type** **SchedulingInfo** provides the **format** of **one** **SystemInformation** (click 5.7.5) message, by indicating its **contents** and delivery **timing**.

  The constant **maxSI-Message** is **32**, which is the **maximum quantity** of **SystemInformation** **message formats** that can be configured to **each UE**.

- **si-RequestConfig** **SI-RequestConfig** (click 5.5.3.4): optional, common

  It is the **Uplink** (**not SUL**) configuration of **Msg1 resources** that the **UE** uses for requesting **SystemInformation** **messages** for which **si-BroadcastStatus** from **SchedulingInfo** (click 5.5.3.2) is set to **notBroadcasting**. It applies to **all other SIBs** sent in the **cell**.

- **si-RequestConfigSUL** **SI-RequestConfig** (click 5.5.3.4): optional, common

  It is the **SUL** configuration of **Msg1 resources** that the **UE** uses for requesting **SystemInformation** (click 5.7.5) **messages** for which **si-BroadcastStatus** from **SchedulingInfo** (click 5.5.3.2) is set to **notBroadcasting**. It applies to **all other SIBs** sent in the **cell**.

- **si-WindowLength** **ENUMERATED**: mandatory, common

  It is the length of the **SI scheduling window** (click 5.5.2) in **slots**. Value **s5** corresponds to **5 slots**, value **s10** corresponds to **10 slots** and so on. It applies to **all other SIBs** sent in the **cell**.

  The network **always** configures **si-WindowLength** to be **shorter than or equal** to **si-Periodicity** from **any** of the **SchedulingInfo** (click 5.5.3.2), within **schedulingInfoList** (see above). The values **s2560-v1710** and **s5120-v1710** are only applicable for a **subcarrier spacing** of **480 kHz**.

  Its **data type** is the **ENUMERATED {s5, s10, s20, s40, s80, s160, s320, s640, s1280, s2560-v1710, s5120-v1710}**.

- **systemInformationAreaID** **BIT STRING (SIZE (24))**: optional, common

  If it is **present**, it is **identity** of the **system information area** (click 3.5.4) to which the **cell belongs to**.

  If a given **other SIB** is configured with **areaScope** from **SIB-TypeInfo** (click 5.5.3.3) or **areaScope-r16** from **PosSIB-Type-r16**, respectively, the parameters that it contains apply to **all** the **cells** in this **system information area**.

This **24-bit** identity for **system information areas** is **unique** within each **PLMN** or **SNPN**.

For more detail on these parameters, refer to the table "*SI-SchedulingInfo field descriptions*" of TS 38.331.

#### 5.5.3.1.2 SI-SchedulingInfo ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
SI-SchedulingInfo ::= SEQUENCE {
  schedulingInfoList SEQUENCE (SIZE (1..maxSI-Message)) OF SchedulingInfo,
  si-WindowLength ENUMERATED {s5, s10, s20, s40, s80, s160, s320,
                              s640, s1280, s2560-v1710, s5120-v1710},
  si-RequestConfig    SI-RequestConfig OPTIONAL,                -- Cond MSG-1
  si-RequestConfigSUL SI-RequestConfig OPTIONAL,             -- Cond SUL-MSG-1
  systemInformationAreaID BIT STRING (SIZE (24)) OPTIONAL,           -- Need R
  ...
}
```

where maxSI-Message is 32, which is the **maximum quantity** of SystemInformation (click 5.7.5) **message formats** that can be configured to **each UE**.

### 5.5.3.2 SchedulingInfo data type

It is **only broadcast** (or **unicast**) within SIB1 (click 5.3.4).

For its **ancestor tree up to** SIB1, click 5.5.3.

**All** its parameters are **common** (click 5.1.3).

It contains the **scheduling configuration**

**Each** of its **instances** configures **1 format** of SystemInformation (click 5.7.5) **message**, indicating its **contained SIBs**, if it is **periodically broadcast** or **on-demand**, and the **broadcasting period**. It corresponds to the **cell** where SIB1 is sent. It is **restricted** to **other SIBs** defined in **R15** and **R16** (i.e., up to SIB14-r16).

#### 5.5.3.2.1 SchedulingInfo parameters

Its **three parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- si-BroadcastStatus ENUMERATED {broadcasting, notBroadcasting}: mandatory, common

  It indicates if the SystemInformation (click 5.7.5) **message format** configured by the SchedulingInfo instance is **periodically broadcast** or **on-demand**. When set to broadcasting, the indication is **valid** until the **end** of the **BCCH modification period**. When SIB19-r17 is scheduled, the si-BroadcastStatus for the **mapped** SIB19-r17 is set to broadcasting.

  ***If*** si-SchedulingInfo-v1700 from SIB1-v1700-IEs is **present**, and either si-RequestConfig or si-RequestConfigSUL (both from SI-SchedulingInfo) or si-RequestConfigRedCap-r17 (from SI-SchedulingInfo-v1740) is configured, ***then***, the **network** ensures that the **total quantity** of **specific** SystemInformation (click 5.7.5) **messages** in the **list** of **concatenated** SystemInformation messages, does **not exceed** the **limit** of maxSI-Message (which is 32). The **specific** SystemInformation **messages** subject to this **limit** are: the ones that have si-BroadcastStatus

(from SchedulingInfo) set to notBroadcasting configured by schedulingInfoList from SI-SchedulingInfo (within si-SchedulingInfo) **plus** the ones containing **type1 SIB** configured by schedulingInfoList2-r17 from SI-SchedulingInfo-v1700 (within si-SchedulingInfo-v1700).

A modification of si-BroadcastStatus should not result in system information change notifications in Short Message transmitted with **P-RNTI** over **DCI** (refer to clause 6.5 of TS 38.331).

- si-Periodicity ENUMERATED: mandatory, common/

  It indicates the **broadcast periodicity** of the SystemInformation (click 5.7.5) **message format** configured by the SchedulingInfo instance. The **periodicity** is expressed in **radio frames**. Value rf8 corresponds to **8 radio frames**, value rf16 corresponds to **16 radio frames**, and so on.

  Its **data type** is the ENUMERATED {rf8, rf16, rf32, rf64, rf128, rf256, rf512}.

- sib-MappingInfo SIB-Mapping: mandatory, common

  It does **not** have a **field description** in the table "*SchedulingInfo field descriptions*" of TS 38.331.

  It is then **assumed** that it has the **same semantics** as its **version-different parameter** sib-MappingInfo-r17 from SchedulingInfo2-r17 that has **data type** SIB-Mapping-v1700.

  It indicates the **list** of **SIBs** that are contained in the SystemInformation (click 5.7.5) **message format** configured by the SchedulingInfo instance. For **each SIB** in the list, it indicates with areaScope if it applies to a **cell** or to an **SI area**

  Its **data type** SIB-Mapping is a SEQUENCE (SIZE (1..maxSIB)) OF SIB-TypeInfo (click 5.5.3.3). The constant maxSIB is 32, which is the **maximum quantity** of **other SIBs** or of **posSIBs** that can be contained in a single SystemInformation **message**.

For more detail on these parameters, refer to the table "*SchedulingInfo field descriptions*" of TS 38.331.

#### 5.5.3.2.2 SchedulingInfo ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
SchedulingInfo ::= SEQUENCE {
  si-BroadcastStatus ENUMERATED {broadcasting, notBroadcasting},
  si-Periodicity ENUMERATED {rf8, rf16, rf32, rf64, rf128, rf256, rf512},
  sib-MappingInfo SIB-Mapping
}

SIB-Maping := SEQUENCE (SIZE (1..maxSIB)) OF SIB-TypeInfo
```

### 5.5.3.3 SIB-TypeInfo data type

It is **only broadcast** (or **unicast**) within SIB1 (click 5.3.4).

For its **ancestor tree up to** SIB1, click 5.5.3.

**All** its parameters are **common** (click 5.1.3).

**Each** of its **instances** indicates the **type** of **other SIB**. It also indicates if **this SIB** applies to **all** the **cells** in a **SI area** (click 5.3.5) or **only** to the **cell** where it is sent. It is **restricted** to **other SIBs** defined in **R15** and **R16** (i.e., up to `SIB14-r16`).

#### 5.5.3.3.1 `SIB-TypeInfo` parameters

This **data type** does **not** have a **table** of **field descriptions** in TS 38.331.

Its **three parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `areaScope` **`ENUMERATED {true}`**: optional, common

  It does **not** have a **field description** in TS 38.331, but its **same-name parameters** from `SchedulingInfo` (click 5.5.3.2) and from `PosSI-SchedulingInfo` have a **field description**, with a **similar semantics**.

  When **present**, it indicates that a given **other SIB** applies to **all** the **cells** in a **SI area** (click 5.3.5). When **absent**, the given **other SIB** applies **only** to the **cell** where it is **sent**.

- `type` **`ENUMERATED`**: mandatory, common

  It does **not** have a **field description** in TS 38.331. It is then **assumed** that it indicates the **type of other SIB** of this **instance** of this data type.

  Its **data type** is an **`ENUMERATED`** that can identify the **SIB type** (click 5.7.6.1) from **2** to **14**. This covers **all** the **SIBs** up to **R16**, but it does **not** cover **SIBs** introduced in **R17** or **later**. These **SIBs** are covered with other new parameters.

- `valueTag` **`INTEGER (0..31)`**: optional, common

  It does **not** have a **field description** in TS 38.331, but some explanations are given about it.

#### 5.5.3.3.2 `SIB-TypeInfo` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
SIB-TypeInfo ::= SEQUENCE {
  type ENUMERATED {sibType2, sibType3, sibType4, sibType5, sibType6,
                   sibType7, sibType8, sibType9, sibType10-v1610,
                   sibType11-v1610, sibType12-v1610, sibType13-v1610,
                   sibType14-v1610, spare3, spare2, spare1,... },
  valueTag INTEGER (0..31) OPTIONAL,                    -- Cond SIB-TYPE
  areaScope ENUMERATED {true} OPTIONAL                          -- Need S
}
```

### 5.5.3.4 `SI-RequestConfig` data type

It is **only broadcast** (or **unicast**) within `SIB1` (click 5.3.4).

For its **ancestor tree up to** `SIB1`, click 5.5.3.

Its **4 parent data types** are `SI-SchedulingInfo` (click 5.5.3.1), `SI-SchedulingInfo-v1700`, `SI-SchedulingInfo-v1740` and `PosSI-SchedulingInfo-r16`.

**All** its parameters are **common** (click 5.1.3).

`SI-RequestConfig` contains the **configuration** for **system information request** based on **Msg1** (click 9.3.2).

#### 5.5.3.4.1 `SI-RequestConfig` parameters

Its **five parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `rach-ConfigSI` `RACH-ConfigGeneric` (click 9.5.4): mandatory (within optional), common

  It does **not** have a **field description** in TS 38.331. It is assumed that it configures **parameters** for the **random access procedure** for the case of **System Information request**.

- `rach-OccasionsSI` **`SEQUENCE`**: optional, common

  It contains the configuration of **dedicated RACH occasions** for **system information**. If it is **absent**, the **UE** uses `rach-ConfigCommon` from `BWP-UplinkCommon` (click 2.2.8.9) corresponding to the configuration of the **Uplink initial bandwidth part**.

  Its **`SEQUENCE`**: contains `rach-ConfigSI` (see below) and `ssb-perRACH-Occasion` (see above).

- `si-RequestPeriod` **`ENUMERATED`**: optional, common

  It is the **periodicity** of the `SI-RequestConfig`, expressed in **quantity of association periods**.

  Its **data type** is the **`ENUMERATED`** `{one, two, four, six, eight, ten, twelve, sixteen}`.

- `si-RequestResources` **`SEQUENCE (SIZE(1..maxSI-Message)) OF`** `SI-RequestResources` (click 5.5.3.5): mandatory, common

  It contains the configuration for **on-demand other SIBs and posSIBs**.

  The constant `maxSI-Message` is `32`, which is the **maximum quantity** of `SystemInformation` (click 5.7.5) **message formats** that can be configured to **each UE**.

  If the **list** contains **only one entry**, the configuration is used for **all** `SystemInformation` **messages** for which `si-BroadcastStatus` from `SchedulingInfo` (click 5.5.3.2) or `posSI-BroadcastStatus` from `PosSchedulingInfo-r16` is set to `notBroadcasting`.

  If the **list** contains **more than one entry**, the configuration depends on whether `si-SchedulingInfo-v1700` from `SIB1-v1700-IEs` is **present** or **not**, as follows:

  - If it is **not present** and this `SI-RequestConfig` is used for **on-demand** system information request in `SI-SchedulingInfo` (click 5.5.3.1) or in `PosSI-SchedulingInfo`, then:

    The **first entry** in the **list** corresponds to the **first** `SystemInformation` (click 5.7.5) **message** in `schedulingInfoList` from `SI-SchedulingInfo` (click 5.5.3.1) or in `posSchedulingInfoList-r16` from `PosSI-SchedulingInfo-r16` for which `si-BroadcastStatus` from `SchedulingInfo` (click 5.5.3.2) or `posSI-BroadcastStatus` from

PosSchedulingInfo-r16 (respectively) is **set** to notBroadcasting; the **same** for the **second** and **successive entries** in the **list**.

- If it **is present** and this SI-RequestConfig is configured in SI-SchedulingInfo for **on-demand** system information request, then:

  The **UE** generates a **list** of **concatenated** SystemInformation (click 5.7.5) **messages** by **appending** the SystemInformation **messages** containing **type1 SIB** configured by schedulingInfoList2-r17 from SI-SchedulingInfo-v1700 in si-SchedulingInfo-v1700 from SIB1-v1700-IEs to the SystemInformation **messages** configured by schedulingInfoList from SI-SchedulingInfo (click 5.5.3.1) in si-SchedulingInfo from SIB1 (notice there is also si-SchedulingInfo-v1700 and si-SchedulingInfo-v1740).

  The **first entry** in the **list** corresponds to the **first** SystemInformation **message** for which si-BroadcastStatus from SchedulingInfo (click 5.5.3.2) is set to notBroadcasting; the **same** for the **second** and **successive entries** in the **list**.

- If it **is present** and SI-RequestConfig is configured in PosSI-SchedulingInfo for **on-demand** system information request, then:

  The **UE** generates a **list** of **concatenated** SystemInformation (click 5.7.5) **messages** by **appending** the SystemInformation messages containing **type2 SIB** configured by schedulingInfoList2-r17 from SI-SchedulingInfo-v1700 in si-SchedulingInfo-v1700 from SIB1-v1700-IEs to the SystemInformation **messages** configured by posSchedulingInfoList-r16 from PosSI-SchedulingInfo-r16 in posSI-SchedulingInfo-r16 from SIB1-v1610-IEs.

  The **first entry** in the **list** corresponds to the **first** SystemInformation **message** in schedulingInfoList from SI-SchedulingInfo (click 5.5.3.1) or in posSchedulingInfoList-r16 from PosSI-SchedulingInfo-r16 for which si-BroadcastStatus from SchedulingInfo (click 5.5.3.2) or posSI-BroadcastStatus from PosSchedulingInfo-r16 (respectively) is **set** to notBroadcasting; the **same** for the **second** and **successive entries** in the **list**.

A change of si-RequestResources should **not** result in **system information change notification**.

- ssb-perRACH-Occasion **ENUMERATED**: mandatory (within optional), common

  It does **not** have a **field description** in the table "*SI-RequestConfig field descriptions*" of TS 38.331. However, it has **two same-name parameters both** of which have its **same data type** and the following field descriptions:

  - From table "*BeamFailureRecoveryConfig field descriptions*" of TS 38.331:

    It is the **quantity** of **SSBs** per **RACH occasion** for **contention-free beam failure recovery**, (refer to clause 8.1 of TS 38.213).

- From table "*CFRA field descriptions*" of TS 38.331:

  It is the **quantity** of **SSBs** per **RACH occasion**.

  Its **data type** is the **ENUMERATED{oneEighth, oneFourth, oneHalf, one, two, four, eight, sixteen}**

For more detail on these parameters, refer to the table "*SI-RequestConfig field descriptions*" of TS 38.331.

#### 5.5.3.4.2 SI-RequestConfig ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
SI-RequestConfig ::= SEQUENCE {
  rach-OccasionsSI SEQUENCE {
    rach-ConfigSI RACH-ConfigGeneric,
    ssb-perRACH-Occasion ENUMERATED {oneEighth, oneFourth, oneHalf, one,
                                     two, four, eight, sixteen}
  } OPTIONAL,                                                    -- Need R
  si-RequestPeriod ENUMERATED {one, two, four, six, eight,
                               ten, twelve, sixteen} OPTIONAL,   -- Need R
  si-RequestResources SEQUENCE (SIZE (1..maxSI-Message))
                      OF SI-RequestResources
}
```

Where the constant **maxSI-Message** is **32**, which is the **maximum quantity** of **SystemInformation** **message formats** that can be configured to **each UE**.

### 5.5.3.5 SI-RequestResources data type

It is **only broadcast** (or **unicast**) within **SIB1** (click 5.3.4).

For its **ancestor tree up to** **SIB1**, click 5.5.3.

Its **only parent data type** is **SI-RequestConfig** (click 5.5.3.4).

**All** its parameters are **common** (click 5.1.3).

**Each** of its **instances** contains **parameters** for the **random access procedure** (click 9) for the case of **System Information request**.

#### 5.5.3.5.1 SI-RequestResources parameters

Its **three parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- **ra-AssociationPeriodIndex** **INTEGER (0..15)**: optional, common

  It is the **index** of the **association period** in the **si-RequestPeriod** (from the **enclosing instance** of **SI-RequestConfig**, click 5.5.3.4) in which the **UE** can send the **SI request** for **SI message(s)** corresponding to this **instance** of **SI-RequestResources**, using the **RA preamble** indicated by **ra-PreambleStartIndex** (see below) and the **PRACH transmission occasions** indicated by **ra-ssb-OccasionMaskIndex** (see below).

- `ra-PreambleStartIndex INTEGER (0..63)`: mandatory, common

  It indicates the **RA preamble index** to be used. The **RA preamble** to be used **depends** on the **quantity** "***N***" of **SSBs** that are associated with **each PRACH transmission occasion** as follows:

  - ***If*** "$N \geq 1$", ***then***,

    For the ***i*-th SSB** (where $i \in \{0, \ldots, N-1\}$) the **RA preamble index** is:

    `ra-PreambleStartIndex` + ***i***

  - ***If*** "$N < 1$", ***then***,

    The **RA preamble index** is `ra-PreambleStartIndex`.

- `ra-ssb-OccasionMaskIndex INTEGER (0..15)`: optional, common

  It does **not** have a **field description** in the table "*SI-RequestResources field descriptions*" of TS 38.331.

  It is then **assumed** that it is the explicitly signaled **PRACH Mask Index** for **random access resource selection**. The **mask** is **valid** for **all** the corresponding **SSB resources**.

  It has **3 same-name parameters**, with the **same data type** and **similar semantics**, from `BeamFailureRecoveryConfig` (click 9.5.4), from `CFRA` (click 9.5.11), and from `CFRA-TwoStep-r16` (click 9.5.12.

For more detail on these parameters, refer to the table "*SI-RequestResources field descriptions*" of TS 38.331.

#### 5.5.3.5.2 `SI-RequestResources` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
SI-RequestResources ::= SEQUENCE {
  ra-PreambleStartIndex INTEGER (0..63),
  ra-AssociationPeriodIndex INTEGER (0..15) OPTIONAL                -- Need R
  ra-ssb-OccasionMaskIndex INTEGER (0..15) OPTIONAL                 -- Need R
}
```

## 5.6 Other RRC parameters and variables

In addition to the **UE parameters** (click 5.6), TS 38.331 also contains the specification of **UE parameters** that are **sent** from the **UE** to the **NR node** (click 5.6.1) and the specification some **UE variables** (click 5.6.2).

### 5.6.1 RRC parameters sent by the UE to the NR node

In addition to **UE parameters** (click 5.6), TS 38.331 also contains the **ASN.1 specification** of **UE parameters** that are **sent** from the **UE** to the **NR node**. These **parameters** are delivered within **RRC messages** that belong to the **RRC message classes** (click 5.8.5):

- **RRC UL-CCCH-Message class** (click 5.8.5.7).
- **RRC UL-CCCH1-Message class** (click 5.8.5.8).
- **RRC UL-DCCH-Message class** (click 5.8.5.9).

**Each RRC messages** within **each** of these **message classes** delivers a **request** or **answers** an answer from the **UE** to the **NR node**.

For example, the **`UECapabilityInformation` RRC message** is **sent** by the **UE** as an **answer** to having **received** from the **NR node** the **`UECapabilityEnquiry` RRC message**. The **`UECapabilityInformation` RRC message** informs the **NR node** of the **capabilities** (click 4.9.1) of the **UE**.

### 5.6.2 RRC UE variables

In addition to **UE parameters**, TS 38.331 also contains the **ASN.1 specification** of **UE variables**.

The specification of these **UE variables** is intended to facilitate the specification of the **UE behavioral requirements**. However, unless explicitly specified otherwise, it is up to **UE implementation** how to store the variables. The **data type names** of **UE variables** start with "**`Var`**". **Each UE variable** is a **top named data type** (click 5.9.6) in the **RRC ASN.1 specification** (click 5.7.2).

Within the **ASN.1** specification of **UE variables**, the **optionality** of the parameters is used **only** to indicate that the values may **not always** be **available**.

For more detail on **UE variables**, refer to clause 7.4 of TS 38.331.

## 5.7 RRC parameter specification

The **higher-layers UE parameters** (click 5.1.1) contain **almost all** the **UE parameters**. **Almost all** of the **higher-layers UE parameters** are defined as part of the **RRC protocol specification** in TS 38.331. For this reason, we will also call the **higher-layer parameters** the "**RRC UE parameters**".

The TS 38.331 document has **over 1,300 pages** with the **ASN.1** (click 5.9) **specification** and the **textual semantics** of **thousands** of **RRC UE parameters**, **data types** and **constants**. The **ASN.1 specification** has well over **20,000 lines**.

In addition to being **quite large**, the TS 38.331 document is **quite cryptic**.

Regarding **data types** and **constants**, you may find useful the **explanation** on their **data identification** and **semantics** found in 5.7.1.

Regarding **parameters**, the **explanation** on their **identification** and **instantiation** is done in 5.7.2, while a **detailed explanation** on their **semantics** may be found in 5.7.3.

You may also find useful the **explanations** on **`SIB1`** parameters (click 5.7.4.1), on the parameters of **other SIBs** and **posSIBs** (**`SystemInformation`**, click 5.7.5) and on the **non-SI** parameter **structure** (click 5.7.8).

Finally, we also perform a description on the **UE internal variables** (click 5.6.2) contained in TS 38.331, although these are **implementation dependent**.

### 5.7.1 RRC UE data types and constants

**Data type and constant identification**

Recall from 5.9.2 that **data type names** and **constant names** are **unique** within **each ASN.1 module**, so their **identification** is **straightforward**. Notice though that you may have **version-different data type names** (click 5.7.2.2), which has some **implications**.

**Data type semantics**

**Some data types** have a **brief semantics description** in TS 38.331. These **semantics descriptions** may be found in **clause 6**, organized in the following **groups** of **data types** (i.e., information elements):

- Clause 6.2.2: **RRC messages**
- Clause 6.3.1: **System Information Blocks**
- Clause 6.3.1a: **positioning System Information Blocks**
- Clause 6.3.2: **general RRC**
- Clause 6.3.3: **UE capability**
- Clause 6.3.4: **other**
- Clause 6.3.5: **Sidelink**
- Clause 6.3.5: **Multicast Broadcast Service**

If you want to **locate** the **description** of a **specific data type**, you may **search** (**whole words**, **case sensitive** and **12pt arial-italics font**) for the **data type name** through **all** the TS 38.331 **file**, in a **file format** and **viewer** that **allows** to **search** for **specific fonts**. To cite one such field format and viewer, **MS-Word** allows to **search** for **specific fonts** (while acrobat **does not**).

Some **data types** also have a **field descriptions table**, that contains the **semantics description** (i.e., field description) of **some fields**. The way to associate a **field description** to a **specific parameter** is far from trivial (click 5.7.35.7.3.1).

For more detail on **PDU specification naming conventions**, refer to annex "*A.3.1.2 ASN.1 identifier naming conventions*" of TS 38.331.

**Constant semantics**

We call **constants** to **ASN.1 named values** (click 5.9.2). Do not confuse them with what TS 38.331 calls constants (refer to clause 7.3 of TS 38.331).

A **listing** of **constant names**, with their **corresponding data type** and **value**, may be found in clause "*6.4 RRC multiplicity and type constraint values*" of TS 38.331. **Most** of the **constants** have a **brief description** of their **semantics**.

If you want to **locate** a **specific constant**, you may **search** (**case sensitive**) for the **data type name** followed by **a few spaces** through **all** the TS 38.331 **file**.

## 5.7.2 RRC UE parameters: identification and instances

The way **each** of the **thousands** of **RRC UE parameters** defined in TS 38.331 is **uniquely identified** (and **referred to** in other **TS** documents) is **not obvious** and is presented in 5.7.2.1.

Related with **parameter identification**, it is important to understand the **implications** of parameters that have the **same-name** and/or a **version-different name** (click 5.7.2.2).

**Each** of the **thousands** of **RRC UE parameters** is **defined only once**, within **its enclosing data type**. However, **each parameter** is **usually instantiated many times** throughout the **RRC ASN.1 specification** (click 5.7.2.3).

### 5.7.2.1 RRC UE parameter identification

Each **parameter** defined within TS 38.331 is **uniquely** identified by its **parameter name** (i.e., its **ASN.1 field name**) and the **name** of its **enclosing named data type** (i.e., the name the information element where the **field** is contained).

**Data type names** are **unique** within each **ASN.1** module (click 5.9). **Field names** are **not unique** within each **ASN.1 module**, and actually there are **many fields** with the **same name** and/or a **version-different name** (click 5.7.2.2) in TS 38.331. However, in TS 38.331 the **field names** are **unique** within the specification of **each named data type**. Consequently, the **combination** of a **parameter name** and its **enclosing data type name** provides a **unique identifier** for each **parameter**. Notice that in **ASN.1** it is **not** mandatory that **field names** are **unique** within the specification of **each named data type**, but in TS 38.331 they have been made **unique**.

Summarizing, in order to **uniquely identify** an **RRC UE parameter**, it is **required** to provide the **parameter name plus** its enclosing **named data type**.

In spite of this **requirement**, most **TS documents** refer to each **RRC parameter** with **only** the **field name**. This is **ambiguous**, because **many fields** have the **same name** or a **version-different name** (click 5.7.2.2) in TS 38.331. Moreover, different TS documents refer an **RRC parameter** or **data type** suppressing its **version-suffix** (click 5.7.2.2) instead of using its **full name**. This **ambiguous identification** of **parameters** and **data types** performed in **TS documents** is a source of mistakes that you should be careful to avoid.

To **avoid** the said **ambiguity**, along this book we will **identify each RRC parameter** by its **field name and** the **name** of **its enclosing named data type**. In other words, we will **identify each parameter** as "`parameter-Name` **from** `Its-Enclosing-Data-Type`", where the **RRC ASN.1 specification** of `Its-Enclosing-Data-Type` **contains** an **ASN.1** field named `parameter-Name`. Do **not confuse** the **enclosing data type** of the **parameter** with the **data type assigned** to the **parameter itself**, which are **always different data types**.

Notice that **ASN.1** specifications in TS 38.331 contain **nameless fields**, but these are **not** used as **parameters**, and therefore do **not** pose an **identification problem**.

For more detail on **parameter identification**, refer to clause 6.1 of TS 38.331.

### 5.7.2.2 RRC same-name and version-different UE parameters and data types

Having parameters with the **same name** or **version-different names** and/or **data types** with **version-different names** may initially surprise you, but it is allowed (click 5.9.5) by **ASN.1**, and it is **not infrequent** in TS 38.331. This is **usually** done on purpose by the **TS** authors intending to **identify** parameters and/or **data types** that have the **same** or **very similar semantics** (click 5.7.3), although this is **not always** the case.

**These similarities** between **parameter names** and/or between **data type names** create a **significant potential** for **confusion** and on our view, it is a **bad practice**.

We now **explain** some frequent **cases** of **similar names** of parameters and/or **data type**s, with the intention of **reducing** the **potential** for **confusion**, in particular in conjunction with section 5.7.3.

**same-name parameters**

They **are parameters** whose **parameter name** (i.e., same field name) is **exactly** the **same**. Notice that even if they have the **same parameter name**, they are **different parameters**, because they are **enclosed** in **different data types** (click 5.7.2.1).

**same-name parameters** may have the **same**, **similar** or **different semantics** (click 5.7.3.4).

For example, **2 same-name parameters** are `format1` from `PUCCH-Config` (click 7.9.4) and `format1` from `PUCCH-Resource` (click 7.9.7). They are **two different** parameters with **exactly** the **same name**. In this example, they have **different semantics**.

As another example, **2 same-name parameters** are `controlResourceSetId` from `ControlResourceSet` (click 6.12.6) and `controlResourceSetId` from `SearchSpace` (click 6.12.8). They are **two different** parameters with **exactly** the **same name**. In this example, they have the **same semantics**. For more detail on this example, click 5.7.3.4.5.

**ASN.1 data type** and **constant names must** be **unique** within **each ASN.1** (click 5.9) **module**. Therefore, there **cannot** be **same-name data types nor same-name constants**.

**Version-suffix**

It is a **text string** at the **end** of some **parameter**, **data type** or **constant names** that indicates (after a hyphen) a **release** or **version number**. The **hyphen prefixing** the **release** or **version number** is **included** in the **version-suffix**.

On **most cases**, the **version-suffix** is at the **very end** of the name.

Two examples of these cases are **version-suffixes** "`-v1610`" (version number) and "`-r16`" (release number), corresponding to parameter names `controlResourceSetId-v1610` and `controlResourceSetId-r16` (respectively).

On a **few cases**, the **version-suffix** is at the **end**, but **not** at the **very end**, of the **name**.

An example of these cases is **version-suffix** "`-v1530`", which is **not** at the **very end** of `RRCReconfigurationComplete-v1530-IEs`.

**Version-different parameters**, **data types** or **constants**

They are **parameters**, **data types** or **constants whose names** are **exactly** the **same**, **except** for the **version-suffix**.

An example of **3 version-different parameters** are `controlResourceSetId` and `controlResourceSetId-v1610` **both** from `ControlResourceSet` (click 6.12.6) and `controlResourceSetId-r16` from `SearchSpaceExt-r16`. The **version-suffixes** of the two latter parameters are, respectively, "`-v1610`" (version number) and "`-r16`" (release number).

An example of **3 version-different data types** are `ControlResourceSetId`, `ControlResourceSetId-v1610` (click 6.12.7) and `ControlResourceSetId-r16`. The **version-suffixes** of the two latter **data types** are, respectively, "`-v1610`" (version number) and "`-r16`" (release number).

**Version-different parameters**, **data types** or **constant**s usually have **very related semantics** (click 5.7.3) and have been introduced in different **5G releases** or **version numbers**.

**Version-prefix and Search-prefix**

The **version-prefix** of several **version-different names** is the result of **removing** the **version-prefix** from the **version different names**. The **version-prefix may be** the **same** as one of the **version-different** names.

The **search-prefix** of several **version-different names** is the **text prefix** of the said names **up to**, but **not** including, the **version-suffix**. The **search-prefix** is **never** the **same** as one of the **version-different** names.

The **version-prefix** and **search-prefix** are used in the procedure to determine the **semantics** of **parameters**.

On **most cases**, the **version-suffix** is at the **very end** of the **name**. In these cases, the **version-prefix** is the **same** as the **search-prefix**. For example, the **version-prefix** of `controlResourceSetId`, `controlResourceSetId-v1610` and `controlResourceSetId-r16` is `controlResourceSetId`, which is the same as their **search-prefix**.

However, on a **few cases**, the **version-suffix** is **not** at the **very end** of the **name**. In these cases, the **version-prefix** is **not** the **same** as the **search-prefix**. For example, the **version-prefix** of `RRCReconfigurationComplete-IEs`, `RRCReconfigurationComplete-v1560-IEs` and `RRCReconfigurationComplete-v1610-IEs` is `RRCReconfigurationComplete-IEs`, which is **different** to their **search-prefix**, which is `RRCReconfigurationComplete`.

More examples of **version-prefix** and **search-prefix** are:

- For the **version-different** parameters `controlResourceSetId`, `controlResourceSetId-v1610` and `controlResourceSetId-r16`, their **version-prefix** and **search-prefix** is `controlResourceSetId`.
- For the **version-different** parameters `controlResourceSetId-v1610` and `controlResourceSetId-r16`, their **version-prefix** and **search-prefix** is `controlResourceSetId`.
- For the **version-different** data types `ControlResourceSetId`, `ControlResourceSetId-16` and `ControlResourceSetId-v1610`, their **version-prefix** and **search-prefix** is `ControlResourceSetId`.
- For the **version-different** data types `RRCReconfigurationComplete-IEs` and `RRCReconfigurationComplete-v1530-IEs`, their **version-prefix** is `RRCReconfigurationComplete-IEs` and their **search-prefix** is `RRCReconfigurationComplete`. Notice that they are different because in this case the **version-suffix** is **not** at the **very end** of the name.
- For the **version-different** data types `RRCReconfigurationComplete-v1530-IEs`, `RRCReconfigurationComplete-v1700-IEs` and `RRCReconfigurationComplete-v1720-IEs`, their **version-prefix** is `RRCReconfigurationComplete-IEs` and their **search-prefix** is `RRCReconfigurationComplete`. Notice that they are different because in this case the **version-suffix** is **not** at the **very end** of the name.

**Case-different parameters or data types**

They are **parameters** (or **data types**) with **exactly** the **same name**, except for the **case** of **one** (or **more**) letter(s).

We **consider** that this is due to **typos** in TS 38.331 and these cases **should not exist**.

An example of **2 case-different parameters** is `ssb-PeriodicityServingCell` from `ServingCellConfigCommonSIB` (click 3.9.3) and `ssb-periodicityServingCell` from `ServingCellConfigCommon` (click 3.9.6). Their only difference is the **case** of "`P`" in "`Periodicity`".

We **consider** that this is a **typo** in TS 38.331 and `ssb-PeriodicityServingCell` should actually be `ssb-periodicityServingCell`.

#### 5.7.2.3 RRC UE parameter instances

For every **RRC named data type**, **one** of its **named data type instances** (click 5.9.7) may contain **one value** of that **data type**. Consequently, **each** of its **named data type instances** may contain **one** (or **zero**, in case of an optional parameter) **parameter instances** for **each** and **every** of the **parameters contained** within that **named data type**.

For every **RRC named data type T**, the **quantity** of **instances** of **parameter** "***P***" (within its enclosing **named data type E(P)**) that **T may contain** is the same as the quantity of **instances** of **E(P)** that **T** may contain.

The **quantity of instances** of **E(P) within T** is the **quantity** of **parameter instances within T** whose **data type** is **E(P)**. The **quantity** of **parameter instances within T** whose **data type** is **E(P)** is the **quantity** of **parameters** within the **ASN.1 specification** of **T**, **plus** the **summatory** of **parameter instances** whose **data type** is **E(P)** corresponding to **all** the **named data type instances within T**. For more detail on the **quantity of instances**, click 5.9.7.

### 5.7.3 RRC UE parameters: semantics

The **RRC UE parameters** are **specified** in **TS 38.331** using over **20,000 lines** of **ASN.1**. The **ASN.1 language** does **not** have semantics: it just provides the **abstract specification** of **parameters** (i.e., fields), **data types** (i.e., information elements) and **constants** (i.e., named values). **ASN.1** may also provide some associated **encoding rules**.

Although the **ASN.1 language** does **not** have **semantics**, the **ASN.1 structure** of the **data type** of each **parameter** provides **valuable information** about its **contents** and **system implications**. Additionally, the TS 38.331 document (and other **TS** documents) **explicitly define semantics** for most **parameters**, most **parameter instances**, some **data types** and most **constants**. Their **semantics** is **defined** by detailing the **system implications** of their respective **structure** and **values**.

The **semantics** of **each RRC UE parameter** is **described** in a **distributed way** within **TS 38.331** and **other TS documents**. **Within TS 38.331**, the **semantics** of **each RRC UE parameter** is **described** also in a **distributed way** in the **ASN.1 specification** of the **parameter**'s **data type**, along the **field descriptions tables** of **many data types**, along the **semantics description** of **some data types** and **constants** (click 5.7.1) and along its **different sections**. It is therefore required to take into account **all** the said **documents** and **sections** to really understand the **full semantics** of **any** given **RRC UE parameter**.

Among the documents and sections just indicated, two **particularly relevant** pieces of **RRC UE parameter semantics** are the **ASN.1 specification** of the **parameter**'s **data**

**type** and the **parameter**'s **field description**. One would expect to **locate** the **parameter**'s **field description** in the **field descriptions table** of the **parameter**'s **enclosing data type**, but there are lots of **unexpected cases**.

Consequently, the way to **locate** the **parameter**'s **data type** and **field description** is **not obvious** and it is explained in 5.7.3.1.

Most of the **unexpected cases** that you may encounter when looking for a **parameter**'s **field description** are presented in 5.7.3.2.

Regarding the cases of a **parameter**'s **field description** that also applies to **another parameter** are presented in 5.7.3.3.

Be aware that **same-name** and/or **version-different** (click 5.7.2.2) **parameters** can be **easily confused**, and this is analyzed in 5.7.3.4.

In addition to the **intrinsic semantics** of **each parameter**, **each parameter instance incorporates** some **specific semantics qualification** depending on **where** it has been **instantiated**, as described in 5.7.3.5.

Notice also that **some ASN.1 comments** may have **semantics**, as presented in 5.7.3.6.

In case the **UE** has **not yet received** a given **RRC UE parameter value**, in TS 38.331 it is stated what **specified/default value** should be used by the **UE** (click 5.7.3.7).

In the context of this section, and in respect to TS 38.331, it is relevant to clarify the following relevant **terminology issues**.

In TS 38.331, an **ASN.1 structural element** containing **single** or **multiple fields** is called "**Information Element**" (**IE**) or "**Information Element type**" (**IE type**). However, we will call it "**data type**" (click 5.9.2), for consistency with many other contexts. The same applies to the term "**value**" of TS 38.331 **ASN.1 specifications**, that we will call "**constant**", for consistency with many other contexts.

What most **5G** TS documents call configuration **parameters**, are called **fields** in TS 38.331. We will call them **parameters** for **consistency** with most TS documents.

As a final remark in this introduction, whenever you are handling an **RRC UE parameter**, it is convenient that you are aware of the following information about it:

- What is the **parameter identification** (i.e., the **name** of the **parameter** and the **name** of its **enclosing named data type**)? (click 5.7.2.1)
- What is its **data type**? (click 5.9.2)
- Is it **optional** or **mandatory**? (click 5.9.9)
- What is its **semantics**, as defined in **TS 38.331** and **other TS documents**? (as explained in this section)
- Is it **common**, **dedicated**, **common or dedicated** or **common and dedicated**? (click 5.1.3)
- What **parameter group** does it belong to? (click 5.2)
- What are its **ancestor trees**? (click 5.9.8)
- What is the **semantic qualification** of **each** of **its instances** within its **ancestor trees**? (click 5.7.3.4)

For more detail on **RRC UE parameter semantics**, refer to clause 6.1 of TS 38.331

#### 5.7.3.1 Procedure to locate a parameter's data types and field description

When along a given **TS** document, you **encounter** a reference to a **parameter name**, you usually want to **locate** its **enclosing data type** to **uniquely identify** the parameter. You also want to **locate** its **data type** and its **field description**, because they both **jointly** provide **valuable information** about the **parameter**'s **semantics**.

**Locating** the said information may be **difficult**, because there are many **unexpected cases** of **field descriptions tables** and of **field descriptions** (click 5.7.3.2).

In order to **locate** the **said information**, we suggest an approximate **procedure** designed to handle **most** of the **unexpected cases** (click 5.7.3.2), which consists of the following **steps**:

1. **Find all the same-name and version-different parameters for the parameter name that you are looking for**
2. **Find all the field descriptions corresponding to the parameter name that you are looking for**
3. **For each parameter record from step 1, add to it the field description from the best matching field description record (if any) from step 2**
4. **Process possible remaining field description records**
5. **Attempt to add a field description to each parameter record lacking it**
6. **Decide which parameter(s) from step 5 correspond to the parameter name that you are looking for**

**Each step** is now **explained** more detail:

1. **Find all the same-name and version-different parameters for the parameter name that you are looking for**

   1.1. **Search** for **same-name and version-different** (click 5.7.2.2) **parameters**

   **Search** (**whole words** and **case sensitive**) for the **search-prefix** (click 5.7.2.2) of the **parameter name** that you are **looking for** through **all** the **ASN.1 code** within TS 38.331.

   You search for the **search-prefix** because in this way with **only one search** you get **all** the **same-name** and **version-different** parameters.

   In order to restrict the **search** to the **ASN.1 code** of TS 38.331 you have **two** options. One option is **searching** in the **word version** of TS 38.331, **setting** the **font** to the one used for the **ASN.1 code**. The other option is **extracting** the **ASN.1 code** of TS 38.331 into a **text file**, and then do **all** the **searches** over that **text file**. We **recommend** the **second** option. It requires some **initial effort** to **automatically** produce the **text file** using some **stream editor**, but it **very well pays off** because of the **much easier** and **faster searches** over the **text file instead of** over the **word file**. Another advantage of the **second** option is that you may do **regular expression searches** and/or **normalize** the **code format** (e.g., having **only one space** before each "`::=`" keyword or ). Both aspects will greatly facilitate searches (e.g., to search for the **ASN.1 specification** of `NamedDataType` you search for "`NamedDataType ::=`" and you find it at the **first shot**.

**1.2. Ignore all irrelevant matches**

The search will **match** parameter names that are **neither same-name nor version-different** parameters of the **parameter name** that you are **looking for**. **All** of **these matches** should be **ignored**.

**1.3. Record all relevant matches**

For **each and every relevant match**, **record** the **complete parameter name**, its **enclosing data type** and its **data type**

**1.4. Optionally check** for **typos**

In case you want to be **very thorough**, you **may check** for **matches hidden** by **typos** in the **document**, by **repeating** the search from **1.1 without case sensitivity**. If the search returns **case-different result(s)** (click 5.7.2.2), you have **found** a **match hidden** by a **typo**. This additional search may be a little time consuming because the search will return many data types and other matches that should be ignored, but it provides the safest results.

The **result** of **step 1** is a **list** of **all** the **same-name**, **version-different** and **case-different** (if any) **parameters** in respect to the **parameter name** that you are **looking for**. **Each record** in the **list** contains **one complete parameter name** and its **enclosing data type**, that together **uniquely** identify the parameter, plus the **data type** of the parameter.

For the remaining part of this procedure, the **complete parameter name** and its **enclosing data type name** in **each** of these records will be called "**parameter name**" and "**enclosing data type**" (respectively). These **records** and this **list** of records will be called "**parameter records**" and "**parameter list**" (respectively).

Notice that this **parameter list** is **built** by a **search** over the **ASN.1 specification** of TS 38.331. Consequently, **every record** in the **parameter list** correspond to an **actual parameter** of TS 38.331. Therefore, **every pair** of **parameter name** and its **enclosing data type** in a **parameter record** actually **exist** in the **ASN.1 specification** of TS 38.331. This is **not** so in the **field descriptions list** from the next step.

As another conclusion, the **parameter list may** contain **several records** with the **same parameter name or** with the **same enclosing data type**, but it **cannot** contain any two records with the **same parameter name and** the **same enclosing data type**.

**2. Find all the field descriptions corresponding to the parameter name that you are looking for**

**2.1 Search** for **field descriptions**

**Search** (**whole words**, **case sensitive** and **arial-bold-italics font**) for the **search-prefix** (click 5.7.2.2) of the **parameter name** that you are **looking for** through **all** the TS 38.331 **file**, in a **file format** and **viewer** that **allows** to **search** for **specific fonts**. To cite one such field format and viewer, **MS-Word** allows to **search** for **specific fonts** (while acrobat **does not**).

For an example, click 5.7.3.1.2.

**2.2 Ignore all irrelevant matches**

The search will **match** parameter names that are **neither same-name** nor **version-different** parameters of the **parameter name** that you are **looking for**. **All** of **these matches** should be **ignored**.

**2.3 Record all relevant matches**

For **each and every relevant match**, **record** the **field description**, the **complete parameter name** of the **field description and all** the **data type(s)** in the **heading** of the **field descriptions table** that contains the **match**.

In case the **field description entry** has **several parameter names**, **each parameter name** produces **one record.**

For an example, click 5.7.3.1.2.

In case the **field descriptions table** has a **heading** with **several data type names**, **all** the **data type names** are **included** in the **record**.

The **result** of **step 2** (in **addition** to the result of **step 1**) is a **list** of **all** the **field descriptions** whose associated **parameter name** is **version-different** of the **parameter name** that you are **looking for**. **Each record** in the **list has one field description**, **one field description parameter name** and **one or more data type names** (the ones in the corresponding table heading).

For the remaining part of this procedure, the **complete parameter name** and the **data type name(s)** in **each** of these records will be called "**field description parameter name**" and "**field description data type(s)**". These **records** and this **list** of records will be called "**field description records**" and "**field descriptions list**" (respectively).

Notice that this **field descriptions list** is **built** by a **search** over the **field descriptions tables** of TS 38.331 (and **not** over the **ASN.1 specification**). Consequently, some **field description parameter names** and/or **some field description data types may**, or **may not**, **exist** in the **ASN.1 specification** in TS 38.331.

As another conclusion, the **field descriptions list** may contain **several records** with the **same field description parameter name or** with the **same field description data type(s)**, but it **cannot** contain any two records with the **same field description parameter name and** the **same field description data type(s)**.

For an example, click 5.7.3.1.2.

**3. For each parameter record from step 1, add to it the field description from the best matching field description record (if any) from step 2**

Take **each parameter record** and **compare** it with **each** and **every field description record**. In case the **field description record** has **more than one field description data type**, then **each parameter record** is compared with the **field description record one time** for **each field description data type**. **Compare** the **parameter names** and the **data types** in **each** of the **two records** (i.e., doing **two pairwise comparisons**).

When doing these **two pairwise comparisons**, **check** one by one the following conditions **3.1** to **3.2**, in that **same order**.

**3.1** The corresponding **field description data type** is the **same** as the **enclosing data type and** the **field description parameter name** is the **same** as the **parameter name**.

**3.2** The corresponding **field description data type** is the **same** as the **enclosing data type and** the **field description parameter name** is the **version-prefix** of the **parameter name**.

**3.3** The corresponding **field description data type** is the **version-prefix** of the **enclosing data type and** the **field description parameter name** is the **same** as the **parameter name**.

For an example, click 5.7.3.1.2.

**3.4** The corresponding **field description data type** is the **version-prefix** of the **enclosing data type and** the **field description parameter name** is the **version-prefix** of the **parameter name**.

For an example, click 5.7.3.1.2.

If **no field description record satisfies any condition**, **nothing** is **done**.

Otherwise, among **all** the **field description records** (and **field description data types**) that satisfy a **condition**, select the **field description record** (and **field description data type**) that satisfies the **higher condition** (**3.1** being **highest** and **3.4 lowest**), that is called "**best match**". **Add** the **field description** in the **best matching field description record** to its **matching parameter record** and **remove** the **field description data type** from the **field description record**. In case there is **no remaining field description data type** in the **field description record**, remove the **field description record** from the **field descriptions list**.

For an example, click 5.7.3.1.2.

The **result** of **step 3** is the **parameter list** from **step 1** where **most** or **all** records have a **field description**, **plus** a **subset** of the **field descriptions list** from **step 2**.

**4. Process possible remaining field description records**

**Apply** this **step only** if there are **any remaining records** in the **field descriptions list**.

One case in which there are **remaining field description records** when arriving to this **step 4** is when the **field description** for a parameter is **not** included in the **field descriptions table** of its **enclosing data type** (or the **table** does **not exist**), and it is **rather** included in the **field descriptions table** of **one** of its **ancestor data types**.

The actions to perform in this step are:

- Take **each remaining field description record** and **search** the **descendant data types** of its **field description data type**(**s**) for a **match** with **one** of the **enclosing data types**.
- If there is a **match**, **add** the **corresponding field description** to the **most appropriate parameter record**(**s**) and **remove** the **field description record**. Notice that the "most appropriate" is **subjective** and is based on **semantics interpretation**.

After the above, if the **field descriptions list** is **not empty** (the present procedure is not guaranteed to cover all the possible cases), **process** it applying the **spirit** of

the **present procedure**. This is, check if **any** of the **field descriptions** from the **field descriptions list** should be **added** to **any** of the **parameter records** that **lacks** a **field description**.

The **result** of **step 4** is the **parameter list** from **step 1** where **most** or **all** the **parameter records** have a **field description**.

5. **Attempt to add a field description to each parameter record lacking it**

   **Apply** this **step only** if there is **any parameter record** from **step 4** that **lacks** a **field description**.

   Since there are **parameter records** that **lack** a **field description**, this step **attempts** to **add** to each of them the **field description** from **another parameter record**. The cases where this can be done are discussed in 5.7.3.3.

   After this step, the **parameter records** that do **not** have a **field description** imply that there is **no field description** for that **parameter** in the whole TS 38.331.

6. **Decide which parameter(s) from step 5 correspond to the parameter name that you are looking for**

   **Apply** this **step only** if there is **more than one parameter record** out of **step 5**.

   **Subjectively decide** what **parameter record(s)** out of **step 5** correspond to the **parameter** that you are **looking for**. This **decision** is based on the **surrounding text** of the **TS document** where you **encountered** the **parameter name**. That **surrounding text** should provide some **semantics** to the **parameter name** that **compared** with the **semantics** of the **parameter record(s)** (**field description**, **data type** and **enclosing data type**) allows you to determine **which parameter record(s)** correspond to the **parameter** that you are **looking for**.

   For an example of **only one parameter record** out of **step 5**, click 5.7.3.1.1.

   For an example of **several parameter records** out of **step 5**, click 5.7.3.1.2.

   For a description on **data types** and **semantics** of **same-name** or **version-different** parameters, click 5.7.3.4.

With **step 6**, the **procedure** is **finalized**.

#### 5.7.3.1.1 Example of procedure for `si-RequestConfig`

The example is that you are **looking for parameter name** `si-RequestConfig`.

**Step 1. Find all the same-name and version-different parameters for the parameter name that you are looking for**

- **Substep 1.1. Search** for **same-name and version-different** (click 5.7.2.2) **parameters**

  **Search** (**whole words** and **case sensitive**) for the **search-prefix** (which is `si-RequestConfig`) of the **parameter name** that you are **looking for** through **all** the **ASN.1 code** within TS 38.331.

  The result is the following **parameter**:

  `si-RequestConfig` from `SI-SchedulingInfo`

- **Substep 1.2. Ignore all irrelevant matches**

  There are **no irrelevant matches**, so **none** is **ignored**.

- **Substep 1.3. Record all relevant matches**

  The result is the following **parameter record**:

  `si-RequestConfig` with **enclosing data type** `SI-SchedulingInfo`, and whose **data type** is `SI-RequestConfig`.

- **Substep 1.4. Optionally check** for **typos**

  **Search** (**whole words** and **case insensitive**) for `si-RequestConfig` through **all** the **ASN.1 code** within TS 38.331.

  Several matches of the **data type** `SI-RequestConfig` are found, but **no case-different parameter match** for `si-RequestConfig` is **found**.

**Step 2. Find all the field descriptions corresponding to the parameter name that you are looking for**

- **Substep 2.1. Search** for **field descriptions**

  **Search** (**whole words**, **case sensitive** and **arial-bold-italics font**) for the **search-prefix** (which is `si-RequestConfig`) of the **parameter name** that you are **looking for** through **all** the TS 38.331 **file**.

  The result is the following **match** in **one entry** of the **field descriptions table entry** of `SI-SchedulingInfo`.

  `si-RequestConfig`, "*Configuration of Msg1 resources that the UE uses for requesting SI-messages for which si-BroadcastStatus is set to notBroadcasting.*".

- **Substep 2.2. Ignore all irrelevant matches**

  There are **no irrelevant matches**, so **none** is **ignored**.

- **Substep 2.3. Record all relevant matches**

  Since the **matching entry** only has **one matching parameter names**, it produces only **one field description record**, which is:

  `perRAInfoList-v1660` with **field description data type** `SI-SchedulingInfo` and its **field description** is "*Configuration of Msg1 resources that the UE uses for requesting SI-messages for which si-BroadcastStatus is set to notBroadcasting.*".

**Step 3. For each parameter record from step 1, add to it the field description from the best matching field description record (if any) from step 2**

**Comparing** the **only parameter record** with the **only field description record** it satisfies condition 3.1 (**same data type name** and **same parameter name** in **both records**). The **resulting field descriptions list** is **empty** and the the **resulting parameter list** is:

- `si-RequestConfig` with **enclosing data type** `SI-SchedulingInfo`, and whose **data type** is `SI-RequestConfig`, and its **field description** is "*Configuration of Msg1 resources that the UE uses for requesting SI-messages for which si-BroadcastStatus is set to notBroadcasting.*".

**Step 4. Process possible remaining field description records**

This **step** is **not applied** because the **field descriptions list** is **empty**.

**Step 5. Attempt to add a field description to each parameter record lacking it**

This **step** is **not applied** because there is **no parameter record** that **lacks** its **field description**.

**Step 6. Decide which parameter(s) from step 5 correspond to the parameter name that you are looking for**

Since you are looking for **parameter name** `si-RequestConfig`, and there is **only one parameter record** with the **exact same parameter name**, you can **safely associate** the **parameter record** to the **parameter name** that you are **looking for**. Therefore, in this particular example it is **not required** to **analyze** the **context** and **semantics**, as it is **required** in the **general case**.

#### 5.7.3.1.2 Example of procedure for `perRAInfoList-v1660`

The example is that you are **looking for parameter name** `perRAInfoList-v1660`.

**Step 1. Find all the same-name and version-different parameters for the parameter name that you are looking for**

- **Substep 1.1. Search** for **same-name and version-different** (click 5.7.2.2) **parameters**

  **Search** (**whole words** and **case sensitive**) for the **search-prefix** (which is `perRAInfoList`) of the **parameter name** that you are **looking for** through **all** the **ASN.1 code** within TS 38.331.

  The result is the following **4 parameters** (listed in the same order as they were found):

  - `perRAInfoList-r17` from `FailureReportSCG`
  - `perRAInfoList-r16` from `ConnEstFailReport-r16`
  - `perRAInfoList-r16` from `RA-InformationCommon-r16`
  - `perRAInfoList-v1660` from `RA-InformationCommon-r16`

- **Substep 1.2. Ignore all irrelevant matches**

  There are **no irrelevant matches**, so **none** is **ignored**.

- **Substep 1.3. Record all relevant matches**

  The result is the following **4 parameter records** (listed in the same order as they were found):

  - `perRAInfoList-r17` with **enclosing data type** `FailureReportSCG`, and whose **data type** is `PerRAInfoList-r16`.
  - `perRAInfoList-r16` with **enclosing data type** `ConnEstFailReport-r16`, and whose **data type** is `PerRAInfoList-r16`.
  - `perRAInfoList-r16` with **enclosing data type** `RA-InformationCommon-r16`, and whose **data type** is `PerRAInfoList-r16`.

- `perRAInfoList-v1660` with **enclosing data type** `RA-InformationCommon-r16`, and whose **data type** is `PerRAInfoList-v1660`.

As indicated at the end of **step 1** from 5.7.3.1, this **parameter list** contains **2 parameter records** with the **same parameter name** (`perRAInfoList-r16`) and **2 other parameter records** with the **same enclosing data type** (`RA-InformationCommon-r16`) but it **does not** contain two records with the **same parameter name and** the **same enclosing data type**.

- **Substep 1.4. Optionally check** for **typos**

  **Search** (**whole words** and **case insensitive**) for `perRAInfoList` through **all** the **ASN.1 code** within TS 38.331.

  Several matches of **unrelated results** are found, but **no case-different parameter match** for `perRAInfoList` is **found**.

**Step 2. Find all the field descriptions corresponding to the parameter name that you are looking for**

- **Substep 2.1. Search** for **field descriptions**

  **Search** (**whole words**, **case sensitive** and **arial-bold-italics font**) for the **search-prefix** (which is `perRAInfoList`) of the **parameter name** that you are **looking for** through **all** the TS 38.331 **file**.

  The result is the following **match** in **one entry** of the **field descriptions table entry** for "*RA-InformationCommon*":

  - `perRAInfoList`, `perRAInfoList-v1660`, "*This field provides detailed information about each of the random access attempts in the chronological order of the random access attempts. If perRAInfoList-v1660 is present, it shall contain the same number of entries, listed in the same order as in perRAInfoList-r16.*".

- **Substep 2.2. Ignore all irrelevant matches**

  There are **no irrelevant matches**, so **none** is **ignored**.

- **Substep 2.3. Record all relevant matches**

  Since the **matching entry** has two **matching parameter names**, it produces **2 field description records**, which are:

  - `perRAInfoList` with **field description data type name** "*RA-InformationCommon*" and its **field description** is "*This field provides detailed information about each of the random access attempts in the chronological order of the random access attempts. If perRAInfoList-v1660 is present, it shall contain the same number of entries, listed in the same order as in perRAInfoList-r16.*"
  - `perRAInfoList-v1660` with **field description data type name** "*RA-InformationCommon*" and its **field description** is "*This field provides detailed information about each of the random access attempts in the chronological order of the random access attempts. If perRAInfoList-v1660 is present, it shall contain the same number of entries, listed in the same order as in perRAInfoList-r16.*".

As indicated at the end of **step 2** from 5.7.3.1, this **field descriptions list does not** contain any two **field description records** with the **same field description parameter name and** the **same field description data type(s)**.

**Step 3. For each parameter record from step 1, add to it the field description from the best matching field description record (if any) from step 2**

In this example, **every field description record** only has **one field description data type**, so this step is **easier** to apply. **Comparing each** of the **4 parameter records** with **each** of the **2 field description records** we get the following result.

- The **first parameter record** with **each** of the **2 field description records** satisfy **none** of the **conditions 3.1** to **3.4**, so the **first parameter record** does **not** get a **field description**. This happens because **none** of the **field description data types** is the **same** as, nor a **version-prefix** of, the **enclosing data type** `FailureReportSCG`.

  Consequently, nothing is done.

- The **second parameter record** with **each** of the **2 field description records** satisfy **none** of the **conditions 3.1** to **3.4**, so the **second parameter record** does **not** get a **field description**. This happens because **none** of the **field description data types** is the **same** as, nor a **version-prefix** of, the **enclosing data type** `ConnEstFailReport-r16`.

  Consequently, nothing is done.

- The **third parameter record** with the **first field description record** satisfies **condition 3.4**. This happens because the **field description data type name** "*RA-InformationCommon*" is the **version-prefix** of the **enclosing data type** `RA-InformationCommon-r16` **and** the **field description parameter name** `perRAInfoList` is a **version-prefix** of the **parameter name** `perRAInfoList-r16`. The **third parameter record** with the **second field description record** satisfies **no condition**.

  Consequently, the **field description** from the **first field description record** is **added** to the **third parameter record** and the **first field description record** is **deleted**. The **field descriptions list** now only has **one record** (the **initially second record**).

- The **fourth parameter record** with the **initially second field description record** satisfies **condition 3.3**. This happens because the **field description data type name** "*RA-InformationCommon*" is the **version-prefix** of the **enclosing data type** `RA-InformationCommon-r16` **and** the **field description parameter name** `perRAInfoList-v1660` is the **same** as the **parameter name**.

  Consequently, the **field description from** the **now first and only field description record** is **added** to the **fourth parameter record** and the **field description record** is **deleted**. The **resulting field descriptions list** is **empty**.

The **resulting parameter list** is:

- `perRAInfoList-r17` with **enclosing data type** `FailureReportSCG`, and whose **data type** is `PerRAInfoList-r16`.

- **perRAInfoList-r16** with **enclosing data type** **ConnEstFailReport-r16**, and whose **data type** is **PerRAInfoList-r16**.
- **perRAInfoList-r16** with **enclosing data type** **RA-InformationCommon-r16**, whose **data type** is **PerRAInfoList-r16** and its **field description** is "*This field provides detailed information about each of the random access attempts in the chronological order of the random access attempts. If perRAInfoList-v1660 is present, it shall contain the same number of entries, listed in the same order as in perRAInfoList-r16.*".
- **perRAInfoList-v1660** with **enclosing data type** **RA-InformationCommon-r16**, whose **data type** is **PerRAInfoList-v1660** and its **field description** is "*This field provides detailed information about each of the random access attempts in the chronological order of the random access attempts. If perRAInfoList-v1660 is present, it shall contain the same number of entries, listed in the same order as in perRAInfoList-r16.*".

**Step 4. Process possible remaining field description records**

This **step** is **not applied** because the **field descriptions list** is **empty**.

**Step 5. Attempt to add a field description to each parameter record lacking it**

In this example, there is no clear criterion to assign **either** of the **two existing field descriptions** (from **parameters three** and **four**) to the **two parameters** (**one** and **two**) that **lack** it.

**Step 6. Decide which parameter(s) from step 5 correspond to the parameter name that you are looking for**

Since you are looking for **parameter name** **perRAInfoList-v1660**, and the **fourth parameter record** is the **only one** with the **exact same parameter name**, you can **safely associate** the **fourth parameter record** to the **parameter name** that you are **looking for**. Therefore, in this particular example it is **not required** to **analyze** the **context** and **semantics**, as it is **required** in the **general case**.

Notice that if you were **looking** for **parameter name** **perRAInfoList-r16**, the whole procedure would have been **exactly the same** (because its **search-prefix** is the same as the one of **perRAInfoList-v1660**), until this **step 6**. In **step 6**, there would be **2 parameter records** (the **second** and the **third**) with exactly the **same parameter name** as the one you are **looking for**, but with two different enclosing data **types** and possibly **different semantics**. It would then be **required** to **analyze** the **context** and **semantics** of the **parameter name** that you are **looking for** and **comparing** them with the **semantics** of **two candidate parameter** records, **determine which one**(s) correspond to the **parameter name** that you are **looking for**.

### 5.7.3.2 Unexpected cases of field descriptions tables and field descriptions

The **expected** case of **field descriptions tables** in TS 38.331 is that **each data type** has **its fields description table** that **contains one entry** for **each** of **its enclosed parameters**.

However, TS 38.331 contains **many unexpected cases**, such as the following ones:

a) A **field descriptions table heading** has **more than one data type name**.

The **field descriptions** in the **table** correspond to **parameters** from **one or more** of the **heading**'s **data types** (or possibly from **version-different data types**).

For example, the table "*SRS-Resource, SRS-PosResource field descriptions*" has **2 data type names**. Its **field descriptions** correspond to **parameters** from **SRS-Resource** and/or from **SRS-PosResource-r16** (notice that the **latter** is a **version-different data type** from the second name in the **table heading**, and that the **data type SRS-PosResource** does **not exist**).

All the other examples that we have found are similar to the previous one:

- "*SRS-ResourceSet, SRS-PosResourceSet field descriptions*", where the **latter data type** does **not exist** and corresponds to **SRS-PosResourceSet-r16**.
- "*PUCCH-FormatConfig*, PUCCH-*FormatConfigExt field descriptions*", where the **latter data type** does **not exist** and corresponds to **PUCCH-FormatConfigExt-r17**.
- "*SRS-ResourceSet, SRS-PosResourceSet field descriptions*", where the **latter data type** does **not exist** and corresponds to **SRS-PosResourceSet-r16**.
- "*PUCCH-Resource, PUCCH-ResourceExt field descriptions*", where the **latter data type** does **not exist** and corresponds to **PUCCH-ResourceExt-v1610**.

b) A **data type lacks** its **field descriptions table** (i.e., **no table under** its **exact name**) but it has it under a **version-different name**

For example, **RA-InformationCommon-r16** **lacks** its **field descriptions table** for its **exact name**, but it has it under its **version-prefix name** as "*RA-InformationCommon field descriptions*". Notice that data type **RA-InformationCommon** does **not exist**.

As another (more extreme) example, the following **6 version-different data types lack their field descriptions table**:

- **RRCReconfigurationComplete-v1530-IEs**
- **RRCReconfigurationComplete-v1560-IEs**
- **RRCReconfigurationComplete-v1610-IEs**
- **RRCReconfigurationComplete-v1640-IEs**
- **RRCReconfigurationComplete-v1700-IEs**
- **RRCReconfigurationComplete-v1720-IEs**

while their **version-different data type** **RRCReconfigurationComplete-IEs**. has a **field descriptions table** which applies to **all of them**. It is worth pointing out that **all** of these **version-different data types** do **not** have **any parameter** with the **same name** (except for the generic **nonCriticalExtension**). However, the **table** provides **field descriptions** for **most** (**not all**) of their parameters. Curiously, even though the **table** is under the heading "*RRCReconfigurationComplete-IEs field descriptions*" it does **not** contain a **field description** for **any** of the **parameters** of **RRCReconfigurationComplete-IEs**.

c) A **data type lacks** its **field descriptions table** in the **whole** TS 38.331 (does **not** have it even under a **version-different name**)

For example, **data type SIB-TypeInfo** **lacks** its **field descriptions table** in the **whole** TS 38.331.

As another example, **data type** `PagingRecord-v1700`, lacks its **field descriptions table** while its **version-different data type** `PagingRecord` has it. Notice that this example is different from **case b**) above because the table "*PagingRecord field descriptions*" corresponds to `PagingRecord` and therefore the **table** does **not correspond** to `PagingRecord-v1700`. It could be the case that the **table** for `PagingRecord` contains **field descriptions** for **parameters enclosed** in `PagingRecord-v1700`. If this were the case, it would be an example of **case e**) below, but it still would **not** be an example of **case b**) above.

d) There is **more than one parameter name** in a **field description entry**.

The **field description** in that **entry** corresponds to **all** the **parameter names** (or possibly to **version-different names**) in the **field description entry**.

For example, the **field descriptions table** of `BWP-UplinkDedicated` (click 2.2.8.10) has a **field description entry** with the **2 parameter names** `cp-ExtensionC2` and `cp-ExtensionC3`, corresponding to the **2 version-different parameters** `cp-ExtensionC2-r16` and `cp-ExtensionC3-r16` (respectively).

There are similar examples for fields (`csi-RS-Index` and `csi-RS-Index-v1660`), (`s-SearchThresholdP` and `s-SearchThresholdP2`) and (`ntn-NeighCellConfigList` and `ntn-NeighCellConfigListExt`).

As a somehow extreme example, the **field descriptions table** of `CellGroupConfig` (click 3.7.3) has one entry with **4 parameter names**: `simultaneousU-TCI-UpdateList1`, `simultaneousU-TCI-UpdateList2`, `simultaneousU-TCI-UpdateList3` and `simultaneousU-TCI-UpdateList4`.

e) A **field descriptions table** of a **data type** contains a **field description** for a **parameter name not enclosed** (**nor** as **version-different name**) by the **data type** (not even as a version-different name)

Such a **field description** corresponds to a **parameter** (possibly under a **version-different name**) from some **other data type** (usually a **descendant data type**).

This case is the **reciprocal** of **case f**).

For example, the **field descriptions table** of `SchedulingInfo` includes a **field description** for `areaScope`, that is **not** a **parameter** enclosed in `SchedulingInfo`. This **field description** corresponds to parameter `areaScope` from `SIB-TypeInfo`. Notice that `SIB-TypeInfo` is a **descendant** of `SchedulingInfo`.

f) A **parameter lacks** its **field description** in the **table** of its **enclosing data type** but it has it under a **version-different parameter name**

For example, **parameter** `dmrs-BundlingPUCCH-Config-r17` **lacks** its **field description** in the **field descriptions table** of its **enclosing data type** `PUCCH-Config`. However, the said **table** contains a **field description** for its **version-different parameter** `dmrs-BundlingPUCCH-Config`, that is **not enclosed** in `PUCCH-Config`. Actually, `dmrs-BundlingPUCCH-Config` does **not exist**.

Recall that it is **also possible** to have a **field description** for the **complete parameter name** (i.e., including the **version-suffix**).

For example, the **parameters** `perRAInfoList-v1660`, `csi-RS-Index`, `csi-RS-Index-v1660`, `interFrequencyConfig-NoGap-r16` and `intra-SlotFH-r17` have a **field description** for their **complete name**.

g) A **parameter lacks** its **field description** in the **table** of its **enclosing data type** but it has it in the **table** of **other data type** (possibly under a **version-different parameter name**)

The **other data type** is usually an **ancestor** of the **enclosing data type**.

This case is the **reciprocal** of **case e**).

For example, parameter `areaScope` **lacks** its **field description** in the **table** of its **enclosing data type** `SIB-TypeInfo`, but it has it in the **table** of `SchedulingInfo`. Notice that `SchedulingInfo` is an **ancestor** of `SIB-TypeInfo`.

h) A **parameter name** has **multiple different field descriptions**, each in a **different field descriptions table**.

This should not be unexpected, because we already explained in 5.7.2.2 that **several different parameters** may have the **same parameter name**. However, we think it is worth pointing this out explicitly in this section.

For example, the **parameter name** `resourceMapping` has **2 different field descriptions** in the **tables** "*SRS-Resource, SRS-PosResource field descriptions*" and "*ZP-CSI-RS-Resource field descriptions*".

As another example, parameters `format1`, `format2` and `format3` from `PUCCH-Config` (click 7.9.4) have their **field descriptions** while `format1`, `format2` and `format3` from `PUCCH-Resource` (click 7.9.7) also have their **field descriptions**. As a comment, they have different data types: the former have **data type** `PUCCH-FormatConfig` (click 7.9.10), while the latter have **data types** `PUCCH-format1`, `PUCCH-format2` and `PUCCH-format3` (click 7.9.8), respectively.

As another (more extreme) example, the **parameter name** `nrofPRBS` has **5 different field descriptions** in the **tables** "*SSI-ResourceConfigCLI field descriptions*", "*PUCCH-format3 field descriptions*", "*PUCCH-Resource, PUCCH-ResourceExt field descriptions*", "*PUCCH-ConfigCommon field descriptions*", and "*CSI-RS-CellMobility field descriptions*".

i) A **parameter lacks** its **field description** in the **whole** TS 38.331.

For example, **parameters** `rach-ConfigSI` (from `SI-RequestConfig`), `schedulingInfoList` (from `SI-SchedulingInfo`) and `valueTag` (from `SIB-TypeInfo` ) **lack** their **field descriptions** in the **whole** TS 38.331.

Notice that there may be **combinations** of **more than one** individual case of the **above bullet points**. A few examples are:

- **Combination of cases c) and g)**

The data type `SI-SchedulingInfo-v1740` **lacks** its **field descriptions table**, in the **whole** TS 38.331, so this falls in **case c**) above. The **field description** of is **enclosed parameter** `si-RequestConfigRedCap-r17`, may be found in the **field descriptions table** of **another data type** (`SI-SchedulingInfo`), under a **version-different name** (`si-RequestConfigRedCap`), so this fall in **case g**) above. Notice

that **SI-SchedulingInfo** does **not** enclose parameter **si-RequestConfigRedCap**, nor **si-RequestConfigRedCap-r17**.

- **Combination of cases b) and d)**

  The **data type** **RA-InformationCommon-r16** **lacks** its **field descriptions table**, but it has it under its **version-different name** "*RA-InformationCommon*", so this falls in **case b)** above (notice that there is **no data type** **RA-InformationCommon**). The said **table** has a **field description entry** with the **2 field names** **perRAInfoList** and **perRAInfoList-v1660**, so it falls in **case d)** above. The **second field name** corresponds to parameter **perRAInfoList-v1660** enclosed in **RA-InformationCommon-r16**.

- **Combination of cases b), d) and f)**

  The **data type** **RA-InformationCommon-r16** **lacks** its **field descriptions table**, but it has it under its **version-different name** "*RA-InformationCommon*", so this falls in **case b)** above (notice that there is **no data type** **RA-InformationCommon**). The said **table** has a **field description entry** with the **2 field names** **perRAInfoList** and **perRAInfoList-v1660**, so it falls in **case d)** above. The **first field name** corresponds to its **version-different** parameter **perRAInfoList-r16** so it falls in **case f)** above.

- **Combination of cases b) and g)**

  The **data type** **PosSI-SchedulingInfo-r16** **lacks** its **field descriptions table**, but it has it under its **version-different name** **PosSI-SchedulingInfo**, so this falls in **case b)** above (notice that there is **no data type** **PosSI-SchedulingInfo**). The said **table** has a **field description entry** for **areaScope**, which is **not** a **parameter enclosed** in **PosSI-SchedulingInfo-r16**. The said **entry** applies to the **version prefix-parameter** **areaScope-r16** **enclosed** in **PosSIB-Type-r16**, so this falls in **case g)** above. Notice that **PosSI-SchedulingInfo-r16** is an **ancestor** of **PosSIB-Type-r16**.

Finally, in some cases it may be **assumed** that a **parameter** gets **its field description** from the **field description** of **another parameter**: see a description in 5.7.3.3.

### 5.7.3.3 A parameter has the same field description as another parameter

Section 5.7.3.2 presented some **unexpected cases** on **where/how** to find the **field description** of a **given parameter**. This section **rather** presents **cases** where it may be **assumed** (or **not**) that a **parameter** hast the **same field description** as **another existing parameter**.

Some cases where it **can** be **assumed** that a **parameter** has the **same field description** as **another existing parameter** are:

- A **field description explicitly** indicates that it is **applicable** to another **parameter**

  For example, the **field description** of **searchSpaceSetZero** and of **controlResourceSetZero**, both from **PDCCH-ConfigCommon** (click 6.12.4), explicitly indicates that they have the **same semantics** as their respective **same-name parameters** from **PDCCH-ConfigCommonSIB1**, within **MIB** (click 5.3.3).

- A **parameter lacks** a **field description** for its **full name** and it is **assumed** that it is the **same** as the one of a given **version-different parameter**

- **Two different same-name parameters** are **assumed** to have the **same semantics**

Some cases where it **cannot** be **assumed** that a **parameter** has the **same field description** as **another existing parameter** are:

- **Same-name or version-different parameters** with **different data type**

  For example, there is **no field description** for `downlinkConfigCommon` in the **table** of its **enclosing data type** `ServingCellConfigCommonSIB` (click 3.9.3). At the same time, there is a **field description** for its **same-name parameter** from `ServingCellConfigCommon` (click 3.9.6). However, the **data type** of the **former** is `DownlinkConfigCommonSIB`, while the **data type** of the **latter** is `DownlinkConfigCommon`, which has several differences. They have **different data types**, and therefore the **field description** of the **latter cannot** be applied to the **former**.

- **Same-name or version-different parameters** with the **same or similar data type** but **different semantics**

  For example, `nrofSymbols` **lacks** its **field description** in the **table** of its **enclosing data type** `SRS-Resource`, which is "*SRS-Resource, SRS-PosResource field descriptions*". At the same time, there is a **field description** for `nrofSymbols-r16` from `RSSI-ResourceConfigCLI-r16`. Their data type is similar (an `INTEGER` with an overlapping range). However, their **semantics** is **different**, so it **cannot** be **assumed** that **field description** of the **latter** is the **same** as the one of the **former**. Notice that the **field description** for `nrofSymbols-r16` is under its **version-prefix name** `nrofSymbols`, and that the table for its **enclosing data type** `RSSI-ResourceConfigCLI-r16` is under its **version-prefix name** `RSSI-ResourceConfigCLI`.

### 5.7.3.4 Semantics of same-name and version-different parameters

Depending on the **data type** of **parameters** within a **set** of **same-name** and/or **version-different parameters** (click 5.7.2.2), we have the following **three cases**:

- **Different data types**

  **Data types** are **considered different** when the **values** they may hold are **sufficiently different**.

  For an example, click 5.7.3.4.1.

  When the **parameters** have **different data types** they have **different semantics**.

- **Similar data types**

  **Data types** are **considered similar** when the **values** they may hold are a **subset** of **one another**.

  For some examples, click 5.7.3.4.2, 5.7.3.4.3, 5.7.3.4.6 and 5.7.3.4.7.

- **Same data types**

  **Data types** are **considered** the **same** when their **ASN.1 specification** is completely equivalent (i.e., they can **hold exactly** the **same values**).

  For some examples, click 5.7.3.4.4 and 5.7.3.4.5.

Deciding if **data types** are **different**, **similar** of the **same** has some degree of **subjectivity**. For example, a `BOOLEAN`, an `INTEGER(0..1)` and an `ENUMERATED{n0, n1}` can be considered **different**, **similar** and even the **same**, depending on **context**.

Regarding the **data type** of parameters, recall from 5.9.9 that it can be a **containing data type**. This is, an `OCTET STRING (CONTAINING DataType)`. We consider this case as if the **data type** of the parameter would be `DataType`.

Depending on the **semantics** of **parameters** within a **set** of **same-name** and/or **version-different parameters** (click 5.7.2.2), we have the following **three cases**:

- **Different semantics**

  Having **different semantics**, there is a **low** potential for **confusion** between the **parameters** in the **set**. Since they have **different semantics**, it is **reasonably clear** that they are **different parameters**.

  When the **parameters** have **different data types** they have **different semantics**.

  For some examples, click 5.7.3.4.1, 5.7.3.4.2 and 5.7.3.4.6.

- **Similar semantics**

  Having **similar semantics**, there is a **substantial** potential for **confusion** between the **parameters** in the **set**. Since their semantics is **similar** it is **reasonably easy** to get **confused** between them. Even if being aware that they are **different parameters**, it is **easy** to **mistakenly** consider that they have the **same semantics**.

  For some examples, click 5.7.3.4.3, 5.7.3.4.4 and 5.7.3.4.7.

- **Same semantics**

  Having the **same semantics**, there is a **high** potential for **confusion** between the **parameters** in the **set**. Since they have the **same semantics**, you may **mistakenly** believe that they are the **same parameter**, when they are actually **different** parameters.

  For an example, click 5.7.3.4.5.

  You may think that them being the same or different parameters is irrelevant, because they have the same semantics, but it is **not irrelevant**. Since **different parameters** are **located** in **different places** of the **ASN.1 specification**, even having the **same semantics** they most likely have **relevant differences**. A few examples of possible **relevant differences** between the **parameters** within such a **set** are: some may be **common** while **others** are **dedicated**; some may be **optional** while **others** are **mandatory**; they may have a **different quantity** of **instances**.

The combination of the **name relationship** (click 5.7.2.2) and the **two categorizations** performed above leads to **12 possible cases** presented in the following **table** (where you can click on **a few** cells to see an **example** of the corresponding case):

| | | Parameters' data type and semantics | | | | | |
|---|---|---|---|---|---|---|---|
| | | Different data type | Similar data type | | Same data type (inc. CONTAINING) | | |
| | | Different semantics | Different semantics | Similar semantics | Different semantics | Similar semantics | Same semantics |
| Parameters' Names | Same-name | 5.7.3.4.1 | 5.7.3.4.2 | 5.7.3.4.3 | | 5.7.3.4.4 | 5.7.3.4.5 |
| | Version-different name | | 5.7.3.4.6 | 5.7.3.4.7 | | | |

Notice that you may have **hybrid cases** corresponding to two or more of the **individual cases** presented in this table.

For example, the **5 same-name parameters** `nrofSymbols` from `PUCCH-format0` to `PUCCH-format4` (click 7.9.8) have **similar data types** (`INTEGER` with ranges `(1..2)`, `(4..14)`, `(1..2)`, `(4..14)` and `(4..14)`) and **similar semantics** (**starting symbol** of **PUCCH** `formats`), click 5.7.3.4.3. At the same time, they have **another same-name parameter** from `SRS-Resource`, with a **similar data type** (`ENUMERATED{n1, n2, n4}`) and **different semantics** (**SRS** mapping), click 5.7.3.4.2. Furthermore, they **also** have a **version-different parameter** `nrofSymbols-r16` from `RSSI-ResourceConfigCLI-r16` with **similar data type** (`INTEGER(1..14)`) and **different semantics** (quantity of symbol intervals for a **CLI-RSSI** resource), click 5.7.3.4.6.

Another example is `nrofPRBs`, which has **several same-name parameters** and several **version-different parameters** in the **same** and **different contexts**.

Another example is `subcarrierSpacing` (click 2.2.10.5).

Notice that the question of **two parameters** having **different**, **similar** or the **same semantics** is somehow subjective, and **not obvious** (click 5.7.3).

The **very important** conclusion is that **great attention** should be paid to **avoid confusing different parameters**, **in particular**, when they have **strong similarities**.

#### 5.7.3.4.1 Example of same-name parameters with different data types and different semantics

The **2 same-name parameters** `format1` from `PUCCH-Config` (click 7.9.4) and from `PUCCH-Resource` (click 7.9.7). The **former parameter** has **data type** `PUCCH-FormatConfig` (click 7.9.10) and the **latter parameter** has **data type** `PUCCH-format1` (click 7.9.8). These two **data types** are **clearly different** (you may compare their **ASN.1 specifications**). Therefore, **both parameters** have **different data types** and **different semantics**.

#### 5.7.3.4.2 Example of same-name parameters with similar data types and different semantics

The **6 same-name parameters** `nrofSymbols` from `PUCCH-format0` to `PUCCH-format4` (click 7.9.8) and from `SRS-Resource`. These **6 parameters** have **similar data types**: `INTEGER` with ranges `(1..2)`, `(4..14)`, `(1..2)`, `(4..14)` and `(4..14)`, plus

**ENUMERATED{n1, n2, n4}** (respectively). The **first 5 parameters** have the **same semantics** among themselves (click 7.9.8) but **different** in respect to the **sixth parameter**. The **semantics** of the **first 5 parameters** is the **starting symbol** of the **5 respective PUCCH formats** while the **semantics** of the **sixth parameters** is **SRS mapping**.

#### 5.7.3.4.3 Example of same-name parameters with similar data types and similar semantics

The **5 same-name parameters** **nrofSymbols** from **PUCCH-format0** to **PUCCH-format4** (click 7.9.8). These **5 parameters** have **similar data types**: **INTEGER** with ranges **(1..2)**, **(4..14)**, **(1..2)**, **(4..14)** and **(4..14)** (respectively). These **5 parameters** have **similar semantics**: the **starting symbol** of the **5 respective PUCCH formats**.

#### 5.7.3.4.4 Example of same-name parameters with same data type and similar semantics

The **2 same-name parameters** **useInterlacePUCCH-PUSCH-r16** from **BWP-UplinkCommon** (click 2.2.8.9) and from **BWP-UplinkDedicated** (click 2.2.8.10). These **2 parameters** have the **same data type** **ENUMERATED {enabled}**. The **semantics** of these **2 parameters** is **similar**, but **not the same**. Moreover, they have a **relevant difference**: the **former parameter** is **common** while the **latter parameter** is **dedicated**.

#### 5.7.3.4.5 Example of same-name parameters with same data type and same semantics

The **2 same-name parameters** **controlResourceSetId** from **ControlResourceSet** (click 6.12.6) and from **SearchSpace** (click 6.12.8). These **2 parameters** have the **same data type** **ControlResourceSetId** (click 6.12.7) and the **same individual semantics**: both **parameters** identify a **CORESET**.

In spite of having the **same individual semantics**, they have a **relevant difference**: the **former** identifies the **CORESET number** to its **containing CORESET specification**, while the **latter** identifies the **CORESET number** to which is **associated** its **containing search space specification**.

#### 5.7.3.4.6 Example of version-different parameters with similar data types and different semantics

The **2 version-different parameters** **nrofSymbols** from **SRS-Resource** and **nrofSymbols-r16** from **RSSI-ResourceConfigCLI-r16**. The **former parameter** has **data type** **ENUMERATED{n1, n2, n4}** and the **latter parameter** has **data type** (**INTEGER(1..14)**, which can be considered **similar**. However, the **semantics** of these **2 parameters** is **different**: the **former parameter** is **SRS mapping** while the **latter parameter** is the **quantity of symbol intervals** for a **CLI-RSSI** resource

#### 5.7.3.4.7 Example of version-different parameters with similar data types and similar semantics

The **3 version-different parameters** **controlResourceSetId** and **controlResourceSetId-v1610** **both** from **ControlResourceSet** (click 6.12.6) and **controlResourceSetId-r16** from **SearchSpaceExt-r16**. The **first parameter** has **no suffix** (indicating that it was present since the first **5G** release **R15**),

the **second parameter** has **suffix** "**-v1610**" (indicating that it was introduced in version **16.10**) and the **third parameter** has **suffix** "**-r16**" (indicating that it was introduced in **R16**).

The **first parameter** has **data type** **ControlResourceSetId** (click 6.12.7), the **second parameter** has **data type** **ControlResourceSetId-v1610** (click 6.12.7) and the **third parameter** has **data type** **ControlResourceSetId-r16** (click 6.12.7). These **3 data types** are an **INTEGER** with ranges **(0..11)**, **(0..15)** and **(12..15)** (respectively).

These **3 version-different parameters** have **similar data types** and **similar semantics**: the three of them **identify** a **CORESET**. The **difference** between their **semantics**, is that each of them **identifies** a **CORESETs** for **different operation** configurations.

### 5.7.3.5 RRC semantic qualification of UE parameter instances

The **parameter identification** (click 5.7.2.1) and **semantics** (click 5.7.3) is associated to the pair of **the parameter name** and its **enclosing data type name**. The **parameter semantics applies equally** to **each** and **every** of its **parameter instances** (click 5.7.2.3).

In addition to the **semantics** of a parameter, **each** and **every** of its **instances** has some **associated semantic qualification** over the **parameter semantics**. This **semantic qualification** of **each parameter instance** depends on the **location**, **within** an **ancestor tree**, where it has been instantiated.

Let us illustrate this with a simple example over the **optional parameter** **bwp-Dedicated** from **BWP-Uplink** (click 2.2.8.8). Its data type is **BWP-UplinkDedicated** (click 2.2.8.10). **Part** of the **ASN.1 specification** of **BWP-Uplink** is:

```
BWP-Uplink ::= SEQUENCE {
  other parameters (not shown)
  bwp-Dedicated BWP-UplinkDedicated OPTIONAL,          -- Cond SetupOtherBWP
  other parameters (not shown)
}
```

The **intrinsic semantics** of **bwp-Dedicated** consists of providing **many dedicated configuration parameters** for an **Uplink bandwidth part**. This **semantics** is **kept no matter where** you instantiate it.

The parameter **bwp-Dedicated** is **instantiated** when its **enclosing data type** **BWP-Uplink** is **instantiated**. The said **data type** is **instantiated** in **uplinkBWP-ToAddModList** from **UplinkConfig** (click 2.2.8.4). The **data type** of **uplinkBWP-ToAddModList** is a **SEQUENCE OF** **BWP-Uplink**. To show this, we copy here **part** of the **ASN.1 specification** of **UplinkConfig** is:

```
UplinkConfig ::= {
  other parameters (not shown)
  uplinkBWP-ToAddModList SEQUENCE (SIZE (1..4)) OF BWP-Uplink
  other parameters (not shown)
}
```

The **instantiating parameter** **uplinkBWP-ToAddModList** is **optional** and contains a **SEQUENCE OF** **BWP-Uplink**, with **1** to **4** elements. Therefore, the **instantiating parameter** contains from **0** to **4** **parameter instances** of **bwp-Dedicated**. **Each** of these **instances** is **associated** to **one specific Uplink additional bandwidth part**. Therefore, the **semantic qualification** of each **bwp-Dedicated** **instance** is that it provides the **dedicated configuration parameters** of **the specific Uplink additional bandwidth part** where it is **contained**.

Furthermore, parameter **uplinkBWP-ToAddModList** from **UplinkConfig** (click 2.2.8.4) is instantiated **twice**: in parameters **uplinkConfig** and **supplemenaryUplink**, both from **ServingCellConfig**, (click 3.9.7). **Part** of the specification of **ServingCellConfig** is:

```
ServingCellConfig ::= SEQUENCE {
  other parameters (not shown)
  uplinkConfig            UplinkConfig OPTIONAL,                -- Need M
  supplementaryUplink     UplinkConfig OPTIONAL,                -- Need M
  other parameters (not shown)
}
```

As you may see, it is **optionally** possible to **configure from 0 to 4 Uplink bandwidth parts** for a **normal Uplink channel** and/or for a **SUL channel**. Therefore, the **semantic qualification** of each **bwp-Dedicated instance** is that it provides the **dedicated configuration parameters** of **the specific Uplink** and/or **SUL additional bandwidth part** where it is **contained**.

Finally, regarding the **specified/default values** for **UE** parameters, click 5.7.3.7.

### 5.7.3.6 RRC semantics of ASN.1 comments

**Comments** in **ASN.1** (by definition) **do not** have **semantics** (click 5.9.9). However, in the **TS 38.331 ASN.1 specifications**, some comments **do have semantics**, as follows:

- **Comment** "`-- Need R`": **Release**

  If the **parameter** is **absent**, the **UE releases** the **current value**. It is used for **parameters** whose value is **stored** by the **UE** (i.e., **not one-shot** parameters).

- **Comment** "`-- Need M`": **Maintain**

  If the **parameter** is **absent**, the **UE maintains** the **current value**. It is used for **parameters** whose value is **stored** by the **UE** (i.e., **not one-shot** parameters).

- **Comment** "`-- Need N`": **No action**

  If the **parameter** is **absent**, the **UE** takes **no action**. If the **parameter** is **present**, it refers to **one-shot configuration** that is **not maintained**. It is used for **parameters** whose value is **not stored** and whose presence causes a **one-shot** action by the **UE**.

- **Comment** "`-- Need S`": **Specified**

  If the **parameter** is **absent**, the **parameter description** or **procedure specifies** the **UE behavior**. Notice that it is **not** the **same** that the **parameter** is **absent** than the **parameter** is **not configured**.

- **Comment** "`-- Cond conditionTag`": **Conditionally present**

  **Presence** of the parameter is **specified** in a **table** placed after the **ASN.1** segment.

- **Comment** "`-- CondC conditionTag`": **Configuration condition**

  **Presence** of the parameter is **conditional** to **other** configuration **settings**.

- **Comment** "`-- CondM conditionTag`": **Message condition**

  **Presence** of the parameter is **conditional** to **other parameters** included **in** the same **message**.

Notice that in TS 38.331, "`CondC`" and "`CondM`" are not used.

For more detail on **RRC semantics** of **ASN.1 comments**, refer to clause 6 and annexes A6 and A7 of TS 38.331.

#### 5.7.3.7 RRC specified and default radio configuration

**Specified** and **default radio configurations** consist of the **default values** for **UE physical layer parameters** that are listed in clause 9 of TS 38.331.

**Specified radio configurations** are **fixed** while **default radio configurations can** be **modified** using **dedicated signaling**.

For the case of **UE parameters not listed** in clause 9 of TS 38.331, their **default value** shall be set such as the corresponding features are not configured (i.e., `release` or `false`) unless explicitly stated otherwise.

If an **optional parameter** has a **`DEFAULT` value** (click 5.9.3) in its **ASN.1** specification, the **UE only** applies it **if** the **enclosing parameter** (i.e., an instance of the enclosing data type) is **present**. Otherwise, the **UE** applies the **default value** from the **specified** and **default configurations**, as indicated above. One example is the **optional** parameter `q-OffsetFreq` from, `InterFreqCarrierFreqInfo`, whose **ASN.1 specification** indicates the **`DEFAULT`** value `dB0`.

When a **parameter** has a **default value** indicated in its **field description** (click 5.7.3), the **UE only** applies it **if** the **enclosing parameter** (i.e., an instance of the enclosing data type) is **present**. Otherwise, the **UE** applies the **default value** from the **specified** and **default configurations**, as indicated above. One example is `txDirectCurrentLocation` from `SCS-SpecificCarrier` (click 2.2.6.12), whose **field description** indicates the default value of `3,300`.

Notice that the **default radio configurations only** list default values for parameters up to TS 38.331 version **V15.3.0**. Therefore, for **all** parameters introduced in **later TS versions**, their default value is `release` or `false` unless explicitly specified otherwise. If the **UE** is to apply **default configuration** while it is configured with some critically extended parameters, the **UE** shall apply the **original version** of those parameters with only default values. In general, the signaling should preferably support a `release` option for fields introduced **after** version **V15.3.0**. The "**value not applicable**" should be used restrictively, mainly limited to for fields which value is relevant only if another field is set to a value other than its default. For parameters in `ServingCellConfig`, their default values are specified in the corresponding specification instead of in clause 9.2 of TS 38.331.

For more detail on **specified** and **default UE configurations**, refer to clause 9 of TS 38.331.

### 5.7.4 `SIB1` data type

For its **delivery mode**, **ancestor trees** and **other characteristics**, click 5.3.5.

**All** its parameters are **common** (click 5.1.3).

It can be an **RRC message** of the `BCCH-DL-SCH-Message` **class** (click 5.8.5.2) or a **data type** contained **within other RRC messages** (possibly as an **`OCTET STRING`**).

It contains **many relevant common parameters** for **UE configuration**.

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_Mib_Sib.html
- https://www.techplayon.com/5g-nr-system-information-type1-sib-1/
- https://www.linkedin.com/pulse/5g-cell-selection-process-bikas-singh/
- https://www.techplayon.com/5g-nr-cell-global-identity-planning/
- http://howltestuffworks.blogspot.com/2019/10/5g-nr-system-information-block-1-sib1.html

#### 5.7.4.1 SIB1 parameters

The **ancestor tree** (click 5.9.8) up to **SIB1** contains around a **hundred instances** of **common** (click 5.1.3) parameters that are **received** by **every**. These parameters **configure** the following **elements**:

- **Carrier parameters** (click 2.2.6.2)
- **Initial bandwidth part parameters** (click 2.2.8.1)
- **Network, node and cell identity parameters** (click 3.6.1)
- **Serving cell parameters** (click 3.9.1)
- **Other SIBs and posSIBs scheduling parameters** (click 5.5.3)
- **PDCCH parameters** (click 6.12.2)
- **PDSCH parameters** (click 6.18.2)
- **PUCCH parameters** (click 7.9.1)
- **PUSCH parameters** (click 7.15.2)
- **Initial access parameters** (click 8.11).
- **Random access procedure** parameters (click 9.5.1).

Notice that the **parameters** within **SIB1** (click 5.3.4), within **other SIBs** (click 5.3.5) and within **posSIBs** (click 5.3.5) do **not** contain **any** parameter with **data type** **PhysCellId** to convey the **PCI** (click 3.5.3.2) of the **cell** to which they **apply**, because they **always apply** to the **cell** where they are **received**. However, **ServingCellConfigCommon** (click 3.9.6) **contains** parameter **physCellId** with **data type** **PhysCellId** to convey the **PCI** (click 3.5.3.2) of the **cell** to which its parameters **apply**. **Each** instance of **ServingCellConfigCommon** contains the **common configuration parameters** for **one specific cell**. The **data type** **CellGroupConfig** (click 3.7.3) contains several instances of **ServingCellConfigCommon**.

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- **cellSelectionInfo** **SEQUENCE**: optional, common

  It contains several **minimum threshold** parameters for **cell search** and **cell access**.

  Its **data type** is a **SEQUENCE** with **several** parameters. **Some** of them are **q-RxLevMin** and **q-QualMin** (see below).

- `cellAccessRelatedInfo CellAccessRelatedInfo` (click 3.6.4): mandatory, common

  It does **not** have a **field description** in TS 38.331, but there is some information about it in the **running text**.

  It is then **assumed** that it provides several **identifier** parameters for **cell search** and **cell access**.

- `servingCellConfigCommon ServingCellConfigCommonSIB` (click 3.9.3): optional, common

  It configures **every UE** with some **common parameters** during their **initial access** to their **primary cell**.

- `q-QualMin Q-QualMin`: optional, common

  It is used to determine if the **quality** is **high enough** to consider the **cell selection** procedure **successful** (click 8.1). It corresponds to "$Q_{qualmin}$" in TS 38.304.

  Its **data type** is an `INTEGER(-43..-12)`.

  If `q-QualMin` is **absent**, the **UE** applies the value of **negative infinity**.

- `q-RxLevMin Q-RxLevMin`: mandatory (within `OPTIONAL`), common

  It is used to determine if the **incoming power** is **high enough** to consider the **cell selection** procedure **successful** (click 8.1). It corresponds to "$Q_{rxlevmin}$" in TS 38.304.

  Its **data type** is an `INTEGER(-70..-22)`.

- `si-SchedulingInfo SI-SchedulingInfo` (click 5.5.3.1): optional, common

  It indicates whether **other SIBs** and **posSIBs** are delivered via **periodic broadcast** or **only** requested **on-demand** (click 5.5.1). It also indicates their availability and **scheduling** (e.g., **periodicity**, **SI-window** size). It also provides information for the **UE** to perform requests for **other SIBs** and **posSIBs**.

  If it indicates **no on-demand System Information**, the **UE** will continue decoding the **other SIBs** received by **broadcast**.

  If it indicates **on-demand** System Information, the **UE** action depends on its **RRC state**:

  - If it is **RRC_IDLE** or **RRC_INACTIVE** (click 5.8.2)

    Instruct the lower layer to initiate a **random access procedure** (click 9). Acquire the **requested SI messages** when the **acknowledgement** for the **SI request** is received.

  - If it is **RRC_CONNECTED** (click 5.8.2)

    The process is not defined yet.

For more detail on these parameters, refer to the table "*SIB1 field descriptions*" of TS 38.331.

#### 5.7.4.2 SIB1 ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
SIB1 ::= SEQUENCE {
  cellSelectionInfo SEQUENCE {
    q-RxLevMin Q-RxLevMin,
    q-RxLevMinOffset INTEGER (1..8) OPTIONAL,                    -- Need S
    q-RxLevMinSUL Q-RxLevMin OPTIONAL,                           -- Need R
    q-QualMin Q-QualMin OPTIONAL,                                -- Need S
    q-QualMinOffset INTEGER (1..8) OPTIONAL                      -- Need S
  } OPTIONAL,                                                 -- Cond Standalone
  cellAccessRelatedInfo CellAccessRelatedInfo,
  connEstFailureControl ConnEstFailureControl OPTIONAL,          -- Need R
  si-SchedulingInfo SI-SchedulingInfo OPTIONAL,                  -- Need R
  servingCellConfigCommon ServingCellConfigCommonSIB OPTIONAL,   -- Need R
  ims-EmergencySupport ENUMERATED {true} OPTIONAL,               -- Need R
  eCallOverIMS-Support ENUMERATED {true} OPTIONAL,               -- Need R
  ue-TimersAndConstants UE-TimersAndConstants OPTIONAL,          -- Need R
  uac-BarringInfo SEQUENCE {
    uac-BarringForCommon    UAC-BarringPerCatList OPTIONAL,      -- Need S
    uac-BarringPerPLMN-List UAC-BarringPerPLMN-List OPTIONAL,    -- Need S
    uac-BarringInfoSetList  UAC-BarringInfoSetList,
    uac-AccessCategory1-SelectionAssistanceInfo CHOICE {
      plmnCommon UAC-AccessCategory1-SelectionAssistanceInfo,
      individualPLMNList SEQUENCE (SIZE (2..12)) OF
           UAC-AccessCategory1-SelectionAssistanceInfo OPTIONAL  -- Need S
    }
  } OPTIONAL,                                                    -- Need R
  useFullResumeID ENUMERATED {true} OPTIONAL,                    -- Need R
  lateNonCriticalExtension OCTET STRING OPTIONAL,
  nonCriticalExtension SIB1-v1610-IEs OPTIONAL
}
```

### 5.7.5 SystemInformation data type

For its **delivery mode**, **ancestor trees** and **other characteristics**, click 5.3.5.

**All** its parameters are **common** (click 5.1.3).

It can be an **RRC message** of the **BCCH-DL-SCH-Message** **class** (click 5.8.5.2) or a **data type** contained **within other RRC messages** (possibly as an **OCTET STRING**).

**Each** of its **instances** can contain **either several other SIBs** (i.e., **SIBS** other than **SIB1**) **or several posSIBs** (but **not** a combination of **both**).

#### 5.7.5.1 SystemInformation parameters

Notice that **SIB1** (click 5.3.4), **other SIBs** (click 5.3.5) and **posSIBs** (click 5.3.5) do **not** contain **any** parameter with **data type** **PhysCellId** to convey the **PCI** (click 3.5.3.2) of the **cell** to which their parameters **apply**, because they **always apply** to the **cell** where they are **received**. However, **ServingCellConfigCommon** (click 3.9.6) **contains** parameter **physCellId** with **data type** **PhysCellId** to convey the **PCI** (click 3.5.3.2) of the **cell** to which its parameters **apply**. **Each** instance of **ServingCellConfigCommon** contains the **common configuration parameters** for **one specific cell**. The **data type** **CellGroupConfig** (click 3.7.3) contains several instances of **ServingCellConfigCommon**.

This **data type** does **not** have a **table** of **field descriptions** in TS 38.331.

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `posSystemInformation-r16` `PosSystemInformation-r16-IEs` (click 5.5.3): mandatory (within a `CHOICE`), common

  It contains **one** or **more other posSIBs** that are sent in the **RRC** `SystemInformation` **message**.

- `systemInformation` `SystemInformation-IEs` (click 5.7.6): mandatory (within a `CHOICE`), common

  It contains **one** or **more other SIBs** that are sent in the **RRC** `SystemInformation` **message**.

#### 5.7.5.2 `SystemInformation` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
SystemInformation ::=  SEQUENCE {
  criticalExtensions CHOICE {
    systemInformation SystemInformation-IEs,
    criticalExtensionsFuture-r16 CHOICE {
      posSystemInformation-r16 PosSystemInformation-r16-IEs,
      criticalExtensionsFuture SEQUENCE {}
    }
  }
}
```

### 5.7.6 `SystemInformation-IEs` data type

For its **ancestor trees**, click 5.3.5.

**All** its parameters are **common** (click 5.1.3).

Its **only parent data type** is `SystemInformation` (click 5.7.5).

It contains **other SIBs** (i.e., other than `SIB1`).

This **data type** is a `SEQUENCE OF 1` to `maxSIB` parameters, where **each** such **parameter** is a `CHOICE` between **all** the **other SIBs**.

The **constant** `maxSIB` is `32`, which is the **maximum quantity** of **other SIBs** that can be contained in a single `SystemInformation` **instance**.

#### 5.7.6.1 `SystemInformation-IEs` parameters

This **data type** does **not** have a **table** of **field descriptions** in TS 38.331.

Some of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `sib2` `SIB2`: optional, common

  It contains **cell re-selection parameters**, mainly related to the serving cell.

- `sib3` `SIB3`: optional, common

  contains parameters about the **serving frequency** and **intra-frequency neighboring cells** relevant for cell re-selection (including cell re-selection parameters common for a frequency as well as cell-specific re-selection parameters).

- `sib4 SIB4`: optional, common

  It contains parameters about **other NR frequencies** and **inter-frequency neighboring cells** relevant for cell re-selection (including cell re-selection parameters common for a frequency as well as cell-specific re-selection parameters), which can also be used for **NR** idle/inactive measurements.

- `sib5 SIB5`: optional, common

  It contains parameters about **E-UTRA** frequencies and **E-UTRA** neighboring cells relevant for cell re-selection (including cell re-selection parameters common for a frequency as well as cell-specific re-selection parameters).

- `sib6 SIB6`: optional, common

  It contains an **ETWS** primary notification.

- `sib7 SIB7`: optional, common

  It contains an **ETWS** secondary notification.

- `sib8 SIB8`: optional, common

  It contains a **CMAS** warning notification.

- `sib9 SIB9`: optional, common

  It contains parameters related to **GPS** time and **Coordinated Universal Time** (**UTC**).

- `sib10 SIB10-r16`: optional, common

  It contains the **Human-Readable Network Names** (**HRNN**) of the **NPNs** listed in `SIB1`.

- `sib11 SIB11-r16`: optional, common

  It contains parameters related to **idle/inactive measurements**.

- `sib15 SIB15-r17`: optional, common

  It contains parameters related to **disaster roaming**.

- `sib16 SIB16-r17`: optional, common

  It contains **slice-based cell re-selection** parameters.

- `sib17 SIB17-r17`: optional, common

  It contains parameters related to **TRS** configuration for **UEs** in **RRC_IDLE** or in **RRC_INACTIVE** (click 5.8.2).

- `sib18 SIB18-r17`: optional, common

  It contains parameters related to the **Group IDs for Network selection** (**GINs**) associated with **SNPNs** listed in `SIB1`.

For **Sidelink**, **other system information** also **includes**:

- `sib12 SIB12-r16`: optional, common

  It contains parameters related to **NR Sidelink** communication.

- **sib13 SIB13-r16**: optional, common

  It contains parameters related to SystemInformationBlockType21 for **V2X Sidelink** communication as specified in clause 5.2.2.28 of TS 36.331.

- **sib14 SIB14-r16**: optional, common

  It contains parameters related to SystemInformationBlockType26 for **V2X Sidelink** communication as specified in clause 5.2.2.33 of TS 36.331.

For **non-terrestrial** network, **other system information** also **includes**:

- **sib19 SIB19-r17**: optional, common

  It contains **NTN-specific** parameters for serving cell and optionally **NTN-specific** parameters for neighbor cells as defined in TS 38.331.

For **MBS broadcast**, **other system information** also **includes**:

- **sib20 SIB20-r17**: optional, common

  It contains **MCCH** configuration.

- **sib21 SIB21-r17**: optional, common

  It contains parameters related to **service continuity** for **MBS** broadcast reception.

For more detail on **system information blocks**, refer to clause 6.3.1 of TS 38.331.

#### 5.7.6.2 SystemInformation-IEs ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
SystemInformation-IEs ::= SEQUENCE {
  sib-TypeAndInfo SEQUENCE (SIZE (1..maxSIB)) OF CHOICE {
    sib2 SIB2,
    sib3 SIB3,
    sib4 SIB4,
    sib5 SIB5,
    sib6 SIB6,
    sib7 SIB7,
    sib8 SIB8,
    sib9 SIB9,
    ...,
    sib10-v1610 SIB10-r16,
    sib11-v1610 SIB11-r16,
    sib12-v1610 SIB12-r16,
    sib13-v1610 SIB13-r16,
    sib14-v1610 SIB14-r16,
    sib15-v1700 SIB15-r17,
    sib16-v1700 SIB16-r17,
    sib17-v1700 SIB17-r17,
    sib18-v1700 SIB18-r17,
    sib19-v1700 SIB19-r17,
    sib20-v1700 SIB20-r17,
    sib21-v1700 SIB21-r17
  },
  lateNonCriticalExtension OCTET STRING OPTIONAL,
  nonCriticalExtension SEQUENCE {} OPTIONAL
}
```

### 5.7.7 PosSystemInformation-r16-IEs data type

For its **ancestor trees**, click 5.3.5.

**All** its parameters are **common** (click 5.1.3).

Its **only parent data type** is `SystemInformation` (click 5.7.5).

It contains **posSIBs**.

### 5.7.7.1 `PosSystemInformation-r16-IEs` parameters

This **data type** does **not** have a **table** of **field descriptions** in TS 38.331.

This **data type** is a `SEQUENCE OF 1` to `maxSIB` parameters, where **each** such **parameter** is a `CHOICE` between **all** the **posSIBs**.

The **constant** `maxSIB` is `32`, which is the **maximum quantity** of **posSIBs** that can be contained in a single `SystemInformation` **instance**.

For more detail on **positioning SIBs**, refer to clause 6.3.1a of TS 38.331 and to TS 37.355.

### 5.7.7.2 `PosSystemInformation-r16-IEs` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
PosSystemInformation-r16-IEs ::= SEQUENCE {
  posSIB-TypeAndInfo-r16 SEQUENCE (SIZE (1..maxSIB)) OF CHOICE {
    posSib1-1-r16     SIBpos-r16,
    posSib1-2-r16     SIBpos-r16,
    posSib1-3-r16     SIBpos-r16,
    posSib1-4-r16     SIBpos-r16,
    posSib1-5-r16     SIBpos-r16,
    posSib1-6-r16     SIBpos-r16,
    posSib1-7-r16     SIBpos-r16,
    posSib1-8-r16     SIBpos-r16,
    posSib2-1-r16     SIBpos-r16,
    posSib2-2-r16     SIBpos-r16,
    posSib2-3-r16     SIBpos-r16,
    posSib2-4-r16     SIBpos-r16,
    posSib2-5-r16     SIBpos-r16,
    posSib2-6-r16     SIBpos-r16,
    posSib2-7-r16     SIBpos-r16,
    posSib2-8-r16     SIBpos-r16,
    posSib2-9-r16     SIBpos-r16,
    posSib2-10-r16    SIBpos-r16,
    posSib2-11-r16    SIBpos-r16,
    posSib2-12-r16    SIBpos-r16,
    posSib2-13-r16    SIBpos-r16,
    posSib2-14-r16    SIBpos-r16,
    posSib2-15-r16    SIBpos-r16,
    posSib2-16-r16    SIBpos-r16,
    posSib2-17-r16    SIBpos-r16,
    posSib2-18-r16    SIBpos-r16,
    posSib2-19-r16    SIBpos-r16,
    posSib2-20-r16    SIBpos-r16,
    posSib2-21-r16    SIBpos-r16,
    posSib2-22-r16    SIBpos-r16,
    posSib2-23-r16    SIBpos-r16,
    posSib3-1-r16     SIBpos-r16,
    posSib4-1-r16     SIBpos-r16,
    posSib5-1-r16     SIBpos-r16,
    posSib6-1-r16     SIBpos-r16,
    posSib6-2-r16     SIBpos-r16,
    posSib6-3-r16     SIBpos-r16,
    ... ,
```

```
      posSib1-9-v1700    SIBpos-r16,
      posSib1-10-v1700   SIBpos-r16,
      posSib2-24-v1700   SIBpos-r16,
      posSib2-25-v1700   SIBpos-r16,
      posSib6-4-v1700    SIBpos-r16,
      posSib6-5-v1700    SIBpos-r16,
      posSib6-6-v1700    SIBpos-r16
    },
    lateNonCriticalExtension OCTET STRING OPTIONAL,
    nonCriticalExtension SEQUENCE {} OPTIONAL
  }
```

### 5.7.8 Structure of parameters within the Non-SI parameter group

**Almost all** of the **UE configuration** parameters belong to **Non-SI parameter group**. The **structure** of these **UE** configuration **parameters** is **approximately hierarchical**, with the following **levels** from **top** to **bottom**:

1. **Cell group parameters within Non-SI** (click 5.7.8.1)

   The **configuration** for **each UE** is structured in **up to 2 cell groups**: the **master cell group** and the **secondary cell group** (click 10.3).

   **Instance count**: **up to 2 parameter instances**.

2. **Serving cell parameters within Non-SI** (click 5.7.8.2)

   The **configuration** for **each cell group** is structured in **up to 32 serving cells**: **1 special cell** and `0` to `31` **secondary cells** (click 3.7).

   **Instance count**: **up to** 2•32=**64 parameter instances**.

3. **Configuration type parameters within Non-SI** (click 5.7.8.3)

   The **configuration** for **each serving cell** (**special** or **secondary**) is structured in **2 configuration types**: the **common configuration** and the **common and dedicated configuration** (click 5.1.3).

   **Up to** 2•64=**128 parameter instances**.

4. **Carrier type parameters within Non-SI** (click 5.7.8.4)

   The **configuration** for **each configuration type** (**common** or **common and dedicated**) is structured in **up to 3 carrier types**: **Downlink**, **Uplink** and **SUL** (click 2.2.5).

   **Instance count**: **up to** 3•128=**384 parameter instances**.

5. **Bandwidth part type parameters within Non-SI** (click 5.7.8.5)

   The **configuration** for **each carrier type** (**Downlink**, **Uplink** or **SUL**) is structured in **up to 5 bandwidth parts** (click 2.2.7): **1 initial bandwidth part** and `0` to `4` **additional bandwidth parts** (click 2.2.7.2).

   **Instance count**: **up to** 5•384=**1,920 parameter instances**.

6. **Physical channel parameters within Non-SI** (click 5.7.8.6)

   The **configuration** for **each Downlink bandwidth part** is structured in **2 physical channel parameter sets** (click 5.4): **1** for **PDCCH** and **1** for **PDSCH**.

   The **configuration** for **each Uplink or SUL bandwidth part** is structured in **3 physical channel parameter sets**: **1** for **PRACH**, **1** for **PUCCH** and **1** **PUSCH**.

**Instance count**: **up to** 2•1,920/3+3•2•1,920/3=**5,120 parameter instances**.

To better understand **level 1**, see the **ancestor trees** of `CellGroupConfig` by clicking 3.7.1.

To better understand **levels 2 to 5**, see the **ancestor trees** of **bandwidth parts** up to `CellGroupConfig` by clicking 2.2.8.2.

**Approximately hierarchical** means that **most**, but **not all**, of the parameters to configure a given subsystem are located in its corresponding hierarchical level. In particular, parameters and data types added in **R16** and **R17** (typically ending in "`-r16`", "`-r17`", "`-r1610`", etc) **might not** follow the **hierarchical** structure.

For a description of the **3 stages** in which a **UE progressively gets** its configuration **parameters**, click 5.3.

### 5.7.8.1 Cell group parameters within non-SI

This section is specific for **cell group parameters** within the **non-SI** parameter group (click 5.3.6).

The **configuration** for **each UE** is structured in **up to 2 cell groups**.

These **2 cell groups** are the **master cell group** and the **secondary cell group** (click 10.3).

The **secondary cell group only** exists if the **UE** is configured with **dual connectivity**.

The configuration parameter for the **master cell group** are:

- `masterCellGroup` from `RRCSetup-IEs`, from `RRCResume-IEs` and from `RRCReconfiguration-IEs` (click 5.8.5.4). These **3 data types** are the **children** (respectively) of the `RRCSetup` (click 5.8.5.11), `RRCResume` (click 5.8.5.4) and `RRCReconfiguration` (click 5.8.5.4) **RRC NR messages**.

  The are **3 same-name parameters** (**each** from a **different data type**), that have the **same data type** (`CellGroupConfig`, click 3.7.3) and the **same semantics**.

The configuration parameter for the **secondary cell group** is:

- `secondaryCellGroup` **from** `RRCReconfiguration-IEs` (click 5.8.5.4) which is the **son** of the `RRCReconfiguration` **message** (click 5.8.5.4).

  Its **data type** is `CellGroupConfig` (click 3.7.3).

For the **ancestor trees** of `CellGroupConfig` **up to** some **RRC NR messages**, click 3.7.1.

For the **ancestor trees** of `CellGroupConfig` **up to** some **RRC inter-node messages**, click 3.7.2.

At **this level** of the configuration **structure**, the **UE** has **up to 2 parameter instances**:

- **1** for **each master cell group** and **1** for **each secondary cell group**.

For more detail on **cell group parameters** within the **non-SI** parameter group, click 3.7.1.

For more detail on **cell group parameters**, click 3.7.

### 5.7.8.2 Serving cell parameters within non-SI

This section is specific for **serving cell parameters** within the **non-SI** parameter group (click 5.3.6).

The **configuration** for **each cell group** is structured in **up to 32 serving cells**.

These **32 serving cells** are **1 special cell** and `0` to `31` **secondary cells** (click 3.7).

The **secondary cells only** exist if the **UE** is configured with **carrier aggregation** (click 10.2).

The configuration parameter for the **special cell** is:

- `spCellConfig` from `CellGroupConfig` (click 3.7.3).

The configuration parameter for **all** the **secondary cells** is:

- `sCellToAddModList` from `CellGroupConfig` (click 3.7.3).

The **ancestor tree** (click 5.9.8) of these **2 serving cell parameters** up to `CellGroupConfig` is:

```
                              CellGroupConfig (click 3.7.3)
spCellConfig                    SpCellConfig (click 3.9.5)
sCellToAddModList               SCellConfig (0..maxNrofSCells) (click 3.9.4)
```

where the constant `maxNrofSCells` is `31`.

At **this level** of the **configuration structure**, the **UE** has **up to 64 parameter instances**:

- **1** for each **special cell** and **1** for each of the `0` to `31` **secondary cells**.
- For each **master cell group** and for each **secondary cell group**.

For more detail on **serving cell parameters** within the **non-SI** parameter group, click 3.9.2.

For more detail on **serving cell parameters**, click 3.9.

### 5.7.8.3 Configuration type parameters within non-SI

This section is specific for **configuration type parameters** within the **non-SI** parameter group (click 5.3.6).

The **configuration** for **each serving cell** (**special** or **secondary**) is structured in **2 configuration types**.

These **2 configuration types** are the **common configuration** and the **common and dedicated configuration** (click 5.1.3).

The parameters for the **common configuration** are:

- `spCellConfigCommon` from `ReconfigurationWithSync` (click 3.9.8), within `SpCellConfig` (click 3.9.5), for the **special cell**. Notice that this parameter has the **intermediate** data type `ReconfigurationWithSync` between itself and `SpCellConfig`, that its reciprocal **dedicated** parameter `spCellConfigDedicated` does **not have** (see below). It is convenient that you are aware of this asymmetry between these two parameters.
- `sCellConfigCommon` from `SCellConfig` (click 3.9.4), for each **secondary cell**.

The parameters for the **common and dedicated configuration** are:

- `spCellConfigDedicated` from `SpCellConfig` (click 3.9.5), for the **special cell**.
- `sCellConfigDedicated`, from `SCellConfig` (click 3.9.4) for each **secondary cell**.

The **ancestor tree** (click 5.9.8) of these **4 configuration type parameters** up to `CellGroupConfig` is:

```
                                CellGroupConfig (click 3.7.3)
spCellConfig                      SpCellConfig (click 3.9.5)
  reconfigurationWithSync           ReconfigurationWithSync (0,1) (click 3.9.8)
    spCellConfigCommon                ServingCellConfigCommon (click 3.9.6)
  spCellConfigDedicated             ServingCellConfig (click 3.9.7)
sCellToAddModList                 SCellConfig (0..maxNrofSCells) (click 3.9.4)
  sCellConfigCommon                 ServingCellConfigCommon (0,1) (click 3.9.6)
  sCellConfigDedicated              ServingCellConfig (click 3.9.7)
```

where the constant `maxNrofCells` is `32`.

At **this level** of the **configuration structure**, the **UE** has **up to 128 parameter instances**:

- **1** for the **common configuration** and **1** for the **common and dedicated configuration**.
- For **each special cell** and for **each** of the `0` to `31` **secondary cells**.
- For **each master cell group** and for **each secondary cell group**.

For more detail on **configuration type parameters** within the **non-SI** parameter group, see the section on **serving cell parameters** within the **non-SI** parameter group (click 3.9.2)

For more detail on **configuration type parameters**, see the section on **serving cell parameters** (click 3.9).

### 5.7.8.4 Carrier type (Downlink, Uplink and SUL) parameters within non-SI

This section is specific for **carrier type parameters** within the **non-SI** parameter group (click 5.3.6).

The **configuration** for **each configuration type** (**common** or **common and dedicated**) is structured in **up to 3 carrier types**.

These **3 carrier types** are **Downlink**, **Uplink** and **SUL** (click 2.2.5).

The **SUL configuration only** exists if the **UE** is configured with **supplementary Uplink** (click 2.2.1.2).

The parameters for **Downlink** are:

- `downlinkConfigCommon` from `ServingCellConfigCommon` (click 3.9.6), for the **common configuration**.
- ***`DownlinkConfig`* parameter set** (click 2.2.8.3) from `ServingCellConfig` (click 3.9.7), for the **common and dedicated configuration**.

The parameters for **Uplink** are:

- `uplinkConfigCommon` from `ServingCellConfigCommon` (click 3.9.6), for the **common configuration**.
- `uplinkConfig` from `ServingCellConfig` (click 3.9.7), for the **common and dedicated configuration**.

The parameters for **SUL** are:

- `supplementaryUplinkConfig` from `ServingCellConfigCommon` (click 3.9.6), for the **common configuration**.
- `supplementaryUplink` from `ServingCellConfig` (click 3.9.7), for the **common and dedicated configuration**.

Notice that **each cell** (either **special** or **secondary**) contains **1 specific instance** of each of the **previous parameters** (see the **ancestor tree** right below):

The **ancestor tree** (click 5.9.8) of these **6 carrier type parameters** (or parameter set) up to `CellGroupConfig` is:

```
                                CellGroupConfig (click 3.7.3)
spCellConfig                      SpCellConfig (click 3.9.5)
  reconfigurationWithSync       ReconfigurationWithSync (0,1) (click 3.9.8)
    spCellConfigCommon            ServingCellConfigCommon (click 3.9.6)
      downlinkConfigCommon          DownlinkConfigCommon (click 2.2.6.8)
      uplinkConfigCommon            UplinkConfigCommon (click 2.2.6.9)
      supplementaryUplinkConfig     UplinkConfigCommon (0,1) (click 2.2.6.9)
  spCellConfigDedicated         ServingCellConfig (click 3.9.7)
    the parameters from DownlinkConfig parameter set (click 2.2.8.3)
    uplinkConfig                  UplinkConfig (0,1) (click 2.2.8.4)
    supplementaryUplink           UplinkConfig (0,1) (click 2.2.8.4)
sCellToAddModList               SCellConfig (0..maxNrofSCells) (click 3.9.4)
  sCellConfigCommon             ServingCellConfigCommon (0,1) (click 3.9.6)
    downlinkConfigCommon          DownlinkConfigCommon (click 2.2.6.8)
    uplinkConfigCommon            UplinkConfigCommon (click 2.2.6.9)
    supplementaryUplinkConfig     UplinkConfigCommon (0,1) (click 2.2.6.9)
  sCellConfigDedicated          ServingCellConfig (click 3.9.7)
    the parameters from DownlinkConfig parameter set (click 2.2.8.3)
    uplinkConfig                  UplinkConfig (0,1) (click 2.2.8.4)
    supplementaryUplink           UplinkConfig (0,1) (click 2.2.8.4)
```

where the constant `maxNrofCells` is `32`.

At **this level** of the **configuration structure**, the **UE** has **up to 384 parameter instances**:

- **1** for each **Downlink** plus **1** for each **Uplink**, and/or **1** for each **SUL**.
- For the **common configuration** and for the **common and dedicated configuration**.
- For each **special cell** and for each of the `0` to `31` **secondary cells**.
- For each **master cell group** and for each **secondary cell group**.

For more detail on **carrier type parameters** within the **non-SI** parameter group, see the section on **serving cell parameters** within the **non-SI** parameter group (click 3.9.2)

For more detail on **carrier type parameters**, see the section on **serving cell parameters** (click 3.9).

#### 5.7.8.5 Bandwidth part parameters within non-SI

This section is specific for **bandwidth part parameters** within the **non-SI** parameter group (click 5.3.6).

The **configuration** for **each carrier type** (**Downlink**, **Uplink** or **SUL**) is structured in **up to 5 bandwidth parts** (click 2.2.7).

These **5 bandwidth parts** are **1 initial bandwidth part** and `0` to `4` **additional bandwidth parts** (click 2.2.7.2).

The **additional bandwidth parts only** exist if the **UE** is configured with **them** (click 2.2.7.2).

The parameters for the **initial bandwidth parts**, for their **common configuration** are:

- `initialDownlinkBWP` from `DownlinkConfigCommon` (click 2.2.6.8): **1** instance for **Downlink**.
- `initialUplinkBWP` from `UplinkConfigCommon` (click 2.2.6.9): **1** instance for **Uplink** and **1** instance for **SUL**.

The parameters for the **initial bandwidth parts**, for their **dedicated configuration** are:

- `initialDownlinkBWP` from the ***DownlinkConfig*** **parameter set** (click 2.2.8.3), which is from `ServingCellConfig` (click 3.9.7): **1** instance for **Downlink**.
- `initialUplinkBWP` from `UplinkConfig` (click 2.2.8.4): **1** instance for **Uplink** and **1** instance for **SUL**.

Notice that the **Downlink**, and the **Uplink**/**SUL** parameters, have the **same** (duplicated) **name**.

The parameters for the **Downlink additional bandwidth parts**, for their **common and dedicated configuration**, are:

- `downlinkBWP-ToAddModList` from the ***DownlinkConfig*** **parameter set** (click 2.2.8.3), which is from `ServingCellConfig` (click 3.9.7): **1** instance for **Downlink**.
- `firstActiveDownlinkBWP-ID` from the ***DownlinkConfig*** **parameter set** (click 2.2.8.3), which is from `ServingCellConfig` (click 3.9.7): **1** instance for **Downlink**.

The parameters for the **Uplink and SUL additional bandwidth parts**, for their **common and dedicated configuration**, are:

- `uplinkBWP-ToAddModList` from `UplinkConfig` (click 2.2.8.4): **1** instance for **Uplink** and **1** instance for **SUL**.
- `firstActiveUplinkBWP-ID` from `UplinkConfig` (click 2.2.8.4): **1** instance for **Uplink** and **1** instance for **SUL**.

Notice that **each cell** (either **special** or **secondary**) contains **1 specific instance** of each of the **previous parameters**.

At **this level** of the **configuration structure**, the **UE** has **up to 1,920 parameter instances**:

- **1** for the **initial bandwidth part** and **1** for each of the `0` to `4` **additional bandwidth parts**.
- For each **Downlink**, for each **Uplink**, and/or for each **SUL**.
- For the **common configuration** and for the **common and dedicated configuration**.
- For each **special cell** and for each of the `0` to `31` **secondary cells**.
- For each **master cell group** and for each **secondary cell group**.

For more detail on **bandwidth parts parameters** within the **non-SI** parameter group, click 2.2.8.2.

For more detail on **bandwidth parts parameters**, click 2.2.8.

### 5.7.8.6 Physical channel parameters within non-SI

This section is specific for **physical channel parameters** within the **non-SI** parameter group (click 5.3.6).

The **configuration** for **each Downlink bandwidth part** is structured in **2 physical channel parameter sets** (click 5.4): **1** for **PDCCH** and **1** for **PDSCH**.

The **configuration** for **each Uplink or SUL bandwidth part** is structured in **3 physical channel parameter sets**: **1** for **PRACH**, **1** for **PUCCH** and **1 PUSCH**.

Notice that there is **no** parameter set to configure **PBCH**, because its configuration is mainly static.

The **most relevant data types** for the configuration of these **5 physical channels** are:

- **PDCCH parameter data types**:
  - `PDCCH-ConfigSIB1` (click 8.12.3): common parameters
  - `PDCCH-ConfigCommon` (click 6.12.4): common parameters
  - `PDCCH-Config` (click 6.12.5): dedicated parameters
  - `PDCCH-ServingCellConfig` (click 6.12.10): dedicated parameters

  For more detail on **PDCCH parameters** within the **non-SI** parameter group, click 6.12.3.

  For more detail on **PDCCH parameters**, click 6.12.
- **PDSCH parameter data types**:
  - `PDSCH-ConfigCommon` (click 6.18.5): common parameters
  - `PDSCH-Config` (click 6.18.6): dedicated parameters
  - `PDSCH-ServingCellConfig` (click 6.18.10): dedicated parameters

  For more detail on **PDSCH parameters** within the **Non-SI parameter group**, click 6.18.4.

  For more detail on **PDSCH parameters**, click 6.18.

- **PRACH parameter data types**:
  - `RACH-ConfigCommon` (click 9.5.3): common parameters
  - `RACH-ConfigGeneric` (click 9.5.4): common parameters
  - `RACH-ConfigDedicated` (click 9.5.10): dedicated parameters

  For more detail on **PRACH parameters** within the **non-SI** parameter group, click 9.5.2.

  For more detail on **PRACH parameters**, click 7.4.
- **PUCCH parameter data types**:
  - `PUCCH-ConfigCommon` (click 7.9.3): common parameters
  - `PUCCH-Config` (click 7.9.4): dedicated parameters

  For more detail on **PUCCH parameters** within the **non-SI** parameter group, click 7.9.2.

  For more detail on **PUCCH parameters**, click 7.8.2.
- **PUSCH parameter data types**:
  - `PUSCH-ConfigCommon` (click 7.15.4): common parameters
  - `PUSCH-Config` (click 7.15.5): dedicated parameters
  - `PUSCH-ServingCellConfig` (click 7.15.8): dedicated parameters

  For more detail on **PUSCH parameters** within the **non-SI** parameter group, click 7.15.3.

  For more detail on **PUSCH parameters**, click 7.15.

At **this level** of the **configuration structure**, the **UE** has **up to 5,120 parameter instances**:

- **2** for each **Downlink bandwidth part**, **3** for each **Uplink bandwidth part** and **3** for each **SUL bandwidth part**.
- For each **initial bandwidth part** and for each of the `0` to `4` **additional bandwidth parts**.
- For each **Downlink**, for each **Uplink**, and/or for each **SUL**.
- For the **common configuration** and for the **common and dedicated configuration**.
- For each **special cell** and for each of the `0` to `31` **secondary cells**.
- For each **master cell group** and for each **secondary cell group**.

## 5.8 The RRC protocol

The **RRC** protocol is responsible for the establishment, maintenance, and release of the **radio bearer** between the **UE** and the **NR node**. It handles the **signaling** for various purposes like connection **setup**, **mobility management**, and **control** of **radio resources**.

The **RRC protocol** provides several relevant **services** and **functions** (click 5.8.1).

The **RRC protocol** entity has a few relevant **protocol states** (click 5.8.2).

The **RRC protocol** has several relevant **procedures** which are subject to **very strict processing delay requirements** (click 5.8.3).

The **RRC protocol entity** offers to its user layer several **signaling bearers** (4.5.2.1).

Among other functions, **RRC** provides specific **access control** (click 5.8.4).

The **RRC definition** includes the **ASN.1** (click 5.7.4) **specification** of the **RRC NR messages** (click 5.8.5), of the **RRC inter-node messages** (click 5.8.6) and of the **RRC UE parameters** (click 5.6) that are conveyed within the **RRC messages**. The **RRC definition** also includes the **ASN.1 specification** of **UE variables** (click 5.6.2).

The **RRC UE parameters dynamically configure** the **UE** to **correctly interoperate** with the **up to 64 serving cells** and **up to 2 RAN nodes** to which the **UE** may be connected.

The **RRC protocol** also includes provisions for **UE capability retrieval** (click 5.8.7), **transport** of **NAS messages** (click 5.8.8), **UE assistance information** (click 5.8.9), and **RRC message segmentation** (click 5.8.10).

For an **overview** of **RRC**, refer to clause 7 of TS 38.300.

For the **detailed RRC specification**, refer to TS 38.331.

### 5.8.1 RRC services and functions

The main services and functions of the **RRC sublayer** over the **Uu** interface include:

- **Broadcast of system information** (click 5.2) related to **AS** and **NAS**.
- **Paging** initiated by **5GC** or **NG-RAN**.
- **Establishment**, **maintenance** and **release** of an **RRC connection** between the **UE** and **NG-RAN** including:
  - **Addition**, **modification** and **release** of **carrier aggregation**.
  - **Addition**, **modification** and **release** of **dual connectivity** in **NR** or between **E-UTRA** and **NR**.
- **Security** functions including key management.
- Establishment, configuration, maintenance and release of **radio bearers** (click 4.5.2.1).
- **Mobility** functions including:
  - **Handover** and **context transfer.**
  - **UE cell selection** and **reselection** and **control** of **cell selection** and **reselection.**
  - **Inter-RAT mobility.**
- **QoS** management functions.
- **UE measurement** reporting and control of the reporting.
- **Detection** of and **recovery** from **radio link failure**.
- **NAS** message transfer to/from **NAS** from/to **UE**.

The **Sidelink specific** services and functions of the **RRC sublayer** over the **Uu** interface include:

- **Configuration** of **Sidelink resource allocation** via system information or dedicated signaling.
- **Reporting** of **UE Sidelink** information.
- **Measurement** configuration and reporting related to **Sidelink**.
- **Reporting** of **UE assistance** information for **Sidelink traffic pattern**(**s**).

### 5.8.2 RRC protocol states and transitions

The **RRC protocol** has the following **3 states**, indicating for **each** of them its main characteristics:

- **RRC_IDLE**:

  The **UE** already performed its **initial access** (click 8). The **RRC connection** between the **network** and the **UE** is **not** established. The **UE** does **not** have a **C-RNTI**. The **UE** is in **low-power mode**.

  The **UE periodically monitors** the **control channel** for **incoming messages** or **paging** for:

  - **PLMN** selection.
  - **Broadcast** of **system information**.
  - **Cell re-selection mobility**.
  - Transfer of **broadcast** data to the **UE** over **multicast radio bearers** (**MRBs**, click 5.8.3.4).
  - **Paging** for mobile terminated data is initiated by **5GC** (**CN paging**).
  - **Discontinuous reception** (**DRX**) for **CN paging** is configured by **non-access stratum** (**NAS**).

- **RRC_CONNECTED**:

  The **RRC connection** between the **network** and the **UE** is **established**, and they are **ready** to **exchange data**. The **UE has** a **C-RNTI**.

  - **5GC** to **NG-RAN** connection (**both control** and **user planes**) is established for **UE**.
  - The **UE access stratum context** is stored both in the **NG-RAN** and in the **UE**.
  - **NG-RAN** knows the cell which the **UE** belongs to.
  - Transfer of **unicast** data to/from the **UE**.
  - Transfer of **multicast/broadcast** data to the **UE** over **MRB**s.
  - Network controlled **mobility** including **measurements**.

- **RRC_INACTIVE**:

  The **RRC connection** between the **network** and the **UE** is **suspended** (a **dormant state**) and can be **quickly resumed**. The **UE** remains in this **state** when there are

**no active sessions** or when the **UE** is **temporarily detached** from the **network** but **still registered**.

This state has **even lower power** consumption than the **RRC_IDLE** state.

When **suspending** the **RRC connection**, **both** the **UE** and the **NR node store** the **UE AS context** (including the **C-RNTI**). When **resuming** the **suspended RRC connection**, **both** the **UE** and the **NR node retrieve** the **UE AS context**.

The **UE AS context** is identified with an "**I-RNTI value**". In spite of its name, the "**I-RNTI value**" is **not** a **RNTI** (click 6.9.10.1). The corresponding **parameters** and **data types** are:

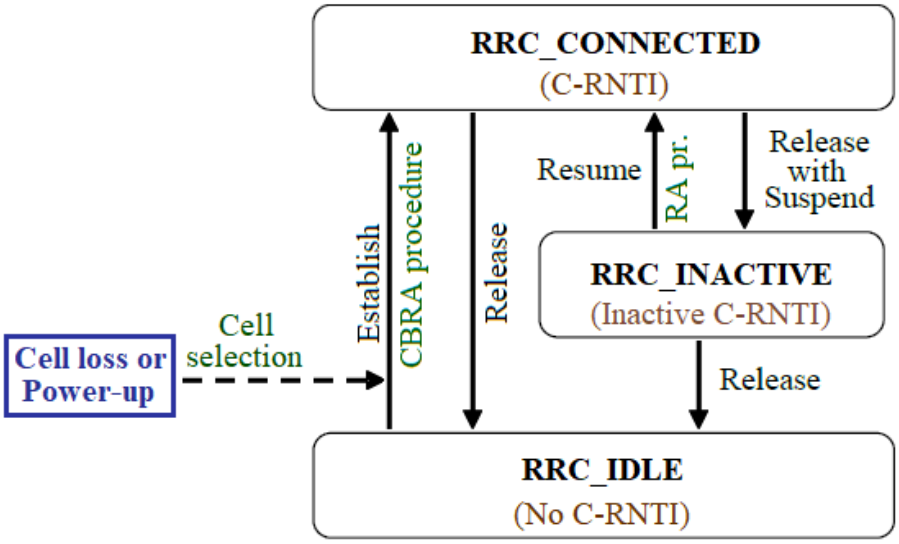
- `fullI-RNTI` from either `SuspendConfig` or `PagingUE-Identity`. Its data type is `I-RNTI-Value`.
- `fullI-RNTI-r17` from `SL-PagingIdentityRemoteUE-r17`. Its data type is `I-RNTI-Value`.
- `shortI-RNTI` from `SuspendConfig`. Its data type is `ShortI-RNTI-Value`.
- `resumeIdentity` from either `RRCResumeRequest-IEs` or `RRCResumeRequest1-IEs`. Its data type is `ShortI-RNTI-Value`.
- `I-RNTI-Value` is a **40-bit bitstring** and `ShortI-RNTI-Value` is a **24-bit bitstring**.

This state did **not** exist in **LTE**.

The **UE periodically monitors** the **control channel** for **incoming messages** or **paging** for:

- **PLMN** selection.
- **Broadcast** of **system information**.
- Cell re-selection **mobility**.
- Transfer of **broadcast** data to the **UE** over **MRB**(s).
- **Paging** is initiated by **NG-RAN** (**RAN paging**).
- **Discontinuous reception** (**DRX**) for **RAN paging** configured by **NG-RAN**.
- **RAN-based notification area** (**RNA**) is managed by **NG-RAN**.
- **5GC** to **NG-RAN** connection (**both C/U-planes**) is established for **UE**.
- The **UE** Inactive **Access Stratum** (**AS**) **context** is stored in **NG-RAN** and the **UE**.
- **NG-RAN** knows the **RNA** which the **UE** belongs to.
- Transfer of **unicast data** and/or **signaling** to/from the **UE** over **radio bearers** configured for **small data transmission** (**SDT**, click 10.9).

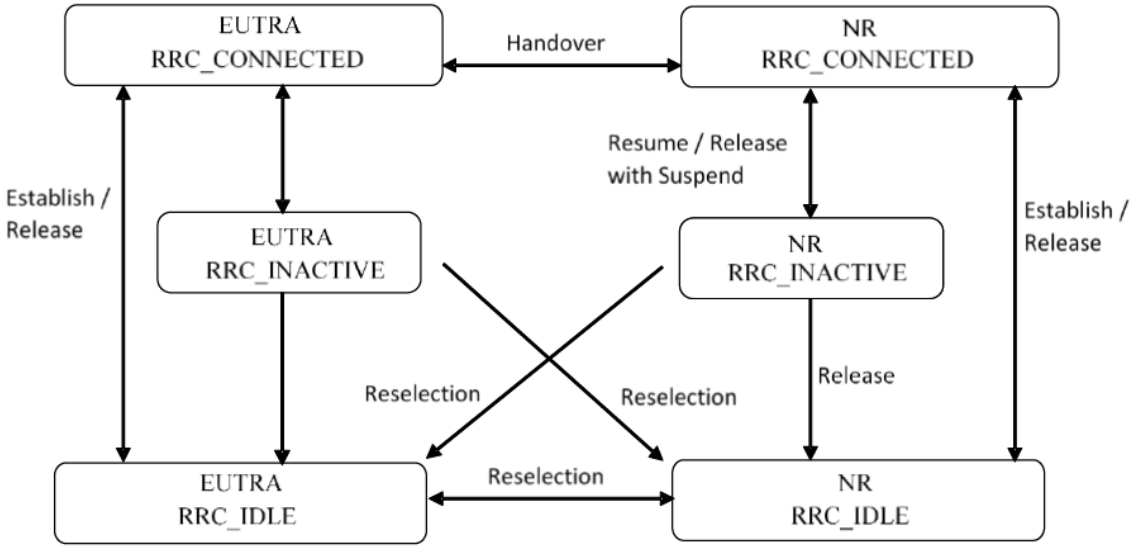
The following figure shows the **RRC** protocol **state machine**:




Some relevant remarks on this figure are:

- The **state** "**Cell loss or Power-up**" and the **transition** "**Cell selection**" are **not** part of **RRC**. They are **only** shown in the figure to relate the **RRC states** with the **initial access procedure** (click 8).
- The text "**CBRA procedure**" indicates that the transition "**Establish**" require a **contention-based random access procedure** (click 9).
- The text "**RA pr.**" indicates that the transition "**Resume**" requires a **random access procedure**.
- The **transition** "**Establish**" involves the **RRC messages** `RRCSetupRequest` (click 5.8.5.10), `RRCSetup` (click 5.8.5.11) and `RRCSetupComplete` (click 5.8.5.12). For more detail, refer to clause 5.3.3 of TS 38.331.
- The **transition** "**Release**" involves the **RRC message** `RRCRelease`. For more detail, refer to clause 5.3.8 of TS 38.331.
- The **transition** "**Release with Suspend**" involves the **RRC message** `RRCRelease` including `suspendConfig` from `RRCRelease-IEs`. For more detail, refer to clause 5.3.8 of TS 38.331.
- The **transition** "**Resume**" involves the **RRC message** `RRCResumeRequest`, `RRCResumeRequest1`, `RRCResume` and `RRCResumeComplete`. This **transition may** include **other RRC messages** (e.g., `RRCSetup`) in **some scenarios**. For more detail, refer to clause 5.3.13 of TS 38.331.
- The **NR RRC** messages `RRCSetupRequest`, `RRCResumeRequest` and `RRCResumeRequest1` (among others) can be sent as **Msg3** (click 9.3.4.2) within the **contention-based random access procedure**.
- There are **other transitions** (e.g., **connection re-establishment**) **not included** in the figure.

The following figure shows the **RRC** protocol **state transitions** in **NR** as well as the mobility procedures supported between **NR/5GC**, **E-UTRA/EPC** and **E-UTRA/5GC**:




For more detail on **RRC protocol states** and **transitions**, refer to clause 4 of TS 38.331.

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_RRC_Overview.html
- https://howltestuffworks.blogspot.com/2019/09/5g-nr-ue-rrc-states-and-state.html
- https://www.sharetechnote.com/html/5G/5G_RRC_Inactive.html
- https://techlteworld.com/rrc-inactive-state-in-5g-nr/

### 5.8.3 RRC procedures and processing delay requirements

The different **RRC procedures** may be structured in the following **groups**:

- **RRC connection control procedures** (click 5.8.3.1).
- **RRC inter-RAT mobility procedures** (click 5.8.3.2).
- **RRC other procedures** (click 5.8.3.3).

Each **RRC procedure** involves several **RRC NR messages** (click 5.8.5) and/or of **RRC inter-node messages** (click 5.8.6).

Because of the very low level at which the **RRC procedures** operate, they are subject to very demanding **processing delay** (click 5.8.3.4) **requirements**.

For more detail on **processing delay requirements** for **RRC procedures**, refer to clause 12 of TS 38.331.

### 5.8.3.1 RRC connection control procedures

The following table shows different **RRC connection control procedures**, the involved **RRC messages**, and the corresponding **processing delay** (click 5.8.3.4) requirement.

| RRC procedure | RRC messages Network => UE | RRC messages UE => Network | Delay [ms] |
|---|---|---|---|
| **RRC reconfiguration** | RRCReconfiguration (click 5.8.5.13) | RRCReconfiguration Complete | 10 |
| **RRC reconfiguration SCell addition or release** | | | 16 |
| **RRC reconfiguration LTE/NR SCG establishment, modification or release** | | | 16 |
| **RRC reconfiguration Intra-NR mobility with LTE/NR SCG establishment, modification or release** | | | 16 |
| **RRC reconfiguration** (Note 1) | DLDedicatedMessage Segment | RRCReconfiguration Complete | 16+ (Nseg-1)•10 |
| **RRC setup** | RRCSetup (click 5.8.5.11) | RRCSetupComplete (click 5.8.5.12) | 10 |
| **RRC release** | RRCRelease | | N/A |
| **RRC re-establishment** | RRCReestablishment | RRCReestablishment Complete | 10 |
| **RRC resume** (Note 2) | RRCResume | RRCResumeComplete | 6 or 10 |
| **RRC resume MCG SCell addition, restoration or release** | RRCResume | RRCResumeComplete | 16 |
| **RRC resume SCG establishment, restoration or release** | RRCResume | RRCResumeComplete | 16 |
| **RRC resume** (Note 1) | DLDedicatedMessage Segment | RRCResumeComplete | 16+ (Nseg-1)•10 |
| **Initial AS security activation** | SecurityModeCommand | SecurityModeComplete/ SecurityModeFailure | 5 |

Some **relevant remarks** on this table are:

- Notice that the **RRC** "…Request" messages (e.g., RRCSetupRequest, RRCResumeRequest) are not included
- This table is a fragment from **table 12.1-1** of TS 38.331.
- **Note 1**: **Nseg** is the **quantity** of **RRC segments**.
- **Note 2**: **Value=6** applies for a **UE** supporting reduced **CP** latency for the case of RRCResume **message only including**: **MAC** and **PHY** configuration, reestablishPDCP and reestablishRLC for **signaling radio bearer 2** (**SRB2**, click 4.5.2.1), **MRB**(s) and **DRB**(s), and **no DRX**, **SPS**, **configured grant**, **CA** or **MIMO re-configuration** will be **triggered** by **this message**. Further, the **PUSCH assignment** for **transmission** of RRCResumeComplete and the **data** is **transmitted** over **common search space** with **DCI Format 0_0**.

  In this scenario, the **RRC procedure delay** can extend **beyond** the **reception** of the **Uplink assignment**, **up to 7 ms**.

**Value = 10** applies to other cases.

### 5.8.3.2 RRC inter RAT mobility procedures

The following table shows different **RRC inter RAT mobility procedures**, the involved **RRC messages**, and the corresponding **processing delay** (click 5.8.3.4) requirement:

| RRC procedure | RRC messages Network => UE | RRC messages UE => Network | Delay [ms] | Notes |
|---|---|---|---|---|
| **Handover to NR** | `RRCReconfiguration` (sent by other RAT) | `RRCReconfiguration Complete` | N/A | 1 |
| **Handover from NR** | `MobilityFromNR Command` | | N/A | 2 |

Some **relevant remarks** on this **table** are:

- This table is a fragment from **table 12.1-1** of TS 38.331.
- **Note 1**: In case of handover from E-UTRA to NR, the performance of this procedure is specified in clauses 5.3.4.2, 5.3.4A.2 and 5.3.5.2 of TS 36.133.
- **Note 2**: The performance of this procedure is specified in clauses 6.1.2.1.2 and 6.1.2.2.2 of TS 38.133.

### 5.8.3.3 RRC other procedures

The following table shows different **RRC other procedures**, the involved **RRC messages**, and the corresponding **processing delay** (click 5.8.3.4) requirement:

| RRC procedure | RRC messages Network => UE | RRC messages UE => Network | Delay [ms] | Notes |
|---|---|---|---|---|
| **UE assistance information** | | `UEAssistanceInformation` | N/A | |
| **UE capability transfer** | `UECapabilityEnquiry` | `UECapabilityInformation` | 80 | |
| **UE capability transfer** | `UECapabilityEnquiry` | `ULDedicatedMessage Segment` | 80 | |
| **Counter check** | `CounterCheck` | `CounterCheckResponse` | 5 | |
| **UE information** | `UEInformationRequest` | `UEInformationResponse` | 15 | |
| **DL Information transfer MR-DC** | `DLInformation TransferMRDC` | | N/A | 1 |
| **IAB other information** | | `IABOtherInformation` | N/A | |
| **Sidelink UE information** | | `SidelinkUEInformationNR` | N/A | |
| **UE Positioning assistance information** | | `UEPositioningAssistance Info` | N/A | |

Some **relevant remarks** on this **table** are:

- This table is a fragment from **table 12.1-1** of TS 38.331.
- **Note 1**: The **UE** shall apply the performance requirements of the **RRC message** included **within** the `DLInformationTransferMRDC` **message**.

### 5.8.3.4 RRC processing delay definition

The **processing delay** is expressed as the **time** (in **ms**) **from** the **end of reception** of the "**Network-to-UE**" **message** on the **UE physical layer up to** when the **UE** shall be **ready**

for the **reception** of the **Uplink transmission** of the "**UE-to-network**" **response message** with **no access delay other** than the **transmission time interval alignment** (e.g., excluding delays caused by scheduling, the **random access procedure** or physical layer synchronization).

The following figure illustrates the definition of the **RRC processing delay**:

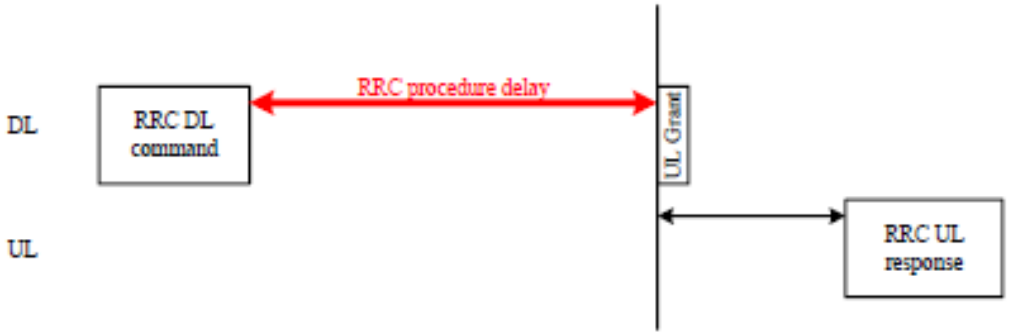



Table 12.1-1 of TS 38.331 defines the specific **processing delay** value for **each** of the different **RRC procedures**.

In case the **RRC procedure triggers bandwidth part switching**, the **RRC procedure delay is** the value defined in the said **table**, **plus** the **bandwidth part switching delay** defined in clause 8.6.3 of TS 38.133.

For more detail on **RRC processing delay requirements**, refer to clause 12 of TS 38.331.

### 5.8.4 RRC access control

**NG-RAN** supports **overload** and **access control** functionality such as **RACH back off**, **RRC Connection Reject**, **RRC Connection Release** and **UE based access barring mechanisms**.

One unified access control framework as specified in TS 22.261 applies to all **UE** states (**RRC_IDLE**, **RRC_INACTIVE** and **RRC_CONNECTED**, click 5.8.2) for **NR**. **NG-RAN** broadcasts **barring control** information associated with **Access Categories** and **Access Identities** (in case of network sharing, the barring control information can be set individually for each **PLMN**). The **UE** determines whether an access attempt is authorized based on the barring information broadcast for the selected **PLMN**, and the selected **Access Category** and **Access Identity(ies)** for the access attempt:

- For **NAS triggered requests**, **NAS determines** the **Access Category** and **Access Identity(ies)**.
- For **AS triggered requests**, **RRC determines** the **Access Category** while **NAS determines** the **Access Identity(ies)**.

The **gNB** handles **access attempts** with establishment causes `emergency`, `mps-PriorityAccess` and `mcs-PriorityAccess` (i.e., **Emergency calls**, **MPS**, **MCS** subscribers) with **high priority** and responds with **RRC Reject** to these access attempts **only** in **extreme network load conditions** that may **threaten** the **gNB stability**.

Unified access control does not apply to **IAB-MTs**.

### 5.8.5 RRC NR messages

**RRC** defines many **NR messages** exchanged across the **NR** radio interface between a **gNB** and **its UEs**. These **NR messages** are grouped into **nine RRC NR message classes** (presented in the following subsections).

**Each RRC message class** is a **top named data type** (click 5.9.6) in the **RRC ASN.1 specification** (click 5.7.2). The **ASN.1 specification** of **each RRC message class** contains a **`CHOICE`** between the different **RRC messages** belonging to it.

**Unicast dedicated signaling** consists of **unicast RRC NR messages**.

In addition to these **NR messages**, **RRC** also defines **inter-node messages** (click 5.8.6).

For more detail on **RRC NR messages**, refer to clause 6.2 of TS 38.331.

Source web pages:

- https://www.nrexplained.com/rrc

#### 5.8.5.1 RRC `BCCH-BCH-Message` class

It is **broadcast** by the **network**.

It **only delivers** the **RRC** `MIB` **message**.

It is **broadcast** (within **SSB**) through the **BCCH logical** channel, the **BCH transport** channel, and the **PBCH physical** channel.

Its **complete ASN.1** (click 5.9) specification is:

```
BCCH-BCH-Message ::= SEQUENCE {
  message BCCH-BCH-MessageType
}
```

where `BCCH-BCH-MessageType` is a **`CHOICE`** that **only** contains the following **RRC message**:

- `MIB` message

  It is **broadcast** by the **network** to distribute `MIB` to the **UEs**.

  For more detail on `MIB` **characteristics**, click 5.3.3.

  For the **parameters contained** within `MIB`, click 8.12.3.

#### 5.8.5.2 RRC `BCCH-DL-SCH-Message` class

It is **broadcast** by the **network**.

It **delivers** one **RRC system information message**: either `SIB1` or `SystemInformation`.

It is **broadcast** through the **BCCH logical** channel, the **DL-SCH transport** channel and the **PDSCH physical** channel.

Its **complete ASN.1** (click 5.9) specification is:

```
BCCH-DL-SCH-Message ::= SEQUENCE {
  message BCCH-DL-SCH-MessageType
}
```

where `BCCH-DL-SCH-MessageType` is a **`CHOICE`** between the following **RRC messages**:

- `SIB1` **message** (click 5.7.4)
- `SystemInformation` **message** (click 5.7.5)

### 5.8.5.3 RRC `DL-CCCH-Message` class

It is **unicast** by the **network**.

It is **unicast** through the **DL-CCCH logical** channel, the **DL-SCH transport** channel and the **PDSCH physical** channel.

Its **complete ASN.1** (click 5.9) specification is:

```
DL-CCCH-Message ::= SEQUENCE {
  message DL-CCCH-MessageType
}
```

where `DL-CCCH-MessageType` is a **`CHOICE`** between the following **RRC messages**:

- `RRCReject` message

  It **rejects** an **RRC connection establishment** (`RRCSetupRequest`, click 5.8.5.10), **connection resume** (`RRCResumeRequest`) or **connection re-establishment** (`RRCReestablishmentRequest`) that has been **requested** by the **UE**.

- `RRCSetup` message (click 5.8.5.11)

### 5.8.5.4 RRC `DL-DCCH-Message` class

It is **unicast** by the **network**.

It **delivers one** among several **RRC control messages** sent by the **network**.

It is **unicast** through the **DL-DCCH logical** channel, the **DL-SCH transport** channel and the **PDSCH physical** channel.

Its **complete ASN.1** (click 5.9) specification is:

```
DL-DCCH-Message ::= SEQUENCE {
  message DL-DCCH-MessageType
}
```

where `DL-DCCH-MessageType` is a **`CHOICE`** between the following **RRC messages**:

- `RRCReconfiguration` message (click 5.8.5.13)
- `RRCResume` message
- `RRCRelease` message
- `RRCReestablishment` message
- `SecurityModeCommand` message
- `DLInformationTransfer` message
- `UECapabilityEnquiry` message
- `CounterCheck` message
- `MobilityFromNRCommand` message

- `DLDedicatedMessageSegment-r16` message
- `UEInformationRequest-r16` message
- `DLInformationTransferMRDC-r16` message
- `LoggedMeasurementConfiguration-r16` message

#### 5.8.5.5 RRC `MCCH-Message` class

It is **broadcast** by the **network**.

It **only delivers** the **RRC** `MBSBroadcastConfiguration-r17` **message**.

It is **broadcast** through the **MCCH logical** channel, the **DL-SCH transport** channel and the **PDSCH physical** channel.

Its **complete ASN.1** (click 5.9) specification is:

```
MCCH-Message ::= SEQUENCE {
  message MCCH-MessageType
}
```

where `MCCH-MessageType` is a `CHOICE` that **only** contains the following **RRC message**:

- `MBSBroadcastConfiguration-r17` message

  It is **broadcast** by the **network** to distribute the configuration of the **multicast-broadcast service** to the **UEs**. It is sent over an **MRB** (click 4.5.2.1), and its **RLC-SAP** is **UM**.

#### 5.8.5.6 RRC `PCCH-Message` class

It is **broadcast** by the **network**.

It **only delivers** the **RRC** `Paging` **message**.

It is **broadcast** through the **PCCH logical** channel, the **PCH transport** channel and the **PDSCH physical** channel.

Its **complete ASN.1** (click 5.9) specification is:

```
PCCH-Message ::= SEQUENCE {
  message PCCH-MessageType
}
```

where `PCCH-MessageType` is a `CHOICE` that **only** contains the following **RRC message**:

- `Paging` message

#### 5.8.5.7 RRC `UL-CCCH-Message` class

It is **unicast** by the **UE**.

It **delivers** one among several **48-bit RRC UE request messages**

It is **unicast** through the **UL-CCCH logical** channel, the **UL-SCH transport** channel and the **PUSCH physical** channel.

Its **complete ASN.1** (click 5.9) specification is:

```
UL-CCCH-Message ::= SEQUENCE {
  message UL-CCCH-MessageType
}
```

where UL-CCCH-MessageType is a **CHOICE** between the following **RRC messages**:

- RRCSetupRequest message (click 5.8.5.10)
- RRCResumeRequest message
- RRCReestablishmentRequest message
- RRCSystemInfoRequest message (click 5.8.5.14)

### 5.8.5.8 RRC UL-CCCH1-Message class

It is **unicast** by the **UE**.

It **only delivers** the **64-bit RRC** RRCResumeRequest1 **message**.

It is **unicast** through the **UL-CCCH logical** channel, the **UL-SCH transport** channel and the **PUSCH physical** channel.

Its **complete ASN.1** (click 5.9) specification is:

```
UL-CCCH1-Message ::= SEQUENCE {
  message UL-CCCH1-MessageType
}
```

where UL-CCCH1-MessageType is a **CHOICE** that **only** contains the following **RRC message**:

- RRCResumeRequest1 message

### 5.8.5.9 RRC UL-DCCH-Message class

It is **unicast** by the **UE**.

It **delivers** one among several **RRC UE control messages**.

It is **unicast** through the **UL-DCCH logical** channel, the **UL-SCH transport** channel and the **PUSCH physical** channel.

Its **complete ASN.1** (click 5.9) specification is:

```
UL-DCCH-Message ::= SEQUENCE {
  message UL-DCCH-MessageType
}
```

where UL-DCCH-MessageType is a **CHOICE** between the following **RRC messages**:

- MeasurementReport message
- RRCReconfigurationComplete message
- RRCSetupComplete message (click 5.8.5.12)
- RRCReestablishmentComplete message
- RRCResumeComplete message
- SecurityModeComplete message
- SecurityModeFailure message
- ULInformationTransfer message
- LocationMeasurementIndication message
- UECapabilityInformation message

- `CounterCheckResponse` message
- `UEAssistanceInformation` message
- `FailureInformation` message
- `ULInformationTransferMRDC` message
- `SCGFailureInformation` message
- `SCGFailureInformationEUTRA` message
- `ULDedicatedMessageSegment-r16` message
- `DedicatedSIBRequest-r16` message
- `MCGFailureInformation-r16` message
- `UEInformationResponse-r16` message
- `SidelinkUEInformationNR-r16` message
- `ULInformationTransferIRAT-r16` message
- `IABOtherInformation-r16` message
- `MBSInterestIndication-r17` message
- `UEPositioningAssistanceInfo-r17` message
- `MeasurementReportAppLayer-r17` message

### 5.8.5.10 `RRCSetupRequest` data type

It is an **NR RRC message** of the `UL-CCCH-MessageType` class (click 5.8.5.7).

It is sent by the **UE** to **request** the **establishment** of an **RRC connection**, within the **RRC procedure** of **connection establishment**.

It can be sent through **RLC-SAP TM**, through **logical binary channel CCCH** and through **signaling radio bearer 0** (**SRB0**, click 4.5.2.1).

It can also be sent as **Msg3** (click 9.3.4.2), within a **contention-based** (click 9.1.2) **random access procedure** (click 9).

For more detail on `RRCSetupRequest`, refer to clause 6.2.2 of TS 38.331.

Its **complete ASN.1** (click 5.9) specification is:

```
RRCSetupRequest ::=  SEQUENCE {
  rrcSetupRequest RRCSetupRequest-IEs
}
```

where the **complete ASN.1 specification** of its **son data type** `RRCSetupRequest-IEs` is:

```
RRCSetupRequest-IEs ::= SEQUENCE {
  ue-Identity InitialUE-Identity,
  establishmentCause EstablishmentCause,
  spare BIT STRING (SIZE (1))
}
```

### 5.8.5.11 `RRCSetup` data type

It is an **NR RRC message** of the `DL-CCCH-Message` class (click 5.8.5.3).

It is sent by the **NR node** to **establish** the **signaling radio bearer 1** (**SRB1**, click 4.5.2.1). It is sent as an answer to an **RRCSetupRequest** (click 5.8.5.10) **message** received from an **UE**, within the **RRC procedure** of **connection establishment**.

It is sent through **RLC-SAP TM**, through **logical binary channel CCCH** and through **SRB0**.

For more detail on **RRCSetup**, refer to clause 6.2.2 of TS 38.331.

Its **complete ASN.1** (click 5.9) specification is:

```
RRCSetup ::=  SEQUENCE {
  rrc-TransactionIdentifier RRC-TransactionIdentifier,
  criticalExtensions CHOICE {
    rrcSetup RRCSetup-IEs,
    criticalExtensionsFuture SEQUENCE {}
  }
}
```

where the **complete ASN.1 specification** of its **son data type** **RRCSetup-IEs** is:

```
RRCSetup-IEs ::= SEQUENCE {
  radioBearerConfig RadioBearerConfig,
  masterCellGroup OCTET STRING (CONTAINING CellGroupConfig),
  lateNonCriticalExtension OCTET STRING OPTIONAL,
  nonCriticalExtension RRCSetup-v1700-IEs
}
```

### 5.8.5.12 RRCSetupComplete data type

It is an **NR RRC message** of the **UL-DCCH-Message** class (click 5.8.5.9).

It is sent by the **UE** to **confirm** the **successful completion** of an **RRC connection establishment**.

It is sent through **RLC-SAP AM**, through **logical binary channel UL-DCCH** and through **signaling radio bearer 1** (**SRB1**, click 4.5.2.1).

For more detail on **RRCSetupComplete**, refer to clause 6.2.2 of TS 38.331.

Its **complete ASN.1** (click 5.9) specification is:

```
RRCSetupComplete ::=  SEQUENCE {
  rrc-TransactionIdentifier RRC-TransactionIdentifier,
  criticalExtensions CHOICE {
  rrcSetupComplete RRCSetupComplete-IEs,
  criticalExtensionsFuture SEQUENCE {}
  }
}
```

where the **complete ASN.1 specification** of its **son data type** **RRCSetupComplete-IEs** is:

```
RRCSetupComplete-IEs ::= SEQUENCE {
  selectedPLMN-Identity INTEGER (1..maxPLMN),
  registeredAMF RegisteredAMF OPTIONAL,
  guami-Type ENUMERATED {native, mapped} OPTIONAL,
  s-NSSAI-List SEQUENCE (SIZE (1..maxNrofS-NSSAI)) OF S-NSSAI OPTIONAL,
  dedicatedNAS-Message DedicatedNAS-Message,
  ng-5G-S-TMSI-Value CHOICE {
  ng-5G-S-TMSI NG-5G-S-TMSI,
  ng-5G-S-TMSI-Part2 BIT STRING (SIZE (9))
  } OPTIONAL,
  lateNonCriticalExtension OCTET STRING OPTIONAL,
  nonCriticalExtension RRCSetupComplete-v1610-IEs OPTIONAL
}
```

### 5.8.5.13 RRCReconfiguration data type

It can be an **NR RRC message** of the DL-DCCH-Message class (click 5.8.5.4) or a **data type** contained **within other RRC messages** (possibly as an **OCTET STRING**).

It is included as an **OCTET STRING** in **seven data types**. This means that RRCReconfiguration is the **son** of the said **seven data types**.

It is sent by the **NR node** to **reconfigure** the **UE** under different circumstances. It may convey information for **measurement configuration**, **mobility control**, **radio resource configuration** (including **RBs**, **MAC main configuration** and **physical channel configuration**) and **AS security configuration**.

As an **RRC message**, it is sent through **RLC-SAP AM**, through **logical binary channel DL-DCCH** and through **signaling radio bearer 1** (**SRB1**, click 4.5.2.1) or **SRB3**.

For more detail on RRCReconfiguration, refer to clause 6.2.2 of TS 38.331.

Its **complete ASN.1** (click 5.9) specification is:

```
RRCReconfiguration ::=  SEQUENCE {
  rrc-TransactionIdentifier RRC-TransactionIdentifier,
  criticalExtensions CHOICE {
    rrcReconfiguration RRCReconfiguration-IEs,
    criticalExtensionsFuture SEQUENCE {}
  }
}
```

where the **complete ASN.1 specification** of its **son data type** RRCReconfiguration-IEs is:

```
RRCReconfiguration-IEs ::= SEQUENCE {
  radioBearerConfig RadioBearerConfig OPTIONAL,                    -- Need M
  secondaryCellGroup OCTET STRING (CONTAINING CellGroupConfig)
                     OPTIONAL,                                   -- Cond SCG
  measConfig MeasConfig OPTIONAL,                                  -- Need M
  lateNonCriticalExtension OCTET STRING OPTIONAL,
  nonCriticalExtension RRCReconfiguration-v1530-IEs OPTIONAL
}
```

### 5.8.5.14 RRCSystemInfoRequest data type

It is an **NR RRC message** of the UL-CCCH-Message class (click 5.8.5.7).

It is sent by the **UE** to **request other SIBs and posSIBs** (SystemInformation, click 5.7.5), **only** in the case that the **NR node** has **configured on-demand System Information delivery** (click 5.3.5).

It can be sent through **RLC-SAP TM**, through **logical binary channel UL-CCCH** and through **signaling radio bearer 0** (**SRB0**, click 4.5.2.1).

It can also be sent as **Msg3** (click 9.3.4.1.2), within a **contention-based** (click 9.1.2) **random access procedure** (click 9).

For more detail on RRCSystemInfoRequest, refer to clauses 5.2.2.3.3, 5.2.2.3.3a and 6.2.2 of TS 38.331.

Its **complete ASN.1** (click 5.9) specification is:

```
RRCSystemInfoRequest ::=  SEQUENCE {
  criticalExtensions CHOICE {
    rrcSystemInfoRequest RRCSystemInfoRequest-IEs,
    criticalExtensionsFuture-r16 CHOICE {
      rrcPosSystemInfoRequest-r16 RRC-PosSystemInfoRequest-r16-IEs,
      criticalExtensionsFuture SEQUENCE {}
    }
  }
}
```

where the **complete ASN.1 specification** of its **children data types** `RRCSystemInfoRequest-IEs` and `RRC-PosSystemInfoRequest-r16-IEs` are:

```
RRCSystemInfoRequest-IEs ::= SEQUENCE {
  requestedSI-List BIT STRING (SIZE (maxSI-Message)),                --32bits
  spare BIT STRING (SIZE (12))
}

RRC-PosSystemInfoRequest-r16-IEs ::= SEQUENCE {
  requestedPosSI-List BIT STRING (SIZE (maxSI-Message)),             --32bits
  spare BIT STRING (SIZE (11))
}
```

### 5.8.6 RRC inter-node messages

**RRC inter-node messages** are exchanged **between network nodes**. These **messages** may **also** be transferred **to**/**from** the **UE** via **another radio access technology** (**RAT**). Consequently, these messages have **similar characteristics** as the **RRC NR messages** that are transferred between a **gNB** and its **UEs**, i.e. the **same transfer syntax** and **protocol extension** mechanisms apply.

**Inter-node messages** are sent either across the **X2**, the **Xn** or the **NG** interface, either **to**/**from** the **gNB**, i.e. a **single** "**logical channel**" is used for **all RRC messages** transferred **across network nodes**. The information could originate from or be destined for another **RAT**.

The **RRC inter-node messages** are:

- `CG-CandidateList`
- `HandoverCommand`

  It is used to transfer the **handover command** as generated by the **target gNB**. For its **parameters**, click 3.7.2.

  **Direction**: **target gNB** to **source gNB/source RAN**.

- `HandoverPreparationInformation`

  It is used to **assist inter-RAT** or **intra-RAT handovers**, particularly when the **handover** involves **another network node** (such as an **NG-RAN** node, **eNB**, or **gNB**) that is **not** the **UE**'s **serving cell**. For its **parameters**, click 3.6.3.

- `CG-Config`

  It is used to **transfer** the **secondary cell group radio configuration** as generated by the **SgNB** or **SeNB**. It can **also** be used by a **Central Unit** (**CU**) to request a **distributed unit** (**DU**) to perform certain actions, e.g. to request the **DU** to perform a **new lower layer configuration**.

**Direction**: **secondary gNB** or **eNB** to **master gNB** or **eNB**. Alternatively, **CU** to **DU**.

- `CG-ConfigInfo`
- `MeasurementTimingConfiguration`
- `UERadioPagingInformation`
- `UERadioAccessCapabilityInformation`

  It is used to transfer **UE radio access capability information**, covering **both upload** to, and **download** from, the **5GC**.

  **Direction**: **ng-eNB** or **gNB** to/from **5GC**.

**Each RRC inter-node message** is a **top named data type** (click 5.9.6) in the **RRC ASN.1 specification** (click 5.7.2).

In addition to these **inter-node messages**, **RRC** also defines **NR messages** (click 5.8.5).

For more detail on **RRC inter-node messages**, refer to clause 11 of TS 38.331.

### 5.8.7 RRC UE capability retrieval framework

The **UE** reports its **UE radio access capability** which are static at least when the network requests. The **NR node** can request the **UE** to report about its capability elements based on band information. The **UE capability** can be represented by a **capability ID**, which may be exchanged in **NAS signaling** over the air and in network signaling instead of the **UE** capability structure.

In **integrated access** and **backhaul** (**IAB**), it is **optional** for an **IAB-MT** to support **UE capability retrieval framework** and the **related signaling**. In case **IAB-MT** does **not** support **UE capability Retrieval framework**, **IAB-MT** capabilities are assumed to be known to the network by other means (e.g., **OAM**).

### 5.8.8 RRC transport of NAS messages

**NR** provides reliable in-sequence delivery of **NAS** messages over **SRBs** in **RRC**, except at **handover** where losses or duplication can occur when **PDCP** is re-established. In **RRC**, **NAS messages** are sent in transparent containers. **Piggybacking** of **NAS messages** can occur in the following scenarios:

- At bearer establishment/modification/release in **Downlink**.
- For transferring the initial NAS message during connection setup and connection resume in **Uplink**.

In addition to the integrity protection and ciphering performed by NAS, NAS messages can also be integrity protected and ciphered by PDCP.

Multiple NAS messages can be sent in a single **Downlink** RRC message during PDU Session Resource establishment or modification. In this case, the order of the NAS messages contained in the RRC message shall be in the same order as that in the corresponding NG-AP message in order to ensure the in-sequence delivery of NAS messages.

NG-RAN node may trigger the NAS Non Delivery Indication procedure to report the non-delivery of the non PDU Session related NAS PDU received from the AMF as specified in TS 38.413.

### 5.8.9 RRC UE assistance information

When configured to do so, the **UE** can **signal** the **network** with the **RRC message** `UEAssistanceInformation` from the `UL-DCCH-Message` **class** (click 5.8.5.9):

- If it **prefers** an **adjustment** in the **connected mode DRX cycle length**, for the purpose of **delay budget reporting**.
- If it is experiencing **internal overheating**.
- If it **prefers** certain **DRX parameter values**, and/or a **reduced maximum quantity** of **secondary component carriers**, and/or a **reduced maximum aggregated bandwidth** and/or a **reduced maximum quantity** of **space-multiplexing layers** and/or **minimum scheduling offsets** "$K_0$" and "$K_2$" for power saving purpose.
- If it **expects not** to **send** or **receive** any more **data** in the near future, and in this case, it can provide its **preference** to **transition out** of **RRC_CONNECTED** (click 5.8.2) where this indication may express its **preferred RRC state**, or alternately, it may **cancel** an **earlier indicated preference** to transition **out** of **RRC_CONNECTED**.
- If it **prefers** (**not**) to be provisioned with **reference time information**.
- If it **prefers** to transition **out** of **RRC_CONNECTED** state for **MUSIM operation and** its **preferred RRC state after transition**.
- If it **wants** to include **assistance information** for **setup** or **release** of **gaps** for **MUSIM** operation.
- The **list** of **frequencies** affected by **IDC problems** (refer to clause 23.4 of TS 36.300).
- Its **RRM measurement relaxation status** indicating whether **RRM measurement relaxation criteria** are met or not.
- Its **RLM measurement relaxation status** indicating whether the **UE** is **applying RLM measurements relaxation**.
- Its **BFD measurement relaxation status** indicating whether the **UE** is **applying BFD measurements relaxation**.

NOTE:Only the Frequency Division Multiplexing (FDM) solution as defined for E-UTRA in clause 23.4 of TS 36.300 is used in NR. The requirements on RRM/RLM/CSI measurements in different phases of IDC interference defined in TS 36.300 are applicable except that for NR serving cell, the requirements in TS 38.133, TS 38.101-**1**, TS 38.101-**2**, TS 38.101-**3** apply.

In the second case, the UE can express a preference for temporarily reducing the quantity of maximum secondary component carriers, the maximum aggregated bandwidth and the quantity of maximum **space-multiplexing layers**. In all cases, it is up to the gNB whether to accommodate the request.

For Sidelink, the UE can report SL traffic pattern(s) to NG-RAN, for periodic traffic.

### 5.8.10 RRC message segmentation

An **RRC message** may be **segmented** in case the size of the **encoded RRC message PDU exceeds** the **maximum PDCP SDU size**. **Segmentation** is performed in the **RRC layer** using a **separate RRC PDU** to **carry each segment**. The **receiver reassembles** the **segments** to form the **complete RRC message**. **All segments** of an **RRC message** are **transmitted before sending another RRC message**. **Segmentation** is **supported** in both **Uplink** and **Downlink** as specified in TS 38.331.

## 5.9 Brief overview of ASN.1

**ASN.1** is used to **formally specify data types**, **parameters**, **variables**, **constants** and related elements of **5G** systems, across many TS documents.

**ASN.1** is an **abstract transfer syntax**: it has **no associated semantics**. **ASN.1** only serves to define the **abstract data types** (i.e., the data structure) and the **possible values** of the **data** that will be **conveyed** between **communicating systems** (e.g., each **gNB** and its **UEs**).

The **ASN.1 data type specifications** are **abstract** because they are independent **both** of the **system**'s **internal data representation** (i.e., word length, big/little endian, one's complement, etc) and of the **on-line data representation** (i.e., encoding rules).

**ASN.1 data type specifications** can be used with several **pre-defined encoding rules**, for the on-line transfer of the actual data. In case you want to define specific encoding rules, **ASN.1** provides a framework and tools to define and test them.

An **ASN.1 data type** is a **set of ASN.1 values**. An **ASN.1 value** of a given **ASN.1 data type** is **one element** of the **set** of **values** of that **data type**.

**ASN.1** allows to **define structured data types** by combining different **built-in data types** and **constants** (click 5.9.1) with **named data types** and **constants** (click 5.9.2). These combinations are done using **ASN.1 structuring keywords** (click 5.9.3).

These **structured data types** may have **multiple levels** (click 5.9.4) of **structuring keywords**.

To cope with this potentially very complex definitions **ASN.1** establishes specific **naming** and **identifier uniqueness rules** (click 5.9.5).

The **structured data types** lead to different **hierarchical relationships** (parent/son, ancestor/descendant and top/bottom) between them (click 5.9.6).

The usage of **named data types** in different places of the **ASN.1 specifications** leads to the concepts of **data type instances** and **parameter instances** (click 5.9.7).

In order to better understand the **complex hierarchical structures** of **ASN.1 data types**, **parameters** and **their instances**, it is very useful to introduce the concept of **ancestor trees** (click 5.9.8).

Finally, **other aspects** of **ASN.1** are presented in 5.9.9.

It is important to point out that in some **5G** documents they use the term "**parameter**" while in others they used "**field**" for the **internal components** of **structured data types**. We prefer to use almost only the term "**parameter**" to avoid possible **misunderstandings** from using **two different terms**.

It is also important to note that in some **5G** documents they use the term "**value**" for what we call "**constant**". We prefer to use "**constant**" because we consider it **less prone to confusion** with the **values** contained in **parameters**.

For more detail on **ASN.1**, refer to **ITU-T** recommendations **X.680** (02/2021) and **X.681**.

Source web pages:

- https://luca.ntop.org/Teaching/Appunti/asn1.html
- https://obj-sys.com/asn1tutorial/node12.html
- https://www.oss.com/asn1/resources/asn1-made-simple/types.html
- https://asn1.io/asn1playground/default.aspx (ASN.1 compiler)

### 5.9.1 ASN.1 simple and built-in data types and constants

**Simple data types**

A **simple data type** is the one whose **ASN.1 specification** (click 5.9.6) contains **neither structuring keywords** (click 5.9.3) **nor named** (click 5.9.2) **structured data types** (click 5.9.3).

A **simple ASN.1 specification** (click 5.9.6) of a **data type** is the one that contains **no structuring keywords** (click 5.9.3).

Therefore, the **specification** of a **simple data type** may **only** contain **either one built-in simple data type or one named simple data type**.

**Built-in (simple) data types**

**ASN.1** provides (among others) the following **built-in simple data types**:

- `BOOLEAN`
- `ENUMERATED`
- `INTEGER`
- `REAL`
- `BIT STRING`
- `OCTET STRING`
- `CHARACTER STRING`
- `NULL`

**Simple constants**

An **ASN.1 simple constant** is the one that does not contain any **structuring keyword** (click 5.9.3).

**Built-in (simple) constants**

**ASN.**1 supports **simple built-in constants** for **many** of its **built-in simple data types**. It is therefore possible to use (among others) `BOOLEAN` constants ("`true`" and "`false`"), `INTEGER` **constants** (e.g., "`34`" or "`-12`") or `REAL` constants (e.g., "`3.1415`" or "`-23.45`").

### 5.9.2 ASN.1 named data types and constants

**ASN.1 allows** to assign a **name** to **data type** or to a **constant** using the **assignment operator** "**::=**".

**Named constants**

A **named constant** is a **constant** that has been **assigned** a **name**. **Constant names** are **unique** within **each ASN.1 module**. **Constant names must start** with a **lower-case letter**. A **named simple constant** is **defined** by providing its **name**, its **data type**, and its **value**.

An example of a **named simple constant definition is:**

```
maxPLMN INTEGER ::= 12
```

where the **constant name** is "**maxPLMN**", its **data type** is "**INTEGER**", and its **value** is the built-in integer constant "**12**".

Most of the **RRC ASN.1** `named constants` are defined in clause "*6.4 RRC multiplicity and type constraint values*".

**Named data types**

A **named data type** is a **data type** that has been **assigned** a **name**. **Data type names** are **unique** within **each ASN.1 module**. **Data type names must start** with an **upper-case letter**.

An example of a **named data type** defined with **one built-in data type** is:

```
AdditionalPCI ::= INTEGER
```

where the **data type name** is `AdditionalPCI`.

An example of a **named data type** defined using **one simple built-in data type**, **one built-in constant** and **one named constant** is:

```
AdditionalPCI ::= INTEGER (1..maxNrofPCI)
```

where the **built-in constant** is "**1**" and the **named constant** is "**maxNrofPCI**". The numbers between parentheses indicate the **range of values** that data type "**AdditionalPCI**" may have, which are from **1** to **maxNrofPCI**.

Although it is unusual, **constants may** have the **same name as** the one of a **named data type**, **except** for their **initial letter**, that **must** have a **different case**. For example, "**servingCellMax**" may be a **constant name** while "**ServingCellMax**" may be a **data type name**.

**General remarks**

**Named data types** and **named constants** can be **structured** (click 5.9.3), in addition to **simple**, as we have seen in this section.

### 5.9.3 ASN.1 structured data types and constants, and parameters

The concepts of this section apply to **both nameless** and **named data types**.

A **structured data type** is the one whose **ASN.1 specification** (click 5.9.6) contains **at least one structuring keyword** (click 5.9.3) **or one named** (click 5.9.2) **structured data type** (click 5.9.3).

A **structured ASN.1 specification** (click 5.9.6) of a **data type** is the one that contains **at least one structuring keyword** (click 5.9.3).

Therefore, the **ASN.1 specification** of a **structured data type** must be a **structured ASN.1 specification** and/or **contain** a **named structured data type**.

**Each** of the different **data types** (**simple** or **structured**) within the **structured specification** of a **data type** is called a **parameter** (i.e., a **field**) A **parameter** may have a **name** (a **named parameter**) or **not** (a **nameless parameter**). **Parameter names must start** with a **lower-case letter**.

Some frequently used **ASN.1 structuring keywords** are:

- **SEQUENCE**

  Defines a **list** of **parameters**, which are usually **named**. Each **parameter** is assigned its specific **data type**, regardless of the **data type** of the other parameters. **All** the **parameter names** in the **SEQUENCE must** be **different**, in order to distinguish the different **named** parameters.

  An example of a **structured data type** using a **SEQUENCE keyword** is:

```
SEQUENCE {
  parameter-name-1 INTEGER,
  parameter-name-2 ENUMERATED{on, off},
  parameter-name-3 OCTET STRING,
  parameter-name-4 INTEGER
}
```

  This **SEQUENCE** contains **4 named parameters** (with names **parameter-name-1** to **-4**), that have (as they **must**) **different names**. **Two** of the **parameters** have the **same data type** (**INTEGER**), while the **other two parameters** have **two** other **different data types** (**ENUMERATED** and **OCTET STRING**).

- **SEQUENCE OF**

  Defines a **list** of **nameless parameters**, **all** of them **having** the **same data type**. You can limit the quantity of elements in the **list** to a certain **range** using the **ASN.1** keyword **SIZE** followed by a **range** between parentheses (e.g., **(SIZE (3..8))**). Notice that a **single value** is a **valid** particular case of a range.

  An example of a **specification** of a **structured data type** using a **SEQUENCE OF keyword** is:

```
SEQUENCE (SIZE (4..10)) OF INTEGER
```

  This **SEQUENCE OF** contains from **4** to **10** **nameless parameters**. **All** the **parameters** have (as they **must**) the **same data type**, which in this example is **INTEGER.**

- **CHOICE**

  Defines **one parameter** out of **several parameters**, which are usually **named**. Each **parameter** is assigned its specific **data type**, regardless of the **data type** of the other parameters.

An example of a **specification** of a **structured data type** using a **CHOICE** is:

```
CHOICE {
  option-1 INTEGER,
  option-2 ENUMERATED{on, off},
  option-3 OCTET STRING,
  option-4 INTEGER
}
```

This **CHOICE** contains **4 named parameters** (option-1 to -4), that have (as they **must**) **different names**. **Two** of the **parameters** have the **same data type** (**INTEGER**), while the **other two parameters** have **two** other **different data types** (**ENUMERATED** and **OCTET STRING**).

- **SET and SET OF**

  They are similar to **SEQUENCE** and **SEQUENCE OF**, but without order (a **set** instead of a **list**). The usage of **SET** and **SET OF** is considered undesirable because they provide a similar functionality to **SEQUENCE** and **SEQUENCE OF**, but with slower encoding.

Notice that **each parameter** within a **SEQUENCE**, **SET** or **CHOICE** usually has a **parameter name**: they are **named parameters**. However, the **parameters** within a **SEQUENCE OF** or a **SET OF do not have** a **parameter name**: they are **nameless** parameters.

Notice also that the **parameter names** within **each and every SEQUENCE**, **SET** or **CHOICE must** be **different**. However, the **parameter names** of **different SEQUENCE**, **SET** or **CHOICE**, **may** be **duplicated**.

**Parameters may** have the **same name as** the one of a **named data type**, **except** for their **initial letter**, that **must** have a **different case**. For example, "servingCellConfigCommon" is a **parameter name** while "ServingCellConfigCommon" is a **data type name**. This happens **very frequently** in **5G ASN.1 specifications**, so is important to be aware in order to avoid misunderstandings.

**Structured constants**

A **structured constant** is the one whose **specification** (click 5.9.6) contains **at least one structuring keyword**.

An example of a **structured specification** of a **constant** is:

```
SEQUENCE {
  parameter-name-1 7,
  parameter-name-2 off,
  parameter-name-3 OCTET STRING,
  parameter-name-4 '0110101000101010'B
}
```

This **structured specification** of a **constant** may be used as a **DEFAULT value**, or in the **specification** of a **named constant**.

**Named structured data types and constants**

**Named data types** and **named constants** can be **structured**, in addition to **simple** (click 5.9.2).

An example of a **specification** of a **named simple data type** is:

```
Name-of-Named-Type ::= SEQUENCE (SIZE (4..10)) OF INTEGER
```

An example of a **structured specification** of a **named data type** is:

```
Name-of-Named-Type ::= SEQUENCE {
  parameter-name-1 INTEGER,
  parameter-name-2 ENUMERATED {on, off},
  parameter-name-3 OCTET STRING,
  parameter-name-4 INTEGER
}
```

An example of a **structured specification** of a **named constant** is:

```
constantName ::= Name-of-Named-Type SEQUENCE {
  parameter-name-1 7,
  parameter-name-2 off,
  parameter-name-3 OCTET STRING,
  parameter-name-4 '0110101000101010'B
}
```

**Optional parameters**

An **optional parameter** is the one whose **assigned data type** is **followed** by the **keywords** "**OPTIONAL**" or "**DEFAULT**". An **optional parameter may**, or **may not**, be included by the sender of a value belonging to the corresponding data type.

The "**DEFAULT**" keyword **must** be followed by a **constant**, which is the **parameter**'s **default value**. When **DEFAULT** is used, and the sender did not include the data type in a given message, the receiver should use for the **parameter** its **default value** in the **ASN.1 specification**.

An example of a **specification** of a **structured data type** with an **optional** parameter is:

```
SEQUENCE {
  parameter-name-1 INTEGER,
  parameter-name-2 ENUMERATED {on, off} DEFAULT on,
  parameter-name-3 OCTET STRING,
  parameter-name-4 INTEGER
}
```

where parameter **parameter-name-2** is **optional**, and its default value is "**on**".

## 5.9.4 ASN.1 multilevel structured specification of data types and constants

The concepts of this section apply to **both nameless** and **named data types**.

The **data types** of the **parameters** in the **structured specification** of a **data type** (click 5.9.3) or a **constant may also be structured specifications**. This implies that the **structured specification** of a **data type** may have **several nested structuring keywords**.

The **quantity of levels** of the **structured specification** of a **data type** is the **largest quantity** of **nested structuring keywords that it contains**.

An example of a **single-level structured specification** of a **data type** is:

```
SEQUENCE {
  plmn-IdentityList PLMN-IdentityList,
  trackingAreaCode TrackingAreaCode,
  ranac RAN-AreaCode,
  cellIdentity CellIdentity
}
```

This **structured specification** contains **only one structuring keyword** (**SEQUENCE**, click 5.9.3). The **data types** of the **four parameters** are **four different named data types** (in purple). These **named data types** are **defined elsewhere within** this same **ASN.1 module**. One or more of these **named data types** may be **multi-level structured data type**, but this does not change that the **structured specification** of this **data type** is a **single-level structured specification**.

An example of a **two-level structured specification** of a **data type** is:

```
SEQUENCE {
  rrc-TransacID RRC-TransacID,
  criticalExt CHOICE {
    mobilityComm MobilityComm-IEs,
    futureExt INTEGER
  }
}
```

Its **first level** is a **SEQUENCE** and contains **parameters** **rrc-TransacID** and **criticalExt**. The **data type** of **parameter** **rrc-TransacID** is the **named data type** **RRC-TransacID**. The **data type of** **criticalExt** has a **structured specification** using **CHOICE**, which creates **one second level**. This **second level**, within **criticalExt**, contains **parameters** **mobilityComm** (with **named data type** **MobilityComm-IEs**) and **futureExt** (an **INTEGER**).

In the general case, there may be **several second levels**, where **each second level** may have **several third levels**, and so on. Therefore, a **structured specification** of a **data type** has a **tree** structure of **structuring keywords**.

### 5.9.5 ASN.1 identifier, names, uniqueness and duplicates

Most **ASN.1 identifiers**, including **data type names**, **parameter names** and **value names**, are **case sensitive**. However, the **ASN.1 module** identifiers and **object** identifiers are **case-insensitive**. For example, **DataType** and **Datatype** are **different data types**, but **MyObject** and **myobject** are **the same object**.

**Data type names must start** with an **upper-case letter**, while **constant names** and **parameter names must start** with a **lower-case letter**.

A **data type name may be** the **same** as a **parameter name**, **except** the **case** of their **first letter** that **must** be **different**. Pay attention to this to avoid confusion.

**Data type names** and **constant names** are **unique** within **each ASN.1 module**.

However, **parameter names must only be unique** within **each individual structuring keyword** (e.g., within each **SEQUENCE**), click 5.9.3. Therefore, it is possible to have **duplicated parameter names** within the **structured specification** of a **data type**, as long as the said restriction is complied with. For example, the following **structured specification** is **correct**:

```
SEQUENCE {
  scs INTEGER,
  scsList-1 SEQUENCE {
    scs INTEGER
  },
  scsList-2 SEQUENCE {
    scs INTEGER
  },
}
```

Although having **duplicate parameter names within** a **structured specification** is **syntactically correct** (as shown in this last example), it is **highly undesirable**. It is therefore a **best practice** to have **unique parameter names within each structured specification** of a **data type**.

Even following the said best practice, **parameter names** may be **duplicated** between the **specifications** of two (or more) **different named data types** within the same **ASN.1 module**. Actually, within the **RRC ASN.1 specifications** of TS 38.331, the said **best practice** is **followed**, and **yet** there are **many duplicated parameter names** enclosed within **different named data types**. For more detail on **same-name** and **version-different parameters**, in the **5G RRC specification** in TS 38.331, click 5.7.2.2.

## 5.9.6 ASN.1 data types: specification, parent/son, ancestor/descendant and top/bottom

The concepts of this section apply to **both** to **nameless** and **named data types**, although they are **most frequently used** for **named data types**.

The combination of **named** and **nameless data types**, with **simple** and **structured data types** leads to **various** cases of **specification** of a **data type** (click 5.9.6.1).

This also leads to different **hierarchical relationships** between **data types**, such as **parent/son** (click 5.9.6.2), **ascendant/descendant** (click 5.9.6.3) and **top/bottom** (click 5.9.6.4).

### 5.9.6.1 Specification of a data type

The **specification** of a **data type** is the **ASN.1 code** that **defines** the **data type**.

Notice that a "**specification** of a **data type**" is the **same** as a "**data type specification**".

Depending on whether the **data type** is **either named or nameless**, and whether a **nameless data type** is assigned either to a **named parameter** or to a **nameless parameter**, we may have the following **three cases**:

- **Specification of a named data type**

  The **ASN.1 code** that defines a **named data type** follows the **data type name** assigned to it, after the **keyword** "::=".

- **Specification of a nameless data type assigned to a named parameter**

  The **ASN.1 code** that defines the **nameless data type** follows the **parameter name** to which it has been **assigned**.

- **Specification of a nameless data type assigned to nameless parameters**

  The **ASN.1 code** that defines the **nameless data type** follows the **keyword** (e.g., **`SEQUENCE OF`** or **`SET OF`**) that **creates** the **nameless parameters** to which the **nameless data type** has been assigned.

Depending on whether the **data type specification** is **either simple or structured**, and whether the **data type itself** is **either simple or structured**, we may have the following **three cases**:

- **Simple specification** of a **nameless simple data type** (assigned to a **named parameter**)

  It is a **simple specification** (click 5.9.1) of a **simple data type** (click 5.9.1).

- **Structured specification** of a **nameless structured data type** (assigned to a **named parameter**)

  It is a **structured specification** (click 5.9.3) of a **structured data type** click 5.9.3).

- **Simple specification** of a **nameless simple or structured data type** (assigned to a **named parameter**)

  It is a **simple specification** (click 5.9.1) of a **simple** (click 5.9.1) **or structured** (click 5.9.3) **data type**.

- The case of a **structured specification** of a **nameless simple data type cannot exist**.

Note that a "**simple specification** of a **data type**" is **not** the **same** as a "**specification** of a **simple data type**". Likewise, a "**structured specification** of a **data type**" is **not** the **same** as a "**specification** of a **structured data type**". See the different cases and examples above. Also, recall from 5.9.3 that the **specification** of a **data type may include data type names**.

The **three former cases** above, in **combination** with the **three later cases** above, are now explained with some **examples**:

- **Specification of a named data type**

  The **ASN.1 code** that defines a **named data type** follows the **data type name** assigned to it, after the keyword "::=".

  Depending on whether the **specification** of the **named data type** is **simple** or **structured**, and whether the **named data type itself** is **simple** or **structured**, we may have

  - **Simple specification** of a **named simple data type**

    An example is:

    ```
    DataTypeName := INTEGER
    ```

  - **Structured specification** of a **nameless structured data type** (assigned to a **named parameter**)

    Two examples are:

    ```
    DataTypeName := SEQUENCE {parA INTEGER ; parB BOOLEAN}

    DataTypeName := SEQUENCE {parA INTEGER ; parB InstantiatedType}
    ```

  - **Simple specification** of a **nameless simple or structured data type** (assigned to a **named parameter**)

    An example is:

    ```
    DataTypeName := InstantiatedType
    ```

This case is a little bit tricky. On the one hand, the **ASN.1 code** that defines the **named data type** `DataTypeName` is **not** the definition of `InstantiatedType` (which is performed elsewhere), but just the **nameless instantiation** of `InstantiatedType`: compare this case with the second example from the previous bullet. On the other hand, the **named data type** `DataTypeName` is **simple** or **structured** if the **instantiated named data type** `InstantiatedType` is **simple** or **structured** (respectively).

- **Specification of a nameless data type assigned to a named parameter**

  The **ASN.1 code** that defines the **nameless data type** follows the **parameter name** to which it has been **assigned**.

  Depending on whether the **specification** of the **nameless data type** is **simple** or **structured**, and whether the **nameless data type itself** is **simple** or **structured**, we may have

  - **Simple specification** of a **nameless simple data type** (assigned to a **named parameter**)

    An example where the **nameless data type** is assigned to **named parameter** `parameterName` is:

    ```
    parameterName INTEGER
    ```

  - **Structured specification** of a **nameless structured data type** (assigned to a **named parameter**)

    Two examples where the **nameless data type** is assigned to **named parameter** `parameterName` are:

    ```
    parameterName SEQUENCE {parA INTEGER ; parB BOOLEAN}

    parameterName SEQUENCE {parA INTEGER ; parB InstantiatedType }
    ```

  - **Simple specification** of a **nameless simple or structured data type** (assigned to a **named parameter**)

    An example where the **nameless data type** is assigned to **named parameter** `parameterName` is:

    ```
    parameterName InstantiatedType
    ```

    This case is a little bit tricky. On the one hand, the **ASN.1 code** that defines the **nameless data type** assigned to `parameterName` is **not** the definition of the **instantiated named data type** `InstantiatedType` (which is performed elsewhere), but just the **nameless instantiation** of `InstantiatedType`: compare this case with the second example from the previous bullet. On the other hand, the **nameless data type** assigned to `parameterName` is **simple** or **structured** if the **instantiated named data type** `InstantiatedType` is **simple** or **structured** (respectively).

- **Specification of a nameless data type assigned to nameless parameters**

  The **ASN.1 code** that defines the **nameless data type** follows the **keyword** (e.g., `SEQUENCE OF` or `SET OF`) that **creates** the **nameless parameters** to which the **nameless data type** has been assigned.

This case may **only** happen **nested** within any of the **three cases** presented in these three bullets. The reason is that the **keyword** that **creates** the **nameless parameters cannot exist** by **itself** and **must rather** be **within the specification** of an **enclosing data type**.

Depending on whether the **specification** of the **nameless data type** is **simple** or **structured**, and whether the **nameless data type itself** is **simple** or **structured**, we may have

- **Simple specification** of a **nameless simple data type** (assigned to **nameless parameters**)

  An example where the **nameless data type** is assigned to **nameless parameters** enclosed **within** `enclosingParam` is:

  ```
  enclosingParam SEQUENCE OF INTEGER
  ```

- **Structured specification** of a **nameless structured data type** (assigned to **nameless parameters**)

  Two examples where the **nameless data type** is assigned to **nameless parameters** enclosed **within** `enclosingParam` are:

  ```
  enclosingParam SEQUENCE OF
                   SEQUENCE {parA INTEGER ; parB BOOLEAN}

  enclosingParam SEQUENCE OF
                   SEQUENCE {parA INTEGER ; parB InstantiatedType}
  ```

- **Simple specification** of a **nameless simple or structured data type** (assigned to **nameless parameters**)

  An example where the **nameless data type** is assigned to **nameless parameters** enclosed **within** `enclosingParam` is:

  ```
  enclosingParam SEQUENCE OF InstantiatedType
  ```

  This case is a little bit tricky. On the one hand, the **ASN.1 code** that defines the **nameless data type** assigned to the **nameless parameters** created by `SEQUENCE OF` is **not** the definition of the **instantiated named data type** `InstantiatedType` (which is performed elsewhere), but just the **nameless instantiation** of `InstantiatedType`: compare this case with the second example from the previous bullet. On the other hand, the **nameless data type** assigned to the **nameless parameters** created by `SEQUENCE OF` is **simple** or **structured** if the **instantiated named data type** `InstantiatedType` is (respectively) **simple** or **structured**.

### 5.9.6.2 Parent and son data types

**Named data type** "**P**" is a **parent** of **named data type** "**S**" if "**S**" **appears** in the **ASN.1 specification** of "**P**".

If "**P**" is a **parent** of "**S**", then "**S**" is a **son** of "**P**", and **vice versa**.

A **parent** is sometimes called a **direct ancestor**. A **son** is sometimes called a **direct descendant**.

#### 5.9.6.3 Ancestor and descendant data types

**Named data type D** is a **descendant** of **named data type A** if **either D** is the **son** of **A or D** is a **descendant** of **any** of the **children named data types** of **A**. Notice that this is a recursive definition.

If **D** is a **descendant data type** of **A**, **then A** is an **ancestor data type of D**, then, and **vice versa**.

#### 5.9.6.4 Top and bottom data types

A **top data type** is the one that is **not** a **descendant** of **any other data type** in the **ASN.1** module. An **ASN.1** module may have **one**, or **more**, **top data types**. In the **RRC ASN.1 specification** in TS 38.331, there are several **top named data types** which are the various **RRC message classes** (click 5.8.5), the **RRC inter-node messages** and the **UE variables**.

A **bottom data type** is the one that does **not have any descendants**. An **ASN.1** module almost always has **several bottom data types**.

When there is no risk of confusion, the term "**top data type**" may be shortened to "**top**" and the term "**bottom data type**" may be shortened to "**bottom**".

### 5.9.7 ASN.1 data type and parameter instances and their quantity

The concepts of this section apply to **both** to **nameless** and **named data types**, although they are **most frequently used** for **named data types**.

**Data type occurrences**

An **occurrence** $\mathbf{f_{ki}(A_k(D))}$ of **data type D** within one of its **ancestor data types** $\mathbf{A_k(D)}$ is **each** of the **different parameters** $\mathbf{f_{ki}}$, within the **expansion** of the **ASN.1 definition** of $\mathbf{A_k(D)}$ **down** to **D**, to which it has been **assigned** the data type **D**.

Notice that we say "**parameters** $\mathbf{f_{ki}}$" and we **do not** say "**parameter names** $\mathbf{f_{ki}}$", which is **very different**. What we want to **highlight** is that **parameters** $\mathbf{f_{ki}(A_k(D))}$ may have **different parameter names** or be **nameless**.

**Data type instances**

An **instance** of **data type D** within one of its **ancestor data types** $\mathbf{A_k(D)}$ is **each** of the **different values** of **data type D** that **may be** found within **one value** of $\mathbf{A_k(D)}$.

The **instances** of **data type D** within $\mathbf{A_k(D)}$ are **not the same** as the **occurrences** $\mathbf{f_{ki}(A_k(D))}$ of **D** within $\mathbf{A_k(D)}$. The reason is that **each occurrence** $\mathbf{f_{ki}(A_k(D))}$ of **D may be converted** to **multiple instances** by **each** upstream **`SEQUENCE OF`** (or **`SET OF`**) and may also be **pruned** by **every** upstream **`OPTIONAL`** or **`CHOICE`** **keyword**.

An **ancestor data type** $\mathbf{A_k(D)}$ of **data type D** may contain **many different instances** of **D**.

**Each** and **every** individual **instance** of the **ancestor data type** $\mathbf{A_k(D)}$ of **D**, within **one** of its **ancestor[2] data types** $\mathbf{A_h(A_k(D))}$, contains **all the instances** of **data type D**, that exist within the **ancestor type** $\mathbf{A_k(D)}$.

**Parameter instances**

The **instances** of a **parameter** are the ones of **its enclosing data type**.

**Quantity of data type instances: addition, multiplication or selection**

The **quantity** of **instances** of **data type D** within one of its **ancestor data types** $A_k$(**D**) is **determined** by **each** of the **occurrences** $f_{ki}(A_k(A))$ of **D**, within **expansion** of the **ASN.1 definition** of $A_k$(**D**) **down** to **D**, **and**, by **all** the **keywords** (e.g., **`SEQUENCE OF`**, **`DEFAULT`**, …) encountered by **each** such **occurrence** in its **upstream path** up to the **root** of $A_k$(**D**).

Because the expansion of $A_k$(**D**) may contain **optional parameters** and **structuring keywords**, the quantity of **instances** of **data type D** within $A_k$(**D**) is, in the general case, a **set** of **subsets** of **integers**. **Each integer** represents the **quantity** of **instances** of **D**. **Each subset** represents one of the **options** arising from the **optional** parameters and the **`CHOICE`** keywords.

To determine the **quantity** of **instances** of **data type D** within $A_k$(**D**) we associate to **each** of its **occurrences one set** of **subsets**, containing **one subset**, that contains only **one element**: the integer "**1**". In case the **parameter** of the **occurrence** is **optional**, include in the set **another subset** with the **integer** "**0**".

Whenever one of the following **keywords** is **encountered** along the **upstream path**, a **resulting set** is produced by **addition**, **multiplication**, **union** or **subset inclusion** over the **arriving set**(**s**) as follows:

- **`SEQUENCE`** or **`SET`**
  - If **only one set** arrives to the **keyword** from its **different branches**

    The **resulting set** is that **arriving set**.
  - If **k sets** arrive from **k** branches,

    The **resulting set** has **one resulting subset** for **each combination of k arriving subsets**, taking **one arriving subset** from **each** of the **k arriving sets**. **Aach** such **resulting subset** has **one resulting element** for **each combination** of **k arriving elements** taking **one element** from **each** of the **k** arriving **subsets** in its **associated combination** of **k arriving subsets**. The **value** of **each such element** is the **addition** of the **k elements** in its **associated combination** of elements.
- **`SEQUENCE OF`** or **`SET OF`**

  The **resulting set** has the **same quantity** of **resulting subsets** as the **arriving set**. Each of the **resulting subsets** has **as many elements** as the **quantity of elements** in the **`SIZE`** clause **multiplied** by the **quantity of elements** of its **corresponding arriving subset**. The **value** of **each such element** is the **multiplication** of **each element** in the **`SIZE`** clause and **each element** in the **corresponding arriving subset**.
- **`CHOICE`**

  The **resulting set** is the **union** of all the **arriving sets** from the **different branches**. If one or more branches **do not contain** any **occurrence** of **D**, the **resulting set** also includes a **subset** with **only one element**: the **integer** "**0**".
- **`OPTIONAL`** or **`DEFAULT`**

  The **resulting set** is the **arriving set** after having included **one subset** with **only one element**: the integer "**0**".

The **quantity** of **instances** of **D** within $\mathbf{A_k(D)}$ is **the resulting set** at the **top keyword** in the definition of $\mathbf{A_k(D)}$.

A **group** of **instances** of **D** within one of its **ancestor data types** $\mathbf{A_k(D)}$ is **optional** if its originating **occurrence** $\mathbf{f_{ki}(A_k(D))}$ is **optional** (i.e., if the **data type** of the parameter $\mathbf{f_{ki}}$ is **followed** by `OPTIONAL` or `DEFAULT`, click 5.9.3). A **group** of **instances** of **D** within one of its **ancestor data types** $\mathbf{A_k(D)}$ is **optional** if its originating **occurrence** $\mathbf{f_{ki}(A_k(D))}$ is **enclosed** within **any** upstream **optional** parameter and/or within **any** upstream `CHOICE`, up to the root of $\mathbf{A_k(D)}$. An instance of **D** is **mandatory** if it is **not optional**. Notice that for a given **group** of **instances** of **D** to be **mandatory**, you have to **analyze** the **complete upstream path** from its originating **occurrence** $\mathbf{f_{ki}(A_k(D))}$ up to the root of $\mathbf{A_k(D)}$. Some **groups** of **instances** of **D** may be **optional**, while other **groups** may be **mandatory**.

**Quantity of parameter instances**

The **quantity** of **instances** of a **parameter** are the ones of **its enclosing data type**.

### 5.9.8 ASN.1 data type and parameter ancestor trees

The concepts of this section apply **only** to **named data types** and **not** to **nameless data types**.

**Data type ancestor trees**

The **ancestor** (click 5.9.6) **trees** $\mathbf{AT(l_1..l_n; L_1..L_n)}$ of **one** or **more named data types** $\{\mathbf{l_1}, \mathbf{l_2}, \ldots, \mathbf{l_n}\}$ (called the **leaves**) up to **zero** or **more other named data types** $\{\mathbf{L_1}, \mathbf{L_2}, \ldots, \mathbf{L_n}\}$ (called the **limits**) consist of **one** or **more trees**, such that **each tree** represents **all** the **ancestors** from **each leave** up to the **highest possible limit**.

The **ASN.1 module**, the **leaves** and the **limits** define the resulting **ancestor trees**. In case the **limits** are **not** defined, then every **tree** is built up to a **top** (click 5.9.6) **data type name** which is the **tree root**.

**Each ancestor tree** is represented as a series of **consecutive text lines**, where **each line** contains **one data type name preceded** by its corresponding **parameter name** and **followed** by the possible **quantity** of **instances** (click 5.9.7) of that **data type name** within its **direct ancestor** (i.e., the **data type name** in the **previous line**). Each **tree leaf** is one of the **leaves** and each **tree root** is the **highest possible limit**. Each **parameter name** is **indented** two spaces in respect to the **parameter name** of its **parent data type**. Each **data type name** is **indented** two spaces in respect its **parent data type name**.

The **ancestor trees** are built **bottom-up**, starting from **each** of the **leaves** up to the **highest possible limit** of **each tree**.

**Parameter ancestor trees**

The **ancestor trees** of a **parameter** are the ones of **its enclosing data type**.

**Characteristics of ancestor trees**

Within the above definition of **ancestor trees**, you may have some **particular cases**, such as the following:

- **Ancestor trees of a few related leaves**

  You obtain it by specifying as **leaves** a **few data type names** that **jointly provide** the configuration of a **certain physical layer element** (e.g., **PRACH**, or

**PUCCH**), and you do **not** specify any **limit**. This usually results in quite focused **ancestor trees**, with a reasonable extension.

- **Ancestor trees of a few related leaves up to a few limits**

  You obtain it by specifying as **leaves** a **few data type names** that **jointly provide** the configuration of a **certain physical layer element** (e.g., **PRACH**, or **PUCCH**), and as **limits** a few of their **ancestors** (click 5.9.6) to prevent the size of the **trees** becoming excessively large. This usually results in very focused **ancestor trees**, with a reasonable extension.

- **Complete ancestor trees**

  You obtain it when the **leaves** are **all** the **bottoms** (click 5.9.6), and you do **not** specify any **limit**. This tends to result in **many large** and **complex trees**, so it is usually **not** very **useful**.

Some **relevant remarks** on the **ancestor trees** are:

- The **ancestor trees** "**AT(a, b, c)** " of **leaves** "{**a**, **b**, **c**}" and **no limits**, **also include** the **ancestor trees** "**AT(x, y, z)** " and **no limits** of **every named data type** "{**x**, **y**, **z**}" that appears within "**AT(a, b, c)** ". This is an extremely important property.
- If we specify **only one limit**, the result is **one ancestor tree** (or zero, in the degenerate case that the **limit** is not an ancestor of any **leave**).
- **Parameter names** and **data type names** are **indented** two spaces among themselves.
- **Parameter name indentation** and **data type indentation** are **aligned**. This implies that when **two parameter names** are **aligned**, the **data type names** in their lines are **also aligned**, and **vice versa**.
- **All** the **text lines** with the **same indentation**, and **without** any **intermediate text line** that is **outdented** in respect to them, represent **branches** in the **tree**. The **tree node** of these **branching text lines** is the **text line directly above** of the **topmost branching text line**.
- The **parameter name** in each **text line** is **assigned either** the **data type name** that is written in its **same line**, or a **`SEQUENCE OF`** that **data type name**.
- A structured data type with a **`SEQUENCE`**, a **`CHOICE`** or several **combinations** of **either** or **both**, is shown with as many **text lines** as the **quantity of times** that the **inner data type name** appears within the whole, each with its corresponding **parameter name**.
- A **`SEQUENCE OF`** is shown by **writing** its **range** to the **right** of the **data type name**. This is **equivalent** to writing **as many copies** of the **tree below this line** as **each** of the **values** in the **range**.
- In every [**sequence** of **text lines indented** two spaces in respect to the **previous text line**], the [**data type name** in **each text line**] is [the **son** of the **data type name** in the **previous text line**} and it is **also** [the **parent** of the **data type name** in the **next text**].
- If you **graft** the **ancestor tree AT($l_1$..$l_n$; L)** of **leaves** {$l_1$, $l_2$,... ,$l_n$} up to **one limit** {**L**}, into **all** the **leaves** of **all** the **ancestor trees AT(L; $L_1$..$L_n$)** with whatever

**limits** (or no **limits**), you **might not get** the **ancestor trees AT($l_1$..$l_n$; $L_1$..$L_n$)**. The reason is that there may be **trees** containing some of the **leaves** {$l_1$, $l_2$,... ,$l_n$ }, **without** containing "***L***", and **rooted** in **data type(s)** contained in **AT(L; $L_1$..$L_n$)**. The said **trees are not** included in the **ancestor tree AT($l_1$..$l_n$; L)**, they **must** be included in the **ancestor tree AT($l_1$..$l_n$; $L_1$..$L_n$)**, and they **would not** be included if applying the **grafting**.

To better understand this procedure, we copy below the **ancestor trees** of `CellGroupConfig`, **up to** the **top RRC NR message classes** (as explained in 3.7.1):

```
                                DL-DCCH-Message (click 5.8.5.4)
message                           DL-DCCH-MessageType (click 5.8.5.4)
  c1                                CHOICE
    rrcReconfiguration              RRCReconfiguration (0,1) (click 5.8.5.4)
      rrcReconfiguration              RRCReconfiguration-IEs (click 5.8.5.4)
        secondaryCellGroup              CellGroupConfig (click 3.7.3)
        nonCriticalExtension            RRCReconfiguration-v1530-IEs (0,1)
          masterCellGroup                 CellGroupConfig (click 3.7.3)
    rrcResume                       RRCResume (0,1) (click 5.8.5.4)
      rrcResume                       RRCResume-IEs
        masterCellGroup                 CellGroupConfig (click 3.7.3)

                                DL-CCCH-Message (click 5.8.5.3)
message                           DL-CCCH-MessageType (click 5.8.5.3)
  c1                                CHOICE
    rrcSetup                        RRCSetup (0,1) (click 5.8.5.11)
      criticalExtensions              RRCSetup-IEs
        masterCellGroup                 CellGroupConfig (click 3.7.3)
```

Some **relevant remarks** on **this example** of **ancestor trees** are:

- Be aware that the **ASN.1** statement "`OCTET STRING (CONTAINING RRCReconfiguration)`" may be found in **several RRC data types**. This means that `RRCReconfiguration` is the **son** of all that **data types**. However, this has **not** been **shown** in the **ancestor trees** above for conciseness's sake.
- This example contains **two blocks** of contiguous lines. **Each** such block is one **ancestor tree**. The **root** of the first **tree** is the **RRC message class** `DL-DCCH-Message` and the **root** of the second **tree** is the **RRC message class** `DL-CCCH-Message`.
- One case of **branching** has as node "`message DL-DCCH-MessageType`" and its **two branches** are "`rrcReconfiguration RRCReconfiguration`" and "`rrcResume RRCResume`".
- One case of a `SEQUENCE OF` happens at parameter name `sCellToAddModList`, which contains a `SEQUENCE OF` data type `SCellConfig` (click 3.9.4), and its `SIZE` is `(1..maxNrofSCells)`. Notice that `maxNrofSCells` is a **constant** defined univocally in the **ASN.1** module.
- **Each tree root** (**each first line**) **never** has a **parameter name**, and it **only** has a **data type name**.

### 5.9.9 ASN.1 other aspects

**Contained data types**

**ASN.1** allows to specify that a given **OCTET STRING** data type **must contain** an **encoded value** of a **specific data type**. For example:

```
nr-SCG OCTET STRING (CONTAINING RRCReconfiguration)
```

This is **roughly equivalent** to specifying that the corresponding parameter name (**nr-SCG**) has been assigned the corresponding data type (**RRCReconfiguration**).

**Parameterized data types**

**ASN.1** supports the specification of **parameterized data types**. One relevant case of an **RRC** parameterized data type is "**SetupRelease**", defined in clause 6.3.0, as follows:

```
SetupRelease { ElementTypeParam } ::= CHOICE {
  release NULL,
  setup ElementTypeParam
}
```

This parameterized data type **SetupRelease** allows the data type **ElementTypeParam** to be used as the **referenced data type** for the **setup** and **release** entries. Refer to A.3.8 for guidelines.

**Extension marker**

In **ASN.1**, an ellipsis "**. . .**" is used to indicate that a "**SEQUENCE**" or "**SET**" data type can have **additional** elements beyond those explicitly listed in the **ASN.1 specification**. This is known as an **extension marker** and is used to allow for **forward compatibility** and **extensibility** of data structures. The **extension marker** is **usually** placed in a text line just **by itself** (e.g., with no other **ASN.1** code).

**Version brackets**

In **ASN.1**, a **pair** of **adjacent left** or **right square brackets** ("**[[**" or "**]]**") indicate the **start** and **end** (respectively) of an **extension addition group**. These double square brackets are called "**version brackets**" because they are usually used to **add new features** in **successive versions** of a given **ASN.1** specification. The right **pair of left brackets** can optionally be followed by a **number** that assigns a **version number** for the **extension addition group**. Each **pair** of **version brackets** is **usually** placed in a text line just **by itself** (e.g., with no other **ASN.1** code).

**Comments**

**All the text** between **two hyphens** "**--**" and the **line end** is an **ASN.1** comment.

**All the text** between "**/***" and "***/**, including multiple lines, is an **ASN.1** comment. This type of **ASN.1 comments** may be **nested**.

The **ASN.1 comments** and **their deliming characters** have no semantics in **ASN.1** and are ignored.

However, **5G documents** do assign **semantics** to some **ASN.1 comments** (click 5.7.3.6).

# 6. NR Downlink physical channels and signals

## 6.1 SSB (PSS, SSS and PBCH)

For an **overview** of the **synchronization signals and PBCH block** (**SSB**), click 6.1.1.

For the **SSB scheduling**, click 6.1.2.

For the **SSB contents**, click 6.1.3.

For the **SSB processing**, click 6.1.4.

### 6.1.1 SSB overview

The **Synchronization Signal and PBCH Block** (usually called **SSB** ) is a **transmission structure** that jointly conveys the **Primary Synchronization Signal** (**PSS**, click 6.19), the **Secondary Synchronization Signal** (**SSS**, click 6.19) and the **Physical Broadcast Channel** (**PBCH**, click 6.2).

The functions of **SSB** are the **ones** of its **components**: **PSS**, **SSS**, and **PBCH**. One relevant function of **SSB** is **beam selection** (click 6.1.2.3).

**Each SSB transmission** takes place in **1** of the **transmission occasions determined** by the **SSB scheduling** (click 6.1.2). **Each slot** can contain **0**, **1** or **several SSB transmissions** (click 6.1.2.2).

**Each SSB** is composed of **4 consecutive symbol intervals** over **20 contiguous Resource Blocks** (i.e., groups of **12 consecutive subcarriers**). The corresponding **240 SSB subcarriers** are indexed from **0** (lowest frequency) to **239** (highest frequency). Depending on the **contents** of **PBCH**, there are **2 types** of **SSB**. For more detail on **SSB contents**, click 6.1.3.

**Each SSB transmission** is subject to a specific **processing**, click 6.1.4.

**SSB** does **not** have **any configuration parameter**, because for **initial access** (click 8) the **UE** has to be able to receive **SSB before any** type of **UE configuration** has taken place.

The **SSB** is sometimes called the "**SS/PBCH block**".

In **LTE** each **SSB** is **2 consecutive symbols** over 7•12=**84 contiguous** subcarriers. Several consecutive **SSBs** are transmitted every **10 ms** (in the first subframe of every frame). If you check the **LTE** channel bandwidths, **all** of them can carry the mandatory **SSB**. The **82 contiguous SSB** subcarriers are **always** located in the **central** part of each **LTE** carrier.

### 6.1.2 SSB scheduling

**Every cell**, including **secondary cells**, **should broadcast** the **SSB**, both for the **UE** to know the **PCI** of the **cell** and for **transmission synchronization** purposes. This is why the **SSB** is the **only always-on** signal in a **cell**. In **secondary cells**, including the **primary-secondary cell**, the **SSB** might **not** contain the **configuration** of **CSS-Type 0** (click 6.8.6), because `SIB1` (click 5.3.4) may be provided through **RRC** (click 5.8) over the **primary cell** (click 3.2.2). Notice though, that there are some **exceptions** of **cells** that do **not broadcast** (click 3.2.1) the **SSB**.

**SSB Tx scheduling**

The **NR node periodically broadcasts** the **SSB** in a **frequency location** that **belongs** to the **synchronization raster**. For more detail on the **frequency Tx scheduling** of **SSB**, click 6.1.2.1. For more detail on the **time Tx scheduling** of **SSB**, click 6.1.2.2.

**SSB Rx scheduling**

**SSB Rx scheduling** is different during **cell search** (click 8.2) and **after** the **RRC** (click 5) **connection establishment** (click 8.11):

- **SSB Rx scheduling during cell search**

  It is a **built-in scheduling** (click 4.11.4.3). The **UE tries**, one after another, **all** the **reference frequencies** in the **synchronization raster** (click 6.1.2.16.1.2.1). Within **each reference frequency**, the **UE continuously scans** for the reception of the **primary synchronization signal** (**PSS**, click 6.19) over a given **antenna port** (click 4.11.5). This **scanning** lasts for a **certain time**, **after** which the **UE** passes to **scan** in **another reference frequency**.

- **SSB Rx scheduling after** the **RRC connection establishment**

  It is a **semi-persistent scheduling** (click 4.11.4.3). The **UE** is configured (using **RRC**) with the **frequency location** and with the **periodicity** of the **SSB** for **each** of the **UE**'s **cells**.

For an **overall** description of **scheduling**, **transmission occasions** and **transmissions**, click 4.11.4.

### 6.1.2.1 SSB frequency-domain resources within band synchronization rasters

The **next-above-middle subcarrier** of **every Cell-Defining SSB** (click 6.1.3) **must** be **centered** in one of the **synchronization frequencies** of the **band synchronization raster**. The **Non Cell-Defining SSBs** (click 6.1.3) **may**, or **may not**, be located in one of the **synchronization frequencies** of the **band synchronization raster**.

This **next-above-middle subcarrier** of the **SSB** is the one with **index 120**. Recall that that **SSB subcarriers** are **indexed** from **0** (lowest frequency) to **239** (highest frequency).

Notice that the **frequency location** of **each cell**'s **channel** (click 2.2.3.2), of **each channel**'s **carrier** (click 2.2.5), of **each UE**'s **bandwidth part** (click 2.2.7.1.1) and of **each cell**'s **SSB** (click 6.1.2.1), can be configured **independently**. However, **every SSB must** be **completely contained within** its **Downlink initial bandwidth part**, **every bandwidth part within** its **carrier** and **every carrier within** its **channel**.

https://www.sharetechnote.com/html/5G/5G_FrameStructure.html

#### 6.1.2.1.1 Band synchronization rasters

**Each band synchronization raster** is a **list** of **exact synchronization frequencies** (**each identified** by **one synchronization number**) in **one** of which the **next-above-middle subcarrier** of every **SSBs** sent in the **band must** be **centered**. Notice that this only applies to **channels** in the **Downlink transmission direction**.

**Every band**'s **synchronization raster** is a **subset** of the **5G global synchronization raster** (click 6.1.2.1.2). It then follows that:

- **Every band**'s **list** of **synchronization frequencies** (denoted $SF_{REF,b}$) is a **subset** of the **5G global synchronization frequencies**"$SS_{REF,3}$" (click 6.1.2.1.2).
- **Every band**'s **list** of **synchronization numbers** (denoted $SN_{REF,b}$) is a **subset** of the **GSCNs** (click 6.1.2.1.2).
- There is a **one-to-one correspondence** between the elements of **each** $SF_{REF,b}$ and the ones of its corresponding $SN_{REF,b}$.
- **Each synchronization number** from **each** $SN_{REF,b}$ identifies the **exact value** of **one and only one synchronization frequency** from its corresponding $SF_{REF,b}$ (and vice versa).

Some **bands** have **more than one synchronization raster**. In these cases, the **synchronization raster** to use depends **either** on the **SSB subcarrier spacing** (click 6.1.2.2.3) **or** on the **Downlink channel bandwidth**.

There are **three** different ways to define a given **synchronization raster**:

- Providing a **range** of **GSCNs** and a "$step_size_{,b}$" value.

  The **list** of **all** the **channel numbers** in the **synchronization raster** is obtained by **adding** the **first GSCN** in the range with a **multiple** (including zero) of "$step_size_{,b}$", as long as the resulting value is **lesser or equal than** the **last GSCN** in the range.

- Providing an **explicit list** of **GSCNs**.
- Providing a **range** of **GSCNs** and a "$step_size_{,b}$" value, plus 2 formulas

When a **synchronization raster** is provided as a **range** of **GSCNs** and a "$step_size_{,b}$", its **synchronization numbers** are **evenly spaced**, and their spacing is "$step_size_{,b}$". If the corresponding **band** is within **frequency segments 2** or **3** (click 6.1.2.1.2), its **synchronization frequencies** are **also evenly spaced**, and their spacing is "$step_size_{,b}$". multiplied by the **frequency step** of the **corresponding segment** (i.e., **1.44** and **17.28 MHz** respectively). However, if the corresponding **band** is within **frequency segment 1** (click 6.1.2.1.2), its **synchronization frequencies** are **evenly spaced only** when "$step_size_{,b}$" is a **multiple** of **3**.

For **all FR1 bands** (except **n34**, **n38**, **n39**, **n46** and **n96**) and **all FR2-1 bands**, **each** of their **synchronization raster**(**s**) is defined with a **range** of **GSCNs** and a "$step_size_{,b}$" value.

For **FR1 bands n46** and **n96**, their **synchronization raster** is defined with an **explicit list** of **GSCNs**.

For **FR1 bands n34**, **n38** and **n39**, one of their **synchronization rasters** is defined with a **range** of **GSCNs** and a "$step_size_{,b}$" value and the other **synchronization raster** is defined with an **explicit list** of **GSCNs**.

For **FR2-2 band n263**, one of its **synchronization rasters** are defined with a **range** of **GSCNs** and a "$step_size_{,b}$" value and each of its other two **synchronization rasters** is defined with an formula.

The definitions of the **per-band synchronization raster(s)** are performed in table 5.4.3.3-1 of TS 38.101-**1** and **-5**, plus tables 5.4.3.3-**1** and **-2** of TS 38.101-**2**. For a sample of these tables, click 6.1.5.

The **location restrictions** imposed by the **synchronization raster**, combined with the **SSB frequency size**, may imply on some cases that there is **only one** possible frequency location for the **SSB** within the corresponding **bandwidth part**.

Restricting the **SSBs' location** (unless provided by **RRC**) to a predefined list of **exact synchronization frequencies** (**each band**'s **synchronization raster**) is done to simplify the design of the radio frequency elements of the **UE** and to accelerate/simplify the **cell search** procedure (click 8.1).

Notice that the **frequency location** of the **channel** (click 2.2.3.3), of the **carriers** (click 2.2.5), of the **UE bandwidth parts** (click 2.2.6.4) and of the **SSBs** (click 6.1), can be configured **independently**. However, **every SSB must** be **completely contained within** its **bandwidth part**, **every bandwidth part within** its **carrier** and **every carrier within** its **channel**.

In **LTE**, the **SSB must** be **centered** in the **carrier**, and is sent with the same subcarrier spacing used in the carrier as a whole. Therefore, in **LTE**, once the carrier location is determined, the **SSB** location is also determined.

For more detail on the **band synchronization raster**, refer to clause 5.4.3 from TS 38.101-**1**, **-2** and **-5**.

Source web pages:

- https://www.nrexplained.com/raster

#### 6.1.2.1.2 5G global synchronization raster

The **5G global synchronization raster** is a **list** of **exact global synchronization frequencies**, **each** of which is **identified** by an **integer number** called **Global Synchronization Channel Number** (**GSCN**).

The **list** of **global synchronization frequencies** in the **raster** is denoted "$\boldsymbol{SS_{REF}}$". The **list** of **GSCN** in the **raster** is denoted ***GSCN***. There is a **one-to-one correspondence** between the elements of "$\boldsymbol{SS_{REF}}$" and the elements of ***GSCN***. Therefore, **each GSCN** from ***GSCN*** **identifies** the **exact value** of **one and only one global synchronization frequency** from "$\boldsymbol{SS_{REF}}$" (and vice versa).

The **subsets** of the **5G global synchronization raster** that are used to locate the **5G SSB** are the **band synchronization rasters** (click 6.1.2.1.1).

The **global synchronization frequencies** within "$\boldsymbol{SS_{REF}}$" are distributed in the **range** from **0 Hz** to **99,999.96 MHz**. The **GSCNs** within ***GSCN*** are **all** the **integers** from **2** to **26,639**.

Obtaining the **global reference frequency identified** by **each NR-ARFCN** (and vice versa) is done with a simple **equation**. The **equation** is **specific** of **each** of the **three**

**frequency segments** "***s***" that have been defined, as indicated in table 5.4.3.1-1 of TS 38.101-**1** and **-2**. The table from each TSs are presented together in the following table:

| ***s*** | **Frequency segment** | **Inc. FR** | $N_s$ | $M_s$ | $GSCN_s$ | $SS_{REF,s}$ |
|---|---|---|---|---|---|---|
| **1** | **0 to 3,000 MHz** | FR1-1 | 1 to 2,499 | 1, 3, 5 | $3N_s + (M_s-3)/2$ | $N_s$ • 1,200 kHz + $M_s$ • 50 kHz |
| **2** | **3,000 to 24,250 MHz** | FR1-2 | 0 to 14,756 | N/A | $7{,}499 + N_s$ | 3,000 MHz + $N_s$ • 1.44 MHz |
| **3** | **24,250 to 100,000 MHz** | FR2 | 0 to 4,383 | N/A | $22{,}256 + N_s$ | 24,250.08 MHz + $N_s$ • 17.28 MHz |

Some **relevant remarks** on this table are:

- Variables "$N_s$" and "$M_s$" are just intermediate parameters that take certain values and produce the corresponding results from the **equations** in the table.
- In **frequency segment 1**, its **global synchronization frequencies** "$SS_{REF,1}$"are **not evenly spaced**. They are **rather successively spaced 100 kHz**, **100 kHz** and **1 MHz**. Therefore, **each unit** of $GSCN_1$ **does not** increase the "$SS_{REF,1}$" frequency in a fixed amount. This is important to recall.
- In **frequency segments 2** and **3**, their **global synchronization frequencies** "$SS_{REF,2}$" and "$SS_{REF,3}$" are **evenly spaced** in **1.44 MHz** and **17.28 MHz** (respectively).
- The column "**Inc. FR**" indicates what **frequency (sub)range** is contained within **each** of the **three frequency segments**.

The resulting ranges of $GSCN_s$ and "$SS_{REF,3}$" are shown in the following table:

| ***s*** | **Frequency segment** | **Inc. FR** | **Range of $GSCN_s$** | **Range of $SS_{REF,s}$** |
|---|---|---|---|---|
| **1** | **0 to 3,000 MHz** | FR1-1 | 2 to 7,498 | 1.25 to 2,999.05 MHz |
| **2** | **3,000 to 24,250 MHz** | FR1-2 | 7,499 to 22,255 | 3,000 to 24,248.64 MHz |
| **3** | **24,250 to 100,000 MHz** | FR2 | 22,256 to 26,639 | 24,250.08 to 99,988.32 MHz |

Although the **GSCNs** and the **NR-ARFCNs** (click 2.2.15) cover the **same frequency range** (0 to 100,000 MHz), they are **very different**. To point just one clear difference, the **range** of **NR-ARFCNs** is from **0** to **3,279,165**, while the **range** of **GSCNs** is from **2** to **26,639**.

For more detail on the **global synchronization raster**, refer to clause 5.4.3 from TS 38.101-**1**, **-2** and **-5**.

Source web pages:

- https://www.nrexplained.com/raster

#### 6.1.2.1.3 Example of SSB frequency configuration

This example continues from 2.2.4.6.

**Configuration example up to now**

From 2.2.1.3, this is b**and n257** in **FR2-1**, it is **TDD** and it spans from **26,500 MHz** to **29,500 MHz**. From 2.2.2.3, we have **one 100 MHz frequency block** from **27,650.08 MHz** to **27,750.08 MHz**. From 2.2.3.4, we have **one 100 MHz channel**, centered in the frequency block. The **channel NR-ARFCN** is **2,074,167**. The **channel** supports **subcarrier spacings** of **60 kHz** and **120 kHz**. From 2.2.3.6.3, the **transmission bandwidth** is **132 RBs** for **60 kHz SCS** and **66 RBs** for **120 kHz SCS**. From 2.2.4.2, subcarrier **396** from the **centered usable CRB range** is centered at **27,700.08 MHz** and its **subcarrier spacing** is **120 kHz**. From 2.2.4.6, **Point A** is placed at **27,652.56 MHz** (which is the **NR-ARFCN 2,073,375**).

**Configuration example in this section**

As indicated in 6.1.4, we check table 5.4.3.3-1 of TS 38.101**-2**, where the **entry** corresponding to **band n257** is:

| NR operating band | SSB subcarrier spacing | SSB pattern | Range of GSCN (First – < $step_size_{,b}$> – Last) |
|---|---|---|---|
| **n257** | 120 kHz | Case D | 22,388 – <1> – 22,558 |
| | 240 kHz | Case E | 22,390 – <2> – 22,556 |

We see that for **band n257** there are **two SSB subcarrier spacing options**: **120 kHz** and **240 kHz**. For this example, we choose the **option** of **240 kHz SSB subcarrier spacing**. This **option** implies that the **synchronization raster** (click 6.1.2.1) is defined by the **GSCN** range from **22,390** to **22,556**, with "$step_size_{,b}$ **= 2**".

The **SSB** is always **20 RBs**, and therefore its **bandwidth** in this example is 20•12•240=**57.6 MHz**.

The **next-above-middle SSB subcarrier must** be **centered** in one of the **reference frequencies** of the **synchronization raster**. The **next-above-middle SSB** subcarrier is the **SSB subcarrier** with index **120** (starting in **0**). The **SSB** bandwidth **below** and **above** the reference frequency are:

$SSB_{BWabove}$ = (120•240-240/2)/1,000 MHz = **28.68 MHz**

$SSB_{BWbelow}$ = (120•240+240/2)/1,000 MHz = **28.92 MHz**

The **only** valid **GSCN** within the range is **22,456**, corresponding to **27,706.08 MHz**. In this case, the **lower** and **upper edges** of **SSB** are:

$SSB_{upper}$ = 27,706.08+28.68 = **27,734.76 MHz** < 27,747.57

$SSB_{lower}$ = 27,706.08-28.92 = **27,677.16 MHz** > 27,652.53

where you may see that the **SSB** falls **within** the **transmission bandwidth**.

The **previous GSCN** (**22,454**) is **not** valid. Its corresponding frequency is 27,671.52 MHz. In this case, the **lower** and **upper** edges of **SSB** are:

$SSB_{upper}$ = 27,671.52+28.68 = 27,700.20 MHz < 27,747.57

$SSB_{lower}$ = 27,671.52-28.92 = 27,642.60 MHz, which is less than 27,652.53 MHz!

where you may see that the **SSB lower edge** is **outside** of the **transmission bandwidth**.

The **next GSCN** (**22,458**) is **not** valid. Its corresponding frequency is 27,740.64 MHz. In this case, the **lower** and **upper** edges of SSB are:

**SSB$_{upper}$** = 27,740.64+28.68 = 27,769.32 MHz, which is more than 27,747.57 MHz!

**SSB$_{lower}$** = 27,740.64-28.92 = 27,711.72 MHz > 27,652.53

where you may see that the **SSB upper edge** is **outside** of the **transmission bandwidth**.

For the value of `subCarrierSpacingCommon` from `MIB` (click 8.12.3), we choose the value `scs30or120`.

From the definition of `offsetToPointA` (click 8.7.3) from `FrequencyInfoDL-SIB` (click 2.2.6.6) we know that:

`offsetToPointA` = ***int***((***f***$_{SC0}^{SSB}$ - **Point A**) / (12 • ***SCS***$_c$)) • (***SCS***$_c$ / ***SCS***$_u$)

where:

- "***f***$_{SC0}^{SSB}$" is the **center frequency** of **subcarrier 0** of **SSB**. In this example, "***f***$_{SC0}^{SSB}$ = 27,677.16 MHz + 0.240 MHz / 2", because it is the **SSB lower edge**, plus **half** the **SSB subcarrier spacing** (for the values, see above). Consequently, "***f***$_{SC0}^{SSB}$ = **27,677.28 MHz**".
- **Point A** is the frequency location of **Point A**. In this example, **Point A** is **27,652.56 MHz** (see above)
- "***SCS***$_c$" is the **subcarrier spacing** of the **RBs** that are used to **calculate** the value of `offsetToPointA`. In this example, "***SCS***$_c$**=120 kHz**", because we are in a **licensed FR2-1 band** and the **value of** `subCarrierSpacingCommon` is `scs30or120` (see above).
- "***SCS***$_u$" is the **subcarrier spacing** used for the **RBs** that are **units** of `offsetToPointA`. In this example, "***SCS***$_u$**=60 kHz**" because we are in a **FR2 band**.

Consequently,

`offsetToPointA` = ***int***((27,677.28 - 27,652.56) / (12 • .120)) • (.120 / .060)

`offsetToPointA` = `34`

From the definition of "***k***$_{SSB}$" (click 8.7.2), we know that:

***k***$_{SSB}$ = (***f***$_{SC0}^{SSB}$ – (**Point A** + `offsetToPointA`• **12** • ***SCS***$_u$)) / ***SCS***$_c$

Therefore:

***k***$_{SSB}$ = (***f***$_{SC0}^{SSB}$ – (**Point A** + `offsetToPointA`• **12** • ***SCS***$_u$)) / ***SCS***$_c$

***k***$_{SSB}$ = (27,677.28 – (27,652.56 + `34`• 12 • .060)) / .120

This implies that "***k***$_{SSB}$" is **2**, and this will be conveyed to the **UE** within `MIB`.

This example continues in 6.1.2.2.3.

### 6.1.2.2 SSB time-domain resources

The **SSBs** are **sent** in **periodical bursts**. **Each burst** has **several SSBs** (**1** for **each beam**) when the **NR node** uses **Tx beam sweeping** (click 6.1.2.3). Otherwise, **each burst** has **only 1 SSB**.

#### 6.1.2.2.1 SSB burst time-domain resources

The **SSB bursts** are sent with a **fixed periodicity**. The **most usual SSB burst** periodicity is **20 ms**, but it is also possible to configure it to **5** or **10 ms**. For special low-power cells it is also possible to send an **SSB burst** every **40**, **80** or **160 ms**. Notice that **broadcasting SSB** is the **only always-on signal** in a **cell**. Therefore, **longer SSB burst** sending **periodicities** enhance network energy performance, but **shorter** sending **periodicities** facilitate faster cell search for **UEs**. The most frequent configuration is sending an **SSB burst** every **20 ms**. The periodicity of **SSB bursts** is configured with `ssb-PeriodicityServingCell` either from `ServingCellConfigCommonSIB` (click 3.9.3) or from `ServingCellConfigCommon` (click 3.9.6).

The **5G UEs** doing **cell search** will **scan each reference frequency** in the **synchronization raste**r for **up to 20 ms**, looking for an incoming **SSB**. You may want to configure a **shorter SSB burst** sending **periodicity** (e.g., **5** or **10 ms**) to enable **faster cell search**, at the cost of more energy consumption. Conversely, you may want a **longer SSB burst** sending **periodicity** (e.g., **80** or **160 ms**) to save energy.

When a **UE** is doing the **cell search**, a sending periodicity larger than **20 ms** does **not** guarantee that the **UE** will receive the **SSB**. This is **not** a **problem** in a **secondary cell**, because the **UE** is expected to get the **SSB** (with **CSS-Type 0 configuration**, leading to `SIB1`) in the **primary cell**. The **UE** may then be informed about the **existence** of the **secondary cells** through **RRC** over the **primary cell**.

**Each SSB burst** is sent in **either** a **first half-frame** or a **second half-frame**. This is explicitly indicated in the **PBCH SI** parameter **half-frame-flag**.

In **LTE**, the **PSS/SSS** was sent every **5 ms** and the **PBCH** every **10 ms**. Using in **5G** a larger **SSB** periodicity than the one of **LTE** implies that the **UE** must stay **longer** at **each searched reference frequency**, that **would** result in a **longer cell search**. This **longer cell search** has been **avoided** in **5G** by having a **sparser synchronization raster** (click 6.1.2.1), thus **requiring** the **UE** to **search** in **fewer reference frequencies**.

For more detail on **SSB burst scheduling**, refer to clause 4.1 of TS 38.213.

#### 6.1.2.2.2 SSB scheduling within each burst: SSB burst Patterns

The **SSBs** within **each SSB burst** are **scheduled** according to a **SSB burst pattern**. **All** the **SSB bursts** in a **cell** have the **same SSB burst pattern**.

**Each SSB burst pattern** defines:

- The **maximum quantity** of **SSBs** in the **burst**
- The **half-frame symbol interval index values** of the **first symbol** of each **SSB** within the **burst**.

The **SSB burst pattern** to use is specified for **each band** and **subcarrier pacing** as **only 1** among several **Cases** (**A**, **B**, **C**, …). The specific **Case** of **SSB Pattern** to be used for **each pair** of **band** and **SSB subcarrier spacing** is stated in table 5.4.3.3-1 of TS 38.101-**1**, -**2** and -**5**. For a sample of these tables, click 6.1.5. There **must** be **only 1 Case** for **each band** and **SSB subcarrier spacing** because the **UE must** univocally know the **symbol location** of **SSB** during **cell search** (click 8.1) in order to achieve **slot** and **symbol synchronization**.

The description of **each Case** indicates in what **symbol interval index** is placed the **first symbol** of **each SSB** in the **burst**. The **symbol interval index** refers to **each half-frame**,

where **index 0** is the **first symbol interval** in **each half-frame**. The **symbol interval index** is computed with the **subcarrier spacing** used in the **active BWP** where the **SSB burst** is sent.

**Case A: FR1 bands and 15 kHz SSB subcarrier spacing**

The **first symbol** of **each SSB** in the burst is placed in the following **symbol indexes** of the **half-frame**: **{2, 8}+14•*n***. The possible values of "***n***" result in the following:

- For **licensed spectrum**,
  - For **FR1-1** bands: **up to 4 SSBs**, where the **first symbol** of **each SSB** is placed in the following **symbol indexes** of the **half-frame**: **2**, **8**, **16** and **22**.
  - For **FR1-2** bands: **up to 8 SSBs**, where the **first symbol** of **each SSB** is placed in the following **symbol indexes** of the **half-frame**: **2**, **8**, **16**, **22**, **30**, **36**, **44** and **50**.
- For **shared spectrum**, as described in TS 37.213, **up to 10 SSBs**, where the **first symbol** of **each SSB** is placed in the following **symbol indexes** of the **half-frame**: **2**, **8**, **16**, **22**, **30**, **36**, **44**, **50**, **58** and **64**.

**Case B: FR1 bands and 30 kHz SSB subcarrier spacing**

The **first symbol** of **each SSB** in the burst is placed in the following **symbol indexes** of the **half-frame**: **{4, 8, 16, 20} + 28•*n***. The possible values of "***n***" result in the following:

- For **FR1-1** bands: **up to 4 SSBs**, where the **first symbol** of **each SSB** is placed in the following **symbol indexes** of the **half-frame**: **4**, **8**, **16** and **20**.
- For **FR1-2** bands: **up to 8 SSBs**, where the **first symbol** of **each SSB** is placed in the following **symbol indexes** of the **half-frame**: **4**, **8**, **16**, **20**, **32**, **36**, **44** and **48**.

Notice that the definitions of Case B and Case C **overlap**. However, there is no ambiguity because (as indicated above), the **Case** of **SSB Pattern** to be used in each **band** is stated in table 5.4.3.3-1 of TS 38.101-**1**, **-2** and **-5**. For a sample of these tables, click 6.1.5.

**Case C: FR1 bands and 30 kHz SSB subcarrier spacing**

The **first symbol** of **each SSB** in the **burst** is placed in the following **symbol indexes** of the **half-frame**: **{2, 8} + 14•*n***. The possible values of "***n***" result in the following:

For **licensed spectrum**:

- For **paired** spectrum bands (**FDD**, **SDL** and **SUL**)
  - For **FR1-1** bands: **up to 4 SSBs**, where the **first symbol** of **each SSB** is placed in the following **symbol indexes** of the **half-frame**: **2**, **8**, **16** and **22**.
  - For **FR1-2** bands: **up to 8 SSBs**, where the **first symbol** of **each SSB** is placed in the following **symbol indexes** of the **half-frame**: **2**, **8**, **16**, **22**, **30**, **36**, **44** and **50**.
- For **unpaired** spectrum bands (**TDD**)
  - For **carrier**'s **center frequency lower than 1.88 GHz**: **up to 4 SSBs**, where the **first symbol** of **each SSB** is placed in the following **symbol indexes** of the **half-frame**: **2**, **8**, **16** and **22**.

- For **FR1 carrier**'s **center frequency higher than or equal 1.88 GHz**: **up to 8 SSBs**, where the **first symbol** of **each SSB** is placed in the following **symbol indexes** of the **half-frame**: **2**, **8**, **16**, **22**, **30**, **36**, **44** and **50**.

For **shared spectrum**, $n \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$. **Up to 20 SSBs**.

**Case D: FR2 bands and 120 kHz SSB subcarrier spacing**

**Up to 64 SSBs**, where the **first symbol** of **each SSB** in the burst is placed in the following **symbol indexes** of the **half-frame**: $\{4, 8, 16, 20\} + 28 \cdot n$, with $n \in \{0, 1, 2, 3, 5, 6, 7, 8, 10, 11, 12, 13, 15, 16, 17, 18\}$.

**Case E: FR2-1 bands and 240 kHz SSB subcarrier spacing**

**Up to 64 SSBs**, where the **first symbol** of **each SSB** in the **burst** is placed in the **symbol indexes** of the **half-frame**: $\{8, 12, 16, 20, 32, 36, 40, 44\} + 56 \cdot n$, with $n \in \{0, 1, 2, 3, 5, 6, 7, 8\}$.

**Case F: FR2-2 bands and 480 kHz SSB subcarrier spacing, and**

**Case G: FR2-2 bands and 960 kHz SSB subcarrier spacing**

**Up to 64 SSBs**, where the **first symbol** of **each SSB** in the burst is placed in the **symbol indexes** of the **half-frame**: $\{2, 9\} + 14 \cdot n$, with $n \in \{0..31\}$.

Up to the **maximum quantity** of **SSBs** indicated in **each** of the **cases A** to **G** above, it is the **NR node** that decides the **actual quantity** of **SSBs** sent in **each burst**.

The **specific case** to be used is **stated** in table 5.4.3.3-1 of TS 38.101-**1** and -**5**, plus tables 5.4.3.3-**1** and -**2** of TS 38.101-**2**. For a sample of these tables, click 6.1.5.

In the table from 6.8.8.1 you may see that the ratio of **SSB-SCS over active BWP-SCS** ranges **from 1/2** (i.e., 15/30) **to 8** (i.e., 960/120) in powers of **2**. Therefore, the **4 SSB symbols** may occupy **from 8 to 1/2 active BWP symbols** in powers of **2**, respectively.

**SSB time scheduling for shared spectrum**

The **SSB time scheduling** for **shared spectrum** is more complex. A distinction is made between a **candidate SSB index** and an **SSB index**. Parameter `ssb-positionsInBurst` either from `ServingCellConfigCommonSIB`, (click 3.9.3) or from `ServingCellConfigCommon` (click 3.9.6) is required for the procedure. It may also be required either of parameters `discoveryBurstWindowLength-r16` or `discoveryBurstWindowLength-v1700`, both from `ServingCellConfigCommonSIB`, or either of parameters `discoveryBurstWindowLength-r16` or `discoveryBurstWindowLength-r17`, both from `ServingCellConfigCommon`.

For more detail on **SSB burst patterns**, refer to clause 4.1 of TS 38.213.

#### 6.1.2.2.3 Example of SSB time configuration

This example continues from 6.1.2.1.3.

**Configuration example up to now**

From 2.2.1.3, this is b**and n257** in **FR2-1**, it is **TDD** and it spans from **26,500 MHz** to **29,500 MHz**. From 2.2.2.3, we have **one 100 MHz frequency block** from **27,650.08 MHz** to **27,750.08 MHz**. From 2.2.3.4, we have **one 100 MHz channel**, centered in the frequency block. The **channel NR-ARFCN** is **2,074,167**. The **channel**

supports **subcarrier spacings** of **60 kHz** and **120 kHz**. From 2.2.3.6.3, the **transmission bandwidth** is **132 RBs** for **60 kHz SCS** and **66 RBs** for **120 kHz SCS**. From 2.2.4.2, subcarrier **396** from the **centered usable CRB range** is centered at **27,700.08 MHz** and its **subcarrier spacing** is **120 kHz**. From 2.2.4.6, **Point A** is placed at **27,652.56 MHz** (which is the **NR-ARFCN 2,073,375**). From 6.1.2.1.3, the **SSB subcarrier spacing** is **240 kHz**, the **SSB** is placed at **27,706.08 MHz**, the value of `subCarrierSpacingCommon` from `MIB` (click 8.12.3) is `scs30or120`, the value of `offsetToPointA` from `FrequencyInfoDL-SIB` (click 2.2.6.6) is `34` and "$k_{SSB}$" is **2**.

**Configuration example in this section**

Checking (click 6.1.2.1.3) we see that for **SSB subcarrier spacing** of **240 kHz**, the **SSB temporal pattern** is "**Case E**". "**Case E**" implies that **up to 64 SSBs** can be sent in **each burst**, where **each SSB** starts in the following half-frame symbol indexes:

**{8, 12, 16, 20, 32, 36, 40, 44} + 56•*n***, with ***n*** ∈ **{0, 1, 2, 3, 5, 6, 7, 8}**.

For this example, we choose that the **NR node** uses **Tx beam sweeping** (click 6.1.2.3), sending **10 SSBs** in each **SSB burst**. Therefore, **one SSB** is sent in **each** of the **half-frame symbol indexes**:

**{8, 12, 16, 20, 32, 36, 40, 44, 64, 68}**

We **chose** that the configured sending frequency for **SSB bursts** is every **20 ms**.

This example is the **last one** of the **series** of **examples** showing the **top down configuration** of the **gNB**. **Another series** of **examples** showing the **bottom up configuration** of the **UE** starts in 8.2.1.

### 6.1.2.3 SSB time and space scheduling: beam sweeping and selection

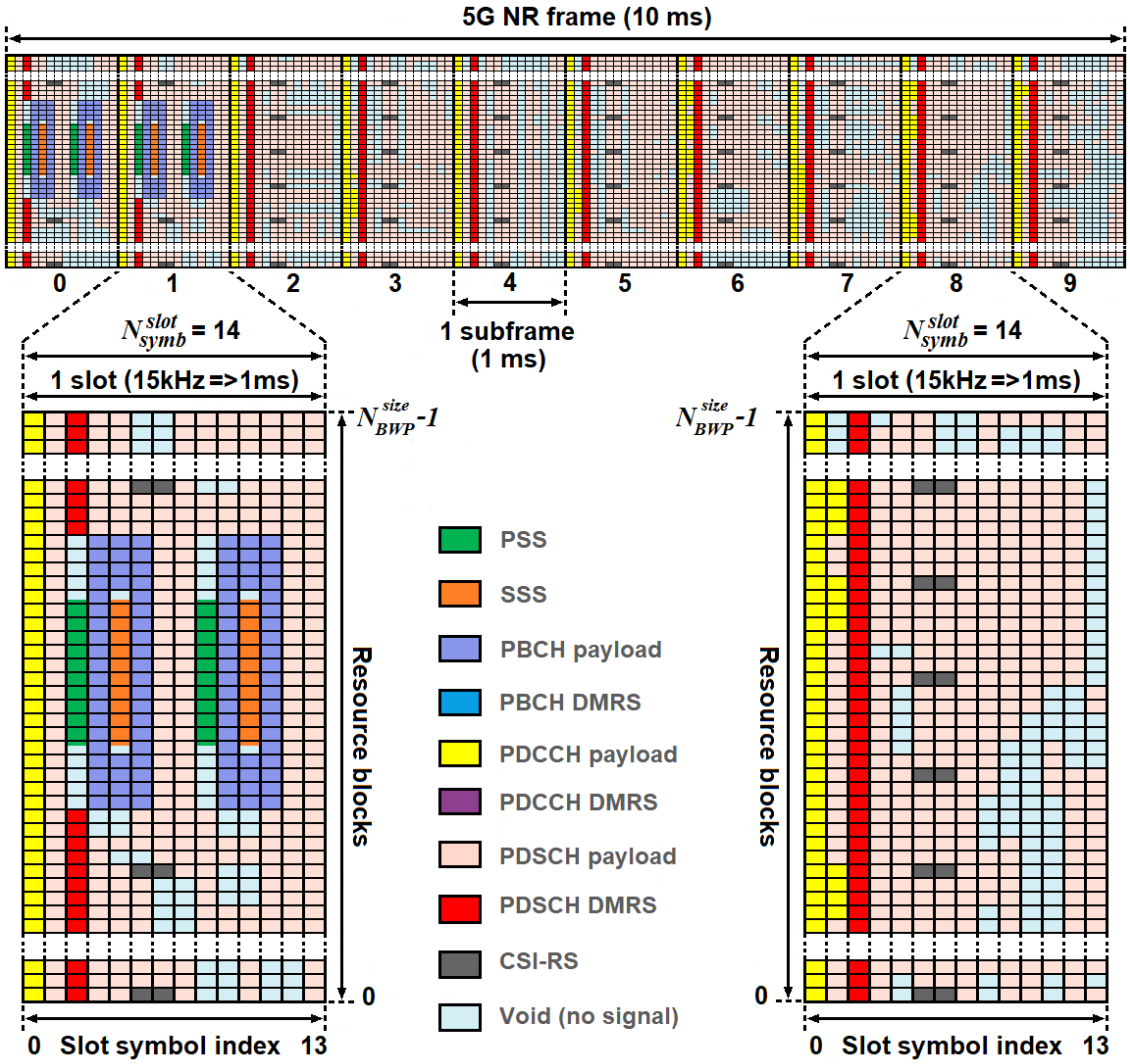
When the **NR node** uses **Tx beam sweeping** (click 2.4.1.5), **each SSB burst** (click 6.1.2.2) contains **1 SSB** for **each beam**, and this is used by the **UE** to perform **beam selection**. The **NR node Tx beam sweeping** consists of the **NR node** sending a **single beam**, but in **different directions** at **different symbol intervals**. In this case, **each SSB burst** contains a **rapid sequence** of **SSB**s, within a **half-frame**. **Each SSB** within the **burst** is sent at a **different symbol interval and** in a **different beam**. In order for the **UE** to **distinguish each beam**, **each SSB carries** a **different value** of the **PBCH SI** parameter `ssb-time-domain-index`.

**Each SSB** sent within **each burst** is **sequentially indexed**, starting from **0** for the **first SSB**. The value of this **SSB index** is broadcast in **PBCH SI** parameter `ssb-time-domain-index` (click 5.3.2). This allows the **UE** to identify **each incoming beam** with its value of `ssb-time-domain-index`. Notice that `ssb-time-domain-index` is large enough to **index** each individual **SSB** within the **burst**, up to the maximum quantity required:

- **Up to 4 SSBs** (**up to 4 beams**): the parameter `ssb-time-domain-index` has **2 bits**.
- **Up to 8 SSB** (**up to 8 beams**): the parameter `ssb-time-domain-index` has **3 bits**.
- **Up to 64 SSBs** (**up to 64 beams**): the parameter `ssb-time-domain-index` has **6 bits**.

In this way, the **UE** can do **beam selection** based on what **beam** is **received** with the **highest receive power**.

The following figure shows an **approximate** representation of **one Downlink frame** with **1 SSB burst** for **NR node Tx beam sweeping**.

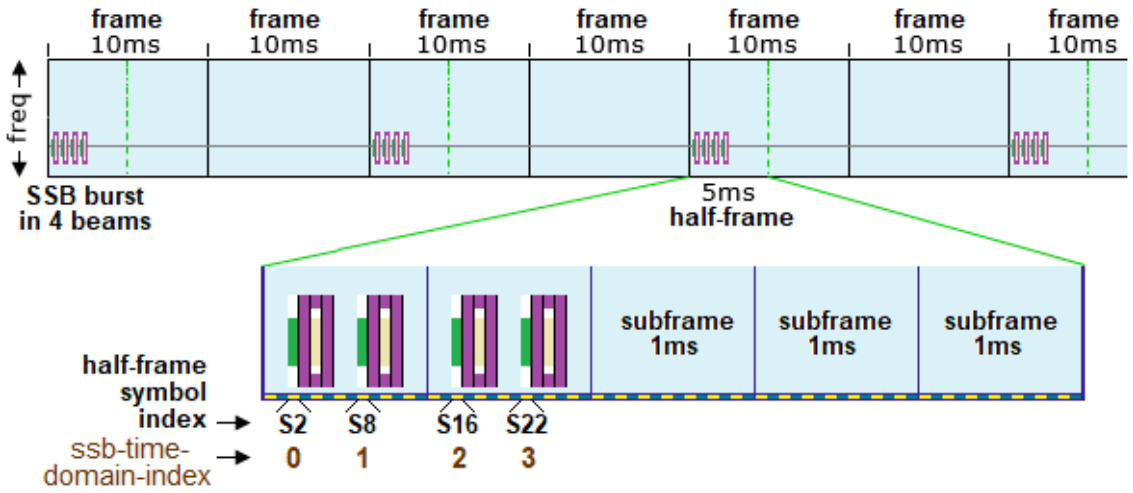



Some relevant remarks on this figure are:

- In this example, the **subcarrier spacing** of both the **carrier** and the **SSB** is **15 kHz**.
- Since the **subcarrier spacing** is **15 kHz**, the **slot duration** is **1 ms**.
- The **frame** contains the **SSB burst** in the **first half-frame**.
- The **SSB burst** corresponds to **pattern Case A**, with **4 SSBs** (the **maximum** possible in **Case A**).
- **Each** of the **4 SSBs** has its **first symbol** located in the **symbol indexes** (of the **half-frame**) **2**, **8**, **16** and **22**. The respective **values** of **ssb-time-domain-index** from **PBCH system information** (click 5.3.2) are **0**, **1**, **2** and **3**.

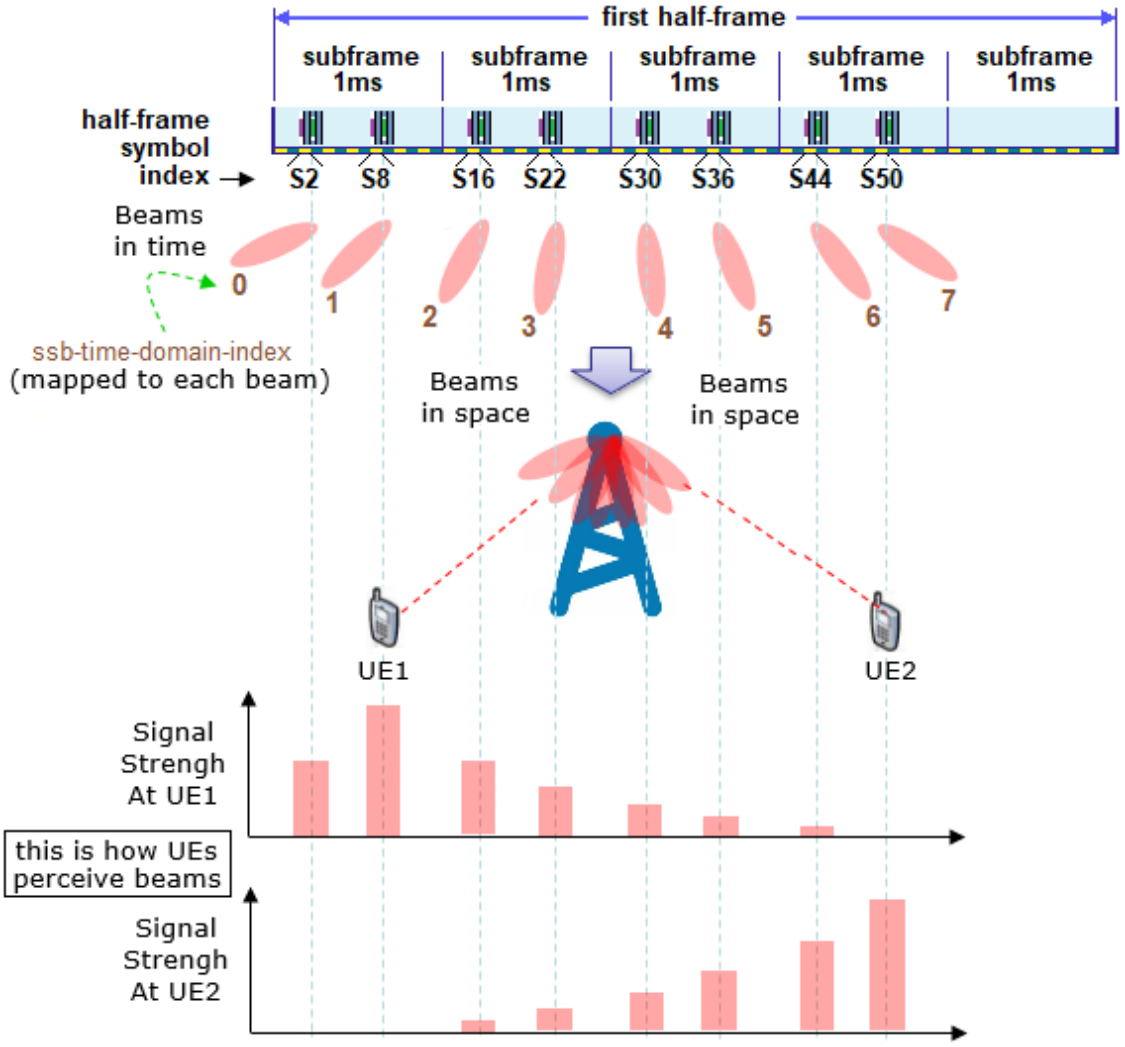
The following figure shows an **approximate** representation of a **Downlink frame** with **one SSB burst** for **NR node Tx beam sweeping**. In this example, the **subcarrier** spacing of both the **carrier** and the **SSB** is **15 kHz**, the **SSB burst** is sent in the **first half-frame**, with **SSB pattern Case A**, where the **first symbol** of each of the **4 SSBs** in the **burst** is located in the corresponding **half-frame** symbols **2**, **8**, **16** and **22**, and their respective

values of the **PBCH system information** parameter **ssb-time-domain-index** are **0**, **1**, **2** and **3**:




The following figure shows **an approximate** representation of a **Downlink frame** with **1 SSB burst** for **Tx beam sweeping**, **showing** the **beam position** for **each SSB**. In this example, the **subcarrier spacing** of the **carrier** is **15 kHz** and the one of the **SSB** is **30 kHz**. The **SSB burst** is sent in the **first half-frame**, with **SSB pattern Case C**, where the **first symbol** of **each** of the **8 SSBs** in the **burst** has the **half-frame symbol index 2**, **8**, **16**, **22**, **30**, **36**, **44** and **50**. The respective values of the **PBCH system information** (click 5.3.2) parameter **ssb-time-domain-index** (click 6.4.1) are **0**, **1**, **2**, **3**, **4**, **5**, **6** and **7**.

As you may see, **two SSB symbols** fit into **each CRB range symbol**, due to the difference in subcarrier spacing:




Notice how the **received signal strength** at the different **UEs** is very different in respect to **each** of the **8 beam positions**.

The **NR node** needs to know **what** is the **beam** that has been **selected** by the **UE**, in order to use the **said beam** when transmitting to this **UE**. The **NR node** learns the **ssb-time-domain-index value** of the **beam selected** by the **UE** from the **frequency-time resources** that the **UE used** to transmit its **RA preamble** (click 7.3.1) through **PRACH**. For more detail on how the **UE** determines the **frequency-time resources** to transmit its **RA preamble** (i.e., the **PRACH scheduling**) based on the **ssb-time-domain-index** value of the **selected beam**, click 7.2.2.

For more detail on **SSB location** in **time** and **space** (**Tx beam sweeping**), refer to clause 4.1 of TS 38.213.

### 6.1.3 SSB contents

**Each SSB** is composed of the following elements:

- **Primary Synchronization Signal** (**PSS**) (click 6.19)

  It is sent in the **first SSB symbol** over **127 subcarriers** of the **SSB**. The subcarriers are **56** to **182**, both included, where the **lowest frequency subcarrier** of the **SSB** is assigned index **0**. Notice these subcarriers are **not centered** within the **SSB** and are **not** a **multiple** of **12**.

- **Secondary Synchronization Signal** (**SSS**) (click 6.19)

  It is sent in the **third SSB symbol** over the **same subcarriers** as the **PSS**.

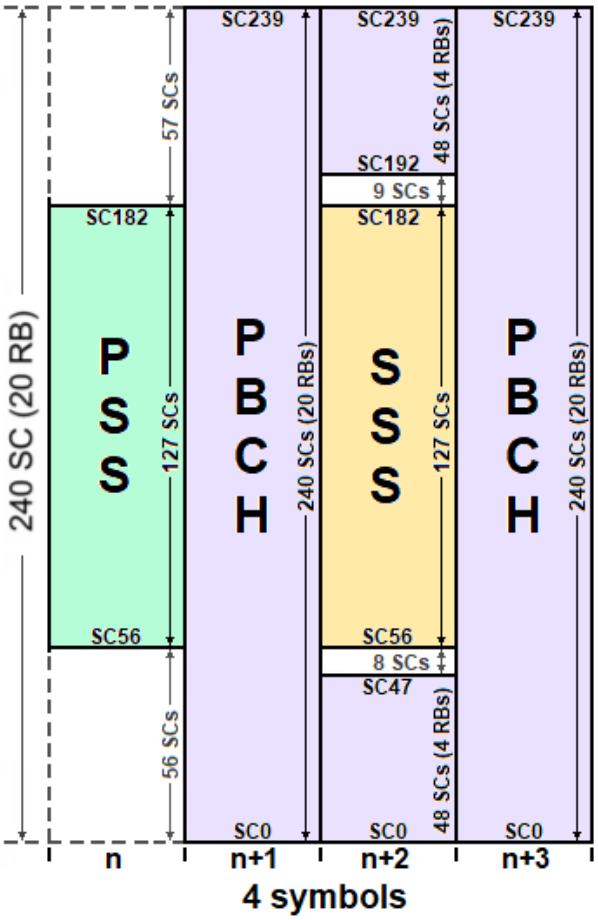
- **Physical Broadcast Channel** (**PBCH**) (click 6.2)

  It is sent in the **second** and **fourth SSB symbols** over the **20** Resource Blocks of the **SSB**, plus, the **third SSB symbol** over the **4 highest** and **4 lowest** Resource Blocks of the **SSB**. It contains the **Master Information Block** (`MIB`, click 5.3.3) plus the **PBCH system information** (click 5.3.2).

- **Unused**

  The **56 Resource Elements** (**REs**) below the **PSS** and the **57** REs above it, plus, the **8 REs** below the **SSS** and the **9 REs** above it are **set to zero**. Set to **zero** means that the corresponding complex-valued symbols are set to zero.

The following figure shows the structure of the **SSB**:



The **UE** may assume that the complex-valued symbols corresponding to **REs** that are part of a **common resource block** partially or fully **overlapping** with an **SSB** and not used for SS/PBCH transmission are **set to zero** in the **OFDM** symbols partially or fully overlapping with **OFDM** symbols where the **SSB** is transmitted.

Depending on the **contents** of **PBCH**, there are **2 types** of **SSB** that can **exist** in **different carriers**, or even in the **same carrier**:

- **Cell-defining SSB** (**CD-SSB**):

  It is the **SSB** that provides **CSS-Type 0 configuration**, which allows the **UE** to receive `SIB1` (click 5.3.4). Notice that `SIB1` contains the **NCGI** (click 3.5.3.1) of the cell, hence the name "**cell-defining**". This **SSB** is **always** configured in the **initial bandwidth part**. The **UE** detects its **location** through **cell search** (click 8.2) over the **synchronization raster** (click 6.1.2.1). Consequently, **this SSB must** be **located** in a **synchronization frequency** from the **synchronization raster**.

- **Non cell-defining SSB** (**NCD-SSB**):

  It is a **SSB** that does **not** contain **CSS-Type 0 configuration** (consequently, `SIB1` cannot be received in the cell). It can be used in **NSA** deployments with **EN-DC**. Even in **SA**, **one or more** of these **SSB** types can **coexist** within the same **bandwidth part**. The **UE** will know their location via **RRC Reconfiguration**. Consequently, **this SSB may**, or **may not**, be **located** in a **synchronization frequency** from the **synchronization raster** (click 6.1.2.1).

### 6.1.4 SSB processing

The **SSB** is **transmitted** in a **single antenna port** (click 4.11.5).

The **3 components** of **SSB**, which are **PSS** (click 6.19), **SSS** (click 6.19) and **PBCH** (click 6.2), **always** use the **normal cyclic prefix** (click 2.3.1).

The **3 components** of **SSB**, which are **PSS**, **SSS** and **PBCH**, use the **same subcarrier spacing**, which is the **SSB subcarrier spacing**.

The **SSB subcarrier spacing** is **optionally** configured for **each cell** with `ssbSubcarrierSpacing` from `ServingCellConfigCommon` (click 3.9.6), **either** within `SIB1` (click 5.3.4) **or** within the **non-SI parameter group** (click 5.3.6). The **data type** of this parameter is `SubcarrierSpacing` (click 2.2.10.5.1).

It can **only** be configured with the **following values**, that depend on the **frequency range** where the **band** is located:

- **FR1**: **15** or **30 kHz**
- **FR2-1**: **120** or **240 kHz**
- **FR2-2**: **120**, **480**, or **960 kHz**

Moreover, the **SSB subcarrier spacing**(**s**) allowed for **each band** is/are indicated in table 5.4.3.3-1 of TS 38.101-**1**, **-2** and **-5**. For a sample of these tables, click 6.1.5.

**Not all** the **combinations** between **SSB subcarrier spacing**, **CORESET#0** (**PDCCH**) **subcarrier spacing** and **control/shared subcarrier spacing** are allowed. For the **allowed combinations**, click 2.1.5.

For more detail on **5G subcarrier spacings**, click 2.2.10.3.

**Other SSB processing** is presented within **each** of its components: **PSS processing** (click 6.19.2), **SSS processing** (click 6.19.3) and **PBCH processing** (click 6.5).

Source web pages:

- http://howltestuffworks.blogspot.com/2019/10/5g-nr-synchronization-signalpbch-block.html
- http://howltestuffworks.blogspot.com/2019/11/5g-nr-resource-blocks.html
- http://sharetechnote.com/html/5G/5G_SS_Block.html
- https://www.sharetechnote.com/html/5G/5G_FrameStructure.html

### 6.1.5 Tables of per-band SSB subcarrier spacing, SSB pattern and synchronization raster

The **per-band SSB subcarrier spacing** is indicated in the **second column** of table 5.4.3.3-1 of TS 38.101-**1**, **-2** and **-5**.

The **per-band SSB pattern** is indicated in the **second** and **third columns** of table 5.4.3.3-**1** of TS 38.101-**1**, **-2** and **-5**, and of the combination of tables 5.4.3.3-**1** and **-2** of TS 38.101-**2**.

The **per-band synchronization raster** is indicated in the **second** and **fourth columns** of table 5.4.3.3-1 of TS 38.101-**1** and **-5**, and of the combination of tables 5.4.3.3-**1** and **-2** of TS 38.101-**2**.

A **fragment** from table 5.4.3.3-1 of TS 38.101-**1** is:

| NR operating band | SSB subcarrier spacing | SSB pattern | Range of GSCN (First – <*step_size,b*> – Last) |
|---|---|---|---|
| **n1** | 15 kHz | Case A | 5,279 – <1> – 5,419 |
| **…** | … | … | … |
| **n5** | 15 kHz | Case A | 2,177 – <1> – 2,230 |
| | 30 kHz | Case B | 2,183 – <1> – 2,224 |
| **…** | … | .. | |
| **n48** | 30 kHz | Case C | 7,884 – <1> – 7,982 |
| **n50** | 30 kHz | Case C | 3,590 – <1> – 3,781 |
| **…** | … | … | … |
| **n79** | 30 kHz | Case C | 8,480 – <16> – 8,880[7] |
| | | | 8,475 – <1> – 8,884[8] |
| **…** | … | … | … |
| **n90** | 15 kHz | Case A | 6,246 – <16> – 6,717[10] |
| | | | 6,245 – <1> – 6,718[11] |
| **…** | … | … | … |
| **n104** | 30 kHz | Case C | 9,882 – <7> – 10,358 |
| **n105** | 15 kHz | Case A | 1,535 – <1> – 1,624 |

The combination of tables 5.4.3.3-1 and 5.4.3.3-2 of TS 38.101-**2** is:

| NR operating band | SSB subcarrier spacing | SSB pattern | Range of GSCN (First – < *step_size,b* > – Last) |
|---|---|---|---|
| **n257** | 120 kHz | Case D | 22,388 - <1> - 22,558 |
| | 240 kHz | Case E | 22,390 - <2> - 22,556 |

| NR operating band | SSB subcarrier spacing | SSB pattern | Range of GSCN (First – < *step_size*$_{,b}$ > – Last) |
|---|---|---|---|
| **n258** | 120 kHz | Case D | 22,257 - <1> - 22,443 |
| | 240 kHz | Case E | 22,258 - <2> - 22,442 |
| **n259** | 120 kHz | Case D | 23,140 – <1> – 23,369 |
| | 240 kHz | Case E | 23,142 – <2> – 23,368 |
| **n260** | 120 kHz | Case D | 22,995 - <1> - 23,166 |
| | 240 kHz | Case E | 22,996 - <2> - 23,164 |
| **n261** | 120 kHz | Case D | 22,446 - <1> - 22,492 |
| | 240 kHz | Case E | 22,446 - <2> - 22,490 |
| **n262** | 120 kHz | Case D | 23,586 – <1> – 23,641 |
| | 240 kHz | Case E | 23,588 – <2> – 23,640 |
| **n263** | 120 kHz | Case D | 24,156 + 6 • N – 3 • ***floor***((N+5)/18), N = 0 to 137 |
| | 480 kHz | Case F | 24,162 + 24 • N – 12 • ***floor***((N+4)/18), N = 0 to 33 |
| | 960 kHz | Case G | 24,162 – <6> – 24,954 |

Some **relevant remarks** on these tables in respect to the **per-band SSB subcarrier spacing** are:

- **FR1 bands n5**, **n24**, **n34**, **n38**, **n39**, **n41**, **n66**, **n101**, **n255** and **all FR2 bands** have **more than one option** for the **SSB subcarrier spacing**.
- **Every band** that lists **more than 1 SSB subcarrier spacing** also has **more than 1 synchronization raster** (click 6.1.2.1.1).
- The **60 kHz subcarrier spacing cannot** be used for **SSB**.
- The **240 kHz subcarrier spacing can be** used for **SSB** in some bands, although it **cannot** be used for any other **Downlink** transmission. The reason to support **240 kHz SSB subcarrier spacing** is to enable a **very short** time **duration** for each **SSB**. This is relevant when the **NR node** uses **Tx beam sweeping** (click 6.1.2.3) over **many beams** with a corresponding large quantity of **time multiplexed SSBs**.
- The **960 kHz subcarrier spacing can** be used for a **Non Cell-defining SSB** (click 6.1.3) but **not** for a **Cell-defining SSB** (i.e., **cannot** be used for **initial access**).
- The columns "**SSB pattern**" and "**Range of GSCN**" are **not** used for the **per-band SSB subcarrier spacing**.

Some **relevant remarks** on these tables in respect to the **per-band SSB pattern** are:

- A few bands have **more than one option** for the **SSB pattern** (e.g., band **n5** in the table fragment above), but there is **only one** option when combined with the **SSB subcarrier spacing**. This is mandatory because the **UE must** univocally know the **symbol location** of **SSB** during **cell search** (click 8.1) in order to achieve **slot** and **symbol synchronization**.
- The column "**Range of GSCN**" is **not** used for the **per-band SSB pattern**.

Some **relevant remarks** on these tables in respect to the **per-band synchronization raster(s)** are:

- **FR1 bands n5**, **n24**, **n34**, **n38**, **n39**, **n41**, **n66**, **n79**, **n90**, **n101**, **n255** and **all FR2-1 bands** have **2 synchronization rasters**. **FR2-2 band n263** has **3 synchronization rasters**. In **all** these **bands** (except **FR1 n79** and **n90**) the **band synchronization raster** to use is **univocally determined** by the **band** and the **SSB subcarrier spacing**. **Cell search** in a **band** with **more than one synchronization raster** is more difficult (click 8.2).
- In **FR1 band n79**, the **synchronization raster** with "$step_size_{,b}$**=16**"is used when the **channel bandwidth** is **larger** than **or equal** to **40 MHz** (note 7 in the table from the TS). Otherwise, the other raster is used (note 8 in the table from the TS).
- In **FR1 band n90**, the **synchronization raster** with "$step_size_{,b}$**=16**" is used when the **channel bandwidth** is **larger** than **or equal** to **10 MHz** (note 10 in the table from the TS). Otherwise, the other raster is used (note 11 in the table from the TS).
- The column "**SSB pattern**" is **not** used for the **per-band synchronization raster**.

## 6.2 PBCH overview

The **Physical Broadcast Channel** (**PBCH**) is used by the **NR node** to **broadcast** the **PBCH System Information** (click 5.3.2) and the **Master Information Block** (**MIB**, click 5.3.3) to **all** the **UEs**, including to the **non-active ones**.

**PBCH** is **jointly broadcast** with **PSS** and **SSS** (click 6.19) as the "**Synchronization signal** and **PBCH block**" (**SSB**, click 6.1).

The most relevant **functions** of **PBCH** are:

- Provide the **UE** with **frame**, **subframe**, **slot** and **symbol interval synchronization**.
- Allow the **UE** to perform **beam selection** (click 6.1.2.3).
- Provide the **UE** with **MIB**.

**Each PBCH transmission** takes place in **each** and **every transmission occasion determined** by the **PBCH scheduling** (click 6.3). **Each slot** can contain **0**, **1** or **several PBCH transmissions**.

**Each PBCH transmission** contains the **PBCH system information** (click 5.3.2) and **MIB** (click 5.3.3).

**Each PBCH transmission** (both **payload** and **PBCH DM-RS**) is subject to a specific **processing** (click 6.5) for its **transmission** and **reception**.

For the **payload** and **DM-RS contents** of **PBCH**, click 6.4.

**PBCH** does **not** have any **configuration** (click 6.6) **parameter**.

**PBCH** information has to be received by **UEs** that might **not** yet have any **Uplink channel**. Since the **information** that **PBCH** conveys is **unidirectional**, it **cannot improve reliability** based on **ARQ**. This is why **PBCH cannot** use the **HARQ protocol**. It therefore **improves reliability** with **very robust modulation** and **coding**.

Among other **techniques** to **improve reliability**, **PBCH** uses **scrambling**, **QPSK**, **polar coding**, **CRC** and **sequence-based encoding**. For more detail on **PBCH reliability techniques**, click 6.5.

## 6.3 PBCH scheduling

**PBCH** is **jointly broadcast** with **PSS** and **SSS** (click 6.19) as the "**Synchronization signal** and **PBCH block**" (**SSB**, click 6.1).

The **PBCH scheduling** is therefore the **same** as the one of **SSB** (click 6.1.2), where the **PBCH REs** are located in **fixed positions** within the **SSB** (i.e., in **fixed positions** in respect to **PSS** and **SSS**).

**Each PBCH** instance is (20+20+4+4)•12=**576 Resource Elements** (**REs**), of which **432** (3/4) are **PBCH payload REs** and **144** (1/4) are the **PBCH DM-RS** (click 4.11.3.2.1) **REs**.

**PBCH scheduling of payload PREs and DM-RS PREs**

The **scheduling** of **PBCH DM-RS REs** within the **PBCH REs** depends on the value of the Physical Cell Identity (**PCI**). Being **v**=PCI ***mod*** 4, the **PBCH DM-RS Resource Elements** are:

- **Second** and **fourth symbols** of **SSB**, for **SSB** subcarriers **0+v**, **4+v**, **8+v**,…, **236+v**.
- **Third symbol** of **SSB**, for **SSB** subcarriers: **0+v**, **4+v**, **8+v**,…, **44+v and 192+v**, plus **196+v**, **200+v**,…, **236+v**.

The **remaining REs** in the **PBCH** are the **PBCH payload REs**.

For an **overall** description of **scheduling**, **transmission occasions** and **transmissions**, click 4.11.4.

## 6.4 PBCH payload and DM-RS contents

**PBCH contains 2 types** of **information**:

- **PBCH System Information parameter group**

  For the **data representation** of the **PBCH System Information parameter group**, and for the **parameters** that it contains, click 6.4.1.

  For **all** the **characteristics** of the **PBCH System Information parameter group**, click 5.3.2.

- **Master Information Block** (**MIB**) **parameter group**

  For the **data representation** of the **MIB parameter group**, click 6.4.2.

  For the **parameters** within the **MIB parameter group**, click 8.12.3.

  For **all** the **characteristics** of the **MIB parameter grou**, click 5.3.3.

For the **PBCH processing** of the **PBCH contents**, click 6.5.

### 6.4.1 PBCH contents: PBCH system information parameters

This section does **not** describe the **parameters** that **configure** **PBCH** (as it is the case with **most other sections describing parameters**) and it rather describes the **parameters** that are **broadcast within** **PBCH SI**.

The **PBCH SI parameters** are **generated** by the **physical layer itself**. This is **very unusual**, because **almost all UE parameters** are provided by **higher layers** (i.e., **protocol layers above** the **physical layer**).

The **PBCH SI parameters** have **10** or **11 bits** (see below). It is **10 bits** in **FR1-1** bands and **11 bits** in **FR1-2** or **FR2** bands. There are **8 payload-sent bits** and **2** or **3 sequence-sent bits**. There are **8 payload bits** are **processed** (click 6.5) and **transmitted** within the **PBCH payload REs**. The **2** or **3 sequence bits** are **directly transmitted** as **variations** of the **scrambling sequence** of **PBCH DM-RS**.

The **PBCH system information** parameters are:

- **1 dual use bit**

  It is part of the **8 payload-sent bits** of **PBCH SI**. Its **dual use** is as follows:

  - In **FR1**: it is the **1** most significant **bit** for `ssb-SubcarrierOffset` (click 8.7.1) from `MIB` (click 8.12.3). Therefore, in **FR1** `ssb-SubcarrierOffset` is **extended** with this **dual use bit** to become **5 bits long**. In this case, **2 other bits** of the **8 payload-sent bits** of **PBCH SI** remain **reserved** for future use.
  - In **FR2**: it is used, together with **2 other bits** from the **8 payload-sent bits** of **PBCH SI**, to convey the **3 most significant bits** of parameter **ssb-time-domain-index** (see next bullet).

- **ssb-time-domain-index** parameter (2, 3 or 6 bits)

  It contains the **sequential index** of **each** of the **SSB instances** (click 6.1.2.2.3) within a **SSB burst** when the **NR node** uses **Tx beam sweeping** (click 6.1.2.3). This parameter has the following variants:

  - In **FR1-1**: it is **2 bits**. They are the **2 sequence-sent bits**. These **2 bits** allow for **SSB** instances in **up to 4 beams**.
  - In **FR1-2**: it is **3 bits**. They are the **3 sequence-sent bits**. These **3 bits** allow for **SSB** instances in **up to 8 beams**.
  - In **FR2**: it is **6 bits**. Its **3 least significant bits** are the **3 sequence-sent bits**, while its **3 most significant bits** are part of the **8 payload-sent bits**. These **6 bits** allow for **SSB** instances in **up to 64 beams**.

- **half-frame-flag** parameter (1 bit):

  It indicates if the **SSB** is placed in the **first half-frame** or in the **second half-frame**. It is part of the **8 payload bits** of **PBCH SI**. This parameter, together with the **PBCH SI** parameter **ssb-time-domain-index** (see above) are used to achieve frame synchronization.

- **system-frame-number** parameter (4 bits):

  It conveys the **4 least significant bits** of the **10-bit system frame number** (**SFN**). It is part of the **8 payload-sent bits** of **PBCH SI**. The **6 most significant bits** of the **10-bit SFN** are **broadcast** in `systemFrameNumber` from `MIB` (click 8.12.3).

For **all** the **characteristics** of the **PBCH SI parameter group**, click 5.3.2.

For the **PBCH processing** of the **PBCH SI parameter group**, click 6.5.

For more detail on **PBCH System Information**, refer to TS 38.211 and TS 38.212 and TS 38.331.

Source web pages:

- https://wirelessbrew.com/5g-nr/5g-nr-master-information-block-mib/
- https://www.sharetechnote.com/html/5G/5G_PBCH.html
- http://howltestuffworks.blogspot.com/2019/10/5g-nr-pbch-and-master-information-block.html
- https://rfmw.em.keysight.com/wireless/helpfiles/n7631/Content/Main/Channel%20Setup%20DL.htm
- https://www.sharetechnote.com/html/5G/5G_SS_Block.html#SSB_Transmission_Pattern
- https://www.sharetechnote.com/html/5G/5G_FrameStructure.html#SS_PBCH_FrequencyDomainResourceAllocation

### 6.4.2 PBCH contents: `MIB` representation

The **master information block** (`MIB`) parameters are **provided** to the **physical layer** by **higher layers** (i.e., **protocol layers** above the **NR node physical layer**).

They are provided through the **logical broadcast control channel** (**BCCH**) and through the **transport broadcast channel** (**BCH**).

The `BCCH-BCH-Message` that contains `MIB` has **24 bits**. These **24 bits** are **processed** (click 6.5) and **transmitted** within the **PBCH payload REs**.

Notice that the **ASN.1 encoding** of `MIB` (click 8.12.3) occupies **23 bits**. However, the `BCCH-BCH-Message` received by the **physical layer** (through **BCCH/BCH**) **occupies 24 bits**. The **additional 1 bit** is required because the **RRC message class** `BCCH-BCH-Message` (that contains the `MIB` **message**) has an **ASN.1** `CHOICE` (click 5.9.3), and the `CHOICE` requires **1 additional ASN.1 encoding bit**.

For the **parameters** of the `MIB` **parameters group**, click 8.12.3.

For **all** the **characteristics** of the `MIB` **parameter group**, click 5.3.3.

For the **PBCH processing** of the `MIB` **parameter group**, click 6.5.

## 6.5 PBCH payload and DM-RS processing

**PBCH conveys** the following **information**:

- **2** or **3 bits** of the **PBCH System Information**, which are sent as **variations** of the **scrambling sequence** of the **PBCH DM-RS**.
- **8 bits** of the **PBCH System Information**, which are sent as **PBCH payload REs**.
- **24 bits** of the `BCCH-BCH-Message` that contains the **Master Information Block** (`MIB`), which are sent as **PBCH payload REs**.

These **8+24 bits** that are sent as **PBCH payload REs** are **processed** into **432 PBCH REs** as follows:

1. **Join** the **24 bits** that contain `MIB` and the **8 bits** from the **PBCH system information** (clause 7.1.1 of TS 38.212), resulting in **32 bits**.

2. **Scramble** these **32 bits** (clause 7.1.2 of TS 38.212).
3. **Trail** a **24-bit CRC** for error detection, resulting in **56 bits** (clause 7.1.3 of TS 38.212).
4. **Polar-encode** (**FEC**), resulting in **512 bits** (clause 7.1.4 of TS 38.212).
5. **Rate-match**, resulting in **864 bits** (clause 7.1.5 of TS 38.212).
6. **Scramble** the **864 bits** (clause 7.3.3.1 of TS 38.211).
7. **Modulate** with **QPSK**, resulting in **432 symbols** (clause 7.3.3.2 of TS 38.211).
8. **Map** the **432 symbols** to the **432 Resource Elements** of the **PBCH payload** (clauses 7.3.3.3 and 7.4.3 of TS 38.211).

**PBCH** uses **no HARQ**, **no ARQ**: just **polar encoding FEC** and **CRC** error detection. It is also possible to adjust the **power** of the **SSB** in respect to the rest of the carrier and of the **SSS** in respect to the rest of the **SSB**. The usual configurable range of power adjustment is -40 to 40 dB.

In addition to the above **432 Resource Elements** of the **PBCH payload**, **PBCH** also contains **144 Resource Elements** of **PBCH DM-RS** (click 6.3). The **sequences** used for **PBCH DM-RS** convey **2** or **3 bits** of the **PBCH System Information** (click 6.4.1). The **PBCH DM-RS** can also aid with **beamforming**.

The **subcarrier spacing** and the **cyclic prefix** of **PBCH** are the ones of **SSB** (click 6.1.4).

The **PBCH decoding** just follows the **reverse process** of its **encoding** because the encoding process does not have any options or variations.

Notice that **PBCH** uses a **single layer** transmission scheme (**no space-division multiplexing**). Notice further that **PBCH**, **PSS** and **SSS** within the same **SSB** are transmitted in the same **antenna port** (click 4.11.5).

For the **NR node functional blocks**, the **NR node physical functions**, the **payload processing** of the **physical binary channels**, the **protocol architecture** of the **New Radio** and the **data flow** through the **protocol architecture**, click 4.3.

For more detail on **PBCH payload encoding** and **decoding**, refer to clause 7.3.3 of TS 38.211.

## 6.6 PBCH parameters

**PBCH** does **not** have **any** configuration **parameter**, because for **initial access** (click 8) the **UE** has to be able to receive **PBCH before any** type of **UE configuration** has taken place.

## 6.7 PDCCH overview

The **Physical Downlink Control Channel** (**PDCCH**) is used to send **Downlink Control Information messages** (**DCI messages**, click 6.9) from the **NR node** to **each UE** (or **group** of **UEs**). These **DCI messages** convey **control information** for the **Physical layer**.

The most relevant **functions** of **PDCCH** are:

- **Assign each UE** the **specific PDSCH** (click 6.13) **resources** for **unicast Downlink reception**. This is done by sending to the **UE** a **DCI Format 1_0**, **1_1** or **1_2 message** (click 6.9.5, 6.9.6 or 6.9.7).

- **Assign** a **group** of **UEs** the **specific PDSCH** (click 6.13) **resources** for **multicast** or **broadcast Downlink reception**. This is done by sending to the **group** of **UEs** a **DCI Format 4_0**, **4_1** or **4_2 message** (click 6.9.9.1, 6.9.9.2 or 6.9.9.3).
- **Inform all** its **UEs** about the **scheduling** of **SIB1** (click 5.3.4).
- **Assign each UE** the **specific PDSCH** (click 6.13) **resources** for **unicast Downlink reception** of **Msg2** and **Msg4** during a **random access procedure** (click 9).
- **Adjust each UE**'s **Uplink power** in **PUSCH** and in **PUCCH** (Uplink power control).
- **Various relevant control functions:**
  - **HARQ ACK feedback configuration** of the **UE**.
  - **Trigger CSI Reporting** by the **UE**.
  - **Activation/Deactivation** of **Configured grants**:
  - **Beam Indication** to the **UEs** (critical in **beamforming** and **mmWave** scenarios).
  - **CORESET configuration** for **mini-slot PDDCH**.
  - **Cross-carrier scheduling** (click 10.2.3) and **PDCCH monitoring** on **multiple carriers**.

**Each PDCCH transmission** takes place in **1** of the **transmission occasions determined** by the **PDCCH scheduling** (click 6.8). **Each slot** can contain **0**, **1** or **several PDCCH transmissions** (click 6.9).

**Each PDCCH transmission** contains **1 Downlink Control Information** (**DCI**) **message** (click 6.9).

**Each DCI message** is subject to a **specific processing** (click 6.9) for its **transmission** and **reception** through **PDCCH**.

For the **contents** of **PDCCH DM-RS**, click 6.11.

**PDCCH** is **configured** (click 6.12) with several **parameters** and **data types**.

**PDCCH** and **PUCCH** are the **2 only physical channels** that **do not** carry data from **any transport** channel. They **only** convey **hard real-time control information** for the **physical layer**. Since the **information** they convey is **hard-real time**, they **cannot improve reliability** through **retransmission**. This is why they do **not** use the **HARQ protocol**. They **rather** improve **reliability** with **extremely robust modulation**, **coding** and **diversity**.

Among other **techniques** to **improve reliability**, **PDCCH** uses **scrambling**, **QPSK**, **polar coding** and **CRC**. For more detail on **PDDCH reliability techniques**, click 6.9.

The **PDCCH description** in this book covers almost nothing about **PDCCH** in **shared spectrum**.

For more detail on **PDCCH**, refer to:

- Clauses 4, 5 and 7.3.2 (**Phys.** and **modulation**) of TS 38.211.
- Clauses 4.2, 5 and 7.3 (**mux**., **coding** and **DCI**) of TS 38.212.

- Clause 10 (**DCI reception**) of TS 38.213.
- Clauses 5.2.1 and 5.2.3 (**overall Tx**) and 10.2 (**scheduling**) of TS 38.300.

## 6.8 PDCCH scheduling

**PDCCH Tx scheduling**

The **NR node decides**, for **each** of its **PDCCH transmissions**, its **frequency resources** (click 6.8.3), its **time resources** (click 6.8.2), its **aggregation level** (click 6.8.1.2), its **masking RNTI** (click 6.9.10.2) and its **destination UE** (or **group** of **UEs**).

The **NR node must** select the values of the said **elements** within a **set** of **values** that has been **previously configured** to the **destination UE** (or **group** of **UEs**). This **set** is the **reception occasions** of the **UE**. These **reception occasions** are configured to **each UE** as a **set** of **search spaces** (click 6.8.5).

**Each PDCCH transmission must** belong to **1** of the **search spaces configured** to the **destination UE** (or **group** of **UEs**).

**Each PDCCH transmission** contains **1** and **only 1 DCI message** (click 6.9), that is **addressed** to **1 UE** (or **group** of **UEs**).

**PDCCH Rx scheduling**

It is a **semi-persistent scheduling** (click 4.11.4.3).

Since the **UE only** knows the **reception occasions** (i.e., its **search spaces**) and **not** the **actual receptions**, it **blindly decodes** (click 6.8.4) **each** and **every possible reception occasion**. **Each frequency-time location** where a **PDCCH reception may** be **found** is called a **PDCCH candidate** (click 6.8.2.3).

The **same frequency resources** and the **same symbol intervals** that are defined in a **given search space may** contain **several PDCCH transmissions**. Some relevant characteristics of these **PDCCH transmissions** over the **same symbol intervals** and **within** the **same frequency resources** are:

- **Each** and **every** of them **spans all** the **symbol intervals** that are defined in the **time resources**.
- **Each** and **every** of them **must** be **frequency-aligned** to the **overall CCE structure** (click 6.8.1.1) of the **frequency resources**.
- **Each** of them may have its **own aggregation level** (click 6.8.1.2).
- They may be **contiguous** or **non-contiguous** within the **overall CCE structure** of the **frequency resources**.
- They may be **destined** to the **same** or to **different UEs** (or **groups** of **UEs**).
- They are **not required** to **jointly span all** the **frequency** and **time resources**. defined by the **search space**.

The **REs** that **belong** to the **union** of **all** the **search spaces** configured to **all** the **UEs** in the **cell**, and that are **not used** by **any actual PDCCH transmission** are **set** to **zero** (**no** transmitted **signal**).

For an **overall** description of **scheduling**, **transmission occasions** and **transmissions**, click 4.11.4.

## 6.8.1 Structure of frequency & duration resources

### 6.8.1.1 Control Channel Elements (CCEs)

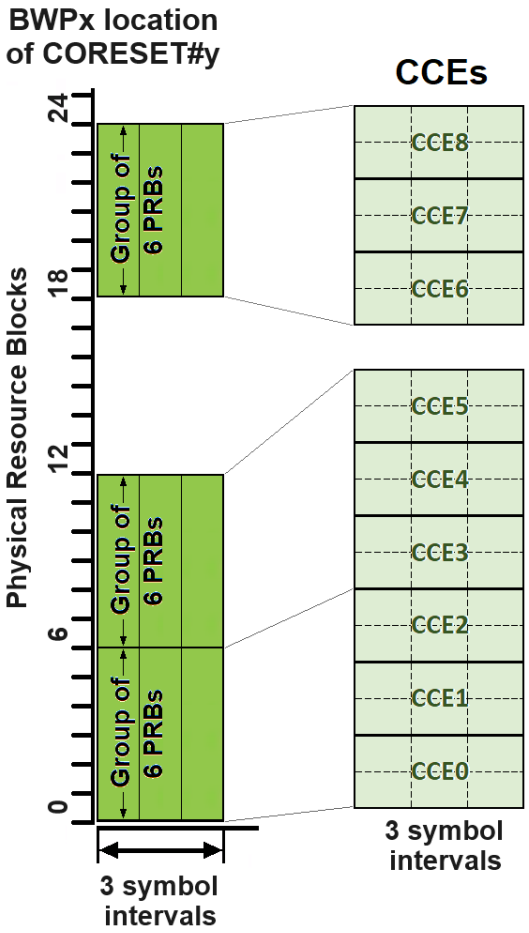
The **PDCCH frequency resources** (click 6.8.2.1) and the **PDCCH duration** (click 6.8.3.1) of **each CORESET** (click 6.8.7) are **jointly structured** in terms of **Control Channel Elements** (**CCEs**). **Each CCE** is **6 Resource Element Groups** (**REGs**, click 1.1.4), where these **6 REGs span all** the **1** to **3 symbol intervals** (click 6.8.3.1) of **every PDCCH transmission** associated with the **CORESET**. Therefore, **each CCE** is **either** of the following:

- **6 contiguous RBs** over **1 symbol interval**.
- **3 contiguous RBs** over **2 symbol intervals**.
- **2 contiguous RBs** over **3 symbol intervals**.

Recall from 6.8 that **each CCE** may be **used** for a **PUCCH transmission** or be **empty** (i.e., its **72 REs** are **set** to **zero**).

The **CCEs** of **each CORESET** are **logically indexed** from **0** to "$N_{CCE}^{CORESET}$-1". This **logical indexing** is **independent** of **each CCE** being **used** or **not**, and it is also **independent** of the **actual frequency location** of **each used CCE**.

The following figure shows the **CCE structure** and **numbering** of the **CORESET example** from 6.8.2.1:



Some **relevant remarks** on this **CORESET** example are:

- There are **9 CCES**, indexed **0** to **8**, following the procedure that has been described above.
- **Each small box** enclosed with dotted lines represents **1 Resource Element Group** (**REG**, click 1.1.4).
- The **units** of the **vertical axis** are **Physical Resource Blocks** (**PRBs**) of the **UE**'s **currently active bandwidth part**.
- The **units** of the **horizontal axis** are **symbol intervals**.
- The **duration** of this **CORESET** is **3 symbol intervals** (the maximum possible).
- The **frequency resources** of this **CORESET** are as follows: its **first range** of **6 contiguous PRBs** is located in **PRBs 0** to **5**, its **second range** is located in **PRBs 6** to **11** (**contiguous** to the **previous block**) and its third **range** is located in **PRBs 18** to **23** (**not contiguous** to the **previous range**).

#### 6.8.1.2 Aggregation level

The **aggregation level** indicates **how many consecutive CCEs** (click 6.8.1.1) are used for **each PDCCH transmission** (i.e., for **each DCI message**, click 6.9). The **aggregation level** may be **1**, **2**, **4**, **8** or **16 CCEs**.

A **larger aggregation level** is used to send a **larger DCI message** and/or for adding a **larger redundancy** to the **DCI message**. The **larger** the **aggregation level**, the **larger** the **DCI message** that can be sent and/or the **larger redundancy** that can be added. The **larger redundancy**, the **stronger error protection**.

### 6.8.2 PDCCH frequency-domain resources

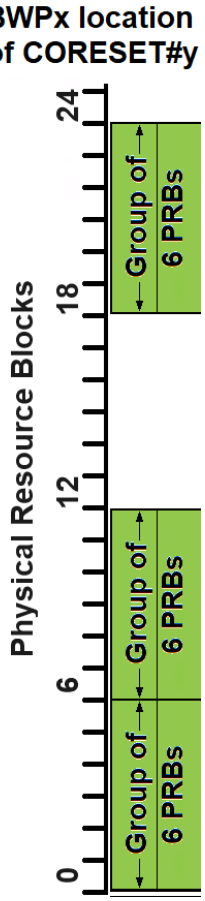
The **NR node decides** the **frequency location** of **each** and **every PDCCH transmission**. **Each PDCCH transmission must** be **located** in **frequency** within the **instance** of **frequency resources** defined in the **CORESET** (click 6.8.7) that is **associated** to **1** of the **search spaces** (click 6.8.5), that have been **configured** to the **destination UE** of **this PDCCH transmission**.

#### 6.8.2.1 PDCCH frequency-domain resource allocation

**Each instance** of **PDCCH frequency resources** is a **set** of **non-overlapping ranges** of **groups** of **6 contiguous Physical Resource Blocks** (**PRBs**). If **several ranges** are contained in the **set**, they are **not contiguous**. The **groups** of **6 contiguous PRBs must** be aligned with the **PRBs** of the **UE**'s **bandwidth part**: the **first group** is **PRBs 0** to **5**, the **second group** is **PRBs 6** to **11** and so on.

**Each CORESET** (click 6.8.7) defines **1** and **only 1 instance** of **PDCCH frequency resources**.

The following figure shows an example of the **frequency resources** of a **CORESET**:




Some **relevant remarks** on this **CORESET** example are:

- There are **3 groups of 6 PRBs**.
- The **units** of the **vertical axis** are **Physical Resource Blocks** (**PRBs**) of the **UE**'s **currently active bandwidth part**.
- The **frequency resources** of this **CORESET** are as follows: its **first range** of **6 contiguous PRBs** is located in **PRBs 0** to **5**, its **second range** is located in **PRBs 6** to **11** (**contiguous** to the **previous block**) and its third **range** is located in **PRBs 18** to **23** (**not contiguous** to the **previous range**).

The **parameter** to configure the **ranges** of **6 contiguous PRBs** of a **CORESET** is:

- ***If*** it is the **CORESET#0** in the **initial bandwidth part** (click 2.2.7.2), the **ranges** of **6 contiguous PRBs** are configured with `controlResourceSetZero` from `MIB` (click 8.12.3), as explained in 6.8.8.3.
- ***Else***, it is a **non-zero CORESET**, and the **ranges** of **6 contiguous PRBs** are configured with `frequencyDomainResources` from `ControlResourceSet` (click 6.12.6). For the **non-zero CORESET** of the **example** above, the value of `frequencyDomainResources` would be `11010…0`.

### 6.8.2.2 CCE RB to PRB mapping

The **mapping** of the **used RBs** (i.e., the **RBs** of the **used CCEs**) to the **actually transmitted PRBs** may be **either non-interleaved** (click 6.8.2.2.1) or **interleaved** (click 6.8.2.2.2).

For the **CCEs** of **CORESET#0**, **interleaved mapping** is **always** used.

For the **CCEs** of a **non-zero CORESET**, the **CORESET** mandatory parameter `cce-REG-MappingType` from `ControlResourceSet` (click 6.12.6) determines **which** of both **mappings** is used.

The **precoding** (click 2.4.2) for **both interleaved** and **non-interleaved mapping** depends on the **value** of `precoderGranularity` from `ControlResourceSet` (click 6.12.6), as follows:

- If it is `sameAsREG-bundle`:

  The **same precoding** is used within a **REG bundle**.

- If it is `allContiguousRBs`:

  The **same precoding** is used across **all REGs** within the set of **contiguous PRBs** in the **CORESET**'s **frequency resources**, and **no resource element** in the **frequency resources overlaps** with a **SSB**, **nor** with **LTE cell-specific** reference signals as indicated by **either** `lte-CRS-ToMatchAround`, `lte-CRSPatternList1`, or `lte-CRS-PatternList2`.

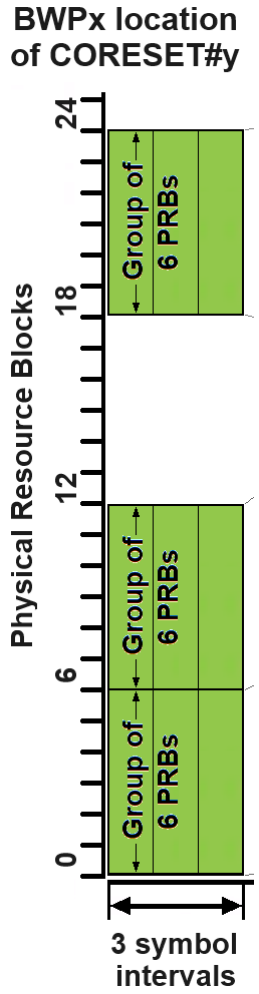
Do **not confuse** the **interleaved mapping** of **used RBs** to **PRBs** with the **CRB interlaces** (click 2.2.4.7).

#### 6.8.2.2.1 CCE non-interleaved mapping

In the **non-interleaved frequency mapping**, the **used RBs** (i.e., the **RBs** of the **used CCEs**, click 6.8.1.1) are **transmitted** in the **same frequency location** as they have been **defined** within the **CORESET**'s **frequency resources** (click 6.8.2).

Recall from 6.8 that **each unused REG** is **set** to **zero**.

The following figure shows the **PRB** where **each RB** from the **CORESET**'s **frequence resources** would be transmitted, following the same example from sections 6.8.1.1 and 6.8.2.1:




#### 6.8.2.2.2 CCE interleaved mapping

In the **interleaved frequency mapping**, the **used RBs** (i.e., the **RBs** of the **used CCEs**, click 6.8.1.1) are **transmitted** in **different frequency locations** to the ones where they have been **defined** within the **CORESET** (click 6.8.7). The **interleaved mapping distributes** in a **pseudo-random** way the **RBs** of the **used CCEs**, across **all** the **RBs within** the **frequency resources** of the **CORESET**.

The **frequency interleaving** of the **used RBs** is performed in terms of **PDCCH REG bundles**. A **REG bundle** is composed of "***L***" **consecutive REGs**.

**Each PDCCH REG bundle must** occupy **all** the **PDCCH transmission symbol intervals** "***S***", because otherwise they would **not** be aligned to **Resource Blocks**, and it would **not** be possible to interleave their **RBs** in frequency. This **restricts** the possible values of "***L***", depending on the value of "***S***". The **interleaving algorithm further restricts** the possible values of "***L***", and the result is:

- ***If*** "***S***=1" or "***S***=2", ***then*** "***L***" may be 2 or 6.
- ***If*** "***S***=3", ***then*** "***L***" may be 3 or 6.

The value of "***L***" is a **parameter** for the **interleaving function** (see below).

The **REG bundles** within a **CORESET** are **indexed** (click 6.8.2.2.3) starting with **0** for the **lowest frequency REG bundles** and increasing **1 unit upwards**.

**Each CCE** contains "**6/*L*" REG bundles**. The **REG bundles** of **CCE** with index "***j***" are **transmitted** in the following **REG bundle locations**:

$$\mathbf{CCE_j} \in \{f(j\cdot 6/L), f(j\cdot 6/L+1), \ldots, f(j\cdot 6/L+L-1)\}$$

where "***f***()" is an **interleaving function**. The **interleaving** function "***f***()" performs an **interleaved mapping** of the "**6/*L*" contiguous REG bundles** of **each CCE** to "**6/*L*" non-contiguous interleaved REG bundles**, where **each interleaved REG bundle** is **placed** in **its specific frequency location**. The result is that the **REG bundles** contained in the different **CCEs** are "**shuffled**" in frequency, so that the **CCEs are spread through**.

The **interleaving** function "***f***()" is defined as:

$$f(x) = (r\cdot C + c + n_{shift}) \ mod \ (N_{REGbundle}^{CORESET})$$

where the different symbols have the following meaning:

| Symbol | Meaning | Value(s) | Comments |
|---|---|---|---|
| $N_{REGbundle}^{CORESET}$ | **Quantity** of **REG bundles** in the **CORESET**. | $N_{REG}^{CORESET}/L$ | $N_{REG}^{CORESET}$ is the **quantity** of **REGs** in the **CORESET**. |
| ***x*** | **Index** of a **REG bundle**, **and** the **argument** to ***f***(). | **0** to $N_{REGbundle}^{CORESET}$-1 | The result of ***f***() is an **integer always within** the **range** of ***x***. |
| ***r*** | A **variable** to compute ***f***(). | 0 to ***R***-1 | |
| ***R*** | Interleaver size. | {2, 3, 6} | Configuration **parameter** for ***f***(). |
| ***c*** | A **variable** to compute ***f***(). | 0 to ***C***-1 | |
| ***C*** | Interleaver size. | $N_{REGbundle}^{CORESET}/R$ | The **UE cannot** handle the case where ***C*** is **not** an **integer**. |
| $n_{shift}$ | Interleaver shift. | 0 to 274 or **0** to **1,007** | Configuration **parameter** for ***f***(). |
| ***L*** | **Quantity** of **REGs** in each **REG bundle**. | Depending on ***S***, 2, 3 or 6 | Configuration **parameter** for the interleaving. |

Notice that the **function argument** "***x***" does **not** appear in the definition of the **function**. The reason is that the value of "***x***" has the following relationship with **variables** "***r***" and "***c***":

$$x = c\cdot R + r$$

The result of this relationship may be better understood with the following examples:

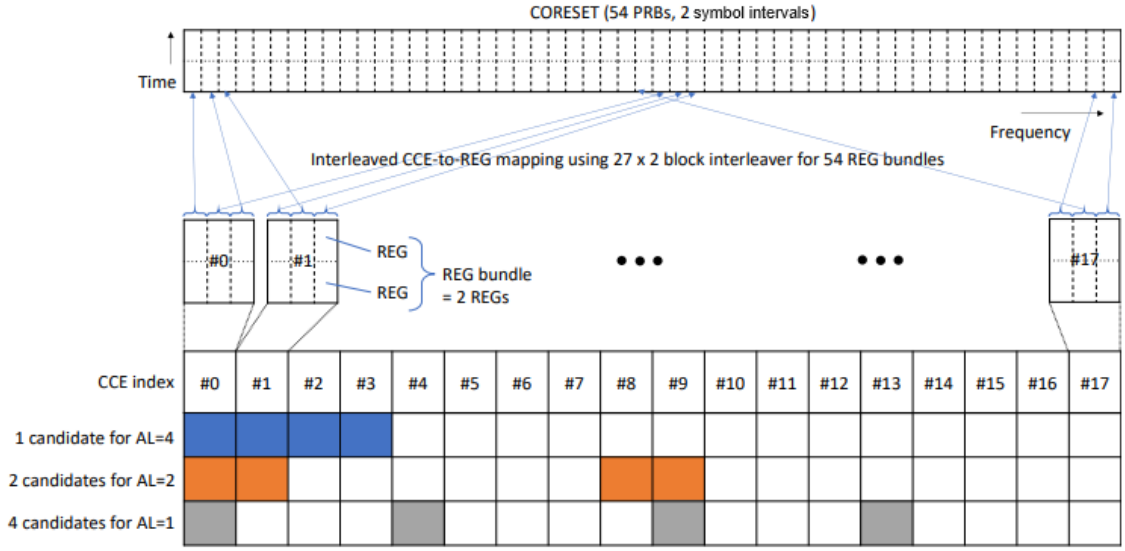
| $N_{REGbundle}^{CORESET}$=4, ***R*** = 2 => ***C*** =2 | | | |
|---|---|---|---|
| ***c*** | ***cR*** | ***r*** | ***x*** |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 1 | 2 | 0 | 2 |
| 1 | 2 | 1 | 3 |

| $N_{REGbundle}^{CORESET}$=6, ***R*** = 3 => ***C*** =2 | | | |
|---|---|---|---|
| ***c*** | ***cR*** | ***r*** | ***x*** |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 0 | 2 | 2 |
| 1 | 3 | 0 | 3 |
| 1 | 3 | 1 | 4 |
| 1 | 3 | 2 | 5 |

Some **relevant remarks** on these tables are:

- Notice that the **range** of "***x***" is from **0** to "$N_{REGbundle}^{CORESET}$-1".
- Recall that "***C***" **must** be an **integer**.

The following figure illustrates **interleaved mapping**:




For **CORESET#0**, the **interleaving** configuration is:

| Interleaver parameter | Meaning | Value(s) | Comments |
|---|---|---|---|
| ***L*** | **Quantity** of **REGs** in each **REG bundle**. | 6 | A **REG bundle** is a **whole CCE.** |
| ***R*** | Interleaver size. | {2, 3, 6} | Configuration **parameter** for ***f*** |
| $n_{shift}$ | Interleaver shift. | **0** to **1,007** | It is the value of the cell **PCI** (click 3.5.3.2) |

Some **relevant remarks** on the **CORESET#0 interleaving configuration** are:

- The **UE** may assume **normal cyclic prefix** when **CORESET#0** is configured by `MIB` or by `SIB1`.
- The **UE** may assume the **same precoding** being used within **each REG bundle**.

For **non-zero CORESETs**, the **interleaving** configuration is:

| Interleaver parameter | Meaning | Value(s) | RRC parameter |
|---|---|---|---|
| ***L*** | **Quantity** of **REGs** in each **REG bundle**. | {2, 3, 6} | `reg-BundleSize` from `ControlResourceSet` (click 6.12.6). |
| ***R*** | Interleaver size. | {2, 3, 6} | `interleaverSize` from `ControlResourceSet` (click 6.12.6). |
| $n_{shift}$ | Interleaver shift. | 0 to 274 | `shiftIndex` from `ControlResourceSet` (click 6.12.6). |

The **interleaved mapping** is used for **frequency diversity**.

For more detail on **interleaved mapping**, refer to clause 7.3.2.2 of TS 38.211.

Recall from 6.8 that **each resulting interleaved REG** that corresponds to an **unused REG** is **set** to **zero**.

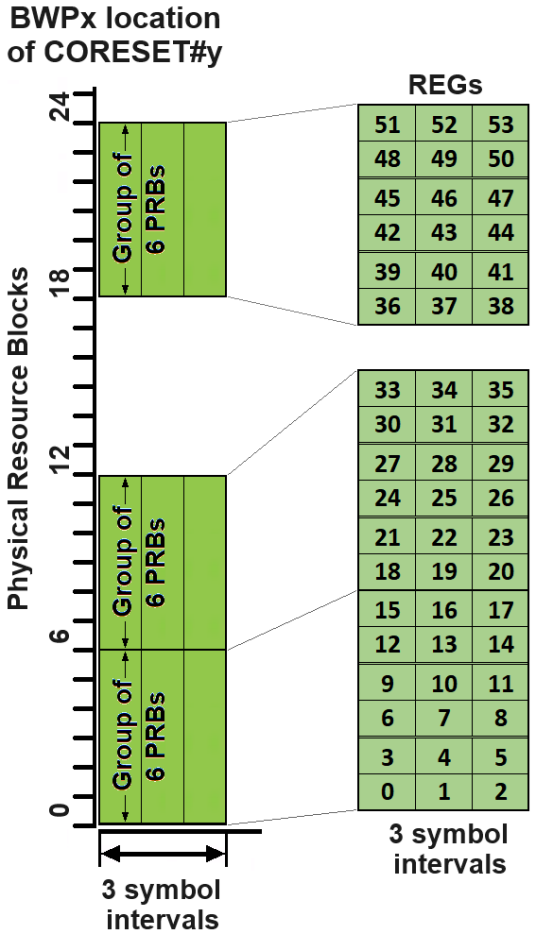
Source web pages:

- https://medium.com/@bhagwan.sahai/5gnr-interleaved-cce-reg-mapping-in-coreset-175de874a132

#### 6.8.2.2.3 PDCCH REG indexing

The **Physical REGs** (click 1.1.4) of **each CORESET** (i.e., of its **frequency resources** and **duration**) are **indexed**, starting with **0** for the **lowest frequency** in the **first symbol interval**, and increasing **1 unit first** in **symbol intervals**, and **second** in **frequency**. Notice that this **index** is **unaffected** by the **frequency locations** of the **REGs** of the **CORESET** and it is **only** dependent on the **total frequency size** and **time duration** of the **CORESET**. **Each CORESET** is associated to a **search space** (click 6.8.5) to which **each PUCCH transmission** belongs.

The following figure shows the **REG** (click 1.1.4) **indexing** of the **CORESET** example from section 6.8.1.1:




Some **relevant remarks** on this figure are:

- There are **54 REGs**, indexed **0** to **53**, following the procedure that has been described above.
- **Each box** of **6 REGs** enclosed with double lines represents **1 CCE** (click 6.8.1.1).

- The **units** of the **vertical axis** are **Physical Resource Blocks** (**PRBs**) of the **UE**'s **currently active bandwidth part**.
- The **units** of the **horizontal axis** are **symbol intervals**.
- The **duration** of this **CORESET** is **3 symbol intervals** (the maximum possible).
- The **frequency resources** of this **CORESET** are as follows: its **first range** of **6 contiguous PRBs** is located in **PRBs 0** to **5**, its **second range** is located in **PRBs 6** to **11** (**contiguous** to the **previous block**) and its third **range** is located in **PRBs 18** to **23** (**not contiguous** to the **previous range**).

#### 6.8.2.3 PDCCH candidates

**Each PDCCH candidate** is a **specific frequency location** and **aggregation level** (click 6.8.1.2), within a given **monitoring occasion** (click 6.8.3.2) and with the corresponding **duration** (click 6.8.3.1), where a **UE may** find a **PDCCH transmission** destined to **itself**.

**Each** and **every PDCCH candidate must** be associated with **1** of the **search spaces configured** to the **UE**. The corresponding **search space** defines:

- The **time duration** of the **PDCCH candidate**. The **time duration** is defined in the **CORESET** associated to the **search space**.
- The **instance** of **frequency resources** within which the **PDCCH candidate must** be **located**. The **instance** of **frequency resources** is defined in the **CORESET** associated to the **search space**.
- The **set** of **possible aggregation levels**, to which the **aggregation level** of the **PDCCH candidate must belong**.
- The **set** of **monitoring occasions**, to which the **monitoring occasion** of the **PDCCH candidate must belong**.

For **each** of the **UE**'s **search spaces** (denoted "$s$"), the **UE computes** the **frequency location** "$CCE_{idx}()$" of **each** and **every PDCCH candidate** at **aggregation level** "$L$", with the **pseudo-random** expression:

$$CCE_{idx}(m_{s,nCI}^{(L)}) =$$
$$i + L \cdot \{ \quad [Y_{p(s),n(\mu,s,f)} + floor((m_{s,nCI}^{(L)} \cdot N_{CCE,p(s)}) / (L \cdot M_{s,max}^{(L)})) + n_{CI}\ ]$$
$$mod\ [floor(N_{CCE,p(s)} / L)]\ \}$$

where:

- "$m_{s,nCI}^{(L)}$" is the **index** of the **PDCCH candidates** at **aggregation level** "$L$". It **ranges** from **0** to "$M_{s,nCI}^{(L)}$-1". **Each** of its values results in a **CCE index** "$CCE_{idx}$" that indicates the **location** of the corresponding **PDCCH candidate**.
- "$M_{s,nCI}^{(L)}$" is the **quantity** of **PDCCH candidates** at **aggregation level** "$L$". This **quantity** is configured in the **search space** "$s$" for a **serving cell** corresponding to "$n_{CI}$".
- "$M_{s,max}^{(L)}$" depends on the **class** of **search space** "$s$":
  - If "$s$" is a **common search space**:
    $$M_{s,max}^{(L)} = M_{s,0}^{(L)}$$

- If "$s$" is a **UE-specific search space**:

  $M_{s,max}^{(L)}$ is the **maximum** "$M_{s,nCI}^{(L)}$" among **all** the possible values of "$n_{CI}$".

- "$n_{CI}$" is the **carrier indicator** field value ***if*** the **UE** is configured with a **carrier indicator field** by `crossCarrierSchedulingConfig` (from `ServingCellConfig`, click 3.9.7) for the **serving cell** on which **PDCCH** is **monitored**, **except** for scheduling of the **serving cell** from the **same serving cell** (in which case "$n_{CI}$=0"). ***Else***, including for **any common search space**, "$n_{CI}$=0".
- "$i$" is an **index** that **ranges** (in steps of **1 unit**) from **0** to "$L$-1".
- "$p(s)$" is the **CORESET** associated to the **search space** "$s$".
- "$n(\mu,s,f)$" is the **frame slot index** (within **frame** with **index** "$f$") that contains the present **monitoring occasion**, for **subcarrier spacing** "μ", referred to the **search space** "$s$".
- "$N_{CCE,p(s)}$" is the **quantity** of **CCEs** defined by **CORESET** "$p(s)$" and, if any, per **RB set**.
- "$Y_{p(s),n(\mu,s,f)}$" is a variable that depends on several factors, as follows:
  - If "$s$" is a **common search space**:

    $Y_{p(s),n(\mu,s,f)} = 0$

  - If "$s$" is a **UE-specific search space**, the variable is computed with the **recursive** expression:

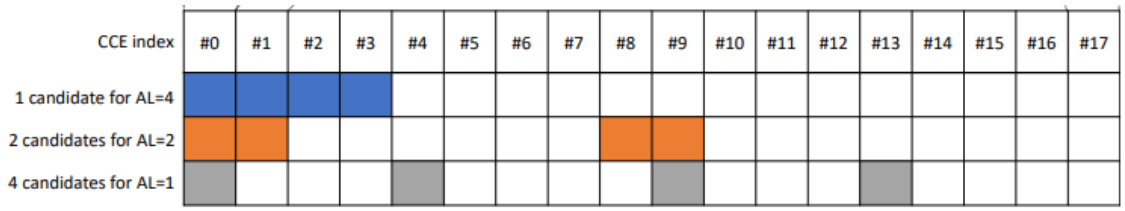
    $Y_{p(s),n(\mu,s,f)} = (A_{p(s)} \bullet Y_{p(s),n(\mu,s,f)-1})\ mod\ D$

    where "$Y_{p(s),-1}$" is the **UE**'s **C-RNTI** (click 6.9.10.2), "$D$=65537" and "$A_{p(s)}$" depends on the **result** of "$p(s)$ **mod** 3" as follows:

    - ***If*** the **result** is **0**, ***then***, "$A_{p(s)}$= 39827".
    - ***If*** the **result** is **1**, ***then***, "$A_{p(s)}$= 39829".
    - ***If*** the **result** is **2**, ***then***, "$A_{p(s)}$= 39839".

Be aware that the above description applies to a quite frequent case, but clause 10.1 of TS 38.213 has **several pages** of **exceptions** and **special cases** in respect to the **frequency location** of the **PDCCH candidates**.

The following figure illustrates the **frequency** (**CCE**) **location** of the **PDCCH candidate**(**s**) for **each** of **3** different **aggregation levels** (AL):




For more detail on the **location** of the **PDCCH candidates**, refer to clause 10.1 of TS 38.213.

### 6.8.3 PDCCH time-domain resources

The **NR node decides** the **time location** of **each** and **every PDCCH transmission**. **Each PDCCH transmission must start** in a **symbol interval** corresponding to **1** of the **configured monitoring occasions** (click 6.8.3.2) and have a **duration** (click 6.8.3.1) of the configured **quantity** of **symbol intervals**. **Both** the **monitoring occasions** and the **duration must** belong to **1** of the **search spaces** (click 6.8.5) that have been **configured** to the **destination UE** currently **active bandwidth part**. **Each search space** contains a **set** of **periodic monitoring occasions**. The **duration** is defined in the **CORESET** (click 6.8.7) that is **associated** to the said **search space**.

#### 6.8.3.1 PDCCH duration

**All** the **PDCCH transmissions** associated to a given **search space** (click 6.8.56.8.4) have the **same time duration**. This **time duration** may be **1**, **2** or **3 contiguous symbol intervals**. The value of the **time duration** is defined in the **CORESET** (click 6.8.7) that is associated to the **search space**. The **parameter** that configures the **time duration** depends on the **type** of **CORESET**:

- ***If*** it is the **CORESET#0** of the **initial bandwidth part** (click 2.2.7.2), ***then***, the **time duration** of is configured with `controlResourceSetZero`, as explained in 6.8.8.2.
- ***Else***, the **time duration** is configured with `duration` from `ControlResourceSet` (click 6.12.6). For the **CORESET** of the **examples** from 6.8.1.1 and 6.8.2.2.3, the value of `duration` would be `3`.

The **time duration** of **PDCCH** of **each non-zero CORESET** is explained in 6.8.1 and of **CORESET#0** is explained in 6.8.8.3.

#### 6.8.3.2 PDCCH monitoring occasions

The **periodical slots** that contain the **monitoring occasions** of a **search space** are **jointly** configured with `monitoringSlotPeriodicityAndOffset` and `duration`.

The **indexes** of the **specific symbol intervals** (within **each such slot**) where the **monitoring occasions** are located are configured with `monitoringSymbolsWithinSlot`.

These **3 parameters** are from `SearchSpace` (click 6.12.8).

### 6.8.4 Blind decoding

**Blind decoding** consists of the **UE decoding each** and **every** of its possible **PDCCH candidates** (click 6.8.2.3), with **all** the corresponding **RNTIs** (click 6.9.10), to **detect each** and **every PDCCH candidate** that contains **1 DCI message** (click 6.9) addressed to the **UE**.

The **blind decoding** procedure performed by the **UE** may be defined as

**For each** of the **UE**'s **search spaces** (click 6.8.5), and

**for each** of the search space's **monitoring occasions** (click 6.8.3.2), and

**for each** of the search space's **aggregation levels** (click 6.8.1.2), and

**for each** of the aggregation level's **PDCCH candidate**, and

**for each** of the **UE**'s **RNTI(s)** (click 6.9.10) that can be used for the **DCI Formats** (click 6.9) that are configured (click 6.8.5.2) in the **search space**, do:

**decode** the **PDCCH candidate** and check if its **unmasked CRC** is **correct**. If it is **correct**, **process** the incoming **DCI message**.

At **each monitoring occasion**, the **UE may find none** or **several actual PDCCH candidates** because **not all** the **locations** of the **PDCCH candidates** will contain **received signal**. **Each** and **every actual PDCCH candidate** is **decoded**, and it **may** contain, or **not**, **1 DCI message addressed** to the **UE**. Consequently, at **each monitoring occasion** the **UE may receive none** or **several DCI messages addressed** to the **UE**.

The **blind decoding** process is **usually parallelized** because it has **very strict computing time restrictions** arising from the **hard-real time nature** of the **control information** conveyed **through PDCCH**.

The **UE knows neither** the **time location**, **nor** the **frequency location**, **nor** the **aggregation level**, nor the **masking RNTI** of **each actual PUCCH transmission** that contains a **DCI message** that is **addressed** to the **UE**. The **UE only** knows the **set** of **possible values** that determine a **PUCCH transmission** that contains a **DCI message** that is **addressed** to the **UE**. This is why **blind decoding** is **required**.

## 6.8.5 PDCCH search spaces

**Each** of the **UE**'s **search spaces** defines the **periodical locations** and **characteristics** of **PDCCH candidates** over which the **UE** performs **blind decoding** (click 6.8.4) in order to **retrieve** the **DCI messages** (click 6.9) that are **addressed** to the **UE.**

**Each search space** can be characterized by:

- Its **search space class** (**common** or **UE-specific**, click 6.8.5.1).
- Its **search space type** (click 6.8.5.5).
- Its **periodical monitoring occasions** (click 6.8.3.2).
- The possible **Formats** (click 6.9) of the **DCI messages** that are contained in its **PDCCH candidates**.
- The possible **aggregation level**(**s**) (click 6.8.1.2) of its **PDCCH candidates**, and for **each** of them, the **quantity** of **PDCCH candidates**.
- Its **associated CORESET** (click 6.8.7).
- Its **search space identifier** (click 6.8.5.3).
- Its **search space role** (click 6.8.5.4).

**Each search space** has **1**, and **only 1**, **associated CORESET**. However, **each CORESET** may be **associated** to **several search spaces**.

For the **search spaces configuration** of **each UE**, click 6.8.5.2.

A **search space** is sometimes called (e.g., in TS 38.213 and TS 38.214) a "**search space set**" and/or a "**PDCCH search space**". Within this book, it is almost always called "**search space**" and few times "**PDCCH search space**". Notice however that there is **DCI message field** that is called "***Search space set group switching flag …***".

You may find a couple **examples** of **search spaces** in section 6.8.5.6.

For more detail on **search spaces**, refer to clause 10.1 of TS 38.213.

Source web pages:

- https://www.nrexplained.com/pdcch_assign
- https://www.sharetechnote.com/html/5G/5G_SearchSpace.html
- https://www.rfwireless-world.com/5G/5G-NR-Search-Space-types.html

### 6.8.5.1 Search space classes

**Each** of the **UE**'s **search space must** belong to **either** of the following **two classes**:

- **Common Search Spaces** (**CSS**)

  **Each** of them is **searched** and **decoded** by **all** the **UEs** in the corresponding **cell**. They are used to receive **system information block 1** (`SIB1`, click 5.3.4), **other SIBs and posSIBs** (click 5.3.5) and the signaling of the **random access procedure** (click 9). They can be the **search space zero** (click 6.8.6) or **non-zero common search spaces**.

- **UE-specific Search Spaces** (**USS**)

  **Each** of them is **searched** and **decoded** by the **UE** to which it has been **configured** (via **RRC**). Their usage therefore requires that the **UE** has completed network attachment and **RRC** establishment. The are always **non-zero search spaces**.

Aside from these **2 classes** of **search spaces**, there is the following **relevant distinction** between **search spaces**. **Each** of the **UE**'s **search space** may be **search space zero** or a **non-zero search space**:

- **Search space zero**

  It is the **common search space** that **always** has **identifier** `0` and it has several **special** characteristics (click 6.8.6).

- **Non-zero search spaces**

  They are **search spaces** with an identifier other than `0`. They can be **common** or **UE-specific**.

### 6.8.5.2 Search spaces configuration (and quantities) of an UE

**All** the **Downlink bandwidth parts** in **each** of the **UE**'s **cells** are configured with a **joint total** of **up to** `40` **search spaces** (including **search space zero**). This **maximum quantity** derives from the **maximum quantity** of **search space identifiers** which is `40` (click 6.8.5.3).

**Each** of the **UE**'s **Downlink bandwidth parts** is configured with the following **search spaces**:

- **Up to 1 search space zero**

  It **must** be configured in the **Downlink initial bandwidth part** (**BWP#0**, click 2.2.7.2). It **may be** configured in the **Downlink initial bandwidth part** of **other UE**'s **cells**.

  **Each search space zero** is configured **either** with `searchSpaceZero`, with **data type** `SearchSpaceZero` (click 8.12.3) or with an **instance** of `SearchSpace` (click 6.12.8), as its is described in 6.8.6.1.

- **Up to `4` non-zero common search spaces**

  They are configured with `commonSearchSpaceList`, `commonSearchSpaceListExt-r16` **or** `commonSearchSpaceListExt2-r17`, the three of them from `PDCCH-ConfigCommon` (click 6.12.4). Their **data types** are (respectively) a `SEQUENCE(SIZE (1..4)) OF SearchSpace`, a `SEQUENCE(SIZE (1..4)) OF SearchSpaceExt-r16` and a `SEQUENCE(SIZE (1..4)) OF SearchSpaceExt-v1700`. Therefore, **each non-zero common search space** is configured with an **instance** of **either** of the **3** said **data types**.

- **Up to `10` UE-specific search spaces**

  They are configured with `searchSpacesToAddModList` from `PDCCH-Config` (click 6.12.5). Its **data type** is a `SEQUENCE(SIZE (1..10)) OF SearchSpace` (click 6.12.8). Therefore, **each UE-specific search space** is configured with an instance of `SearchSpace` (click 6.12.8). Recall from 6.8.5.3 that **UE-specific search spaces** are **always non-zero search spaces**.

For more detail on **PDCCH parameters**, click 6.12.

Notice that the **maximum quantity** of **physical layer elements** (**cells**, **bandwidth parts**, **CORESETs**, **search spaces**, etc) that can be **configured** to **each UE** is **not** the same as the **maximum quantity** that a **NR node** can handle.

### 6.8.5.3 Search space identification

**Each search space** is **identified** with an **instance** of `SearchSpaceId`, which is an `INTEGER (0..maxNrofSearchSpaces-1)`. The constant `maxNrofSearchSpaces-1` is `39`, which is the **maximum quantity**, minus **1**, of **UE**'s **search spaces** in **each** of the **UE**'s **cells**.

The **search space identifier uniquely** identifies **each** of the **UE**'s **search spaces** within **each** of the **UE**'s **serving cells**.

### 6.8.5.4 Search space roles

A **search space** can be assigned a specific **role**, by assigning its **identifier** to one of the following **parameters** from `PDCCH-ConfigCommon` (click 6.12.4), **all** of them with **data type** `SearchSpaceId` (click 6.12.9):

- `searchSpaceSIB1`
- `searchSpaceOtherSystemInformation`
- `pagingSearchSpace`
- `ra-searchSpace`
- `searchSpaceMCCH-r17`
- `searchSpaceMTCH-r17`
- `pei-SearchSpace-r17`

It is also possible to assign the **role** of **small data transmission** (**SDT**, click 10.9) to an **existing** or **newly defined search space**, with the following parameter, that is also from **PDCCH-ConfigCommon** (click 6.12.4):

```
sdt-SearchSpace-r17 CHOICE {
  newSearchSpace SearchSpace,
  existingSearchSpace SearchSpaceId} OPTIONAL,                    -- Need R
```

### 6.8.5.5 Search space types

Within **each search space class**, there are different **types** of **search spaces** depending on the **DCI message Format** transported, the **RNTI** used to mask the **CRC** of the **DCI message** and the **role** assigned to the **search space**. The following table (based on clause

10.1 of TS 38.213) summarizes the **space types** indicating for **each** of them the **cell type(s)**, the **DCI Format(s)** and the **RNTI(s) characteristics**:

| Class | Type | searchSpaceId = | Cell type | RNTI(s) | DCI Format(s) | DCI message schedules |
|---|---|---|---|---|---|---|
| CSS | CSS-Type 0 | 0, even if configured by searchSpaceMCCH-r17 or searchSpaceMTCH-r17, both from PDCCH-ConfigCommon (within SIB1) | PCell | SI-RNTI | 1_0 | **Broadcast** of SIB1. |
| | | | | G-RNTI for broadcast | 4_0 | **Dynamically** scheduled **MBS PTM** transmission. |
| | | | | MCCH-RNTI | | **Dynamically** scheduled **MCCH signaling** and **MCCH change notification**. |
| | CSS-Type 0A | searchSpaceOtherSystemInformation from PDCCH-ConfigCommon (within SIB1) | PCell | SI-RNTI | 1_0 | **Broadcast** of **other** SI |
| | CSS-Type 0B | searchSpaceMCCH-r17 or searchSpaceMTCH-r17, both from PDCCH-ConfigCommon (within SIB1) | PCell | G-RNTI for broadcast | 4_0 | **Dynamically** scheduled **MBS PTM** transmission |
| | | | | MCCH-RNTI | | **Dynamically** scheduled **MCCH signaling** and **MCCH change notification** |
| | CSS-Type 1 | ra-SearchSpace from PDCCH-ConfigCommon (within SIB1) | SpCell | MSGB-RNTI | 1_0 | **RA Response** (**MsgB**) in **2-step CFRA**. |
| | | | | RA-RNTI | | **RA Response** (**Msg2**) in **4-step CBRA** |
| | | | | TC-RNTI | | **Contention Resolution** (**Msg4** in **4-step CBRA** or **MsgB** in **2-step CBRA**) |
| | CSS-Type 1A | sdt-SearchSpace-r17 from PDCCH-ConfigCommon (within SIB1) | SpCell | C-RNTI, CS-RNTI | 1_0, 1_1, 1_2 | Click 6.9.10.2 |
| | CSS-Type 2 | pagingSearchSpace from PDCCH-ConfigCommon (within SIB1) | PCell | P-RNTI | 1_0 | **Paging** and **System Information change** notification. |
| | CSS-Type 2A | pei-SearchSpace-r17 from PDCCH-ConfigCommon (within SIB1) | PCell | PEI-RNTI | 2_7 | **Paging Early Indication**. |
| | CSS-Type 3 | searchSpaceId from any **common** search space defined within PDCCH-Config | SpCell | C-RNTI, MCS-C-RNTI, CS-RNTI(s) | 1_0, 1_1, 1_2 | **Configured** or **Dynamically** scheduled **unicast** or **multicast** **Downlink** transmission. |
| | | | | PS-RNTI | 2_6 | **DCP** to indicate whether to start sl-drx-onDurationTimer-r17 for associated **DRX** cycle. |
| | | | Any | INT-RNTI, SFI-RNTI, TPC-xxx-RNTI, CI-RNTI | 2_0, 2_1, 2_2, 2_3, 2_4 | Click 6.9.10.2 |
| | | searchSpaceMCCH-r17 or searchSpaceMTCH-r17, both from PDCCH-ConfigCommon (within SIB1) | SCell | G-RNTI for broadcast | 4_0 | **Dynamically** scheduled **MBS PTM** transmission. |
| | | | | MCCH-RNTI | 4_0 | **Dynamically** scheduled **MCCH** signaling and **MCCH change** notification. |
| | | searchSpace from PDCCH-Config, within CFR-ConfigMulticast-r17 | Any | G-RNTI | 4_1, 4_2 | **Dynamically** scheduled **MBS PTM** transmission. |
| | | | | G-CS-RNTI | | **Configured** scheduled **multicast** transmission. |

| Class | Type | searchSpaceId = | Cell type | RNTI(s) | DCI Format(s) | DCI message schedules |
|---|---|---|---|---|---|---|
| **USS** | **UE-specific** | searchSpaceId from any **UE-specific** search space defined within PDCCH-Config | **Any** | **C-RNTI, MCS-C-RNTI, SP-CSI-RNTI, CS-RNTI(s), SL-RNTI, SL-CS-RNTI, SL Semi-Persistent Scheduling V-RNTI** | **0_0, 0_1, 0_2, 1_0, 1_1, 1_2, 3_0, 3_1** | **Unicast messages**. Click 6.9.10.2 |

Some **relevant remarks** on this table are:

- The column "**searchSpaceId =**" indicates what **parameter**(s) contain(s) the value of the **identifier** of the corresponding **search space**.
- The **Common Search Spaces cannot** be used to schedule **Uplink** transmissions, hence the corresponding **DCI Formats** and **Usage**.
- The indication "CS-RNTI(s)" comes from the table source (clause 10.1 of TS 38.213) and stands for CS-RNTI, CG-SDT-CS-RNTI and G-CS-RNTI.
- The indication "TPC-xxx-RNTI" stands for TPC-PUCCH-RNTI, TPC-PUSCH-RNTI and TPC-SRS–RNTI.
- The table is missing the AI-RNTI because it only applies to a very specific case of a IAB-node. For more detail, refer to the top of page 219 of TS 38.213.
- For more detail on **MIB**, click 5.3.3.
- For more detail on **SIB1**, click 5.3.4.
- For more detail on **PDCCH-ConfigCommon**, click 6.12.4.
- For more detail on **PDCCH-Config**, click 6.12.5.

The following **relevant tables** are complementary to this table:

- Table with **all** the **DCI message formats** with their related **RNTI**(s), and their **combined function**, click 6.9.
- Table with **all** the **RNTIs**, with a **short individual description**, click 6.9.10.1.
- Table with **all** the **RNTIs** with their related **DCI Formats**, their **mapping** to **channels**, and their **combined function**, click 6.9.10.2.

### 6.8.5.6 Search space configuration examples

These examples correspond to **non-zero search spaces**.

**Example A of search space configuration**

As a **first** example, the **instance** of **SearchSpace** (click 6.12.8) that configures a **search space** contains the following **parameter** values:

```
searchSpaceId = 5,
controlResourceSetId = 6
monitoringSlotPeriodicityAndOffset CHOICE
  sl10 = 2
duration = 4
monitoringSymbolsWithinSlot = 10000001000000
```

```
nrofCandidates SEQUENCE
  aggregationlevel1  = n0
  aggregationlevel2  = n0
  aggregationlevel4  = n0
  aggregationlevel8  = n6
  aggregationlevel16 = n8
searchSpaceType CHOICE
  ueSpecific SEQUENCE
    dciFormats = formats0-0-And-1-0
```

These **parameter values** for this **search space** example imply that:

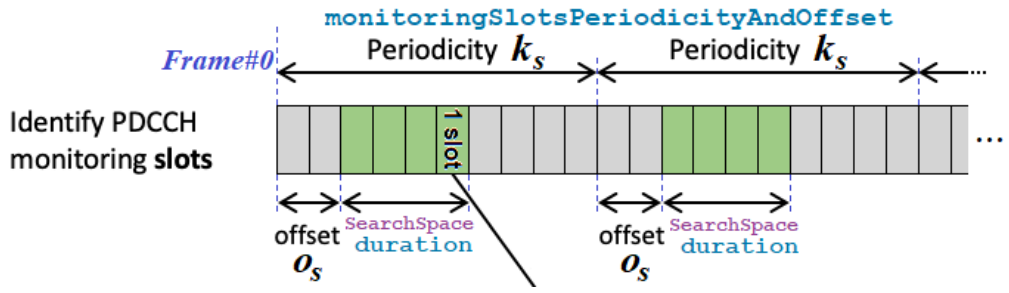
- It is **search space** "5" because "searchSpaceId = 5"
- Its **associated CORESET** is the one with identifier 6, because:

  controlResourceSetId = 6

  Its **associated CORESET** configures this **search space** with:

  - Its **frequency resources** with **parameter** frequencyDomainResources from ControlResourceSet (click 6.12.6). Its **frequency resources** are the **set** of **groups** of **6 consecutive RBs within which each** and **every** of its **PDCCH transmissions** and **PDCCH candidates** take place.
  - Its **PDCCH duration** with **parameter** duration from ControlResourceSet (click 6.12.6). Its **PDCCH duration** is the **quantity** of **consecutive symbol intervals** of **each** and **every** of its **PDCCH transmissions** (click 6.8) and **PDCCH candidates** (click 6.8.2.3). For this example, we consider that its **PDCCH duration** is 3.
  - The **CORESET** also provides the **search space** with a **few other configuration parameters**.
- Its **monitoring periodicity** is **10 slots** because the **choice** of monitoringSlotPeriodicityAndOffset is sl10. This means that **every 10 slots** there is a **repetition** of the **slots** that contain **monitoring occasions**.
- Within **each** of its **10-slot monitoring periods**, the **first slot** that contains **monitoring occasions** is the **third slot** (**slot index 2**) because "sl10=2".
- Within **each** of its **10-slot monitoring periods**, there are **4 consecutive slots** (3rd, 4th, 5th and 6th) that contain **monitoring occasions** because "duration=4".

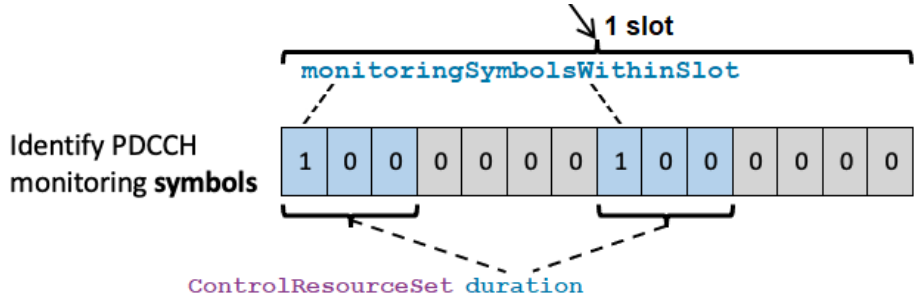
  The following figure illustrates a **monitoring periodicity** of **10 slots**, where the **first slot** that has **monitoring occasions** in **each monitoring period** is the **third slot**, and **4 consecutive slots** containing **monitoring occasions** within **each monitoring period**:



- **Within each slot** that contains **monitoring occasions**, the **monitoring occasions** are **located** in the **1st** and **8th symbol intervals** of the **slot**, because:

  `monitoringSymbolsWithinSlot` = `10000001000000`

  The following figure illustrates, for this **search space**, the **slot symbol intervals** where the **monitoring occasions are** located, and the **slot symbol intervals** where the **PDCCH candidates** associated to this **search space are** located in **time**:




- It does **not contain PDCCH candidates** with **aggregation levels 2**, **3** or **4** because the value of `aggregationLevel1`, of `aggregationLevel2` and of `aggregationLevel4` is `n0`.
- The **aggregation level 8** contains **6 PDCCH candidates** because the value of `aggregationLevel8` is `n6`.
- The **aggregation level 16** contains **8 PDCCH candidates** because the value of `aggregationLevel16` is `n8`.
- It is a **UE-specific search space** because the **choice** of `searchSpaceType` is `ueSpecfic`.
- The **DCI message contained** in **each** of its **PDCCH transmissions** can **only** have **DCI Format 0_0** or **1_0** because the value of `dciFormats` is `formats0-0-And-1-0`.

**Example B of search space configuration**

As a **second** example, the **instance** of `SearchSpace` (click 6.12.8) that configures a **search space** contains the following **parameter** values:

```
searchSpaceId = 11,
controlResourceSetId = 6
monitoringSlotPeriodicityAndOffset => sl16 = 1
duration = 2
monitoringSymbolsWithinSlot = 10000001000100
nrofCandidates =>
  aggregationlevel1  = n0
  aggregationlevel2  = n0
  aggregationlevel4  = n4
  aggregationlevel8  = n6
  aggregationlevel16 = n8
searchSpaceType =>
  ueSpecific =>
    dciFormats = formats0-1-And-1-1
```

This implies that it is a **UE-specific search space**, with identifier `11` and its **associated CORESET** is "`6`" (i.e., the same **CORESET** as the **search space** from the previous example). It can **only** use **aggregation levels 4**, **8** and **16**. For aggregation level **4**, it

contains **4 PDCCH candidates**. For aggregation level **8**, it contains **6 PDCCH candidates**. For aggregation level **16**, it contains **6 PDCCH candidates**. The **DCI message contained** in **each** of its **PDCCH transmissions** can **only** have **DCI Format 0_1** or **1_1**.

It contains **PDCCH monitoring occasions** every **period** of **16 slots** (`sl16`). Within **each** such **period** of **16 slots**, the **first slot** that contains **monitoring occasions** is the **second slot** (`sl16=1`), and there is a total of **2 consecutive slots** (`duration=2`) with **monitoring occasions**. The following figure illustrates this **periodicity** and the **slots** containing **monitoring occasions** within **each period**:

**PDCCH monitoring slots**

| $n_f$ | $n_{slot}$ within a frame | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 0 | | 1 | 2 | | | | | | | |
| 1 | | | | | | | | 7 | 8 | |
| 2 | | | | | | | | | | |
| 3 | | | | 3 | 4 | | | | | |

Some **relevant remarks** on this figure are:

- The symbol "$n_f$" represents the **system frame number**.
- **Subcarrier spacing** is **15 kHz**, and therefore, **each subframe** contains **1 slot**. Consequently, **each frame** contains **10 slots**.

**Within each slot** that contains **monitoring occasions**, the **monitoring occasions** start in the **1st**, **8th** and **12th symbol intervals** of the **slot**, as indicated in the **bitstring** of `monitoringSymbolsWithinSlot`. The **duration** of **each PDCCH candidate** is defined (in **symbol intervals**) in the **CORESET associated** to **this search space**, as is described in 6.8.1. For this example, we will assume that the **duration** of **each PDCCH monitoring occasion** is `3` **symbol intervals**. The following figure illustrates the **symbol intervals** (colored in blue) that contain **PDCCH candidates**, and that **may** contain **PDDCH transmissions**:

**PDCCH candidates**

| $i_{symbol}$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PDCCH transm. | | ? | | No | No | No | No | | ? | | No | | ? | |

## 6.8.6 PDCCH common search space Type 0

### 6.8.6.1 CSS-Type 0 configuration

**Common search space** (**CSS**) **Type 0** may be configured with **either** of **two** same-name parameters `searchSpaceZero`. **Each** of these **two** same-name parameters is **enclosed** in **each** of the following **data types**:

- `PDCCH-ConfigSIB1` (click 8.12.3.1), within `MIB` (click 5.3.3).

  The `searchSpaceZero` parameter enclosed in this data type **must** be used for the **Downlink initial bandwidth part** of the **UE**'s **primary cell** during **initial access** (click 8).

- `PDCCH-ConfigCommon` (click 6.12.4), within `SIB1` (click 5.3.4).

  The `searchSpaceZero` parameter enclosed in this data type **can** be used for **any Downlink initial bandwidth part** of **any UE cell type**.

These **two same-name** parameters have the same **data type**, `SearchSpaceZero` (click 8.12.3), which is an `INTEGER (0..15)` and it is **represented** with **4 bits**.

**Search space zero** may also be configured with an instance of `SearchSpace` (click 6.12.8), as it is done for **non-zero search spaces** (click 6.8.5).

### 6.8.6.2 CSS-Type 0 monitoring occasions

The **UE** gets the **periodicity** "$k_s$" of the **monitoring occasions** of **search space** (click 6.8.5) **zero** from the **multiplexing Pattern** (click 6.8.8.4), which is indicated by `controlResourceSetZero` (click 6.8.8.2), as follows:

- Multiplexing **Pattern 1** (click 6.8.8.4.1): **periodicity** "$k_s$=**2**" **frames**.
- Multiplexing **Pattern 2** (click 6.8.8.4.2) and **Pattern 3** (click 6.8.8.4.3): **periodicity** "$k_s$" is the **same** as the periodicity of **SSB bursts**.

The **UE** gets the specific **slot index** and **symbol index** for the **PDCCH monitoring occasion**(**s**) (click 6.8.3.2) of **search space zero** from the **4 bits** of `searchSpaceZero` (click 6.8.6). The said **4 bits** are an **index** for the **monitoring occasion tables 13-1** to **13-15A**, of TS 38.213.

The specific **time location table** to use, and **how to use it**, depends on:

- The multiplexing **Pattern** (click 6.8.8.1), which is indicated by `controlResourceSetZero` (click 6.8.8.2).
- The **band frequency range** (**FR1**, **FR2-1** or **FR2-2**).
- The **subcarrier spacing** of **SSB** and of **CORESET#0**.
- Whether operation is for **licensed spectrum** or for **shared spectrum**.

The various TS combinations have been organized into the **three Cases**, as it is now described.

- **Case a: multiplexing Pattern 1 and licensed spectrum**

  The **UE** has **2 PDCCH monitoring occasions** within **every other frame**. The **frames** that contain the **2 PDCCH monitoring occasions** are the ones having a **System Frame Number** either **even** or **odd**, matching the result of the following expression:

  $$floor((O \cdot 2^{\mu} + floor(i \cdot M)) / N_{slot}^{frame,\mu})$$

  The **2 PDCCH monitoring occasions** are in **two different slots** within the **frame**. The **frame slot index** of the **second slot** is "$n_0$+**j**" and depends on "**μ**" (the numerology of **CORESET#0**) as indicated below. The **frame slot index** of the **first slot** is "$n_0$" and is computed according to the following expression:

  $$n_0 = (O \cdot 2^{\mu} + floor(i \cdot M)) \bmod N_{slot}^{frame,\mu}$$

  The variables in the two expressions above are:

  - "$i$" is the **burst index** of the **SSB** received in the **beam** that has been **selected** by the **UE** (click 6.1.2.2.3). The said **SSB burst index** is **broadcast** in **ssb-time-domain-index** (click 6.4.1) from **PBCH SI** (click 5.3.2).

- "***O***", "***M***" and the **slot symbol index** of the **first symbol** of the **PDCCH monitoring occasion** are found by inputting the **4 bits** of `searchSpaceZero` (click 6.8.6) in **table 13-11**, **13-12** or **13-12A** . The **specific table** to use depends on the numerology "**μ**" of **CORESET#0**, as follows:
  - For "**μ** ∈{0, 1, 2, 3}", the **frame slot index** of the **second slot** is "$n_0$ **+1**". For **FR1** bands, the **table** to use is **13-11**. For **FR2-1** bands and for **FR2** bands with **120 kHz** subcarrier spacing for both **SSB** and **CORESET#0**, the table to use is **13-12**. For **FR2-2** bands with values of subcarrier spacings other than the ones just indicated, multiplexing **Pattern 1 cannot** be used.
  - For "**μ**=5", the **frame slot index** of the **second slot** is "$n_0$ **+4**". The **table** to use is **13-12A**, with "***X***=1.25".
  - For "**μ**=6", the **frame slot index** of the **second slot** is "$n_0$ **+8**". The **table** to use **is 13-12A**, with "***X***=0.625".

To show one example of these tables, we copy below **table 13-11**, that applies to **multiplexing Pattern 1** and **FR1** bands:

| Index | *O* | Monitoring occasions per slot | *M* | Slot symbol index of PDCCH's first symbol |
|---|---|---|---|---|
| **0** | 0 | 1 | 1 | 0 |
| **1** | 0 | 2 | 0.5 | {0, if ***i*** is even}, {$N_{symb}^{CORESET}$ if ***i*** is odd} |
| **2** | 2 | 1 | 1 | 0 |
| **3** | 2 | 2 | 0.5 | {0, if ***i*** is even}, {$N_{symb}^{CORESET}$, if ***i*** is odd} |
| **4** | 5 | 1 | 1 | 0 |
| **5** | 5 | 2 | 0.5 | {0, if ***i*** is even}, {$N_{symb}^{CORESET}$, if ***i*** is odd} |
| **6** | 7 | 1 | 1 | 0 |
| **7** | 7 | 2 | 0.5 | {0, if ***i*** is even}, {$N_{symb}^{CORESET}$, if ***i*** is odd} |
| **8** | 0 | 1 | 2 | 0 |
| **9** | 5 | 1 | 2 | 0 |
| **10** | 0 | 1 | 1 | 1 |
| **11** | 0 | 1 | 1 | 2 |
| **12** | 2 | 1 | 1 | 1 |
| **13** | 2 | 1 | 1 | 2 |
| **14** | 5 | 1 | 1 | 1 |
| **15** | 5 | 1 | 1 | 2 |

Notice that there is a column labeled "***Quantity of search spaces per slot***", having values **1** or **2**. The value **2** indicates that on some slots there are **two CORESET#0 instances**, which may mislead you to think that the **UE** has to **search two CORESET#0 PDCCH monitoring occasions** in **each** of the slots "$n_0$" and "$n_0$**+j**". This is **not true**, and the **UE only searches one CORESET#0 PDCCH monitoring occasion** in **each** of the slots "$n_0$" and "$n_0$**+j**". The meaning of the value **2** is that some slots do contain **two CORESET#0 instances**, but they correspond to **two different values** of the **PBCH SI parameter** `ssb-time-domain-index` "***i***", and therefore, the **two** corresponding **CORESET#0 PDCCH monitoring occasion instances** are **not** searched by the same **UE**. When the value of "***M***" is **0.5**, the result of the expression

that computes "$n_0$" (see above) is **the same** for **two different values** of the **PBCH SI parameter** ssb-time-domain-index "***i***". Therefore, **two different values** of the **PBCH SI parameter** ssb-time-domain-index "***i***" result in the **same value** of the slot index "$n_0$". In order to be able to transmit a different **CORESET#0 instance** for each value of "***i***", the **NR node** transmits **two different CORESET#0 instances** in the **same slot**, but in **different slot symbols**. This is why in the rows where the quantity of search spaces per slot is **2**, the column indicating the **slot symbol index** of the **CORESET#0 instances** contains **two values**, that depend on the value of "***i***". Therefore, **each UE** uses **only one** of the two values to determine its **single CORESET#0 PDCCH monitoring occasion**.

- **Case b: multiplexing Pattern 2 or 3 and licensed spectrum**

  The **UE** searches **one CORESET#0 monitoring occasion**, located in **one slot**, in the **same frame** as the **SSB**. The periodicity of the **SSB burst** and of **CORESET#0** are therefore the same. If the **UE** has selected **SSB** with index "***i***" (contained in **PBCH system information** parameter ssb-time-domain-index), the **UE** determines the **frame slot index** "$n_c$" and the **slot symbol index** of the **first symbol** of the **CORESET#0 PDCCH monitoring occasion** based on tables **13-13** through **13-15A**.

  In these tables, the meaning of the different symbols is as follows (summarized from one long and confusing sentence from the first paragraph of clause 13 of TS 38.213):

  - "***SFNc***" is the **system frame number** where **CORESET#0** is located.
  - "$n_c$" is the **frame slot index** of the slot where **CORESET#0** is located. The slot index within the frame is based on the subcarrier spacing of **CORESET#0**.
  - "$SFN_{SSB,i}$" is the **system frame number** of the **frame** that overlaps in time with the **SSB** with index "***i***".
  - "$n_{SSB,i}$" is the **frame slot index** of the **slot** that overlaps in time with the **SSB** with index "***i***". The frame slot index is based on the subcarrier spacing of **CORESET#0**.
  - Notice that the term "**overlaps**" is due to the possible case where the **subcarrier spacing** of **SSB** is **different** to the one of **CORESET#0**.

  To show one example on how to use these tables, we copy below **table 13-13**. This table applies to **multiplexing Pattern 2**, **SSB SCS = 120 kHz** and **CORESET#0 SCS = 60 kHz**:

| Index | PDCCH monitoring occasions (SFN and slot number) | Slot symbol index of CORESET#0 first symbol ($k$ = 0, 1, …, 15) |
|---|---|---|
| **0** | $SFN_c = SFN_{SSB,i}$<br>$n_c = SFN_{SSB,i}$ | **0**, **1**, **6**, **7** *in*<br>***i*** = *4k*, ***i*** = *4k+1*, ***i*** = *4k+2*, ***i*** = *4k+3* |
| **1 to 15** | Reserved | |

  This table implies that **CORESET#0** is in the **same slot** as the **SSB** and that the **slot symbol index** of the **first symbol** of **CORESET#0** is as follows:

  ***i mod*** 4 = 0 => slot index of first symbol = **0**
  ***i mod*** 4 = 1 => slot index of first symbol = **1**
  ***i mod*** 4 = 2 => slot index of first symbol = **6**

***i mod*** 4 = 3 => slot index of first symbol = **7**

As another example to show how to use these tables, we copy below **table 13-14**. This table applies to **multiplexing pattern 2**, **SSB SCS = 240 kHz** and **CORESET#0 SCS = 120 kHz**:

| Index | PDCCH monitoring occasions (SFN and slot number) | Slot symbol index of CORESET#0 first symbol ($k$ = 0, 1, …, 7) |
|---|---|---|
| 0 | $SFN_c = SFN_{SSB,i}$<br>$n_c = n_{SSB,i}$ or<br>$n_c = n_{SSB,i}$ -1 | **0**, **1**, **2**, **3**, **0**, **1** *in* $i = 8k$, $i = 8k+1$, $i = 8k+2$, $i = 8k+3$, $i = 8k+6$, $i = 8k+7$ ($n_c = n_{SSB,i}$)<br>**12**, **13** in $i = 8k+4$ and $i=8k+5$ ($n_c = n_{SSB,i} - 1$) |
| 1 to 15 | Reserved | |

This table means that the **location** of **CORESET#0** depends on the value of the **ssb-time-domain-index** "***i***", as follows:

- In all the following cases, **CORESET#0** is in the **same slot** as the **SSB** and the **slot symbol index** of the **first symbol** of **CORESET#0** is as indicated:

  ***i mod*** 8 = 0 or 6 => slot index of first symbol = **0**
  ***i mod*** 8 = 1 or 7 => slot index of first symbol = **1**
  ***i mod*** 8 = 2 => slot index of first symbol = **2**
  ***i mod*** 8 = 3 => slot index of first symbol = **3**

- In all the following cases, **CORESET#0** is in the **previous slot** of the **SSB** and the **slot symbol index** of the **first symbol** of **CORESET#0** is as indicated:

  ***i mod*** 8 = 4 => slot index of first symbol = **12**
  ***i mod*** 8 = 5 => slot index of first symbol = **13**

Notice that the two tables in the last two examples correspond to **multiplexing Pattern 2**, which can **only** be used in **FR2 bands**, and therefore, the **ssb-time-domain-index** "***i***" ranges from **0** to **63** (click 6.4.1).

- **Case c: for shared spectrum**

  This case is somehow complex. You may refer to its two specific sub-cases in clause 13 of TS 38.213.

As we have seen in the explanation above, the **UE** only explores the **PDCCH monitoring occasions** for **CORESET#0** corresponding to the **ssb-time-domain-index** "***i***" of the beam that the **UE** has selected. However, it is very important to highlight that the **NR node** should transmit **all** the **CORESET#0 instances**, corresponding to **all** the **SSBs** that were **transmitted** in the **SSB burst**. This is, the **NR node** should transmit **all** the **CORESET#0 instances** corresponding to **all** the values of the **ssb-time-domain-index** "***i***" that the NR node has **transmitted**.

As indicated in clause 10.1 of TS 38.213, for the case of a **CSS-Type 0** where `searchSpaceSIB1` from `PDCCH-ConfigCommon` (click 6.12.4) is **absent**, the

**aggregation levels**, and **each** corresponding **quantity** of **PDCCH candidates** (click 6.8.2.3), are stated in table 10.1-1:

| CCE aggregation level | Quantity of PDCCH candidates |
|---|---|
| **4** | 4 |
| **8** | 2 |
| **16** | 1 |

As it has been shown, multiplexing **Pattern 1** only allows for **2 PDCCH monitoring occasions** for **CORESET#0** every **2 frames** (i.e., every **20 ms**). This is **not** very suitable when the **NR node** uses **Tx beam sweeping** (click 6.1.2.3), where an **SSB burst** may contain up to **64 SSBs**, **each** sent in its **specific beam direction**. Therefore, multiplexing **Patters 2** and **3** were introduced for **FR2 bands**, both of which allow for **up to 64 CORESET#0 instances** being sent, **each** in its specific beam direction. Notice that the information sent in PDCCH **CORESET#0** in **all** the **PDCCH monitoring occasions** is the same (i.e., a **DCI message** to locate `SIB1` in **PDSCH**), but sending one for **each beam direction** maximizes the possibility of correct reception by **UEs** located in **different beam directions**.

For more detail on **search space zero**, refer to clause 13 of TS 38.213.

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_CommonSearchSpace_Type0_PDCCH.html
- https://www.linkedin.com/pulse/5g-nr-controlresourcesetzero-corseset0-search-space-zero-chelikani/

## 6.8.7 PDCCH COntrol REsource SETs (CORESETs)

**Each** of the **CORESETs** configured to **each bandwidth part** of **each UE** defines **1 instance** of **frequency resources** (click 6.8.2.1) and the **time duration** (click 6.8.3.1) for the **UE**'s **PDCCH candidates** (click 6.8.2.3). **Each CORESET** also defines the **type** of **mapping** (click 6.8.2.2) from the **RBs** of its **CCEs** (click 6.8.1) to the **actually transmitted PRBs** of the **bandwidth part**.

**Each CORESET** may be **associated** to **several search spaces** (click 6.8.5). However, **each search space** has **1**, and **only 1**, **associated CORESET**.

**Each CORESET** has a **unique identifier** (click 6.8.7.1) that allows to **reference** it from the **search spaces**. There are different types (click 6.8.7.2) of CORESETs.

The **UE**'s **configuration** of **non-zero CORESETs** is presented in 6.8.7.3.

The **characteristics** and **UE**'s **configuration** of **CORESET#0** are presented in 6.8.8.

Neighboring **NR nodes** usually coordinate their **CORESET frequency locations** to **minimize** the mutual interference caused to their **Downlink control information**.

For more detail on **CORESETs**, refer to clause 7.3.2.2 of TS 38.211, clauses 9 and 10 of TS 38.213 and clauses 5.1 and 6.1 of TS 38.214.

Source web pages:

- https://www.techplayon.com/5g-nr-coreset-control-resource-set/

- https://www.sharetechnote.com/html/5G/5G_ResourceAllocationUnit.html
- https://www.nrexplained.com/pdcch_assign
- https://es.mathworks.com/videos/5g-explained-coreset-and-pdcch-in-5g-nr-1559890449525.html
- https://www.youtube.com/watch?v=v6lS-48isyc
- https://www.telecomhall.net/t/how-coreset-is-configured-in-5g-nr/20120/4

#### 6.8.7.1 CORESET identification

**Each CORESET** is **identified** with an **instance** of **either** data type `ControlResourceSetId`, `ControlResourceSetId-r16` or `ControlResourceSetId-v1610` (click 6.12.7). **Each** of the said **data types** is an `INTEGER` with a certain **range**.

This **CORESET identifier uniquely** identifies **each** of the **UE**'s **CORESET** within **each** of the **UE**'s **serving cells**.

#### 6.8.7.2 CORESET types

**Each** of the **UE**'s **CORESETs** may be **common or dedicated**:

- **Common CORESETs**

  **Each** of them has the **same configuration** (except the identifier) for **all** the **UEs** in **each serving cell**. They can be **CORESET#0** or **non-zero common CORESETs**.

- **Dedicated CORESETs**

  **Each** of them is **specific** for a **UE**, although **several UEs** may **share some/all** of their **dedicated CORESETs**. They are always **non-zero CORESETs**.

**Each** of the **UE**'s **CORESETs** may be **CORESET#0** or a **non-zero CORESET**:

- **CORESET#0**

  It is a **common CORESET** that **always** has **identifier** `0` and it has several **special** characteristics (click 6.8.7).

- **Non-zero CORESETs**

  They are the **CORESETs** with identifiers other than `0`. They can be **common** or **dedicated**.

#### 6.8.7.3 CORESET UE configuration and quantity

**All** the **Downlink bandwidth parts** in **each** of the **UE**'s **cells** are configured with a **joint total** of **up to** `12` or `16` **CORESETs** (including **CORESET#0**). These two **maximum quantities** derive from the **maximum quantity** of **CORESET identifiers**. The **CORESET identifier** is **either** `controlResourceSetId` or `controlResourceSetId-v1610`, **both** from `ControlResourceSet` (click 6.12.6).

Which of the **two values** (`12` or `16`) is the **maximum quantity** depends on `coresetPoolIndex-r16` from `ControlResourceSet` (click 6.12.6), as follows:

- ***If*** `coresetPoolIndex-r16` is **absent**, **or** if it **is present** but its value is the **same** for **all** the **UE**'s **CORESETs**, then:

  The **maximum quantity** of **CORESETs** configured in **all** the **Downlink bandwidth parts** of **each UE**'s **cell** is `12`, because in this case `controlResourceSetId` is the **CORESET identifier**, and its range is from `0` to `maxNrofControlResourceSets-1`, whose value is `11`.

- ***Else***:

  The **maximum quantity** of **CORESETs** configured in **all** the **Downlink bandwidth parts** of **each UE**'s **cell** is `16`, because in this case `controlResourceSetId-v1610` is the **CORESET identifier**, and its range is from `0` to `maxNrofControlResourceSets-r16-1`, whose value is `15`.

**Each** of the **UE**'s **Downlink bandwidth parts** is configured with the following **CORESET types** and **quantities**:

- **Up to 1 CORESET#0**

  It **must** be configured in the **Downlink initial bandwidth part** (**BWP#0**, click 2.2.7) of the **UE**'s **primary cell**. It **may be** configured in the **Downlink initial bandwidth part** of **other UE**'s **cells**.

  Although it **may only** be **configured** in **Downlink initial bandwidth parts**, it can be **used** in **Downlink additional bandwidth parts**, **if** the conditions defined in clause 10 of TS 38.213 are satisfied.

  **Each CORESET#0** is configured with `controlResourceSetZero` as described in 6.8.8.2.

- **Up to 1 non-zero common CORESET**

  **Each non-zero common CORESET** is configured with `commonControlResourceSet` from `PDCCH-ConfigCommon` (click 6.12.4) or with `commonControlResourceSetExt-r17` from `CFR-ConfigMCCH-MTCH-r17`. The data type of **either** parameter is `ControlResourceSet` (click 6.12.6).

- **Up to `3` or `5` (non-zero) dedicated CORESETs**

  The **maximum quantity** being `3` or `5` depends (same as the **maximum quantity** of **CORESETs** in the **pool**, see above) on `coresetPoolIndex-r16` from `ControlResourceSet` (click 6.12.6), as follows:

  - If `coresetPoolIndex-r16` is **absent**, **or** if it is **present** but its value is the **same** for **all** the **UE**'s **CORESETs**, then it is **up to `3` dedicated CORESETs**. These **up to `3` dedicated CORESETs** are configured with `controlResourceSetToAddModList` from `PDCCH-Config` (click 6.12.5). Its **data type** is a `SEQUENCE(SIZE (1..3)) OF ControlResourceSet` (click 6.12.6).
  - Else, it is **up to `5` dedicated CORESETs**. The **up to `2`** dedicated **CORESETs** (in **addition** to the `3` ones from the previous bullet) are configured with `controlResourceSetToAddModListSizeExt-v1610` also from

`PDCCH-Config` (click 6.12.5). Its **data type** is a `SEQUENCE (SIZE (1..2)) OF ControlResourceSet` (click 6.12.6).

Notice that **each dedicated CORESET** is configured with an instance of `ControlResourceSet` (click 6.12.6).

For more detail on **PDCCH parameters**, click 6.12.

For more detail on `coresetPoolIndex-r16` refer to clause 10.1 from TS 38.213 (in particular, page 153).

Notice that the **maximum quantity** of **physical layer elements** (**cells**, **bandwidth parts**, **CORESETs**, **search spaces**, etc) that can be **configured** to **each UE** is **not** the same as the **maximum quantity** that a **NR node** can handle.

## 6.8.8 PDCCH CORESET#0

For the **general** characteristics of **CORESETs**, that also apply to **CORESET#0**, click 6.8.3.1.

### 6.8.8.1 CORESET#0 subcarrier spacing

For **FR1** and **FR2-1** bands, the **UE** gets the **subcarrier spacing** of **CORESET#0** out of `subCarrierSpacingCommon` (1 bit) from `MIB` (click 8.12.3) while for **FR2-2** bands, it is the **same** as the **SSB subcarrier spacing** (click 6.1.4). Parameter `subCarrierSpacingCommon` indicates, in **FR1** bands, either **15 kHz** or **30 kHz**, while in **FR2-1** bands it indicates either **60 kHz** or **120 kHz**.

The **CORESET#0 subcarrier spacing** must be one of the **control/shared subcarrier spacings** configured in the **channel** (click 2.2.3.2). This is further **restricted** by the **allowed multiplexing patterns** (click 6.8.8.4).

**Not all** the **combinations** between **SSB subcarrier spacing**, **CORESET#0** (**PDCCH**) **subcarrier spacing** and **control/shared subcarrier spacing** are allowed. For the **allowed combinations**, click 2.1.5.

### 6.8.8.2 CORESET#0 configuration

**CORESET#0** may be configured with **either** of **two same-name** parameters `controlResourceSetZero` **from** the **data types**:

- `PDCCH-ConfigSIB1` (click 8.12.3.1), within `MIB` (click 8.12.3).

  The `controlResourceSetZero` parameter within `MIB` **must** be used for the **Downlink initial bandwidth part** of the **UE**'s **primary cell** during **initial access**.

- `PDCCH-ConfigCommon` (click 6.12.4), within `SIB1` (click 5.3.4).

  The `controlResourceSetZero` parameter within `SIB1` **may** be used for **any Downlink initial bandwidth part** of **any UE cell type**.

These **2 same-name** parameters have the same **data type**, `ControlResourceSetZero` (click 8.12.3), which is an `INTEGER (0..15)` and it is **represented** with **4 bits**.

For the case of **CORESET#0** configured by `controlResourceSetZero` from `PDCCH-ConfigCommon` (click 6.12.4), the **UE** may assume **interleaved CCE-to-PRB mapping** (click 6.8.3.1) with the following configuration:

- **Physical REG bundle size** "***L***" is **2 Physical REGs**.

- Sequence generation parameters "$R = 2$" and "$n_{shift} = N_{ID}^{cell}$" (where "$N_{ID}^{cell}$" is the **Physical Layer Identity**, obtained from **PSS/SSS** in 8.2).
- The **same precoding** being used within a **REG bundle**.

For more detail on this last configuration, refer to clause 7.3.2.2 of TS 38.213.

### 6.8.8.3 CORESET#0 frequency scheduling and PDCCH candidate duration

The **UE** determines the **frequency scheduling**, **PDCCH candidate duration** and other characteristics of **CORESET#0** from the configuration parameters indicated in 6.8.8.2, that provide **4 bits**. These **4 bits** are an **index** for tables **13-1** to **13-10A**, of TS 38.213, that provide the **frequency location** and the **PDCCH candidate duration**. These **tables** also indicate the **multiplexing Pattern** used for **SSB**, **CORESET#0** (**PDCCH**) and **PDSCH** for `SIB1`. The **frequency location** of **CORESET#0** is provided **relative** to the **location** of **SSB**.

The specific **table** to use depends on:

- Whether the band is for **licensed spectrum** or for **shared spectrum**.
- The **minimum channel bandwidth** in the band.
- The **subcarrier spacing** of **SBB** and of **CORESET#0** (**PDCCH**) .

Each entry of the **frequency location** and **size tables** provides the following information:

- **Multiplexing Pattern**

  It indicates the **multiplexing Pattern** of **CORESET#0**. Possible values are **1**, **2** or **3**. For additional information click 6.8.8.1

- **Quantity of RBs**

  It indicates the **frequency size** of **CORESET#0** in **Resource Blocks**. Possible values for are **24**, **48** or **96 RBs**, except for **Pattern 3** that can **only** have **24** or **48 RBs**. The **subcarrier spacing** of these **RBs** is the one of **CORESET#0** (click 6.8.8.1).

- **Duration of PDCCH candidates**

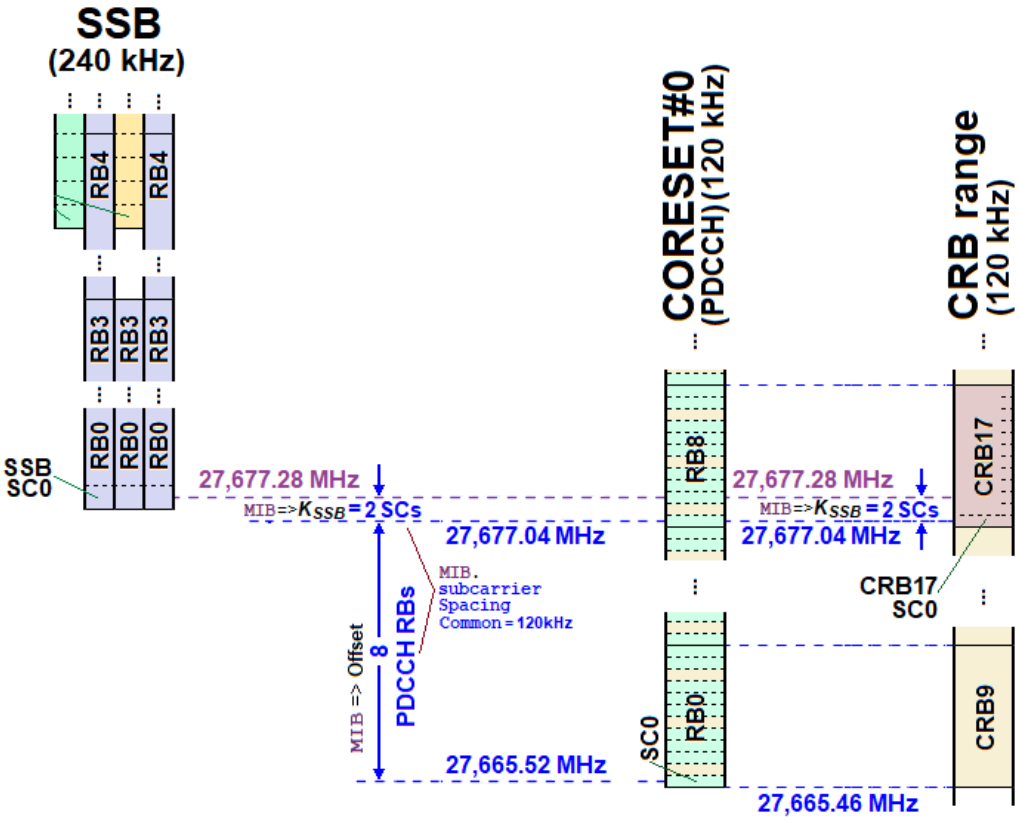
  It indicates the **time duration** of **PDCCH candidates** of **CORESET#0** in **symbol intervals**. Possible values are **1**, **2** or **3 symbol intervals**. Recall that the **time duration** of **each CORESET#0 symbol interval** depends on the **CORESET#0 subcarrier spacing**.

The **tables 13-1** through **13-10A** contain **all** the combinations between the quantity of **CORESET#0 RBs** (**24**, **48** or **96**) and the quantity of **symbol intervals** (**1**, **2** or **3**) of the **PDCCH candidates** of **CORESET#0**

- **Offset**

  For **licensed spectrum** and for **shared spectrum** in **FR2-2** bands, **Offset** is the quantity of **RBs** (with the subcarrier spacing used to calculate `offsetToPointA` (click 8.7.3)) between the **center frequency** of **SC0** of **CORESET#0** and the **center frequency** of **SC0** of the **CRB** (with the **subcarrier spacing** used to calculate `offsetToPointA`) that **overlaps** with **SSB**. Parameter `offsetToPointA` is from `FrequencyInfoDL-SIB` (click 2.2.6.6).

The following figure shows the **frequency location** of **CORESET#0** in respect to the **SSB**, indicating what is **Offset** and what is "$k_{SSB}$" (click 8.7.2):




The value of "$k_{SSB}$" is obtained from `ssb-SubcarrierOffset` (click 8.7.1) from `MIB` (click 8.12.3).

Notice that for **SSB** and **CORESET#0 multiplexing patterns 2** and **3** the value of **Offset** is **negative**, because the **SC0** of **CORESET#0** is **above** the **SC0** of **SSB**. This **negative Offset** has an **absolute** value **similar** to the bandwidth of the **SSB**, because **CORESET#0** is placed in the **first CRB** right above the **SSB**.

For **shared spectrum** in **FR1** or in **FR2-1** bands, the meaning of **Offset** is more complex: refer to clause 13 of TS 38.213.

To show one example of a table of **frequency location** and **PDCCH candidate duration**, we copy below **table 13-1**, that applies to **frequency bands** with **minimum channel** bandwidths of **5 MHz** or **10 MHz**, **licensed spectrum**, and a subcarrier spacing of **15 kHz** for **SSB** and of **15 kHz** for **CORESET#0**:

| Index | SSB and CORESET multiplexing pattern | Frequency resources ($N_{RB}^{CORESET}$) [RBs] | Duration ($N_{symb}^{CORESET}$) [sym. intvs.] | Offset [RBs] |
|---|---|---|---|---|
| **0** | 1 | 24 | 2 | 0 |
| **1** | 1 | 24 | 2 | 2 |
| **2** | 1 | 24 | 2 | 4 |
| **3** | 1 | 24 | 3 | 0 |
| **4** | 1 | 24 | 3 | 2 |
| **5** | 1 | 24 | 3 | 4 |
| **6** | 1 | 48 | 1 | 12 |

| | | | | |
|---|---|---|---|---|
| **7** | 1 | 48 | 1 | 16 |
| **8** | 1 | 48 | 2 | 12 |
| **9** | 1 | 48 | 2 | 16 |
| **10** | 1 | 48 | 3 | 12 |
| **11** | 1 | 48 | 3 | 16 |
| **12** | 1 | 96 | 1 | 38 |
| **13** | 1 | 96 | 2 | 38 |
| **14** | 1 | 96 | 3 | 38 |
| **15** | Reserved | | | |

Notice that the **smallest CORESET#0 bandwidth** is 24•12•15 kHz = **4.32 MHz**, that **fits well** within the **smallest** channel bandwidth (5 MHz) to which this table applies.

**CORESET#0** and **PDSCH** associated with `pdcch-ConfigSIB1` within `MIB` (click 5.3.3) or with `searchSpaceSIB1` from `PDCCH-ConfigCommon` (click 6.12.4) use **normal cyclic prefix** (as indicated in clause 7.3.2.2 of TS 38.211).

For more detail on **CORESET#0 frequency location** and **size**, refer to clause 13 of TS 38.213.

### 6.8.8.4 CORESET#0 multiplexing Patterns

There are **3** different **time** and **frequency locations** of **CORESET#0** and **PDSCH**, **relative** to **SSB**, for the purpose of **getting** `SIB1`. These **3** different **locations** (relative to **SSB**) are called **multiplexing Patterns 1**, **2** and **3**.

**FR1** bands **only** use **multiplexing Pattern 1**. **FR2** bands usually use **multiplexing Patterns 2** or **3**, although they can also use **multiplexing Pattern 1**.

For the **3 multiplexing Patterns**, the **frequency-domain resource allocation** used in the **DCI message 1_0** is **DFA-Type 1** (start-and-length). Also, the **frequency location** of **SIB** within **PDSCH**, indicated in **DCI message 1_0**, is always within the **PRBs** of **CORESET#0**, and it is referred to the said **PRBs**. For more information, click 6.14.3.3.

#### 6.8.8.4.1 Multiplexing Pattern 1

**Multiplexing Pattern 1 must** be used in **FR1** bands. It can be used in **FR2** bands, but this is unusual.

**Multiplexing Pattern 1** uses the **frequency location** and **PDCCH duration tables 13-1** through **13-10A** (all of them) of TS 38.213. From these tables, we can see:

- The **subcarrier spacing** used in **SSB** may be larger than, equal to, or smaller than the one used in **CORESET#0** (which is indicated in `subcarrierSpaceCommon` from `MIB` (click 8.12.3)).
- There is an **overlap** between the **frequency locations** of **SSB** and of **CORESET#0**.
- There is **no overlap** between the **time locations** of **SSB** and of **PDCCH** (of **CORESET#0**).
- The **CORESET#0** bandwidth may be **24**, **48** or **96 CORESET#0 contiguous** Resource Blocks.
- The **duration** of **PDCCH** (of **CORESET#0**) is from **1** to **3 symbol intervals**.

- Therefore, the **size** of **CORESET#0** may be **24**, **48**, **72**, **96**, **144**, **192** or **288 REGs**, which is **4**, **8**, **12**, **16**, **24**, **32** or **48 CCEs**.

**Multiplexing Pattern 1** uses the **PDCCH monitoring occasion tables 13-11**, **13-12** and **13-12A** of TS 38.213. From these tables, we can see:

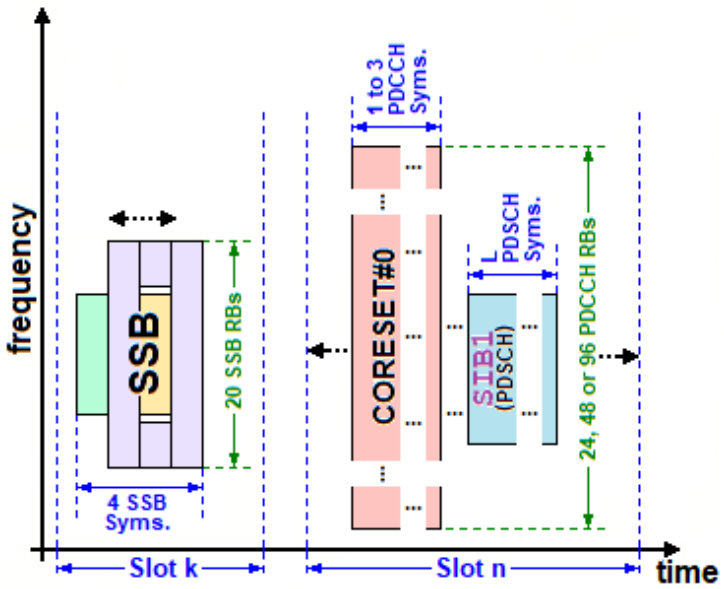
- Each **UE** has **2 monitoring occasions** in every **other frame**. These **2 monitoring occasions** are placed in **two** different **slots** of the frame. Notice that different **UEs** may have different time locations for their **monitoring occasions**, depending on the **PBCH SI parameter** ssb-time-domain-index "***i***" (click 6.4.1)
- The **NR node** may transmit **up to 64 pairs** of **DCI 1_0 messages**, **1 pair** for **each** of the possible values of the **PBCH SI parameter** ssb-time-domain-index "***i***" (click 6.4.1).
- There **must** be **no overlap** between **time locations** of **SSB** and of **PDCCH** (of **CORESET#0**), because there is an **overlap** between the **frequency locations** of **SSB** and of **CORESET#0**.

For more detail on the **PDCCH monitoring occasions** in **search space zero**, click 6.8.6.2.

**Multiplexing Pattern 1** uses "PDSCH time-domain resource allocation" **Default A** with **normal cyclic prefix**, corresponding to table **5.1.2.1.1-2** of TS 38.214. From this table and from tables **13-11** to **13-12A** indicated above, we can see:

- **DCI message 1_0** (within **PDCCH** of **CORESET#0**) and **SSB** are **usually** on **different slots** (actually they may have **different periodicity**), although it is also **possible** to have them in the **same slot**.
- **DCI message 1_0** (within **PDCCH** of **CORESET#0**) and SIB1 (within **PDSCH**) are **always** in the **same slot** because "$K_0$" is always **0** in table **5.1.2.1.1-2**. For more information click 6.14.4.

The following figure shows multiplexing **Pattern 1**.



This figure shows the **frequency location** in a reasonably correct way, but the **actual time location** of **SSB** and of **PDCCH** (of **CORESET#0**) may vary, **independently**, left or right, as indicated by the dotted black arrows.

**Multiplexing Pattern 1** is sometimes said to be "time multiplexing".

#### 6.8.8.4.2 Multiplexing Pattern 2

**Multiplexing Pattern 2** can only be used in **FR2** bands.

**Multiplexing Pattern 2** uses the **frequency location** and **PDCCH duration tables 13-7** and **13-10** of TS 38.213. From these tables, we can see:

- The **subcarrier spacing** used in **SSB** is **twice** the one used in **CORESET#0** (which is indicated in `subcarrierSpaceCommon` from `MIB` (click 8.12.3)). Therefore, the symbol duration in **SSB** is **half** the one of **CORESET#0**.
- There is **no overlap** between the **frequency locations** of **SSB** and of **CORESET#0**.
- There **may** be an **overlap** between the **time locations** of **SSB** and of **PDCCH** (of **CORESET#0**).
- **CORESET#0 starts** in the **first CRB above** the location of **SSB** and extends upwards in frequency. This **first CRB** belongs to the **CRB range** with the **same subcarrier spacing** as **CORESET#0**.
- The **frequency separation** ("**D**" in the figure below) between **CORESET#0** and **SSB** is **0** (i.e., they are contiguous) when "$k_{SSB}$" (click 8.7.2) is **0**. Otherwise, it is the bandwidth of **one CORESET#0 RB** minus the bandwidth indicated by "$k_{SSB}$".
- The **CORESET#0** bandwidth may be **24**, **48** or **96 CORESET#0 contiguous** Resource Blocks.
- The **duration** of **PDCCH** (of **CORESET#0**) is **1 symbol interval**.
- Therefore, the size of **CORESET#0** may be **24**, **48** or **96 REGs**, which is **4**, **8** or **12 CCEs**.

**Multiplexing Pattern 2** uses the **PDCCH monitoring occasion tables 13-13** and **13-14** of TS 38.213. From these tables, we can see:

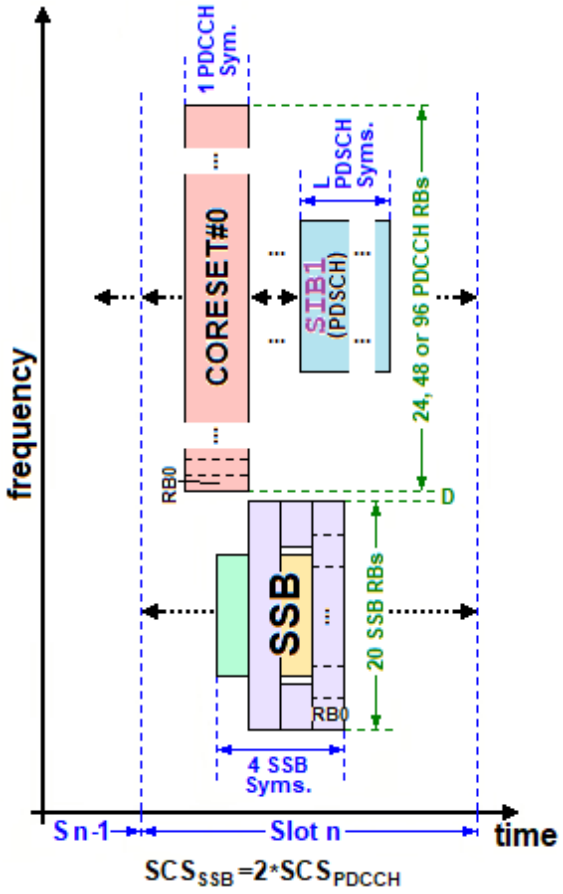
- **Each UE** has **1 monitoring occasion** for **each SSB burst**. Depending on the case, **each monitoring occasion** may be placed in the **same slot**, **or** in the **previous slot**, as the **SSB** of the **beam** selected by the **UE**.
- The **NR node** may transmit **up to 8 DCI 1_0 messages** for **each SSB burst**.
- There **may** be an **overlap** between the **time locations** of **SSB** and of **PDCCH** (of **CORESET#0**) because there is **no overlap** between the **frequency locations** of **SSB** and of **CORESET#0**.

For more detail on the **PDCCH monitoring occasions** in **search space zero**, click 6.8.6.2.

**Multiplexing Pattern 2** uses "PDSCH time-domain resource allocation" **Default B**, corresponding to table **5.1.2.1.1-4** of TS 38.214. From this table and from tables **13-13** and **13-14** mentioned above, we can see:

- **DCI message 1_0** (within **PDCCH** of **CORESET#0**) may **either** be in the **same slot** as the **SSB**, or in the **previous slot** because in tables **13-13** and **13-14** the value of $SFN_c$ and $SFN_{SSB,i}$ are always equal, but "$n_c$" may be equal to "$n_{SSB,i}$" or to "$n_{SSB,i}$-1".
- **DCI message 1_0** (within **PDCCH** of **CORESET#0**) and `SIB1` (within **PDSCH**) may **either** be in the **same slot**, or in **consecutive slots**, because "$K_0$" may be **0** or **1** in table **5.1.2.1.1-4**. For more information click 6.14.4.

The following figure shows multiplexing **Pattern 2**.




This figure shows the **frequency location** in a reasonably correct way, but the **actual time location** of **SSB** and of **PDCCH** (of **CORESET#0**) may vary, **independently**, left or right, as indicated by the dotted black arrows.

**Multiplexing Pattern 2** is sometimes said to be "frequency multiplexing".

#### 6.8.8.4.3 Multiplexing Pattern 3

**Multiplexing Pattern 3** can only be used in **FR2** bands.

**Multiplexing Pattern 3** uses the **frequency location** and **PDCCH duration tables 13-8** and **13-10A** of TS 38.213. From these tables, we can see:

- The **subcarrier spacing** used in **SSB** is the **same** as the one used in **CORESET#0** (which is indicated in `subcarrierSpaceCommon` from `MIB`

(click 8.12.3)). Therefore, the symbol duration in **SSB** is the **same** as the one of **CORESET#0**.

- There is **no overlap** between the **frequency locations** of **SSB** and of **CORESET#0**.
- There is an **overlap** between the **time locations** of **SSB** and of **PDCCH** (of **CORESET#0**).
- **CORESET#0 starts** in the **first CRB above** the location of **SSB** and extends upwards in frequency. This **first CRB** belongs to the **CRB range** with the **same subcarrier spacing** as **CORESET#0**.
- The **frequency separation** ("D" in the figure below) between **CORESET#0** and **SSB** is **0** (i.e., they are contiguous) when "$k_{SSB}$" (click 8.7.2) is **0**. Otherwise, it is the bandwidth of **one CORESET#0 RB** minus the bandwidth indicated by "$k_{SSB}$".
- The **CORESET#0** bandwidth may be **24** or **48 CORESET#0 contiguous** Resource Blocks.
- The **duration** of **PDCCH** (of **CORESET#0**) is **2 symbol intervals**.
- Therefore, the size of each **CORESET#0 instance** may be **48** or **96 REGs**, which is **8** or **12 CCEs**.

**Multiplexing Pattern 3** uses the **PDCCH monitoring occasion tables 13-15** and **13-15A** of TS 38.213. From these tables, we can see:

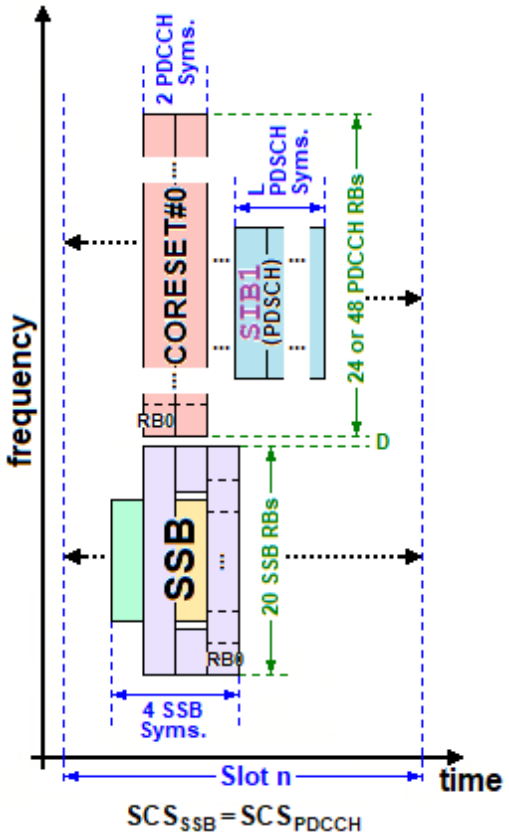
- Each **UE** has **1 monitoring occasion** for **each SSB burst**. **Each monitoring occasion** is placed in the **same slot** as the **SSB** of the beam selected by the **UE**.
- The **NR node** may transmit **up to 8 DCI 1_0 messages** for each **SSB burst**.
- There **may** be an **overlap** between the **time locations** of **SSB** and of **PDCCH** (of **CORESET#0**) because there is **no overlap** between the **frequency locations** of **SSB** and of **CORESET#0**.

For more detail on the **PDCCH monitoring occasions** in **search space zero**, click 6.8.6.2.

**Multiplexing Pattern 3** uses "PDSCH time-domain resource allocation" **Default C**, corresponding to table **5.1.2.1.1-5** of TS 38.214. From this table and from tables **13-15** and **13-15A** mentioned above, we can see:

- **DCI message 1_0** (within **PDCCH** of **CORESET#0**) and **SSB** are **always** in the **same slot** because in tables**13-15** and **13-15A** the values of "$SFN_c$ and "$n_c$ are always **equal** (respectively) to "$SFN_{SSB,i}$" and "$n_{SSB,i}$".
- **DCI message 1_0** (within **PDCCH** of **CORESET#0**) and **SIB1** (within **PDSCH**) are **always** in the same slot because "$K_0$" is always **0** in table **5.1.2.1.1-5**. For more information click 6.14.4.

The following figure shows multiplexing **Pattern 3**.

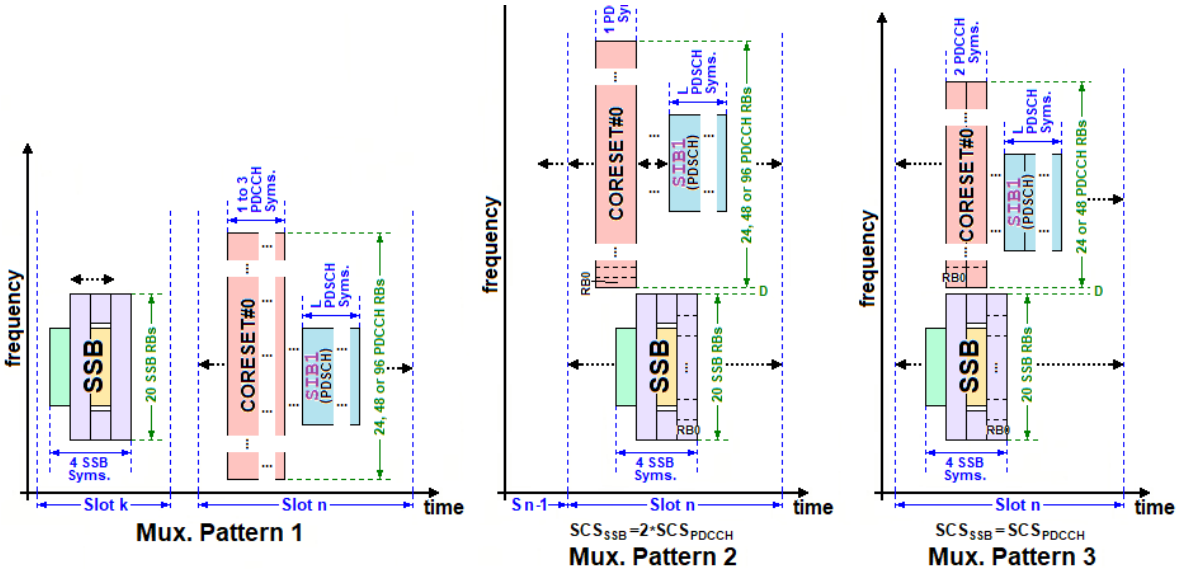



This figure shows the **frequency location** in a reasonably correct way, but the **actual time location** of **SSB** and of **PDCCH** (of **CORESET#0**) may vary, **independently**, left or right, as indicated by the dotted black arrows.

**Multiplexing Pattern 3** is sometimes said to be "time and frequency multiplexing".

#### 6.8.8.4.4 Comparison of the three multiplexing Patterns

The following figure shows the **3** possible **multiplexing Patters**:



This figure shows the **frequency location** in a reasonably correct way, but the **actual time location** of **SSB** and of **PDCCH** (of **CORESET#0**) may vary, **independently**, left or right, as indicated by the dotted black arrows.

Notice that in **Patterns 2** and **3** the **frequency** separation between **SSB** and **CORESET#0** is labelled as "**D**". As indicated above, "**D**" is **0** if "$k_{SSB}$" (click 8.7.2) is **0**, and it is **1RB-$k_{SSB}$** otherwise.

### 6.8.9 PDCCH scheduling of payload PREs and DM-RS PREs

**PDCCH DM-RS** are the **DM-RS** (click 4.11.3.2.1) within **PDCCH**. They are characterized by their **scheduling** within the **PDCCH PREs**, that is described in this section, and by their **contents** (click 6.11).

Out of the **72 REs** of **each Control Channel Element** (**CCE**, click 6.8.1.1), **54 REs** (3/4) are for the **PDCCH payload** and **18 REs** (1/4) are for **PDCCH DM-RS**.

Within each **CCE**, the **PDCCH DM-RS REs** are **interleaved** with the **payload PREs**, in **every symbol**, every **4 subcarriers**, starting in **subcarrier** with index **1**. Therefore, the **PDCCH DM-RS PREs** are placed in **subcarrier indexes 1**, **5** and **9** from **every PDCCH Resource Block**.

## 6.9 PDCCH payload contents: DCI messages

**Each DCI message** is sent by the **NR node** to **1 UE** (or **group** of **UEs**) to signal **1** (or a **few**) **scheduling** (click 4.11.4) **decisions** and for different **control purposes**.

**DCI messages** can have **different Formats**. **DCI Formats 0_x**, **1_x** and **4_x** are used to **activate** and **deactivate configured grants** (in **semi-persistent scheduling**, click 4.11.4.3) and to perform **resource assignments** (in **dynamic scheduling**, click 4.11.4.3). **DCI Formats 0_x** are used for **Uplink scheduling** while **DCI Formats 1_x** and **4_x** are used for **Downlink scheduling**. **DCI Formats 2_x** are used for **different control purposes**. **DCI Formats 3_x** are used for **Sidelink scheduling**.

There are **several Formats** of **Downlink Control Information** (**DCI**) **messages**, according to their **different usages**. They are defined in clause 7.3.1 of TS 38.212. The following is table 7.3.1-1, that summarizes **all** the **DCI Formats**, their possible associated **RNTI** type(s), and the **usage** for **each** such **combination**.

| Purpose | DCI Format | DCI Format usage | RNTI type(s) |
|---|---|---|---|
| PUSCH scheduling | 0_0 | For **1 PUSCH assignment**, of **1 unicast codeword**, in **1 layer**, in **1 cell**. Typically for **fallback/Msg3-based** or for **contention-based PUSCH scheduling**. | **C-RNTI**, **CS-RNTI**, **MCS-C-RNTI** |
| | | | **TC-RNTI** |
| | 0_1 | **Either** for **a) 1** to **8 PUSCH assignments**, of **1 unicast codeword** in **each assignment**, in **1** to **4 layers**, in **1 cell**; or **b)** for sending **Downlink feedback indication** for **configured grant PUSCH** (**CG-DFI**). | **C-RNTI**, **CS-RNTI**, **MCS-C-RNTI**, **SP-CSI-RNTI** |
| | 0_2 | For **1 PUSCH assignment**, of **1 unicast codeword**, in **1** to **4 layers**, in **1 cell**. It can also **(de)activate** a **PUSCH semi-persistent scheduling**. | **C-RNTI**, **CS-RNTI**, **MCS-C-RNTI**, **SP-CSI-RNTI** |

| Purpose | DCI Format | DCI Format usage | RNTI type(s) |
|---|---|---|---|
| PDSCH scheduling | 1_0 | For **1 PDSCH assignment**, of **1 codeword**, in **1 layer**, in **1 cell**. It can be in **unicast** or in **broadcast** (e.g., for **System Information** delivery). | **C-RNTI**, **CS-RNTI**, **MCS-C-RNTI** |
| | | | **P-RNTI** |
| | | | **SI-RNTI** |
| | | | **RA-RNTI**, **MSGB-RNTI** |
| | | | **TC-RNTI** |
| | 1_1 | For **1** to **8 PDSCH assignments**, of **1** or **2 unicast codeword(s)** in **each assignment**, in **1** to **4 layers**, in **1 cell** and/or for **triggering one shot HARQ-ACK codebook feedback**. | **C-RNTI**, **CS-RNTI**, **MCS-C-RNTI** |
| | 1_2 | For **1 PDSCH assignment**, of **1 unicast codeword**, in **1** to **4 layers**, in **1 cell**. | **C-RNTI**, **CS-RNTI**, **MCS-C-RNTI** |
| Various control purposes | 2_0 | Notifying a **group** of **UEs** of the **slot format**, **available RB sets**, **COT duration** and **search space group switching** | **SFI-RNTI** |
| | 2_1 | Notifying a **group** of **UEs** of the **PRB(s)** and **OFDM symbol(s)** where **UE** may assume no transmission is intended for the **UE** | **INT-RNTI** |
| | 2_2 | Transmission of **TPC commands** for **PUCCH** and **PUSCH** | **TPC-PUCCH-RNTI**, **TPC-PUSCH-RNTI** |
| | 2_3 | Transmission of a **group** of **TPC commands** for **SRS transmissions** by **one or more UEs** | **TPC-SRS-RNTI** |
| | 2_4 | Notifying a **group** of **UEs** of the **PRB(s)** and **OFDM symbol(s)** where **UE cancels** the corresponding **UL transmission** from the **UE** | **CI-RNTI** |
| | 2_5 | Notifying the availability of **soft resources** as defined in clause 9.3.1 of TS 38.473 | **AI-RNTI** |
| | 2_6 | Notifying the **power saving information outside DRX Active Time** for **one or more UEs** | **PS-RNTI** |
| | 2_7 | Notifying **paging early indication** and **TRS availability indication** for **one or more UEs**. | **PEI-RNTI** |
| Sidelink schedlng. | 3_0 | **NR** in **1 cell** | **SL-RNTI**, **SL-CS-RNTI** |
| | 3_1 | **LTE** in **1 cell** | **SL Semi-Persistent Scheduling V-RNTI** |
| PDSCH MBS scheduling | 4_0 | For **1 PDSCH assignment**, of **1 broadcast codeword**, in **1 layer**, in **1 cell**. | **MCCH-RNTI**, **G-RNTI** |
| | 4_1 | For **1 PDSCH assignment**, of **1 multicast codeword**, in **1 layer**, in **1 cell**. | **G-RNTI**, **G-CS-RNTI** |
| | 4_2 | For **1 PDSCH assignment**, of **1** or **2 multicast codeword(s)**, in **1** to **8 layers**, in **1 cell**. | **G-RNTI**, **G-CS-RNTI** |

The following **relevant tables** are complementary to this table:

- Table with **all** the **search space types** with their **cell type**, their related **RNTI**(s), their related **DCI Formats** and their **combined function**, click 6.8.5.5.
- Table with **all** the **RNTIs**, with a **short individual description**, click 6.9.10.1.
- Table with **all** the **RNTIs** with their related **DCI Formats**, their **mapping** to **channels**, and their **combined function**, click 6.9.10.2.

Within **each DCI Format**, there are **different** types of **DCI messages**, depending on what **RNTI** is used to mask the **CRC**.

Regarding **data** delivery, we can establish the following four **categories** of **DCI Formats**, indicating the **RNTIs** used to mask the **DCI message CRC**:

- **Unicast UL/DL: DCI Formats 0_0**, **0_1**, **0_2** (**UL**) and **1_0**, **1_1**, **1_2** (**DL**)

  When their **CRC** is masked by **C-RNTI**, **CG-SDT-CS-RNTI**, **CS-RNTI** or **MCS-C-RNTI**.

- **Multicast DL: DCI Formats 4_1** and **4_2**

  When their **CRC** is masked by **G-RNTI** for **multicast** or by **G-CS-RNTI**.

- **Broadcast DL: DCI Format 4_0**

  When their **CRC** masked by **MCCH-RNTI** or **G-RNTI** for **broadcast**.

- **SL transmissions: DCI Formats 3_0** and **3_1**

  When their **CRC** masked by **SL-RNTI**, **SL-CS-RNTI** or **SL Semi-Persistent Scheduling V-RNTI**.

When the **PDCCH** reception includes two **PDCCH candidates** from two respective **search spaces**, as described in clause 10.1 of TS 38.213, for the purpose of determining the **Downlink RB set** of a **PDSCH** when scheduled by **DCI Format 1_0**, the **CORESET** with **lower ID** among two **CORESETs** associated with two **PDCCH candidates** is used.

For a listing of **all** the **possible fields** for **each DCI Format**, click 6.9.1.

For a **PDCCH overview**, click 6.7.

For the **scheduling** of **PDCCH** and for its **DM-RS**, click 6.8.

For the **payload contents** of **PDCCH**, click 6.9.

For the **configuration parameters** of **PDCCH**, click 6.12.

## 6.9.1 DCI message fields

The following table summarizes the **most relevant DCI message fields** (in alphabetical order) indicating for **each field** in what **DCI Formats** (except **2_4** to **2_7**) it **may** be **present** and its **most frequent bitlengths**:

| | | **DCI message Format** (and **field bitlengths**) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | **PUSCH Unicast** | | | **PDSCH** | | | | | |
| | | | | | **B/U** | **Unicast** | | **Bcst.** | **Multicast** | |
| **DCI Field** | **(q)** | **0_0** 6.9.2 | **0_1** 6.9.3 | **0_2** 6.9.4 | **1_0** 6.9.5 | **1_1** 6.9.6 | **1_2** 6.9.7 | **4_0** 6.9.9.1 | **4_1** 6.9.9.2 | **4_2** 6.9.9.3 |
| ***Antenna ports*** | | | 2 to 5 | 0, 2 to 5 | | 4 to 6 | 0, 4 to 6 | | | 4 to 6 |
| ***Bandwidth part indicator*** | | | 0 to 2 | 0 to 2 | | 0 to 2 | 0 to 2 | | | |
| ***beta_offset indicator*** | | | 0, 2 | 0, 1 | | | | | | |
| ***ChannelAccess-CPext*** | | 2 | 0, 3 | 0, 2 | 2 | 0 to 4 | 0 to 4 | | | |
| ***ChannelAccess-CPext-CAPC*** | | | 0 to 6 | 0 to 6 | | | | | | |
| ***Carrier indicator*** | | | 0, 3 | 0, 3 | | 0 to 3 | 0 to 3 | | | |
| ***CBG flushing out information*** | | | | | | 0, 1 | | | | |

| | | DCI message Format (and field bitlengths) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | PUSCH Unicast | | | PDSCH | | | | | |
| | | | | | B/U | Unicast | | Bcst. | Multicast | |
| DCI Field | (q) | 0_0 6.9.2 | 0_1 6.9.3 | 0_2 6.9.4 | 1_0 6.9.5 | 1_1 6.9.6 | 1_2 6.9.7 | 4_0 6.9.9.1 | 4_1 6.9.9.2 | 4_2 6.9.9.3 |
| *CBG transmission information* | | | 0, 2, 4, 6, 8 | | | 0, 2, 4, 6, 8 | | | | |
| *CSI request* | | | 0 to 6 | 0 to 6 | | | | | | |
| *DFI flag* | | | 0, 1 | | | | | | | |
| *DMRS sequence initialization* | | | 0, 1 | 0, 1 | | 1 | 0, 1 | | | 1 |
| *Downlink assignment index:* | | | | 0 to 6 | 2 | 0, 2, 4, 6 | 0, 1, 2, 4 | | 2 | 0, 2 |
| *1st Downlink assignment index:* | | | 1, 2, 4 | 1, 2 | | | | | | |
| *2nd downlink assignment index:* | | | 0, 2, 4 | 0, 2 | | | | | | |
| *3rd downlink assignment index* | | | 0 to 2 | 0 to 2 | | | | | | |
| *Enabling/disabling HARQ-ACK feedback indication* | | | | | | | | | | 0, 1 |
| *Enhanced Type 3 codebook indicator* | | | | | | 0 to 3 | 0 to 3 | | | |
| *Frequency hopping flag* | | 1 | 0, 1 | 0, 1 | | | | | | |
| *Frequency domain resource assignment* | | Var. | Var. | Var. | Var. | Var. | Var. | Var. | Var. | Var. |
| *HARQ-ACK bitmap* | | | 16, 32 | | | | | | | |
| *HARQ process number* | | 4 | 4, 5 | 0 to 5 | 4 | 4, 5 | 0 to 5 | | 4 | 4 |
| *HARQ-ACK retransmission indicator* | | | | | | 0, 1 | 0, 1 | | | |
| *Identifier for DCI formats* | | 1 | 1 | 1 | 1 | 1 | 1 | | | |
| *Invalid symbol pattern indicator* | | | 0, 1 | 0, 1 | | | | | | |
| *LSBs of SFN* | | | | | 2 | | | | | |
| *MCCH change notification* | | | | | | | | 2 | | |
| *Minimum applicable scheduling offset indicator* | | | 0, 1 | | | 0, 1 | | | | |
| *Modulation and coding scheme* | 0 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 |
| | 1 | | | | | 5 | | | | 5 |
| *New data indicator* | 0 | 1 | 1 to 8 | 1 | 1 | 1 to 8 | 1 | | 1 | 1 |
| | 1 | | | | | 1 to 8 | | | | 1 |
| *New feedback indicator* | | | | | | 0 to 2 | | | | |
| *Number of requested PDSCH group(s)* | | | | | | 0, 1 | | | | |
| *One-shot HARQ-ACK request* | | | | | | 0, 1 | 0, 1 | | | |
| *Open-loop power control parameter set indication* | | | 0 to 2 | 0 to 2 | | | | | | |
| *Padding* | | Var. | | | | | | | | |
| *PDCCH monitoring adaptation indication* | | | 0 to 2 | 0 to 2 | | 0 to 2 | 0 to 2 | | | |
| *PDSCH-to-HARQ_feedback timing indicator* | | | | | 3 | 0 to 3 | 0 to 3 | | 3 | 0 to 3 |
| *PDSCH group index* | | | | | | 0, 1 | | | | |
| *PRACH Mask index* | | | | | 4 | | | | | |
| *PRB bundling size indicator* | | | | | | 0, 1 | 0, 1 | | | 0, 1 |

| | | **DCI message Format** (and **field bitlengths**) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | **PUSCH Unicast** | | | **PDSCH** | | | | | |
| | | | | | **B/U** | **Unicast** | | **Bcst.** | **Multicast** | |
| **DCI Field** | **(q)** | **0_0** 6.9.2 | **0_1** 6.9.3 | **0_2** 6.9.4 | **1_0** 6.9.5 | **1_1** 6.9.6 | **1_2** 6.9.7 | **4_0** 6.9.9.1 | **4_1** 6.9.9.2 | **4_2** 6.9.9.3 |
| ***Precoding information and number of layers*** | | | 0 to 6 | 0 to 6 | | | | | | |
| ***Priority indicator*** | | | 0, 1 | 0, 1 | | 0, 1 | 0, 1 | | | 0, 1 |
| ***PTRS-DMRS association*** | | | 0, 2 | 0, 2 | | | | | | |
| ***PUCCH Cell indicator*** | | | | | | 0, 1 | 0, 1 | | | |
| ***PUCCH resource indicator*** | | | | | 3 | 3 | 0 to 3 | | 3 | 3 |
| ***Random Access Preamble index*** | | | | | 6 | | | | | |
| ***Rate matching indicator*** | | | | | | 0 to 2 | 0 to 2 | | | 0 to 2 |
| ***Redundancy version*** | **0** | 2 | 2 to 8 | 0 to 2 | 2 | 2 to 8 | 0 to 2 | 2 | 2 | 2 |
| | **1** | | | | | 2 to 8 | | | | 2 |
| ***Reserved bits*** | | | | | Var. | | | 14 | 3 | |
| ***SCell dormancy indication*** | | | 0, 5 | | | 0 to 5 | | | | |
| ***Second Precoding information*** | | | 0 to 5 | 0 to 5 | | | | | | |
| ***Second PTRS-DMRS association*** | | | 0, 2 | 0, 2 | | | | | | |
| ***Second SRS resource indicator*** | | | Var. | Var. | | | | | | |
| ***Second TPC command for scheduled PUCCH*** | | | | | | 0, 2 | 2 | | | |
| ***Second TPC command for scheduled PUSCH*** | | | 0, 2 | 0, 2 | | | | | | |
| ***Short Messages*** | | | | | 8 | | | | | |
| ***Short Messages Indicator*** | | | | | 2 | | | | | |
| ***Sidelink assignment index*** | | | 0 to 2 | | | | | | | |
| ***Slot format indicator*** | | | | | | | | | | |
| ***SRS offset indicator*** | | | 0 to 2 | 0 to 2 | | 0 to 2 | 0 to 2 | | | |
| ***SRS resource indicator*** | | | Var. | Var. | | | | | | |
| ***SRS resource set indicator*** | | | 0, 2 | 0, 2 | | | | | | |
| ***SRS request*** | | | 2 | 0 to 3 | | 0 to 2 | 0 to 3 | | | |
| ***SS/PBCH index*** | | | | | 6 | | | | | |
| ***System information indicator*** | | | | | 1 | | | | | |
| ***TB scaling*** | | | | | 2 | | | | | |
| ***Time domain resource assignment*** | | 4 | 0 to 6 | 0 to 6 | 4 | 0 to 6 | 0 to 4 | 4 | 4 | 0 to 4 |
| ***TPC command for PUCCH/PUSCH*** | | | 0, 2 | | | | | | | |
| ***TPC command for scheduled PUCCH*** | | | | | 2 | 2 | 2 | | | |
| ***TPC command for scheduled PUSCH*** | | 2 | 2 | 0, 2 | | | | | | |
| ***Transmission configuration indication*** | | | | | | 0, 3 | 0 to 3 | | | 0, 3 |
| ***TRS availability indication*** | | | | | 1 to 6 | | | | | |
| ***UL-SCH indicator*** | | | 0, 1 | 1 | | | | | | |
| ***UL/SUL indicator*** | | 0, 1 | 0, 1 | 0, 1 | 1 | | | | | |
| ***VRB-to-PRB mapping*** | | | | | 1 | 0, 1 | 0, 1 | 1 | 1 | 0, 1 |
| ***ZP CSI-RS trigger*** | | | | | | 0 to 2 | 0 to 2 | | | 0 to 2 |

Some **relevant remarks** on this table are:

- The **only common field** between **Formats 0_x**, **1_x** and **4_x** (this table) and **Formats 2_x** (**next** table) is "***TRS availability indication***". It may belong to **both Format 1_0** and **Format 2_7**.
- The table shows **all** the **fields** that a **DCI message** with the corresponding **Format** **may** have, but depending on the **masking RNTI** and on **configuration**, the **message** may have **only** a **subset** of the **fields** shown.
- The column heading "(**q**)" indicates the **identifier** of **each** of the **1** or **2 PDSCH transport block**(**s**), which **also identifies** the **PDSCH elements derived** from **each** of the **1** or **2 PDSCH transport block**(**s**) (e.g., **each** the **1** or **2 codeword**(**s**)).
- In the table heading, "**B/U**" stands for "**Broadcast** or **Unicast**" and "**Bcst.**" stands for **Broadcast**.

The following table summarizes **all** the **DCI message fields** for **DCI Formats 2_4** to **2_7**, indicating for **each field** in what **DCI Formats** it **may** be **present** and its **bitlengths**:

| | DCI 2_x message Format Various control purposes | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **DCI Field** | **2_0** | **2_1** | **2_2** | **2_3** | **2_4** | **2_5** | **2_6** | **2_7** |
| ***Availability indicator*** **1 to *A*** | | | | | | ≤128 | | |
| ***Available RB set indicator*** **1 to *R*** | Var. | | | | | | | |
| ***Block number*** **1 to *B*** | | | Var. | Var. | | | Var. | |
| ***COT duration indicator*** **1 to *O*** | Var. | | | | | | | |
| ***Cancellation indication*** **1 to *C*** | | | | | ≤126 | | | |
| ***Paging indication*** | | | | | | | | Var. |
| ***Pre-emption indication*** **1 to *P*** | | 14•***P*** ≤126 | | | | | | |
| ***Search space set group switching flag*** **1 to *F*** | Var. | | | | | | | |
| ***Slot format indicator*** **1 to *S*** | Var. | | | | | | | |
| ***TRS availability indication*** | | | | | | | | 1 to 6 |

Some **relevant remarks** on this table are:

- For **fields** that may be present **several times**, the **bitlength** indicated in **each cell** corresponds to **all** the possible fields, and **not** to just **one** of them.
- The table shows **all** the **fields** that a **DCI message** with the corresponding **Format** **may** have, but depending on the **masking RNTI** and on **configuration**, the **message** may have **only** a **subset** of the **fields** shown.
- **Format 2_0**: the **bitlength** of **each complete DCI 2_0 message** is ≤128 bits.
- **Format 2_2**: **each field** "***Block number***" may contain the **sub-fields** "***Closed loop indicator***" (0 or 1 bit) and "***TPC command***" (2 bits).

- **Format 2_3**: **each field** "***Block number***" may contain, depending on configuration:
  - **Only 1** "***Block number***" with **1 sub-field** "***SRS request***" (0 or 2 bits) and **1** to ***T*** **sub-fields** "***TPC command number***".
  - **Several** "***Block number***" fields, **each** with **1 sub-field** "***SRS request***" (0 or 2 bits) and **1 sub-field** "***TPC command***" (2 bits).
- **Format 2_6**: **each field** "***Block number***" may contain the **sub-fields** "***Wake-up indication***" (1 bit) and "***SCell dormancy indication***" (0 to 5 bits). The **bitlength** of **each complete DCI 2_6 message** is indicated by `sizeDCI-2-6-r16` from `DCP-Config-r16`.
- The **only common field** between **Formats 2_x** (this table) and **Formats 0_x**, **1_x** and **4_x** (**previous** table) is "***TRS availability indication***". It may belong to **both Format 1_0** and **Format 2_7**.

**DCI Formats 3_x** are **not included** because they correspond to **Sidelink**, which is **not covered** in this **book**.

## 6.9.2 DCI Format 0_0 messages

**Each DCI Format 0_0 message** signals **1 PUSCH assignment**, for **1 unicast codeword**, in **1 layer**, in **1 cell**.

Its **fields** and **functionality depend** on the **RNTIs** that are used to **mask** its **CRC**.

- **DCI Format 0_0** with **C-RNTI**, **CS-RNTI** or **MCS-C-RNTI** (click 6.9.2.1).
- **DCI Format 0_0** with **TC-RNTI** (click 6.9.2.2).

For more detail on **DCI Format 0_0**, refer to clause 7.3.1.1.1 of TS 38.212.

### 6.9.2.1 DCI Format 0_0 messages with C-RNTI, CS-RNTI or MCS-C-RNTI

This case is for **every DCI Format 0_0 message** with the **last 16 bits** of its **CRC** masked with **either C-RNTI**, **CS-RNTI** or **MCS-C-RNTI**.

**Each DCI Format 0_0 message** signals **1 PUSCH assignment**, for **1 unicast codeword**, in **1 layer**, in **1 cell**.

In this **Format**, and with the said **RNTIs**, the following table (summarized from TS 38.212) shows its **fields** and **functionality**:

| Field | Bitlength | Comments |
|---|---|---|
| ***Identifier for DCI formats*** | 1 | Set to **0**, to indicate **DCI Format 0_x** |
| ***Frequency domain resource assignment***, (click 7.11.3) | Variable | Indicates the **Resource Blocks** where the **UE** receiving this **DCI message** can transmit its **PUSCH** information |
| ***Time domain resource assignment***, click 7.11.4 | 4 | Indicates the **slot** and **symbol intervals** where the **UE** receiving this **DCI message** can transmit its **PUSCH** information |
| ***Frequency hopping flag*** (click 7.6.5.3) | 1 | Value "**1**" implies **frequency hopping** in this **PUSCH assignment** and value "**0**" implies **no frequency hopping**. |
| ***Modulation and coding scheme index***, (click 7.13) | 5 | Indicates **MCS**, by inputting its value in the corresponding table 6.1.4.1-**1** to **-4** of TS 38.214. |

| Field | Bitlength | Comments |
|---|---|---|
| ***New data indicator***, (click 4.5.6.3) | 1 | Indicates, as an **alternating bit**, the **first transmission** of a **PUSCH codeword**. |
| ***Redundancy version*** (click 4.5.6.6) | 2 | Indicates the type of **LDPC rate-matching** that **must** be applied to this **PUSCH codeword transmission** {**0**} or **retransmission** {**1**, **2**, **3**} |
| ***HARQ process number***, (click 4.5.6.1) | 4 | Indicates the **HARQ process Id** to which this **PUSCH codeword** corresponds. |
| ***TPC command for scheduled PUSCH.*** | 2 | Indicates the **transmission power command** for this **PUSCH assignment**. |
| ***ChannelAccess-CPext***. | 2 | Indicates a combination of **channel access type** and **CP extension**. |
| ***Padding*** | Variable | **Only** if required. |
| ***UL/SUL indicator***, (click 10.4) | 0 or 1 | **If present**, value "**0**" indicates that this **PUSCH assignment** is for **UL**. Value "**1**" indicates for **SUL**. |

For a listing of **all** the **possible fields** for **each DCI Format**, click 6.9.1.

For more detail on **DCI Format 0_0**, refer to clause 7.3.1.1.1 of TS 38.212.

### 6.9.2.2 DCI Format 0_0 messages with TC_RNTI

This case is for **every DCI Format 0_0 message** with the **last 16 bits** of its **CRC** masked with **TC_RNTI**.

**Each DCI Format 0_0 message** signals **1 PUSCH assignment**, for **1 unicast codeword**, in **1 layer**, in **1 cell**.

The **codeword** contains **Msg3** (click 9.3.4), from the **random access procedure** (click 9).

The **message fields** in **this case** are the **same** as in the **previous case** (click 6.9.2.1). The **semantics** of **each** of the **fields** is also the **same**, **except** for the **following fields**, where **some differences** exist:

- ***Frequency domain resource assignment***
- ***Modulation and coding scheme index*** ($I_{MCS}$)
- ***HARQ process number***

  This field is **reserved** because the **random access procedure always** uses **HARQ process number 0**.
- ***UL/SUL indicator***

The **differences** in the **semantics** of these **fields** arise from the delivery of **random access procedure messages** in **this case**, in respect to the delivery of **data** in the **previous case** (click 6.9.2.1).

For a listing of **all** the **possible fields** for **each DCI Format**, click 6.9.1.

For more detail on **DCI Format 0_0**, refer to clause 7.3.1.1.1 of TS 38.212.

## 6.9.3 DCI Format 0_1 messages

**Every DCI Format 0_1 message** has the **last 16 bits** of its **CRC masked** with **either C-RNTI**, **CS-RNTI**, **MCS-C-RNTI** or **SP-CSI-RNTI**.

**Each DCI Format 0_1 message** signals **either** a) **1** to **8 PUSCH assignment(s)**, for **1 unicast codeword** in **each assignment**, in **1** to **4 layers**, in **1 cell** and in, **or** b) **Downlink feedback indication** for **configured grant PUSCH** (**CG-DFI**).

In case of **more than 1 PUSCH assignment** (i.e., a **multi-PUSCH assignment**), **each** and **every** of the **2** to **8 PUSCH assignments** have the **same frequency-domain resources** (click 7.11.3) and **different time-domain resources** (click 7.11.4).

**More** than **1 PUSCH assignment** can be **scheduled** with **Format 0_1** if `pusch-TimeDomainAllocationListForMultiPUSCH-r16` from `PUSCH-Config` (click 7.9.4) is **present**. In this case, it is possible to configure **each UE** with `1` to `64` **options** of **time domain allocations**. **Each such option** contains `1` to `8` **time domain PUSCH assignments**. Consequently, **each DCI Format 0_1 message** may **schedule 1** of the said **options**, where **each option** has `1` to `8` **PUSCH assignments**.

The **data type** of the **parameter** indicated above is `PUSCH-TimeDomainResourceAllocationList-r16` (see within 7.15.5.1), which is a `SEQUENCE OF PUSCH-TimeDomainResourceAllocation-r16` (see within 7.15.5.1). This **sequence** contains `1` to `maxNrofUL-Allocations-r16` **instances**, where the value of this **constant** is `64`.

**Each instance** of this latter **data type** configures the **UE** with **1 multi-entry** of **PUSCH time-domain resource allocations** (click 7.11.4) within a **given list** of **multi-entries**. **Each multi-entry** contains a `SEQUENCE OF PUSCH-Allocation-r16`. This **sequence** contains `1` to `maxNrofMultiplePUSCHs-r16` **instances**, where the value of this **constant** is `8`.

**Each instance** of `PUSCH-Allocation-r16` defines the **time-domain resources** of **1 PUSCH assignment**.

The previous **option** ("**multi-scheduling**") of **multiple PUSCH assignments** with **1 DCI Format 0_1 message** seems **incompatible** with **PUSCH HARQ CBG mode** (click 4.5.6.4), because when the said **option** is **set**, there are **not** enough **bits** in the "***CBG transmission information***" **field** to **acknowledge** (click 4.5.6.7.2) the required **quantity** of **CBGs**.

In this **Format**, and with the said **RNTIs**, the following table (summarized from TS 38.212) shows its **mandatory fields** and for **each** of them its **bitlength** and **functionality**:

| Field | Bitlength | Comments |
|---|---|---|
| ***Identifier for DCI formats*** | 1 | Set to **0**, to indicate **DCI Format 0_x** |
| ***Carrier indicator*** | 0 to 3 | If a **UE** is configured with an **instance** of `CrossCarrierSchedulingConfig` for a **serving cell** the ***Carrier indicator*** **value** corresponds to the value indicated by the said **instance**. Refer to clause 10.1 of TS 38.213. |
| ***DFI flag*** | 0, 1 | See below. |

The "***DCI flag***" field has the following **bitlength** and **semantics**:

- **1 bit**, if the **UE** is configured to **monitor DCI Format 0_1** with **CRC** masked with **CS-RNTI** and for **operation** in a cell for **shared spectrum** when `cg-RetransmissionTimer` is **present**.

  For a **DCI format 0_1 message** with its **CRC** masked with **CS-RNTI**, **value** "**0**" indicates **activating** or **releasing type 2 CG transmission** while **value** "**1**" indicates CG-DFI.

  For a **DCI format 0_1 message** with its **CRC** masked with **C-RNTI**, **MCS-C-RNTI** or **SP-CSI-RNTI** and for operation in a cell for **shared spectrum**, the bit is **reserved**.

- **0 bit otherwise**.

***If*** the **DCI Format 0_1 message** signals a **configured grant downlink feedback indication** (**CG-DFI**), ***then***, **all** its **remaining fields** are:

| Field | Bitlength | Comments |
|---|---|---|
| ***HARQ-ACK bitmap*** (click 4.5.6.7). | 16, 32 | **HARQ** (**N**)**ACKs** for **all** the **HARQ processes**. **Each** "**1**" in the **bitmap** indicates **ACK** of the **last transport block** in the corresponding **HARQ process**. **Each** "**0**" indicates **NACK**. **HARQ process indices** start in **0** and go upwards from **MSB** to **LSB**. |
| ***TPC command for scheduled PUSCH.*** | 2 | Indicates the **transmission power command** for **all** the **configured grants**. |
| **All remaining bits** | Variable | Set to "**0**". |

***else***, **all** its **remaining fields** are:

| Field | Bitlength | Comments |
|---|---|---|
| ***UL/SUL indicator*** (click 10.4) | 0 or 1 | Value "**0**" indicates that **all** the **1** to **8 PUSCH assignments** are for **UL**. Value "**1**" indicates for **SUL**. |
| ***Bandwidth part indicator*** (click 2.2.7.4). | 0 to 2 | Indicates the **active bandwidth part** that the **UE** should **switch to**. |
| ***Frequency domain resource assignment*** (click 7.11.3) | Variable | Indicates the **scheduled Resource Blocks**, that are **common** to **all** the **1** to **8 PUSCH assignments**. |
| ***Time domain resource assignment*** (click 7.11.4) | 0 to 6 | Indicates the **scheduled slot** and **symbol intervals** for **each** of the **1** to **8 PUSCH assignments**. |
| ***Frequency hopping flag*** (click 7.6.5.3) | 0, 1 | Value "**1**" implies **frequency hopping** in **all** the **1** to **8 PUSCH assignments**. Value "**0**" implies **no frequency hopping**. |
| ***Modulation and coding scheme index*** (click 4.12.10) | 5 | Indicates the **MCS** for **all 1** to **8 PUSCH assignments**. **MCS** is found by inputting the field's value in the corresponding table 6.1.4.1-**1** to **-4** of TS 38.214. |
| ***New data indicator*** (click 4.5.6.3) | 1 to 8 | Indicates if the **UE** should **do** (or **not**) the **transmission** of a **new codeword** in **each** of **1** to **8 PUSCH assignments**. This is indicated as an **alternating bit**. |

| Field | Bitlength | Comments |
|---|---|---|
| ***Redundancy version*** (click 4.5.6.6.3) | 2 to 8 | Indicates the type of **LDPC rate-matching** for **each** of the **1** to **8 PUSCH assignments**. |
| ***HARQ process number*** (click 4.5.6.1) | 4, 5 | Indicates the **PUSCH HARQ process Id** of **all** the **1** to **8 codewords** (**each** in **1** of the **1** to **8 PUSCH assignments**). |
| ***1st downlink assignment index*** | 1, 2, 4 | **Each** of the **3 fields** indicates how many **HARQ-ACK bits** to **include**.<br>For more detail, click 4.5.6.7.3 and refer to clause 9.1 of TS 38.213. |
| ***2nd downlink assignment index*** | 0, 2, 4 | |
| ***3rd downlink assignment index*** | 0 to 2 | |
| ***TPC command for scheduled PUSCH*** | 2 | Indicates the **transmission power command** for **PUSCH**. |
| ***Second TPC command for scheduled PUSCH.*** | 0, 2 | Indicates a second the **transmission power command** for **PUSCH**. |
| ***SRS resource set indicator*** | 0, 2 | **Sounding Reference Signal** fields. |
| ***SRS resource indicator*** | Variable | |
| ***Second SRS resource indicator*** | 0, Var. | |
| ***Precoding information and number of layers*** | 0 to 6 | |
| ***Second precoding information*** | 0 to 5 | |
| ***Antenna ports*** | 2 to 5 | **Antenna port** information. It also indicates if **single-symbol** or **double-symbol DM-RS** are used. |
| ***SRS request*** | 2 | **Sounding Reference Signal** fields. |
| ***SRS offset indicator*** | 0 to 2 | |
| ***CSI request*** | 0 to 6 | Triggers the **UE transmission** of an **aperiodic CSI** (**A-CSI**) **report**. |
| ***CBG transmission information*** (click 4.5.6.7.2) | 0, 2, 4, 6, 8 | **HARQ** (**N**)**ACKs** for the **CBGs** of **1 transport block** in **1 PUSCH assignment** in the corresponding **HARQ process**. **Each** "**1**" in the **bitmap** indicates **ACK**, and **each** "**0**" indicates **NACK**. **CBG indices** start in **0** upwards from **MSB** to **LSB**. |
| ***PTRS-DMRS association*** | 0, 2 | |
| ***Second PTRS-DMRS association*** | 0, 2 | |
| ***Beta_offset indicator*** | 0, 2 | |
| ***DMRS sequence initialization*** | 0, 1 | Value of variable "$n_{SCID}$" for **PUSCH DM-RS** sequence initialization. |
| ***UL-SCH indicator*** | 0, 1 | |
| ***ChannelAccess-CPext-CAPC*** | 0 to 6 | |
| ***Open-loop power control parameter set indication*** | 0 to 2 | |
| ***Priority indicator*** | 0, 1 | |
| ***Invalid symbol pattern indicator*** | 0, 1 | |

| Field | Bitlength | Comments |
|---|---|---|
| ***Minimum applicable scheduling offset indicator*** | 0, 1 | |
| ***SCell dormancy indication*** | 0 to 5 | |
| ***Sidelink assignment index*** | 0 to 2 | |
| ***PDCCH monitoring adaptation indication*** | 0 to 2 | |

For a listing of **all** the **possible fields** for **each DCI Format**, click 6.9.1.

For more detail on **DCI Format 0_1**, refer to clause 7.3.1.1.2 of TS 38.212.

### 6.9.4 DCI Format 0_2 messages

**Every DCI Format 0_2 message** has the **last 16 bits** of its **CRC masked** with **either C-RNTI**, **CS-RNTI**, **MCS-C-RNTI** or **SP-CSI-RNTI**.

**Each DCI Format 0_2 message** signals **1 PUSCH assignment**, of **1 unicast codeword**, in **1** to **4 layers**, in **1 cell**.

For a listing of **all** the **possible fields** for **each DCI Format**, click 6.9.1.

For more detail on **DCI Format 0_2**, refer to clause 7.3.1.1.3 of TS 38.212.

### 6.9.5 DCI Format 1_0 messages

**Each DCI Format 1_0 message** signals **1 PDSCH assignment**, of **1 codeword**, in **1 layer** and in **1 cell**. It may be for **unicast** or for **broadcast**, depending on what **RNTI** is used to mask its **CRC**.

Its **fields** and **functionality depend** on what **RNTI** is used to **mask** its **CRC**.

You may find a **reasonably detailed description** of **2 cases** in 6.9.5.1 and 6.9.5.2, plus a **listing** of **other cases** in 6.9.5.3.

For a listing of **all** the **possible fields** for **each DCI Format**, click 6.9.1.

For more detail on **DCI Format 1_0**, refer to clause 7.3.1.2.1 of TS 38.212.

#### 6.9.5.1 DCI Format 1_0 messages with C-RNTI, CS-RNTI or MCS-C-RNTI

This case is for **every DCI Format 1_0 message** with the **last 16 bits** of its **CRC** masked with **either C-RNTI**, **CS-RNTI** or **MCS-C-RNTI**, and the field "***Frequency domain resource assignment***" having a value **different** to **all 1s**.

**Each DCI Format 1_0 message** signals **1 PDSCH assignment**, of **1 codeword**, in **1 layer**, in **1 cell** and in **unicast**.

In this **Format**, and with the said **RNTIs**, the following table (summarized from TS 38.212) shows its **fields** and **functionality**:

| Field | Bitlength | Comments |
|---|---|---|
| ***Identifier for DCI formats*** | 1 | Set to **1**, to indicate **DCI Format 1_x** |
| ***Frequency domain resource assignment*** (click 6.14.3) | Variable | Indicates the **Resource Blocks** where the **UE** receiving this **DCI message** should receive its **PDSCH** information |

| Field | Bitlength | Comments |
|---|---|---|
| ***Time domain resource assignment*** (click 6.14.4) | 4 | Indicates the **slot** and **symbol intervals** for the **PDSCH assignment**. |
| ***VRB-to-PRB mapping*** (click 4.12.15) | 1 | Value **0** is **non-interleaved**: **VRBs match PRBs**. Value **1** is for **interleaved**: **VRBs** do **not** match **PRBs**. |
| ***Modulation and coding scheme index*** (click 6.16.1) | 5 | Indicates the **MCS** for the **PDSCH assignment**, by inputting its value in the corresponding table 5.1.3.1-**1** or **-2** of TS 38.214. |
| ***New data indicator*** (click 4.5.6.3) | 1 | Indicates, as an **alternating bit**, the **first transmission** of this **PDSCH codeword**. |
| ***Redundancy version*** (click 4.5.6.6) | 2 | Indicates the type of **LDPC rate-matching** that has been applied to this **PDSCH codeword transmission** (value **0)** or **retransmission** (values **1**, **2** or **3**). |
| ***HARQ process number*** (click 4.5.6.1) | 4 | Indicates the **HARQ process Id** to which this **PDSCH codeword** corresponds. |
| ***Downlink assignment index*** (click 4.5.6.6) | 2 | Tracks **Downlink assignments** for **HARQ-ACK feedback**.<br>For more detail, 0refer to clause 9.1 of TS 38.213. |
| ***TPC command for scheduled PUCCH*** | 2 | Indicates the **transmission power command** for the **assignmented PUCCH**. Refer to clause 7.2.1 of TS 38.213. |
| ***PUCCH resource indicator*** (click 7.6.4.3) | 3 | Indicates the **PUCCH resource** for the **UE** to send an **UCI message** through **PUCCH**. Refer to clause 9.2.3 of TS 38.213. |
| ***PDSCH-to-HARQ_feedback timing indicator*** (click 7.6.1) | 3 | Indicates the **slots** between a **PDSCH Rx codeword** and its corresponding **(N)ACK** being **sent** by the **UE**. |
| ***Channel-Access-CPext*** | 0 or 2 | **If present**, it indicates a combination of **channel access type** and **cyclic prefix extension**. |
| ***Reserved bits*** | 0 or 2 | |

If the value of its "***Frequency domain resource assignment***" field were **all 1s**, it would indicate that this is a **random access specific DCI Format 1_0**. In this case, the message fields are **different** from the table above and may be found in 9.2.6.

For a listing of **all** the **possible fields** for **each DCI Format**, click 6.9.1.

For more detail on **DCI Format 1_0**, refer to clause 7.3.1.2.1 of TS 38.212.

### 6.9.5.2 DCI Format 1_0 messages with SI-RNTI

This case is for **every DCI Format 1_0 message** with the **last 16 bits** of its **CRC** masked with **SI-RNTI**.

**Each DCI Format 1_0 message** signals **1 PDSCH assignment**, of **1 codeword**, in **1 layer**, in **1 cell** and in **broadcast**. The **codeword** conveys **System Information**, **either** `SIB1` (click 5.3.4) or a `SystemInformation` **message** with **other SIBS** or **posSIBs** (click 5.3.5).

In this case, the following table (summarized from TS 38.212) shows its **fields** and **functionality**:

| Field | Bitlength | Comments |
|---|---|---|
| ***Frequency domain resource assignment*** (click 6.14.3) | Variable | Indicates the **first**, and the **quantity of**, **Virtual Resource Blocks** where the **UE** receiving this DCI message should listen for its **PDSCH** information |
| ***Time domain resource assignment*** (click 6.14.4) | 4 | Indicates the **slot** and **symbol intervals** where the **UE** receiving this **DCI message** should listen for its **PDSCH** information |
| ***VRB-to-PRB mapping*** (click 4.12.15) | 1 | Value **0** is **non-interleaved**: **VRBs match PRBs**. Value **1** is for **interleaved**: **VRBs** do **not** match **PRBs**. |
| ***Modulation and coding scheme index*** ($I_{MCS}$) (click 6.16.1) | 5 | Indicates **MCS**, by inputting $I_{MCS}$ in the corresponding table 5.1.3.1-**1** or -**2** of TS 38.214. |
| ***Redundancy version*** ($rv_{id}$) (click 4.5.6.6) | 2 | Indicates the type of **LDPC rate-matching** that has been applied to this **PDSCH codeword transmission** (value **0**) or **retransmission** (values **1**, **2** or **3**). |
| ***System Information indicator*** | 1 | Value "**0**" indicates that the allocated **PDSCH REs** contain `SIB1` (click 5.3.4) and value "**1**" indicates that they contain a `SystemInformation` (click 5.7.5) message with **other SIBs or posSIBs** (click 5.3.5). |
| ***Reserved bits*** | 15 or 17 | 17 bits if **shared spectrum** in **FR1** or if **FR2-2**. 15 bits otherwise. |

Notice that the "***System information indicator***" field may **only** be present in this case (i.e., a **DCI Format 1_0 message** with its **CRC** masked by **SI-RNTI**). This case can **only** be used when the **NR node** has configured **on-demand system information** delivery (click 5.3.4 and 5.3.5), which is somehow unusual.

For a listing of **all** the **possible fields** for **each DCI Format**, click 6.9.1.

For more detail on **DCI Format 1_0**, refer to clause 7.3.1.2.1 of TS 38.212.

### 6.9.5.3 DCI Format 1_0 messages with P-RNTI, RA-RNTI, MSGB-RNTI or TC-RNTI

In **addition** to the cases presented in the **2 previous subsections**, there are **specific cases** for a **DCI Format 1_0 message** with the **last 16 bits** of its **CRC masked** with the following **RNTIs**:

- **DCI Format 1_0** with **P-RNTI**.
- **DCI Format 1_0** with **RA-RNTI** or **MSGB-RNTI**.
- **DCI Format 1_0** with **TC-RNTI**.

**Each DCI Format 1_0 message** signals **1 PDSCH assignment**, of **1 codeword**, in **1 layer**, in **1 cell** and in **unicast** or **broadcast**.

For a listing of **all** the **possible fields** for **each DCI Format**, click 6.9.1.

For more detail on **DCI Format 1_0**, refer to clause 7.3.1.2.1 of TS 38.212.

### 6.9.6 DCI Format 1_1 messages

**Every DCI Format 1_1 message** has the **last 16 bits** of its **CRC masked** with **either C-RNTI**, **CS-RNTI** or **MCS-C-RNTI**.

**Each DCI Format 1_1 message** signals **1** to **8 PDSCH assignment(s)** for **1** or **2 unicast codeword(s)** in **each assignment**, in **1** to **8 layer(s)**, in **1 cell**. It may also be used for **triggering** a **one-shot HARQ ACK-codebook feedback**.

In case of **more than 1 PDSCH assignment** (i.e., a **multi-PDSCH assignment**), **each** and **every** of the **2** to **8 PDSCH assignments** have the **same frequency-domain resources** (click 6.14.3) and **different time resources** (click 6.14.4).

**More** than **1 PDSCH assignment** can be **scheduled** with **Format 1_1** if `pdsch-TimeDomainAllocationListForMultiPDSCH-r17` from `PDSCH-Config` (click 6.18.6) is **present**. In this case, it is possible to configure **each UE** with `1` to `64` **options** of **time domain allocations**. **Each such option** contains `1` to `8` **time domain PDSCH assignments**. Consequently, **each DCI Format 1_1 message** may **schedule 1** of the said **options**, where **each option** has `1` to `8` **PDSCH assignments**.

The **data type** of the **parameter** indicated above is `MultiPDSCH-TDRA-List-r17` (see within 6.18.6.1), which is a `SEQUENCE OF MultiPDSCH-TDRA-r17` (see within 6.18.6.1). This **sequence** contains `1` to `maxNrofDL-AllocationsExt-r17` **instances**, where the value of this **constant** is `64`.

**Each instance** of this latter **data type** configures the **UE** with **1 option** of **PDSCH time-domain resource allocations** (click 6.14.4). **Each option** (i.e., **each instance**) contains a `SEQUENCE OF PDSCH-TimeDomainResourceAllocation-r16`. This **sequence** contains `1` to `maxNrofMultiplePDSCHs-r17` **instances**, where the value of this **constant** is `8`.

**Each instance** of `PDSCH-TimeDomainResourceAllocation-r16` defines the **time-domain resources** of **1 PDSCH assignment**.

The previous **option** ("**multi-scheduling**") of **multiple PDSCH assignments** with **1 DCI Format 1_1 message** seems **incompatible** with **PDSCH HARQ CBG mode** (click 4.5.6.4), because when the said **option** is **set**, there are **not** enough **bits** in the "***CBG transmission information***" **field** to **signal** (click 4.5.6.6.2) the required **quantity** of **retransmitted CBGs**.

In this **Format**, and with the said **RNTIs**, the following table (summarized from TS 38.212) shows its **fields** and for **each** of them its **bitlength** and **functionality**:

| Field | (q) | Bitlength | Comments |
|---|---|---|---|
| ***Identifier for DCI formats*** | | 1 | Set to **1**, to indicate **DCI Format 1_x** |
| ***Carrier indicator*** | | 0 to 3 | **Reserved** when is carried on the **PCell** and the **UE** is configured for scheduling on the **PCell** from an **SCell**, with the same number of bits as that in this format carried by **PDCCH** on the **SCell** for the scheduling on the **PCell**. Refer to clause 10.1 of TS 38.213. |
| ***Bandwidth part indicator*** (click 2.2.7.4) | | 0 to 2 | Indicates the **active bandwidth part** that the **UE** should **switch to**. |

| Field | (q) | Bitlength | Comments |
|---|---|---|---|
| ***Frequency domain resource assignment*** (click 6.14.3) | | Variable | Indicates the **scheduled Resource Blocks**, that are **common** to **all** the **1** to **8 PDSCH assignments**. |
| ***Time domain resource assignment*** (click 6.14.4) | | 0 to 6 | Indicates the **scheduled slot** and **symbol intervals** for **each** of the **1** to **8 PDSCH assignments**. |
| ***VRB-to-PRB mapping*** (click 4.12.15) | | 0, 1 | 0 bit if resource allocation **DFA-Type 0 is** configured or if **interleaved VRB-to-PRB mapping** is **not** configured.<br>1 bit otherwise. Refer to table 7.3.1.2.2-5 or clause 7.3.1.6 of TS 38.214. |
| ***PRB bundling size indicator*** | | 0, 1 | 0 bit if `prb-BundlingType` from `PDSCH-Config` (click 6.18.6) is **absent**, or is **set** to `staticBundling`.<br>1 bit otherwise. Refer to clause 5.1.2.3 of TS 38.214. |
| ***Rate matching indicator*** | | 0 to 2 | |
| ***ZP CSI-RS trigger*** | | 0 to 2 | |
| ***Modulation and coding scheme index*** (click 4.12.10) | **0**<br>**1** | 5 | The **1st field** indicates the **MCS** of **every 1st codeword** in **all** the **1** to **8 PDSCH assignments**. The **2nd field** is the **same** for **every 2nd codeword**. **MCS** is found by inputting the field's value in the corresponding table 5.1.3.1-**1** to **-4** of TS 38.214. |
| ***New data indicator*** (click 4.5.6.3) | **0**<br>**1** | 1 to 8 | The **1st field** indicates if the **1st codeword**, in **each** of the **1** to **8 PDSCH assignments**, is a **new transmission** or a **retransmission**. This is indicated as an **alternating bit**. The **2nd field** indicates the **same** for the **2nd codeword**.<br>Refer to clause 5.1.3 of TS 38.214. |
| ***Redundancy version*** (click 4.5.6.6.3) | **0**<br>**1** | 2 to 8 | The **1st field** indicates the **LDPC rate-matching** of the **1st codeword** in **all** the **1** to **8 PDSCH assignments**. The **2nd field** is the **same** for **every 2nd codeword**. |
| ***HARQ process number*** (click 4.5.6.1) | | 4, 5 | Indicates the **PDSCH HARQ process Id** of **all** the **1** to **16 codewords** (**1** or **2** in **each** of the **1** to **8 PDSCH assignments**). |
| ***Downlink assignment index*** (click 4.5.6.7.3) | | 1, 2, 4 | Tracks **Downlink assignments** for **HARQ-ACK feedback**.<br>For more detail, refer to clause 9.1 of TS 38.213. |
| ***TPC command for scheduled PUCCH*** | | 2 | Indicates the transmission power command for **PUCCH**. Refer to clause 7.2.1 of TS 38.213. |
| ***Second TPC command for scheduled PUCCH*** | | 0, 2 | Indicates the transmission power command for **PUCCH**. Refer to clause 7.2.1 of TS 38.213. |
| ***PUCCH resource indicator*** (click 7.6.4.3) | | 3 | Indicates the **PUCCH resource** for the **UE** to send an **UCI message** through **PUCCH**. Refer to clause 9.2.3 of TS 38.213. |

| Field | (q) | Bitlength | Comments |
|---|---|---|---|
| ***PDSCH-to-HARQ_feedback timing indicator*** (click 7.6.1) | | 0 to 3 | Indicates the **slots** between a **PDSCH Rx codeword** and its corresponding (**N**)**ACK** being **sent** by the **UE**. |
| ***One-shot HARQ-ACK request*** (click 4.5.6.7.1) | | 0, 1 | Indicates the **UE** to send a **one-shot HARQ-ACK feedback**. |
| ***Enhanced Type 3 codebook indicator*** | | 0 to 3 | |
| ***PDSCH group index*** | | 0, 1 | |
| ***New feedback indicator*** | | 0 to 2 | Info for **HARQ-ACK codebook** management. |
| ***Number of requested PDSCH group(s)*** | | 0, 1 | Info for **HARQ-ACK codebook** management |
| ***HARQ-ACK retransmission indicator*** (click 4.5.6.7.1) | | 0, 1 | Indicates the **UE** to **retransmit** the **HARQ-ACK**. |
| ***Antenna ports*** | | 4 to 6 | Antenna port information. It also indicates if **single-symbol** or **double-symbol DM-RS** are used. |
| ***Transmission configuration indication*** | | 0, 3 | |
| ***SRS request*** | | 2 | **Sounding Reference Signal** fields. |
| ***SRS offset indicator*** | | 0 to 2 | |
| ***CBG transmission information*** (click 4.5.6.6.2) | | 0, 2, 4, 6, 8 | Indicates the **CBGs** corresponding to **1** or **2 transport block(s)** that are transmitted in **1 PDSCH assignment**. **Each** "1" in the **bitmap** indicates a transmitted **CBG**, and **each** "0" indicates not transmitted. **CBG indices** start in 0 upwards from **MSB** to **LSB**. |
| ***CBG flushing out information*** (click 4.5.6.6.2) | | 0, 1 | Value **0** indicates to **discard** the **previous CBGs**. Value **1** indicates that **previous CBGs** are **combinable**. |
| ***DMRS sequence initialization*** (click 6.17.1) | | 1 | Value of variable "$n_{SCID}$" for **PDSCH DM-RS** sequence initialization. |
| ***Priority indicator*** | | 0, 1 | |
| ***ChannelAccess-CPext*** | | 0 to 4 | |
| ***Minimum applicable scheduling offset indicator*** | | 0, 1 | |
| ***SCell dormancy indication*** | | 0 to 5 | |
| ***PDCCH monitoring adaptation indication*** | | 0 to 2 | |
| ***PUCCH Cell indicator*** | | 0, 1 | **1 bit** if `pucch-sSCellDyn` is **present**. **0 bit** otherwise.<br>***If*** the **UE** has a `PUCCH-SCell` **instance**, ***then***, `pucch-sSCellDyn` is replaced by `pucchsSCellDynSecondaryPUCCHgroup` for the **secondary PUCCH group**. |

Some **relevant remarks** on this table are:

- The column heading "(**q**)" indicates the **identifier** of **each** of the **1** or **2 PDSCH transport block(s)**, which **also identifies** the **PDSCH elements derived** from

**each** of the **1** or **2 PDSCH transport block(s)** (e.g., **each** of the **1** or **2 codeword(s)**).

For more detail on **DCI Format 1_1**, refer to clause 7.3.1.2.2 of TS 38.212.

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_PDCCH.html#PDCCH_Transport_Process
- https://www.sharetechnote.com/html/5G/5G_DCI.html
- https://www.rfwireless-world.com/5G/5G-NR-PDCCH.html

### 6.9.7 DCI Format 1_2 message

**Every DCI Format 1_2 message** has the **last 16 bits** of its **CRC masked** with **either C-RNTI**, **CS-RNTI** or **MCS-C-RNTI**.

**Each DCI Format 1_2 message** signals **1 PDSCH assignment** for **1 unicast codeword**, in **1 layer**, in **1 cell**.

**DCI Format 1_2** is **much more complex** than **1_0**, with **many more fields** of **variable size**, and **many optional** ones.

For a listing of **all** the **possible fields** for **each DCI Format**, click 6.9.1.

For more detail on **DCI Format 1_2**, refer to clause 7.3.1.2.3 of TS 38.212.

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_PDCCH.html#PDCCH_Transport_Process
- https://www.sharetechnote.com/html/5G/5G_DCI.html
- https://www.rfwireless-world.com/5G/5G-NR-PDCCH.html

### 6.9.8 DCI Format 2_x messages

**DCI messages Formats 2_0**, **2_1**, **2_2**, **2_3**, **2_4**, **2_5**, **2_6** or **2_7** are used for **various control purposes**, as it is presented in the following subsections.

#### 6.9.8.1 DCI Format 2_0 messages

**Every DCI Format 2_0 message** has the **last 16 bits** of its **CRC masked** with **SFI-RNTI**.

**DCI Format 2_0 messages** are used for notifying the **slot format**, **available RB set**, **Channel Occupancy Time (COT) duration**, and **search space group switching**.

The following **fields** are **included**, depending on some parameters (**all** of them from `SlotFormatIndicator`, click 2.3.5.7) being **configured** or **not**, as follows:

- If `slotFormatCombToAddModList` is configured:

  "***Slot format indicator 1***", "***Slot format indicator 2***", …, "***Slot format indicator S***". **Each** such **field** is the **identifier** of the **slot format combination** to be **used** in **one** of the **UE**'s **serving cells**. For more detail, click 2.3.4.2.3.

- If `availableRB-SetsToAddModList-r16` is configured:

  "***Available RB set indicator 1***", "***Available RB set indicator 2***", …, "***Available RB set indicator R***".

- If `co-DurationsPerCellToAddModList-r16` or `co-DurationsPerCellToAddModList-r17` is configured:

  "***COT duration indicator 1***", "***COT duration indicator 2***", …, "***COT duration indicator O***".

- If `switchTriggerToAddModList-r16` is configured:

  "***Search space set group switching flag 1***", "***Search space set group switching flag 2***", …, "***Search space set group switching flag F***".

The **size** of the **DCI Format 2_0 message payload** is configurable by higher layers up to **128 bits**, according to clause 7.3.1.3.1 of TS 38.212.

For a listing of **all** the **possible fields** for **each DCI Format**, click 6.9.1.

For more detail on **Format 2_0**, refer to clause 7.3.1.3.1 of TS 38.212.

### 6.9.8.2 DCI Format 2_1 messages

**Every DCI Format 2_1 message** has the **last 16 bits** of its **CRC masked** with **INT-RNTI**.

**DCI Format 2_1 messages** are used for **notifying the PRB(s)** and **OFDM symbol(s)** where **UE** may assume **no transmission** is intended for the **UE**.

The following **fields** are included:

"***Pre-emption indication 1***", "***Pre-emption indication 2***", …, "***Pre-emption indication P***".

**Each pre-emption indication is 14 bits**. The size of **DCI Format 2_1** is configurable by higher layers **up** to **126 bits**, according to clause 11.2 of TS 38.213.

For a listing of **all** the **possible fields** for **each DCI Format**, click 6.9.1.

For more detail on **Format 2_1**, refer to clause 7.3.1.3.2 of TS 38.212.

### 6.9.8.3 DCI Format 2_2 messages

**Every DCI Format 2_2 message** has the **last 16 bits** of its **CRC masked** with **either TPC-PUSCH-RNTI** or **TPC-PUCCH-RNTI**.

**DCI Format 2_2 messages** are used for the transmission of **TPC commands** for **PUCCH** and **PUSCH**.

The following **fields** are included:

"***Block number 1***", "***Block number 2***",…, "***Block number B***".

Either parameter `tpc-PUSCH` or `tpc-PUCCH`, both from `PDCCH-Config` (click 6.12.5) determines the **index** to the **block number** for an **Uplink** of a **cell**, with the following **DCI message sub-fields** defined within **each field** "***Block number***":

- "***Closed loop indicator***" (0 or 1 bit)
  - For **DCI Format 2_2** with **TPC-PUSCH-RNTI**:
    - **0 bits** if `twoPUSCH-PC-AdjustmentStates` from `PUSCH-PowerControl` is **absent**, in which case the **UE** assumes that each block in the **DCI Format 2_2** is of 2 bits.
    - **1 bit** otherwise, in which case **UE** assumes that each block in the **DCI Format 2_2** is of 3 bits.
  - For **DCI Format 2_2** with **TPC-PUCCH-RNTI**:
    - **0 bits** if `twoPUCCH-PC-AdjustmentStates` from `PUCCH-PowerControl` is **absent**, in which case the **UE** assumes that each block in the **DCI Format 2_2** is of 2 bits.
    - **1 bit** otherwise, in which case the **UE** assumes that each block in the **DCI Format 2_2** is of 3 bits.
- "***TPC command***" (2 bits)

The quantity of information bits in **Format 2_2** shall be equal to or less than the payload size of **Format 1_0** monitored in **common search space** in the same serving cell. If the quantity of information bits in **Format 2_2** is less than the payload size of format 1_0 monitored in **common search space** in the same serving cell, zeros shall be appended to **Format 2_2** until the payload size equals that of **Format 1_0** monitored in **common search space** in the same serving cell.

For a listing of **all** the **possible fields** for **each DCI Format**, click 6.9.1.

For more detail on **Format 2_2**, refer to clause 7.3.1.3.3 of TS 38.212.

### 6.9.8.4 DCI Format 2_3 messages

**Every DCI Format 2_3 message** has the **last 16 bits** of its **CRC masked** with **TPC-SRS-RNTI**.

**DCI Format 2_3 messages** are used for the transmission of a **group of TPC commands** for **SRS transmissions** by one or more **UEs**. Along with a **TPC** command, an **SRS request** may also be transmitted.

The following **fields** are included:

"***Block number 1***", "***Block number 2***",…, "***Block number B***".

Where the starting position of a **block** is determined by parameter `startingBitOfFormat2-3` or by `startingBitOfFormat2-3SUL-v1530`, both from `SRS-TPC-CommandConfig`.

If the **UE** is configured with parameter `srs-TPC-PDCCH-Group` from `SRS-CarrierSwitching`, with `CHOICE` of `typeA`, for an **Uplink** without **PUCCH** and **PUSCH** or an **Uplink** on which the **SRS power control** is **not** tied with **PUSCH power**

**control**, **only 1 block** is configured for the **UE** by higher layers, with the following **sub-fields** defined for the **block**:

- "***SRS request***" (0 or 2 bits). The presence of this field is according to the definition in Clause 11.4 of TS 38.213. If it is **present**, this field is interpreted as defined by Table 7.3.1.1.2-24.
- "***TPC command number 1***", "***TPC command number 2***", ..., "***TPC command number T***", where each **TPC command** applies to a respective **Uplink** carrier provided by parameter `cc-IndexInOneCC-Set` from `SRS-CC-SetIndex`.

If the **UE** is configured with parameter `srs-TPC-PDCCH-Group` from `SRS-CarrierSwitching`, with `CHOICE` of `typeB`, for an **Uplink** without **PUCCH** and **PUSCH** or an **Uplink** on which the **SRS power control** is not tied with **PUSCH power control**, one block or more blocks is configured for the **UE** by higher layers where each block applies to an **Uplink** carrier, with the following **sub-fields** defined for **each block**:

- "***SRS request***" (0 or 2 bits) The presence of this field is according to the definition in clause 11.4 of TS38.213. If it is **present**, this field is interpreted as defined by Table 7.3.1.1.2-24.
- "***TPC command***" (2 bits)

The quantity of information bits in **Format 2_3** shall be equal to or less than the payload size of **Format 1_0** monitored in **common search space** in the same serving cell. If the quantity of information bits in **Format 2_3** is less than the payload size of **Format 1_0** monitored in **common search space** in the same serving cell, zeros shall be appended to **Format 2_3** until the payload size equals that of **Format 1_0** monitored in **common search space** in the same serving cell.

For a listing of **all** the **possible fields** for **each DCI Format**, click 6.9.1.

For more detail on **Format 2_3**, refer to clause 7.3.1.3.4 of TS 38.212.

### 6.9.8.5 DCI Format 2_4 messages

**Every DCI Format 2_4 message** has the **last 16 bits** of its **CRC masked** with **CI-RNTI**.

**DCI Format 2_4 messages** are used for notifying the **PRB(s)** and **OFDM symbol(s)** where **UE cancels** the **corresponding Uplink transmission** from the **UE** according to clause 11.2A of TS 38.213.

The following **fields** are included:

"***Cancellation indication 1***", "***Cancellation indication 2***", …, "***Cancellation indication C***".

The size of **DCI Format 2_4** is configurable (up to **126 bits**) with `dci-PayloadSizeForCI-r16` from `UplinkCancellation-r16`, according to clause 11.2A of TS 38.213. The quantity of bits for each cancellation indication is configurable with `ci-PayloadSize-r16` from `CI-ConfigurationPerServingCell-r16`. For a **UE**, there is at most one cancellation indication for an **Uplink** carrier.

For a listing of **all** the **possible fields** for **each DCI Format**, click 6.9.1.

For more detail on **Format 2_4**, refer to clause 7.3.1.3.5 of TS 38.212.

#### 6.9.8.6 DCI Format 2_5 messages

**Every DCI Format 2_5 message** has the **last 16 bits** of its **CRC masked** with **AI-RNTI**.

**DCI Format 2_5 messages** are used for notifying the **availability of soft resources** as defined in clause 9.3.1 of TS 38.473.

The following **fields** are included:

"***Availability indicator 1***", "***Availability indicator 2***", …, "***Availability indicator A***".

The size of **DCI Format 2_5** is configurable by higher layers up to **128 bits**, according to clause 14 of TS 38.213.

For a listing of **all** the **possible fields** for **each DCI Format**, click 6.9.1.

For more detail on **Format 2_5**, refer to clause 7.3.1.3.6 of TS 38.212.

#### 6.9.8.7 DCI Format 2_6 messages

**Every DCI Format 2_6 message** has the **last 16 bits** of its **CRC masked** with **PS-RNTI**.

**DCI Format 2_6 messages** are used for notifying the **power saving information outside Discontinuous Reception (DRX) Active Time** for one or more **UEs**.

The following **fields** are included:

"***Block number 1***", "***Block number 2***",…, "***Block number B***".

where the starting position of a block is determined by the parameter `ps-PositionDCI-2-6` provided by higher layers for the **UE** configured with the block.

If the **UE** is configured with higher layer parameter `ps-RNTI` and `dci-Format2-6`, **only 1** "***Block number***" field is configured, and it includes the following **sub-fields**:

- ***Wake-up indication*** (1 bit)
- ***SCell dormancy indication***
  - If `dormancyGroupOutsideActiveTime` is **absent**, then it is **0 bits**.
  - Otherwise, it is a **1**, **2**, **3**, **4** or **5 bits bitmap** determined according to the quantity of different **DormancyGroupID(s)** provided by higher layer parameter `dormancyGroupOutsideActiveTime`, where each bit corresponds to one of the **SCell** group(s) configured by higher layer parameter `dormancyGroupOutsideActiveTime`, with **most** to **least** significant bits of the bitmap corresponding to the **first** to **last** configured **SCell** group in ascending order of `DormancyGroupID`.

The size of a **DCI Format 2_6** message is indicated by the higher layer parameter `sizeDCI-2-6`, according to clause 10.3 of TS 38.213.

For a listing of **all** the **possible fields** for **each DCI Format**, click 6.9.1.

For more detail on **Format 2_6**, refer to clause 7.3.1.3.7 of TS 38.212.

#### 6.9.8.8 DCI Format 2_7 messages

**Every DCI Format 2_7 message** has the **last 16 bits** of its **CRC masked** with **PEI-RNTI**.

**DCI Format 2_7 messages** are used for notifying the **paging early indication** and **TRS availability indication** for one or more **UEs**.

The following **fields** are included:

- "***Paging indication***" ($N_{PO}^{PEI}$ $N_{SG}^{PO}$ bit(s))

  where

  - $N_{PO}^{PEI}$ is the **quantity** of **paging occasions** configured by `po-NumPerPEI` as defined in clause 10.4A in TS 38.213.
  - $N_{SG}^{PO}$ is the **quantity** of **sub-groups** of a **paging occasion** configured by `subgroupsNumPerPO`.
  - **Each bit** in the **field** indicates **1 UE subgroup** of a **paging occasion**.

- "***TRS availability indication***" (1 to 6 bits)

  where the **quantity** of bits is **equal** to one plus the **highest value** of **all** the `indBitID`(s) provided by the `trs-ResourceSetConfig` if **present**; 0 bits otherwise.

The size of **DCI Format 2_7** is indicated by parameter `payloadSizeDCI-2-7`, according to clause 10.4A of TS 38.213. The quantity of information bits in **Format 2_7** shall be **equal** to, or **less** than, the **payload** size of **Format 2_7**. If the quantity of information bits in **Format 2_7** is **less** than the size of **Format 2_7**, the remaining bits are **reserved**.

For a listing of **all** the **possible fields** for **each DCI Format**, click 6.9.1.

For more detail on **Format 2_7**, refer to clause 7.3.1.3.8 of TS 38.212.

## 6.9.9 DCI Format 4_0, 4_1 and 4_2 messages

**Each DCI Format 4_0 message** signals **1 PDSCH assignment**, of **1** or **2 codeword(s)**, in **1** to **8 layers**, in **1 cell** and for **multicast/broadcast service** (**MBS**).

### 6.9.9.1 DCI Format 4_0 messages

**Every DCI Format 4_0 message** has the **last 16 bits** of its **CRC** masked **either** with **MCCH-RNTI** or with **G-RNTI** configured by an **instance** of `MBS-SessionInfo-r17`.

**Each DCI Format 4_0 message** signals **1 PDSCH assignment**, of **1 broadcast codeword**, in **1 layer**, in **1 cell**.

This **Format** does **not** have the **fields** used for **HARQ** (click 4.5.6) because **HARQ** is **not** used for **broadcast data**.

For a listing of **all** the **possible fields** for **each DCI Format**, click 6.9.1.

For more detail on **Format 4_0**, refer to clause 7.3.1.5.1 of TS 38.212.

### 6.9.9.2 DCI Format 4_1 messages

**Every DCI Format 4_1 message** has the **last 16 bits** of its **CRC** masked with **G-RNTI** for **multicast** or with **G-CS-RNTI** configured by an **instance** of `MBS-RNTI-SpecificConfig-r17`.

**Each DCI Format 4_1 message** signals **1 PDSCH assignment**, of **1 multicast codeword**, in **1 layer**, in **1 cell**.

For a listing of **all** the **possible fields** for **each DCI Format**, click 6.9.1.

For more detail on **Format 4_1**, refer to clause 7.3.1.5.2 of TS 38.212.

#### 6.9.9.3 DCI Format 4_2 messages

**Every DCI Format 4_2 message** has the **last 16 bits** of its **CRC** masked with **G-RNTI** for **multicast** or with **G-CS-RNTI** configured by an **instance** of `MBS-RNTI-SpecificConfig-r17`.

**Each DCI Format 4_2 message** signals **1 PDSCH assignment**, of **1** or **2 multicast codeword(s)**, in **1** to **8 layers**, in **1 cell**.

For a listing of **all** the **possible fields** for **each DCI Format**, click 6.9.1.

For more detail on **Format 4_2**, refer to clause 7.3.1.5.3 of TS 38.212.

### 6.9.10 Radio Network Temporary identifier (RNTI)

A **RNTI** is a **16-bit identifier** used to **mask** the **last 16 bits** of the **CRC** of **each Downlink Control Information message** (click 6.10.1). The formal name of **5G RNTIs** is "**New Radio RNTI**" or "**NR-RNTI**", but if no ambiguity exists in respect to previous generation **RNTIs**, its name is shortened to just "**RNTI**".

**Each RNTI** is **16 bits**, and **each active UE** needs **at least 1 individual RNTI** (the **C-RNTI**) in **each** of its **cells**. Consequently, **each cell** can contain **at most ~65,000 active UEs**.

#### 6.9.10.1 Types of RNTI

There are **individual RNTIs** (for **one UE**) and **group RNTIs** (for a **group** of **UEs**). There are several **types** of **RNTI** for different procedures and cases. There are **RNTIs** to differentiate/identify a connected **UE** in the **cell**, for a specific **radio channel**, for a **group** of **UEs** in case of **paging**, for a **group** of **UEs** for which **power control** is issued by the **NR node** or to transmit System Information Blocks (**SIBs**) to **all** the **UEs** in the **serving cell**.

The **SI-RNTI**, **P-RNTI**, **MCCH-RNTI** and **PEI-RNTI** are **group RNTIs** that have **fixed** values known to **all UEs**. Other **group RNTIs** are **INT-RNTI** and **SFI-RNTI**. Most of the other **RNTIs** are **individual RNTIs**.

The following table lists **all RNTI types** (from clause 7.1 of TS 38.321):

| RNTI type | RNTI usage |
|---|---|
| **AI-RNTI** | Availability indication RNTI. |
| **CG-SDT-CS-RNTI** | Configured grant Small Data Transmission Configured Scheduling RNTI. |
| **CI-RNTI** | Cancellation Indication RNTI. |
| **C-RNTI** | Cell RNTI. |
| **CS-RNTI** | Configured Scheduling RNTI. It is used for the activation of Uplink and Downlink unicast configured grants. |
| **G-CS-RNTI** | Group Configured Scheduling RNTI. It is used for the activation of Downlink multicast configured grants. |
| **G-RNTI** | Group RNTI. |
| **INT-RNTI** | Interruption RNTI. |
| **MCCH-RNTI** | Multicast Control Channel RNTI. |
| **MCS-C-RNTI** | Modulation and Coding Scheme Cell RNTI. |
| **MSGB-RNTI** | Message B RNTI. |
| **PEI-RNTI** | Paging Early Indication RNTI. |
| **P-RNTI** | Paging RNTI. |
| **PS-RNTI** | Power Saving RNTI. |

| RNTI type | RNTI usage |
|---|---|
| **RA-RNTI** | Random Access RNTI. |
| **Semi-Persistent Scheduling V-RNTI** | Semi-Persistent Scheduling for V2X RNTI. |
| **SFI-RNTI** | Slot Format Indication RNTI. |
| **SI-RNTI** | System Information RNTI. |
| **SL-CS-RNTI** | Sidelink Transmission (Configured Scheduled) RNTI. |
| **SL-RNTI** | Sidelink Transmission RNTI. (Dynamically Scheduled) |
| **SP-CSI-RNTI** | Semi-Persistent Channel State Information RNTI. |
| **TC-RNTI** | Temporary Cell RNTI. |
| **TPC-PUCCH-RNTI** | Transmit Power Control-PUCCH – RNTI. |
| **TPC-PUSCH-RNTI** | Transmit Power Control-PUSCH – RNTI. |
| **TPC-SRS-RNTI** | Transmit Power Control-Sounding Reference Symbols – RNTI. |

The following **relevant tables** are complementary to this table:

- Table with **all** the **search space types** with their **cell type**, their related **RNTI**(s), their related **DCI Formats** and their **combined function**, click 6.8.5.5.
- Table with **all** the **DCI message formats** with their related **RNTI**(s), and their **combined function**, click 6.9.
- Table with **all** the **RNTIs** with their related **DCI Formats**, their **mapping** to **channels**, and their **combined function**, click 6.9.10.2.

The following is table 7.1-1 of TS 38.321 that lists the **hexadecimal** and **decimal** value (or range of values) for each type of **RNTI**:

| RNTI | Value(s) [hexa-decimal] | Value(s) [decimal] |
|---|---|---|
| **N/A** | 0000 | 0000 |
| **RA-RNTI**, **MSGB-RNTI**, **TC-RNTI**, **C-RNTI**, **CI-RNTI**, **MCS-C-RNTI**, **CS-RNTI**, **TPC-PUCCH-RNTI**, **TPC-PUSCH-RNTI**, **TPC-SRS-RNTI**, **INT-RNTI**, **SFI-RNTI**, **SP-CSI-RNTI**, **PS-RNTI**, **SL-RNTI**, **SL-CS-RNTI SL**, **Semi-Persistent Scheduling V-RNTI**, **AI-RNTI**, **G-RNTI**, **G-CS-RNTI**, and **CG-SDT-CS-RNTI** | 0001–FFF2 | 1–65,522 |
| **Reserved** | FFF3–FFFB | 65,522–65,531 |
| **PEI-RNTI** | FFFC | 65,532 |
| **MCCH-RNTI** | FFFD | 65,533 |
| **P-RNTI** | FFFE | 65,534 |
| **SI-RNTI** | FFFF | 65,535 |

Do not confuse the "**I-RNTI value**" (click 5.8.2) with the **RNTIs**. In spite of its name, the "**I-RNTI value**" is **not** a **RNTI**.

Some of the **NR-RNTIs** are similar to the **LTE RNTIs**, but some **new NR-RNTIs** have been introduced to support **new 5G** functions and capabilities.

### 6.9.10.2 RNTIs, DCI Formats, usage and mapping to binary channels

The following is table 7.1-2 of TS 38.321 that lists **all** the **RNTI** types and their usage. It also shows, depending on their usage, their mapping to **Transport** and **Logical** binary channels. We have added the **DCI Formats** corresponding to **each RNTI**:

| RNTI Type | DCI Format(s) | Usage | Transport Channel | Logical Channel |
|---|---|---|---|---|
| **AI-RNTI** | 2_5 | Availability indication on the given cell for an IA-node | N/A | N/A |
| **C-RNTI** | 0_0, 0_1, 0_2 for PUSCH transmissions<br>1_0, 1_1, 1_2 for PDSCH transmissions | Dynamically scheduled unicast PUSCH transmission | UL-SCH | DCCH, DTCH |
| | | Dynamically scheduled unicast PDSCH transmission | DL-SCH | CCCH, DCCH, DTCH |
| | | Triggering of PDCCH Ordered random access | N/A | N/A |
| | | Dynamically scheduled PTP retransmission for initial PTM transmission for multicast MBS. | DL-SCH | MTCH |
| **CG-SDT-CS-RNTI** | 0_0, 0_1, 0_2 for PUSCH Tx<br>1_0, 1_1, 1_2 for PDSCH Tx | Dynamically scheduled unicast transmission (retransmission) | UL-SCH | CCCH, DCCH, DTCH |
| **CI-RNTI** | 2_4 | Cancellation indication in Uplink | N/A | N/A |
| **CS-RNTI** | 0_0, 0_1, 0_2 for PUSCH transmissions<br>1_0, 1_1, 1_2 for PDSCH transmissions | Semi-persistently scheduled PUSCH transmission (activation, reactivation and retransmission) | DL-SCH, UL-SCH | DCCH, DTCH |
| | | Semi-persistently scheduled unicast transmission (deactivation) | N/A | N/A |
| | | Semi-persistently scheduled unicast transmission (PTP retransmission for initial PTM transmission) | DL-SCH | MTCH |
| | | Semi-persistently scheduled unicast transmission (MBS SPS deactivation) | N/A | N/A |
| **G-RNTI** | 4_0, 4_1, 4_2 for PDSCH MBS Tx | Dynamically scheduled MBS PTM transmission | DL-SCH | MTCH |
| **G-CS-RNTI** | 4_1, 4_2 for PDSCH MBS transmissions | Semi-persistently scheduled multicast transmission (activation, reactivation and retransmission) | DL-SCH | MTCH |
| | | Semi-persistently scheduled multicast transmission (deactivation) | N/A | N/A |
| **INT-RNTI** | 2_1 | Indication pre-emption in Downlink | N/A | N/A |
| **MCCH-RNTI** | 4_0 for PDSCH broadcast | Dynamically scheduled MCCH signaling and MCCH change notification | DL-SCH | MCCH |
| **MCS-C-RNTI** | 0_0, 0_1, 0_2 | Dynamically scheduled unicast Uplink transmission | UL-SCH | DCCH, DTCH |
| | 1_0, 1_1, 1_2 | Dynamically scheduled unicast Downlink transmission | DL-SCH | DCCH, DTCH |
| **MSGB-RNTI** | 1_0 | RA Response for 2-step CFRA (MsgB) | DL-SCH | CCCH, DCCH, DTCH |
| **P-RNTI** | 1_0 | Paging and System Information change notification | PCH | PCCH |
| **PEI-RNTI** | 2_7 | Paging Early Indication | N/A | N/A |

| RNTI Type | DCI Format(s) | Usage | Transport Channel | Logical Channel |
|---|---|---|---|---|
| **PS-RNTI** | 2_6 | DCP to indicate whether to start `sl-drx-onDurationTimer-r17` for associated DRX cycle | N/A | N/A |
| **RA-RNTI** | 1_0 | RA Response for CBRA or 4-step CFRA. | DL-SCH | N/A |
| **SFI-RNTI** | 2_0 | Slot Format Indication on the given cell | N/A | N/A |
| **SI-RNTI** | 1_0 | Broadcast of System Information | DL-SCH | BCCH |
| **SL-RNTI** | 3_0 | Dynamically scheduled Sidelink transmission | SL-SCH | SCCH, STCH |
| **SL-CS-RNTI** | 3_0 | Semi-persistently scheduled Sidelink transmission (activation, reactivation and retransmission) | SL-SCH | SCCH, STCH |
| | | Semi-persistently scheduled Sidelink transmission (deactivation) | N/A | N/A |
| **SL Semi-Persistent Scheduling V-RNTI (NOTE 2)** | 3_1 | Semi-persistently scheduled Sidelink transmission for V2X Sidelink communication (activation, reactivation and retransmission) | SL-SCH | STCH |
| | | Semi-persistently scheduled Sidelink transmission for V2X Sidelink communication (deactivation) | N/A | N/A |
| **SP-CSI-RNTI** | 0_1 | Activation of Semi-persistent CSI reporting on PUSCH | N/A | N/A |
| **TC-RNTI** | 0_0 | Scheduled Transmission (Msg3) for 4-step CBRA or 2-step CBRA with fallback. | UL-SCH | CCCH, DCCH, DTCH |
| | 1_0 | Contention Resolution (Msg4 or MsgB) in 4-step or 2-step CBRA (no valid C-RNTI is available) | DL-SCH | CCCH, DCCH, DTCH |
| **TPC-PUCCH-RNTI** | 2_2 | PUCCH power control | N/A | N/A |
| **TPC-PUSCH-RNTI** | 2_2 | PUSCH power control | N/A | N/A |
| **TPC-SRS-RNTI** | 2_3 | SRS trigger and power control | N/A | N/A |
| NOTE 1: The usage of MCS-C-RNTI is equivalent to that of C-RNTI in MAC procedures (except for the C-RNTI MAC CE). | | | | |
| NOTE 2: The MAC entity uses SL Semi-Persistent Scheduling V-RNTI to control semi-persistently scheduled Sidelink transmission on SL-SCH for V2X Sidelink communication as specified in clause 5.14.1.1 of TS 36.321 [22]. | | | | |
| NOTE 3: The usage of CG-SDT-CS-RNTI is equivalent to that of CS-RNTI when there is an CG-SDT procedure ongoing. | | | | |

In this table, the term “Configured scheduled” that is used in TS 38.321 has been replaced by “Semi-persistently scheduled” for coherence with the terminology of most TS documents.

The following **relevant tables** are complementary to this table:

- Table with **all** the **search space types** with their **cell type**, their related **RNTI**(s), their related **DCI Formats** and their **combined function**, click 6.8.5.5.

- Table with **all** the **DCI message formats** with their related **RNTI**(s), and their **combined function**, click 6.9.
- Table with **all** the **RNTIs**, with a **short individual description**, click 6.9.10.1.

A brief description of a **few** relevant **RNTI types** follows:

- **Cell RNTI** (**C-RNTI**)

  It is a **cell-unique identification** used for identifying the **RRC Connection** and **scheduling** which is dedicated to **each UE**. The **NR node** assigns different **C-RNTI** values to different **UE** in the same **cell**. The **NR node** uses the **C-RNTI** of **each UE** to signal to the **UE** its **PDSCH** and **PUSCH assignments**, etc. **C-RNTI** is used by **NR node** to differentiate the **Uplink** transmissions (e.g., **PUSCH**, **PUCCH**) of **each UE** from the ones of **other UEs**. When **carrier aggregation** (**CA**, click 10.2) is configured, the **UE only** has **1 C-RNTI**, that is used in **all** its **serving cells**

- **Paging RNTI** (**P-RNTI**)

  It is used by the **UEs** for the reception of **paging**. It is also a **common RNTI** meaning that it is not allocated to any **UE** explicitly. **P-RNTI** is **65,534** (**0xFFFE**). The **paging message** is carried by **PCCH logical** channel which is mapped to **PCH transport** channel and to **PDSCH**. The **NR node** masks **PDCCH**'s **CRC** with **P-RNTI** for transmission of **PDSCH** that carries **paging information DCI Formats** which carry scheduling information for **paging**.

- **Random Access RNTI** (**RA-RNTI**)

  It is used in **contention-based random access procedures** (**CBRA**, click 9.1.2) and for the **4-step CFRA procedure**. In these **procedures**, the **RA-RNTI** is used by the **NR node** to **mask** the **last 16 bits** of the **CRC** of the **DCI message** that **schedules** the **PDSCH reception** of **Msg3** (click 9.3.4) by the **UE**. For the computation of **RA-RNTI**, click 9.1.7.

- **System Information RNTI** (**SI-RNTI**)

  It is used for **broadcasting system information**. It is a **common RNTI** meaning that, it is not allocated to any **UE** explicitly and is common to all **UEs** in the serving cell. **SI-RNTI** is **65,535** (**0xFFFF**). The only **SI-RNTI** is used to address **all system information messages**. **Broadcast** of **system information** uses **BCCH logical** channel, mapped to **DL-SCH transport** and to **PDSCH**. The **UEs** should know the scheduling information for **PDSCH** which is carrying **system information**. The required scheduling information is contained in Downlink Control Information (**DCI**) **messages** whose **CRC** is masked with **SI-RNTI**. The **UE** starts decoding **PDCCH** masked with **SI-RNTI** at the **start** of **SI Window** (for the **concerned SI message**) until the end of the **SI window**, or until the **SI message** was received excluding the following subframes.

- **Temporary C-RNTI** (**TC-RNTI**)

  It is used during a **random access procedure** (click 9). The **NR node** includes a **TC-RNTI** in the **random access response** (**Msg2**, click 9.3.3 or **MsgB**, click 9.3.7) **sent** to the **UE**.

  ***If*** the **UE** does **not have** a **C-RNTI**,

***then***, the **UE** uses the **TC-RNTI** to **demask** the **last 16 bits** of the **CRC** of the **DCI message** corresponding to **Msg4** (click 9.3.5) and, when the **contention resolution** (**Msg4** or **MsgB**) is **received**, the **UE promotes** its **TC-RNTI** to its **C-RNTI**.

***Else***, the **TC-RNTI** is **ignored**.

- **Transmit Power Control RNTIs** (**TPC RNTIs**)

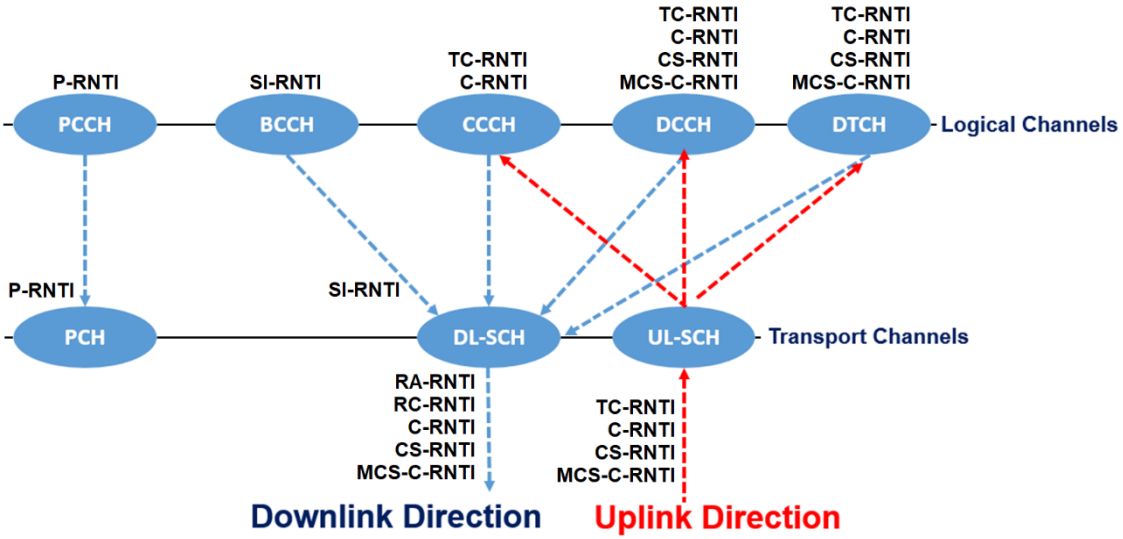
  They are used for **Uplink** power control purpose. There are 3 types of **TPC RNTIs**, namely: **TPC-PUSCH-RNTI**, **TPC-PUCCH-RNTI** and **TPC-SRS-RNTI**. Normally, the **TPC-RNTIs** is/are assigned to a group of **UEs**. The **NR node** may configure the **UE** with **TPC-PUSCH-RNTI**, **TPC-PUCCH-RNTI** and **TPC-SRS-RNTI** via higher layer signaling (**RRC**).

For a listing of **search space types** with their **cell type**, **RNTI(s)**, **DCI Formats** and **usage**, click 6.8.5.5.

For a listing of **DCI Formats** with their **usage** and their associated **RNTI type(s)**, click 6.9.

For a listing of **RNTIs** with their **individual usage**, click 6.9.10.1.

The following figure shows **some** of the **RNTI types** and their association with Logical and Transport channels:




Some of the **NR-RNTIs** are similar to the **LTE RNTIs**, but some **new NR-RNTIs** have been introduced to support **new 5G** functions and capabilities.

Source web pages:

- https://www.techplayon.com/5g-nr-radio-network-temporary-identifier-rnti/
- http://howltestuffworks.blogspot.com/2019/09/rntis-in-5g-nr.html
- https://www.sharetechnote.com/html/5G/5G_RNTI.html

## 6.10 PDCCH payload processing

The **PDCCH subcarrier spacing** of **each PDCCH transmission** is the **same** as the one of the **CORESET associated** to the **search space** of the **PDCCH transmission**. The

**subcarrier spacing** of **each non-zero CORESET** is the same as the one of the **bandwidth part** where it is contained.

During **initial access**, the **subcarrier spacing** for **PDCCH** in the **Downlink initial bandwidth part** is the same as the one of **CORESET#0** (click 6.8.8.1).

The **PDCCH cyclic prefix** (click 2.3.2) is the **same** as the one of the **CORESET associated** to the **search space** of the **PDCCH transmission**. The **cyclic prefix** of **each non-zero CORESET** is the **same** as the one of the **bandwidth part** where it is contained. The **cyclic prefix** of **CORESET#0** is explained in 6.8.8.1.

For the **PDCCH Tx payload processing**, click 6.10.1.

For the **PDCCH Rx payload processing**, click 6.10.2.

For the **DCI messages** sent through **PDCCH**, click 6.9.

For the **NR node functional blocks**, the **NR node physical functions**, the **payload processing** of the **physical binary channels**, the **protocol architecture** of the **New Radio** and the **data flow** through the **protocol architecture**, click 4.3.

### 6.10.1 PDCCH Tx payload processing

The **control system directly on top of PDCCH** (there is **no** higher layer transport channel on top) produces the **Downlink Control Information** (**DCI**) **messages**, and hands them to **PDCCH** for their processing and delivery.

**Each DCI message** to be **sent** is processed by **PDCCH** in the following way:

1. **Size align** the **DCI** message by **padding** or **truncation** (clause 7.3.1 of TS 38.212).
2. **Trail** a **24-bit CRC** for error detection.
3. **Mask** (XOR) the **16 least significant** bits of the **CRC** with the corresponding **RNTI** (click 6.9.10 and clause 7.3.2 of TS 38.212).
4. **Polar-encode** (**FEC**, click 11.2.3) (clause 7.3.3 of TS 38.212) and **rate-match** (clause 7.3.4 of TS 38.212) using the corresponding **aggregation level** (click 6.8.1.2).
5. **Scramble** (clause 7.3.2.3 of TS 38.211).
6. **Map** the **RBs** of the **CCEs** to **PRBs** (click 6.8.2.2).
7. **Modulate with QPSK** (click 11.2.1 and clause 7.3.2.4 of TS 38.211).

**Size alignment** (by **padding** or **truncation**), of **DCI messages** is done to achieve efficient utilization of resources, maintain **compatibility** with the **fixed-size polar code**, and minimize interference, ultimately **improving** the **reliability** and **performance** of **DCI message transmission**.

**DCI messages** are **hard real-time**, because their information has to be used in the **same slot** (or in the next one) where they have been received: **otherwise**, they are **useless**. Therefore, they **cannot** be **retransmitted** and they **cannot** be protected by **HARQ**. However, **PDCCH protects** the **DCI messages** with:

- **Very robust modulation** (**QPSK**).
- **Polar code** forward error correction (**FEC**).

- **Careful CORESET frequency locations** (coordinated between **adjacent cells**) to **minimize cross-interference**.

For the **NR node functional blocks**, the **NR node physical functions**, the **payload processing** of the **physical binary channels**, the **protocol architecture** of the **New Radio** and the **data flow** through the **protocol architecture**, click 4.3.

For more detail on **PDCCH payload processing**, refer to TS 38.211 and TS 38.212.

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_PDCCH.html#PDCCH_Transport_Process
- https://www.sharetechnote.com/html/5G/5G_DCI.html
- https://www.rfwireless-world.com/5G/5G-NR-PDCCH.html

## 6.10.2 PDCCH Rx payload processing

The **processing** performed by **each UE** to **receive its DCI messages** through **PDCCH** is the following.

For **every search space** (click 6.8.5) of its **currently active bandwidth part**, and for **every monitoring occasion** (click 6.8.3.2) in that **search space**, do:

1. **Demodulate** with **QPSK** (click and clause 7.3.2.4 of TS 38.211) **all** the **used payload resource elements** within the **frequency resources** and **symbol intervals configured** in the **CORESET** that is **associated** to the corresponding **search space**.
2. **De-map** the received **PRBs** to the **RBs** of **CCEs** (click 6.8.2.2).
3. **De-scramble** (clause 7.3.2.3 of TS 38.211) the **bit sequence** of **each** and **every** of the **PDCCH candidates** (click 6.8.2.3), using **all** the corresponding **aggregation level**(**s**) (click 6.8.1.2).
4. **Polar-decode** (**FEC**, click 11.2.3) (clause 7.3.3 of TS 38.212) and **rate-dematch** (clause 7.3.4 of TS 38.212) **each** and **every** of the **bit sequences** resulting from **stage 3**.
5. **De-mask** (XOR) the **16** least significant bits of the **CRC** of **each** and **every** of the **bit sequences** resulting from **stage 4** with **each** and **every** of the possibly applicable **RNTIs** (click 6.9.10 and clause 7.3.2 of TS 38.212).
6. **Verify** the **24-bit CRC** for **each** and **every** pair of **received unmasked CRC** resulting from **stage 5** and **discard all** the **cases** where the **CRC** is **wrong**.
7. **Remove** the **CRC** and **deliver upwards all** the **pairs** of **bit sequence** plus **RNTI** from **stage 6**.

**Each case** resulting from **stage 7** should be a **DCI message** that was **masked** with the corresponding **RNTI**.

Notice that the **above** procedure is **equivalent** to the **blind decoding** procedure presented in section 6.8.4.

For the **NR node functional blocks**, the **NR node physical functions**, the **payload processing** of the **physical binary channels**, the **protocol architecture** of the **New Radio** and the **data flow** through the **protocol architecture**, click 4.3.

## 6.11 PDCCH DM-RS contents

**PDCCH DM-RS** are the **DM-RS** (click 4.11.3.2.1) within **PDCCH**. They are characterized by their **scheduling** within the **PDCCH REs** (click 6.8.4) and by their **contents**, that are described in this section

The **UE** shall assume that the **PDCCH DM-RS** sequence "$r_{si}(m)$" is mapped to **resource elements** in **port** "$(sc, si, ap)_{,\mu}$" according to:

$$a_{sc,si}^{(ap,\mu)} = \beta_{DMRS}^{PDCCH} r_{si}(3n+sc')$$

where:

$$sc = n N_{sc}^{RB} + 4 sc' + 1$$

$$sc' = 0, 1, 2$$

$$n = 0, 1, \ldots$$

The reference point for "$sc$=0" is:

- For **CORESET#0** during **initial access**, "$sc$=0" is **subcarrier 0** of the lowest-numbered **RB** in **CORESET#0**.
- For **non-zero CORESETs**, "$sc$=0" is **subcarrier 0** in **common resource block 0**.

The variable "$si$" is the **symbol interval index** within the **slot**.

The **antenna port** "$ap$ = **2000**".

A **UE** not attempting to detect a **PDCCH** in a **CORESET** shall not make any assumptions on the presence or absence of **DM-RS** in the **CORESET**.

In absence of **CSI-RS** configuration, and unless otherwise configured, the **UE** may assume **PDCCH DM-RS** and **SSB** to be **quasi co-located** with respect to **Doppler shift**, (click 12.1.8), **Doppler spread** (click 12.1.8), **average delay**, **delay spread** (click 12.2.3) and, when applicable, **spatial Rx parameters**.

For more detail on **PDCCH DM-RS scheduling**, refer to clause 7.4.1.3.2 of TS 38.211.

## 6.12 PDCCH parameters

The **UE**'s **Physical Downlink Control Channel** (**PDCCH**, click 6.7) is configured with:

- **Common** parameters that are **broadcast** in `MIB`, as described in section 6.12.1.
- **Common** parameters that are **broadcast** (or **unicast**) within `SIB1`, as described in section 6.12.2.
- **Common** and **dedicated parameters** that are **unicast** within the **Non-SI parameter group**, as described in section 6.12.3.

Possibly existing **PDDCH parameters** within **other SIBs** and **posSIBs** (click 5.3.5) are **not addressed** in this book.

For an overall explanation on **UE configuration parameters** click 5. For a brief overview of **ASN.1**, click 5.9.

### 6.12.1 PDCCH parameters within MIB

**PDCCH** is configured with **common parameters** that are **broadcast** in **MIB** (click 5.3.3), which is sent within **SSB** (click 6.1), in the following case:

- In **the primary cell**, **during cell selection**.

For **PDCCH** parameters within **MIB**, the **most relevant related data type** is:

- **PDCCH-ConfigSIB1** (click 8.12.3): common parameters

The following **data types**, whose **only parents** are **PDCCH-ConfigSIB1** (click 8.12.3) and **PDCCH-ConfigCommon** (click 6.12.4), are also relevant:

- **ControlResourceSetZero** (click 8.12.3): common parameters
- **SearchSpaceZero** (click 6.4.2): common parameters

The **ancestor tree** (click 5.9.8) of **PDCCH-ConfigSIB1** up to the **BCCH-BCH-Message class** is:

```
                                BCCH-BCH-Message (click 5.8.5.1)
message                           BCCH-BCH-MessageType (click 5.8.5.1)
  mib                               MIB (click 8.12.3)
    pdcch-ConfigSIB1                  PDCCH-ConfigSIB1 (click 8.12.3)
      controlResourceSetZero            ControlResourceSetZero (click 8.12.3)
      searchSpaceZero                   SearchSpaceZero (click 8.12.3)
```

### 6.12.2 PDCCH parameters within SIB1

**PDCCH** is configured with **common parameters** within **SIB1** (click 5.3.4) in the following case:

- In **the primary cell**, **during initial access**, but **after cell selection**.

For **PDCCH** parameters within **SIB1**, the **most relevant related data type** is:

- **PDCCH-ConfigCommon** (click 6.12.4): common parameters

The following **data types**, whose **only parents** are **PDCCH-ConfigSIB1** (click 8.12.3) and **PDCCH-ConfigCommon** (click 6.12.4), are also relevant:

- **ControlResourceSetZero** (click 8.12.3): common parameters
- **SearchSpaceZero** (click 8.12.3): common parameters

The following **data types**, **children** of **PDCCH-ConfigCommon** (click 6.12.4) and/or of **PDCCH-Config** (click 6.12.5), are also relevant:

- **ControlResourceSet** (click 6.12.6): instantiated as common
- **ControlResourceSetId** (click 6.12.7): instantiated as common
- **SearchSpace** (click 6.12.7): instantiated as common
- **SearchSpaceId** (click 6.12.9): instantiated as common

The **ancestor tree** (click 5.9.8) of this **most relevant related data type** up to `SIB1` is:

```
                                  SIB1 (click 5.7.4)
servingCellConfigCommon             ServingCellConfigCommonSIB (0,1) (click 3.9.3)
  downlinkConfigCommon                DownlinkConfigCommonSIB (click 2.2.6.4)
    initialDownlinkBWP                  BWP-DownlinkCommon (click 2.2.8.6)
      pdcch-ConfigCommon                  PDCCH-ConfigCommon (click 6.12.3)
```

For the **ancestor trees** of `SIB1` **up to** the **top**, click 5.3.4.

## 6.12.3 PDCCH parameters within the Non-SI parameter group

**PDCCH** is configured with **Common** and **dedicated** parameters that are **unicast** within the **non-SI parameter group** (click 5.3.6) in the following cases:

- In **the primary-secondary cell** and in **secondary cells**.
- In **the primary cell**, if configuration changes are done **after initial access**.

The **most relevant related data types** are:

- `PDCCH-ConfigCommon` (click 6.12.4): common parameters
- `PDCCH-Config` (click 6.12.5): dedicated parameters
- `PDCCH-ServingCellConfig` (click 6.12.10): dedicated parameters

The following **data types**, whose **only parents** are `PDCCH-ConfigSIB1` (click 8.12.3.1) and `PDCCH-ConfigCommon` (click 6.12.4), are also relevant:

- `ControlResourceSetZero` (click 8.12.3.1): common parameters
- `SearchSpaceZero` (click 8.12.3.1): common parameters

The following **data types**, **children** of `PDCCH-ConfigCommon` (click 6.12.4) and/or of `PDCCH-Config` (click 6.12.5), are also relevant:

- `ControlResourceSet` (click 6.12.6): instantiated as dedicated
- `ControlResourceSetId` (click 6.12.7): instantiated as dedicated
- `SearchSpace` (click 6.12.8): instantiated as dedicated
- `SearchSpaceExt-r16` (not explained in this book): instantiated as dedicated
- `SearchSpaceExt-v1700` (not explained in this book): dedicated parameters
- `SearchSpaceId` (click 6.12.9): instantiated as dedicated

For **PDCCH** parameters within the **Non-SI parameter group**, we show the following **2 ancestor trees**.

**First**, the **ancestor tree** (click 5.9.8) of the **most relevant related data types** (see above), up to `CellGroupConfig` is:

```
                                CellGroupConfig (click 3.7.3)
spCellConfig                      SpCellConfig (click 3.9.5)
  reconfigurationWithSync           ReconfigurationWithSync (0,1) (click 3.9.8)
    spCellConfigCommon                ServingCellConfigCommon (click 3.9.6)
      downlinkConfigCommon              DownlinkConfigCommon (click 2.2.6.8)
        initialDownlinkBWP                BWP-DownlinkCommon (click 2.2.8.6)
          pdcch-ConfigCommon                PDCCH-ConfigCommon (click 6.12.3)
  spCellConfigDedicated             ServingCellConfig (click 3.9.7)
    begin of DownlinkConfig parameter set (click 2.2.8.3)
    initialDownlinkBWP                BWP-DownlinkDedicated (click 2.2.8.7)
      pdcch-Config                      PDCCH-Config (0,1) (click 6.12.5)
      sl-PDCCH-Config-r16               PDCCH-Config (0,1) (click 6.12.5)
      sl-V2X-PDCCH-Config-r16           PDCCH-Config (0,1) (click 6.12.5)
      cfr-ConfigMulticast-r17           CFR-ConfigMulticast-r17 (0,1)
        pdcch-ConfigMulticast-r17         PDCCH-Config (0,1) (click 6.12.5)
    downlinkBWP-ToReleaseList       BWP-Id (0..4) (click 2.2.8.13)
    downlinkBWP-ToAddModList        BWP-Downlink (0..4) (click 2.2.8.5)
      bwp-Common                        BWP-DownlinkCommon (click 2.2.8.6)
        pdcch-ConfigCommon                PDCCH-ConfigCommon (click 6.12.3)
      bwp-Dedicated                     BWP-DownlinkDedicated (click 2.2.8.7)
        pdcch-Config                      PDCCH-Config (0,1) (click 6.12.5)
        sl-PDCCH-Config-r16               PDCCH-Config (0,1) (click 6.12.5)
        sl-V2X-PDCCH-Config-r16           PDCCH-Config (0,1) (click 6.12.5)
        cfr-ConfigMulticast-r17           CFR-ConfigMulticast-r17 (0,1)
          pdcch-ConfigMulticast-r17         PDCCH-Config (0,1) (click 6.12.5)
    pdcch-ServingCellConfig         PDCCH-ServingCellConfig (0,1) (click 6.12.10)
    end of DownlinkConfig parameter set
sCellToAddModList                 SCellConfig (0..maxNrofSCells) (click 3.9.4)
  sCellConfigCommon                 ServingCellConfigCommon (0,1) (click 3.9.6)
    downlinkConfigCommon              DownlinkConfigCommon (click 2.2.6.8)
      initialDownlinkBWP                BWP-DownlinkCommon (click 2.2.8.6)
        pdcch-ConfigCommon                PDCCH-ConfigCommon (click 6.12.3)
  sCellConfigDedicated              ServingCellConfig (click 3.9.7)
    begin of DownlinkConfig parameter set (click 2.2.8.3)
    initialDownlinkBWP                BWP-DownlinkDedicated (click 2.2.8.7)
      pdcch-Config                      PDCCH-Config (0,1) (click 6.12.5)
      sl-PDCCH-Config-r16               PDCCH-Config (0,1) (click 6.12.5)
      sl-V2X-PDCCH-Config-r16           PDCCH-Config (0,1) (click 6.12.5)
      cfr-ConfigMulticast-r17           CFR-ConfigMulticast-r17 (0,1)
        pdcch-ConfigMulticast-r17         PDCCH-Config (0,1) (click 6.12.5)
    downlinkBWP-ToAddModList        BWP-Downlink (0..maxNrofBWPs) (click 2.2.8.5)
      bwp-Common                        BWP-DownlinkCommon (click 2.2.8.6)
        pdcch-ConfigCommon                PDCCH-ConfigCommon (click 6.12.3)
      bwp-Dedicated                     BWP-DownlinkDedicated (click 2.2.8.7)
        pdcch-Config                      PDCCH-Config (0,1) (click 6.12.5)
        sl-PDCCH-Config-r16               PDCCH-Config (0,1) (click 6.12.5)
        sl-V2X-PDCCH-Config-r16           PDCCH-Config (0,1) (click 6.12.5)
        cfr-ConfigMulticast-r17           CFR-ConfigMulticast-r17 (0,1)
          pdcch-ConfigMulticast-r17         PDCCH-Config (0,1) (click 6.12.5)
    pdcch-ServingCellConfig         PDCCH-ServingCellConfig (0,1) (click 6.12.10)
    end of DownlinkConfig parameter set
```

Some **relevant remarks** on this **first ancestor tree** are:

- For the **ancestor trees** of `CellGroupConfig`, **up to** some **RRC NR messages** click 3.7.1 and **up to** some **RRC inter-node messages** click 3.7.2.
- The constant `maxNrofSCells` is `31`, which is the **maximum quantity** of **UE**'s **secondary cells** (i.e., **other** than the **special cells**) in **each cell group**

- The constant **maxNrofBWPs** is **4**, which is the **maximum quantity** of **UE**'s **additional bandwidth parts** (i.e., **other** than the **initial bandwidth part**).

**Second**, the **additional ancestor tree** (click 5.9.8) of **PDCCH-Config**, **up to** the **top**, is:

```
                                                DL-DCCH-Message (click 5.8.5.4)
message                                           DL-DCCH-MessageType (click 5.8.5.4)
  c1                                                CHOICE
    rrcRelease                                        RRCRelease (click 5.8.5.4)
      rrcRelease                                        RRCRelease-IEs
        suspendConfig                                     SuspendConfig (0,1)
          sdt-Config-r17                                    SDT-Config-r17 (0,1)
            sdt-MAC-PHY-CG-Config-r17                         SDT-CG-Config-r17 (0,1)
                                                                OCTET STRING (CONTAINING
                                                                SDT-MAC-PHY-CG-Config-r17)
                cg-SDT-ConfigInitialBWP-DL-r17 BWP-DownlinkDedicatedSDT-r17
                  pdcch-Config-r17                  PDCCH-Config (0,1) (6.12.5)
```

Some **relevant remarks** on this **second ancestor tree** are:

- This **PDCCH-Config** instance is contained in **RRCRelease message**, in addition to the ones contained in **RRCReconfiguration**, **RRCResume** and **RRCSetup** (click 5.8.5.11) **messages** from the **first ancestor tree**.
- This **PDCCH-Config** instance is devoted to **small data transmission** (**SDT**).

### 6.12.4 PDCCH-ConfigCommon data type

It can be **both broadcast** (or **unicast**) within **SIB1** (click 5.3.4) and **unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to SIB1**, click 6.12.2.

For its **ancestor tree up to CellGroupConfig** (click 3.7.3), click 6.12.3.

**All** its parameters are **common** (click 5.1.3).

**Each** of its **instances** provides the **PDCCH common configuration parameters**.

**Each** of its **instances** configures the **UE** with **most PDCCH elements**, of its corresponding **bandwidth part** (**initial** or **additional**), of **common configuration**, of its corresponding **cell** (**special** or **secondary**), of its corresponding **cell group** (**master** or **secondary**). The "**corresponding**" elements are determined by the **specific place**, within the overall **ASN.1 specification**, where this **data type** has been **instantiated**.

#### 6.12.4.1 PDCCH-ConfigCommon parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- **commonControlResourceSet** **ControlResourceSet** (click 6.12.6): optional, common

  It is **one common CORESET** (additional to **CORESET#0**) which **may be** configured and **used** for **any common** or **UE-specific search space**.

  When this parameter is **broadcast** in **SIB1**, it **is required** that the **bandwidth** of this **CORESET** be **completely within** the **bandwidth** of **CORESET#0**. As an exception, this is **not required** when the **RedCap-specific Downlink initial bandwidth part** does **not contain** the **entire CORESET#0**.

- `commonSearchSpaceList SEQUENCE (SIZE(1..4)) OF SearchSpace` (click 6.12.8): optional, common
- `commonSearchSpaceListExt-r16 SEQUENCE (SIZE(1..4)) OF SearchSpaceExt-r16`: optional, common
- `commonSearchSpaceListExt2-r17 SEQUENCE (SIZE(1..4)) OF SearchSpaceExt-v1700` (optional, common)

  **Each** of them is a **list** of **non-zero common search spaces**. If the **NR node** configures **any** of these parameters, it **replaces** any previous list, i.e. **all** the **entries** of the list are **replaced** and **each search space entry** is considered to be **newly created** and the **conditions** and "`Need codes`" for **setup** of the entry **apply**.

  If the network includes `commonSearchSpaceListExt-r16` or `commonSearchSpaceListExt2-r17`, it includes the **same quantity of entries**, and listed in the **same order**, as in `commonSearchSpaceList`.

- `controlResourceSetZero ControlResourceSetZero` (click 8.12.3): optional, common
- `pagingSearchSpace SearchSpaceId` (click 6.12.9): optional, common

  It is the **identifier** of the **search space** for **paging** (refer to clause 10.1 of TS 38.213). If it is **absent**, the **UE** does **not** receive **paging** in this **bandwidth part** (refer to clause 10 of TS 38.213).

  This parameter is **absent** for the **RedCap-specific Downlink initial bandwidth part**, if it does **not** include the **cell-defining SSB** and the entire **CORESET#0**. In that case, a **RedCap UE** in **RRC_IDLE** or **RRC_INACTIVE** (click 5.8.2) while **small data transmission** (**SDT**) procedure is **not** ongoing, shall monitor **paging** in the **Downlink initial bandwidth part** that includes **CORESET#0**.

- `ra-SearchSpace SearchSpaceId` (click 6.12.9): optional, common

  It is the **identifier** of the **search space** for **random access procedure** (refer to clause 10.1 of TS 38.213). If it is **absent**, the **UE** does **not** receive **random access response** (**RAR**) in this **bandwidth part**. This parameter is **mandatory** in **all** the **UE**'s **Downlink bandwidth part**(**s**) if the conditions described in clause 5.15 of TS 38.321 are met.

- `searchSpaceOtherSystemInformation SearchSpaceId` (click 6.12.9): optional, common

  It is the **identifier** of the **search space** for **other system information** (see clause 10.1 of TS 38.213). If it is **absent**, the **UE** does **not** receive **other system information** in this **bandwidth part**.

  This parameter is absent for the **RedCap-specific Downlink initial bandwidth part**, if it does **not** include the **cell-defining SSB** and the entire **CORESET#0**. In that case, a **RedCap UE** in **RRC_IDLE** or **RRC_INACTIVE** (click 5.8.2) shall monitor **PDCCH** to receive **other system information** using `searchSpaceOtherSystemInformation` in the **Downlink initial bandwidth part** that includes the **cell-defining SSB** and the entire **CORESET#0**.

- **searchSpaceSIB1** **SearchSpaceId** (click 6.12.9): optional, common

  It is the **identifier** of the **search space** for reception of **SIB1** (click 5.3.4). If it is **absent**, the **UE** does **not** receive **SIB1** in this **bandwidth part** (refer to clause 10 of TS 38.213). In the **UE**'s **Downlink initial bandwidth part** of its **PCell**, the network sets this parameter to **0**.

  This parameter is absent for the **RedCap-specific Downlink initial bandwidth part**, if it does **not** include the **cell-defining SSB** and the entire **CORESET#0**. In that case, a **RedCap UE** in **RRC_IDLE** or **RRC_INACTIVE** (click 5.8.2) shall monitor **PDCCH** to receive **SIB1** using **searchSpaceSIB1** in the **Downlink initial bandwidth part** that includes the **cell-defining SSB** and the entire **CORESET#0**.

- **searchSpaceZero** **SearchSpaceZero** (click 8.12.3): optional, common

  These two parameters have the **same data type** and **same semantics** as the respective **same-name parameters** from **PDCCH-ConfigSIB1** (click 8.12.3.1), which are **broadcast** within **MIB** (click 5.3.3).

For more detail on these parameters, refer to the table "*PDCCH-ConfigCommon field descriptions*" of TS 38.331.

### 6.12.4.2 PDCCH-ConfigCommon ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
PDCCH-ConfigCommon ::= SEQUENCE {
  controlResourceSetZero ControlResourceSetZero OPTIONAL,
                                                    -- Cond InitialBWP-Only
  commonControlResourceSet ControlResourceSet OPTIONAL,          -- Need R
  searchSpaceZero SearchSpaceZero OPTIONAL,         -- Cond InitialBWP-Only
  commonSearchSpaceList SEQUENCE (SIZE(1..4)) OF SearchSpace OPTIONAL,
                                                                 -- Need R
  searchSpaceSIB1   SearchSpaceId OPTIONAL,                      -- Need S
  searchSpaceOtherSystemInformation SearchSpaceId OPTIONAL,      -- Need S
  pagingSearchSpace SearchSpaceId OPTIONAL,                      -- Need S
  ra-SearchSpace    SearchSpaceId OPTIONAL,                      -- Need S
  ...,
  [[
  firstPDCCH-MonitoringOccasionOfPO CHOICE {
    sCS15KHZoneT SEQUENCE (SIZE (1..maxPO-perPF)) OF INTEGER (0..139),
    sCS30KHZoneT-SCS15KHZhalfT
                 SEQUENCE (SIZE (1..maxPO-perPF)) OF INTEGER (0..279),
    sCS60KHZoneT-SCS30KHZhalfT-SCS15KHZquarterT
                 SEQUENCE (SIZE (1..maxPO-perPF)) OF INTEGER (0..559),
    sCS120KHZoneT-SCS60KHZhalfT-SCS30KHZquarterT-SCS15KHZoneEighthT
                 SEQUENCE (SIZE (1..maxPO-perPF)) OF INTEGER (0..1119),
    sCS120KHZhalfT-SCS60KHZquarterT-SCS30KHZoneEighthT-SCS15KHZoneSixteenthT
                 SEQUENCE (SIZE (1..maxPO-perPF)) OF INTEGER (0..2239),
    sCS120KHZquarterT-SCS60KHZoneEighthT-SCS30KHZoneSixteenthT
                 SEQUENCE (SIZE (1..maxPO-perPF)) OF INTEGER (0..4479),
    sCS120KHZoneEighthT-SCS60KHZoneSixteenthT
                 SEQUENCE (SIZE (1..maxPO-perPF)) OF INTEGER (0..8959),
    sCS120KHZoneSixteenthT
                 SEQUENCE (SIZE (1..maxPO-perPF)) OF INTEGER (0..17919)
  } OPTIONAL                                                  -- Cond OtherBWP
  ]],
  [[
  commonSearchSpaceListExt-r16
              SEQUENCE(SIZE(1..4)) OF SearchSpaceExt-r16 OPTIONAL -- Need R
  ]],
```

```
    [[
    sdt-SearchSpace-r17 CHOICE {
      newSearchSpace SearchSpace,
      existingSearchSpace SearchSpaceId
    } OPTIONAL,                                                    -- Need R
    searchSpaceMCCH-r17 SearchSpaceId OPTIONAL,                    -- Need R
    searchSpaceMTCH-r17 SearchSpaceId OPTIONAL,                    -- Need S
    commonSearchSpaceListExt2-r17
              SEQUENCE (SIZE(1..4)) OF SearchSpaceExt-v1700 OPTIONAL, -- Need R
    firstPDCCH-MonitoringOccasionOfPO-v1710 CHOICE {
      sCS480KHZoneEighthT
              SEQUENCE (SIZE (1..maxPO-perPF)) OF INTEGER (0..35839),
      sCS480KHZoneSixteenthT
              SEQUENCE (SIZE (1..maxPO-perPF)) OF INTEGER (0..71679)
    } OPTIONAL,                                                    -- Need R
    pei-ConfigBWP-r17 SEQUENCE {
    pei-SearchSpace-r17 SearchSpaceId,
    firstPDCCH-MonitoringOccasionOfPEI-O-r17  CHOICE {
    sCS15KHZoneT SEQUENCE (SIZE (1..maxPEI-perPF-r17)) OF INTEGER (0..139),
    sCS30KHZoneT-SCS15KHZhalfT
                 SEQUENCE (SIZE (1..maxPEI-perPF-r17)) OF INTEGER (0..279),
    sCS60KHZoneT-SCS30KHZhalfT-SCS15KHZquarterT
                 SEQUENCE (SIZE (1..maxPEI-perPF-r17)) OF INTEGER (0..559),
    sCS120KHZoneT-SCS60KHZhalfT-SCS30KHZquarterT-SCS15KHZoneEighthT
                 SEQUENCE (SIZE (1..maxPEI-perPF-r17)) OF INTEGER (0..1119),
    sCS120KHZhalfT-SCS60KHZquarterT-SCS30KHZoneEighthT-SCS15KHZoneSixteenthT
                 SEQUENCE (SIZE (1..maxPEI-perPF-r17)) OF INTEGER (0..2239),
    sCS480KHZoneT-SCS120KHZquarterT-SCS60KHZoneEighthT-SCS30KHZoneSixteenthT
                 SEQUENCE (SIZE (1..maxPEI-perPF-r17)) OF INTEGER (0..4479),
    sCS480KHZhalfT-SCS120KHZoneEighthT-SCS60KHZoneSixteenthT
                 SEQUENCE (SIZE (1..maxPEI-perPF-r17)) OF INTEGER (0..8959),
    sCS480KHZquarterT-SCS120KHZoneSixteenthT
                 SEQUENCE (SIZE (1..maxPEI-perPF-r17)) OF INTEGER (0..17919),
    sCS480KHZoneEighthT
                 SEQUENCE (SIZE (1..maxPEI-perPF-r17)) OF INTEGER (0..35839),
    sCS480KHZoneSixteenthT
                 SEQUENCE (SIZE (1..maxPEI-perPF-r17)) OF INTEGER (0..71679)
    }
    }  OPTIONAL                                          -- Cond InitialBWP-Paging
    ]],
    [[
    followUnifiedTCI-State-v1720 ENUMERATED {enabled}  OPTIONAL      -- Need R
    ]]
  }
```

Where `maxPO-perPF` is `4`, which is the **maximum quantity** of **paging occasions per paging frame**.

### 6.12.5 `PDCCH-Config` data type

Its **parent** is `BWP-DownlinkDedicated` (click 2.2.8.7), among other data types.

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 6.12.3.

**All** its parameters are **dedicated** (click 5.1.3).

**Each** of its **instances** provides the **PDCCH dedicated configuration parameters**.

**Each** of its **instances** configures the **UE** with **most PDCCH elements**, of its corresponding **bandwidth part** (**initial** or **additional**), of **dedicated configuration**, of its corresponding **cell** (**special** or **secondary**), of its corresponding **cell group** (**master** or **secondary**). The "**corresponding**" elements are determined by the **specific place**, within the overall **ASN.1 specification**, where this **data type** has been **instantiated**.

#### 6.12.5.1 `PDCCH-Config` parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `controlResourceSetToAddModList` `SEQUENCE (SIZE (1..3)) OF ControlResourceSet` (click 6.12.6): optional, dedicated
- `controlResourceSetToAddModListSizeExt-v1610` `SEQUENCE (SIZE (1..2)) OF ControlResourceSet` (click 6.12.6): optional, dedicated

  **Each** of them is a **list** of **dedicated CORESETs** to be used by the **UE**.

  The **UE restrictions** on configuration of **CORESETs** per **Downlink bandwidth part** are specified in clause 10.1 of TS 38.213 and in TS 38.306.

  The **UE** shall consider the **CORESET** entries in parameter `controlResourceSetToAddModList` and in parameter `controlResourceSetToAddModListSizeExt-v1610` as a **single CORESET list**. Therefore, any given **CORESET created** with **either** parameter can be **modified** with **either** parameter, and it can also be **deleted** with **either** `controlResourceSetToReleaseList` (see below) or `controlResourceSetToReleaseListSizeExt-r16` (see below).

  In case the network reconfigures a **CORESET** with the same value of `controlResourceSetId` (within either of these two parameters) as the one used in a **CORESET** defined with `commonControlResourceSet` from `PDCCH-ConfigCommon` (click 6.12.4) or with `commonControlResourceSetExt-r17` from `CFR-ConfigMCCH-MTCH-r17`, the configuration from `PDCCH-Config` **always takes precedence** and should **not** be updated by the **UE** based on `PDCCH-ConfigCommon` **nor** based on `SIB20-r17`. The data type of **both** `commonControlResourceSet` and `commonControlResourceSetExt-r17` is `ControlResourceSet` (click 6.12.6): the **first** one is **sent** within `SIB1` (click 5.3.4); the **second** one is **broadcast** within `SIB20-r17` (click 5.5.3).

- `controlResourceSetToReleaseList` `SEQUENCE (SIZE (1..3)) OF ControlResourceSetId` (click 6.12.7): optional, dedicated
- `controlResourceSetToReleaseListSizeExt-r16` `SEQUENCE (SIZE (1..3)) OF ControlResourceSetId-r16` (click 6.12.7): optional, dedicated

  **Each** of them is a **list** of **dedicated CORESETs** to be **released** by the **UE**.

  These parameters **only** applies to **dedicated CORESETs**, which are the ones configured by `controlResourceSetToAddModList` or `controlResourceSetToAddModListSizeExt-v1610`. Therefore, these parameters **cannot release** the **common CORESETs** configured either with `commonControlResourceSet` from `PDCCH-ConfigCommon` (click 6.12.4) or with `commonControlResourceSetExt-r17` from `CFR-ConfigMCCH-MTCH-r17`. The data type of **both** `commonControlResourceSet` and `commonControlResourceSetExt-r17` is `ControlResourceSet` (click 6.12.6): the **first** one is **broadcast** within `SIB1` (click 5.3.4) and **unicast**

within the **Non-SI parameter group** (click 5.3.6) the **second** one is **broadcast** within `SIB20-r17` (click 5.5.3).

The configuration from `PDCCH-Config` **always takes precedence** and should **not** be updated by the **UE** based on `PDCCH-ConfigCommon` **nor** based on `SIB20-r17`.

- `searchSpacesToAddModList` `SEQUENCE(SIZE (1..10)) OF SearchSpace` (click 6.12.8): optional, dedicated
- `searchSpacesToAddModListExt-r16` `SEQUENCE(SIZE (1..10)) OF SearchSpaceExt-r16`: optional, dedicated
- `searchSpacesToAddModListExt-v17` `SEQUENCE(SIZE (1..10)) OF SearchSpaceExt-v1700`: optional, dedicated

  **Each** of them is a **list** of **UE specific search spaces** or **MBS multicast search spaces**.

  The network configures **at most 10 search spaces per bandwidth part per cell** (including **both UE-specific** and **common search spaces**).

  If the network includes parameter(s) `searchSpacesToAddModListExt-r16` and/or `searchSpacesToAddModListExt-v17` , the said parameter(s) **must** include the **same quantity of entries**, and listed in the **same order**, as `searchSpacesToAddModList`.

- `tpc-PUSCH` `PUSCH-TPC-CommandConfig`: optional, dedicated
- `tpc-PUCCH` `PUCCH-TPC-CommandConfig`: optional, dedicated
- `tpc-SRS` `SRS-TPC-CommandConfig`: optional, dedicated

  **Each** of them **enables**, and **configures**, the reception of **group TPC commands** for **PUSCH**, **PUCCH** or **SRS**, respectively.

For more detail on these parameters, refer to the table "*PDCCH-Config field descriptions*" of TS 38.331.

### 6.12.5.2 `PDCCH-Config` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
PDCCH-Config ::= SEQUENCE {
  controlResourceSetToAddModList SEQUENCE(SIZE (1..3)) OF ControlResourceSet
                                 OPTIONAL,                          -- Need N
  controlResourceSetToReleaseList SEQUENCE(SIZE (1..3)) OF
                    ControlResourceSetId OPTIONAL,                  -- Need N
  searchSpacesToAddModList  SEQUENCE(SIZE (1..10)) OF SearchSpace OPTIONAL,
                                                                    -- Need N
  searchSpacesToReleaseList SEQUENCE(SIZE (1..10)) OF SearchSpaceId
                                 OPTIONAL,                          -- Need N
  downlinkPreemption SetupRelease {DownlinkPreemption} OPTIONAL,    -- Need M
  tpc-PUSCH SetupRelease {PUSCH-TPC-CommandConfig} OPTIONAL,        -- Need M
  tpc-PUCCH SetupRelease {PUCCH-TPC-CommandConfig} OPTIONAL,        -- Need M
  tpc-SRS   SetupRelease {SRS-TPC-CommandConfig} OPTIONAL,          -- Need M
  ...,
  [[
```

```
    controlResourceSetToAddModListSizeExt-v1610 SEQUENCE (SIZE (1..2)) OF
                        ControlResourceSet OPTIONAL,                -- Need N
    controlResourceSetToReleaseListSizeExt-r16 SEQUENCE (SIZE (1..5)) OF
                        ControlResourceSetId-r16 OPTIONAL,          -- Need N
    searchSpacesToAddModListExt-r16 SEQUENCE(SIZE (1..10)) OF
                        SearchSpaceExt-r16 OPTIONAL,                -- Need N
    uplinkCancellation-r16 SetupRelease {UplinkCancellation-r16} OPTIONAL,
                                                                    -- Need M
    monitoringCapabilityConfig-r16 ENUMERATED
         {15monitoringcapability, r16monitoringcapability} OPTIONAL, -- Need M
    searchSpaceSwitchConfig-r16 SearchSpaceSwitchConfig-r16 OPTIONAL -- Need R
    ]],
    [[
    searchSpacesToAddModListExt-v1700 SEQUENCE(SIZE (1..10)) OF
                        SearchSpaceExt-v1700 OPTIONAL,              -- Need N
    monitoringCapabilityConfig-v1710 ENUMERATED {r17monitoringcapability}
                                     OPTIONAL,                      -- Need M
    searchSpaceSwitchConfig-r17 SearchSpaceSwitchConfig-r17 OPTIONAL,
                                                                    -- Need R
    pdcch-SkippingDurationList-r17 SEQUENCE(SIZE (1..3)) OF
                        SCS-SpecificDuration-r17 OPTIONAL           -- Need R
    ]]
}
```

### 6.12.6 `ControlResourceSet` data type

It can be **both broadcast** (or **unicast**) within `SIB1` (click 5.3.4) and **unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `SIB1`, click 6.12.2.

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 6.12.3.

Its **3 parent data types** are `PDCCH-ConfigCommon` (click 6.12.4), `PDCCH-Config` (click 6.12.5) and `CFR-ConfigMCCH-MTCH-r17`.

**Each** of its **instances** contains **either all common parameters or all dedicated parameters** (click 5.1.3), depending on where it is **instantiated**.

**Each** of its **instances** configures **one non-zero CORESET** (i.e., other than **CORESET#0**).

#### 6.12.6.1 `ControlResourceSet` parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `cce-REG-MappingType` **`CHOICE`**: mandatory, common **or** dedicated

  It configures the **type** of **mapping** from **control channel elements** (**CCE**) to **resource element groups** (**REG**) (refer to clauses 7.3.2.2 and 7.4.1.3.2 of TS 38.211).

  Its **data type** is a **`CHOICE`** between `interleaved` and `nonInterleaved`.

  The **data type** of `nonInterleaved` is **`NULL`** (no need for additional configuration parameters).

  The **data type** of `interleaved` is a **`SEQUENCE`** (see below).

- `interleaved` **`SEQUENCE`**: choice, common **or** dedicated

  It configures the **type** of **mapping** from **control channel elements** (**CCE**) to **resource element groups** (**REG**) (refer to clauses 7.3.2.2 and 7.4.1.3.2 of TS 38.211).

  Its **`SEQUENCE`** contains `interleaverSize`, `reg-BundleSize` and `shiftIndex` (see below).

- `interleaverSize` **`ENUMERATED`** **`{`**`n2`**`,`** `n3`**`,`** `n6`**`}`**: mandatory (within **`CHOICE`**), common **or** dedicated

  It is the **PDCCH interleaver size** (refer to clause 7.3.2.2 of TS 38.211).

- `controlResourceSetId` `ControlResourceSetId` (click 6.12.7): mandatory, common **or** dedicated

  It is the **identifier** of this **CORESET**. This **identifier** is **unique** within **each** of the **UE**'s **serving cells**. Its **data type** `ControlResourceSetId` is an **`INTEGER`** from `0` to `11`.

  Identifier `0` is reserved for **CORESET#0** (click 6.8.8).

  It is ignored if the optional parameter `controlResourceSetId-v1610` is **present** (see the next bullet).

- `controlResourceSetId-v1610` `ControlResourceSetId-v1610` (click 6.12.7): optional, common **or** dedicated

  It is the **identifier** of this **CORESET**. This **identifier** is **unique** within **each** of the **UE**'s **serving cells**. Its **data type** `ControlResourceSetId-v1610` is an **`INTEGER`** from `12` to `15`.

- `coresetPoolIndex-r16` **`INTEGER`** **`(`**`0..1`**`)`**: optional, common **or** dedicated

  It is the **index** of the **CORESET pool** for this **CORESET**, as specified in clauses 9 and 10 of TS 38.213 and clauses 5.1 and 6.1 of TS 38.214. If it is **absent**, the **UE** applies the value `0`.

- `duration` **`INTEGER`** **`(`**`1..maxCoReSetDuration`**`)`**: mandatory, common **or** dedicated

  It is the **quantity** of **contiguous symbol intervals** of **every PUCCH transmission** associated to this **CORESET** (refer to clause 7.3.2.2 of TS 38.211). The constant `maxCoReSetDuration` is `3`, which is the **maximum quantity** of **contiguous symbol intervals** of each **CORESET**.

- `frequencyDomainResources` **`BIT STRING (SIZE (`**`45`**`))`**: mandatory, common **or** dedicated

  It is the **frequency-domain resources** for **each non-zero CORESET**. **Each bit** corresponds to a **range** of **6 contiguous PRBs**. The **first PRB range starts** from the **first PRB** in the **bandwidth part** (**BWP**) **or** in the **multicast broadcast service** (**MBS**) **common frequency resource** (**CFR**) where the **CORESET** is configured.

  The **first** (**left-most / most significant**) **bit** corresponds to the **first PRB range** in the **BWP** or **MBS CFR** where the **CORESET** is configured, and so on. A **bit** set

to `1` indicates that the corresponding **PRB range belongs** to the **frequency-domain resource** of this **CORESET**. **Bits** corresponding to a **PRB range not fully contained** in the **bandwidth part within** which the **CORESET** is **configured must** be set to `0` (refer to clause 7.3.2.2 of TS 38.211).

The **quantity** of **PRBs** "$N_{PRB}^{CORESET}$" of the **CORESET** is therefore **6 times** the **quantity of bits** set to `1`.

Notice that the **45 bits** allows that the **frequency resources** of a **CORESET** span up to 45•6=270 **PRBs**, or 270•12=3,240 **subcarriers**, within the corresponding **bandwidth part**.

Recall that the **maximum quantity** of **PRBs** in **each** of the **UE**'s **bandwidth parts** is `275`, and therefore, the **non-zero CORESETs cannot** occupy the **5 highest PRBs**.

When **at least 1 search space** is configured with `searchSpaceFreqMonitorLocation-r16` from `SharedSpectrumChAccessParamsPerBand-r16`, **only** the **first** "$N_{RBG,set0}^{size}$" bits are **valid** (refer to clause 10.1 of TS 38.213).

- `precoderGranularity` **`ENUMERATED {`**`sameAsREG-bundle, allContiguousRBs`**`}`**: mandatory, common **or** dedicated

  It is the **precoder granularity** in the **frequency domain** (refer to clauses 7.3.2.2 and 7.4.1.3.2 of TS 38.211).

- `reg-BundleSize` **`ENUMERATED {`**`n2, n3, n6`**`}`**: mandatory (within **`CHOICE`**), common **or** dedicated

  It is the **size** of the **PDCCH REG bundles**: the **quantity** of **Resource Element Groups** (**REGs**) **within each REG bundle**. For more detail, refer to clause 7.3.2.2 of TS 38.211.

- `shiftIndex` **`INTEGER(`**`0..maxNrofPhysicalResourceBlocks-1`**`)`**: optional, common **or** dedicated

  It is the **shift index** for the **PDCCH interleaver**. If the parameter is **absent** the **UE** uses the value of the **physical cell identity** (**PCI**, click 3.5.3.2) of this cell (refer to clause 7.3.2.2 of TS 38.211). The constant `maxNrofPhysicalResourceBlocks-1` is `274`, which is the **maximum quantity**, minus **1**, of **PRBs** in **each** of the **UEs**' **bandwidth parts**.

- `tci-PresentInDCI` **`ENUMERATED {`**`enabled`**`}`**: optional, common **or** dedicated

  If **present**, the **DCI messages Format 1_1** and **4_2** contain the field "***Transmission configuration information***" (click 6.9.1) and its **bitlength** is **3 bits**. If **absent**, the said **DCI messages** do **not** contain the said **field**.

  In case of **cross-carrier scheduling** (click 10.2.3), this field is present in the `ControlResourceSet` instance that corresponds to **cross-carrier scheduling** in **DCI Format 1_1** in the **scheduling cell,** if `enableDefaultBeamForCCS` is **absent** (refer to clause 5.1.5 of TS 38.214).

- `tci-PresentDCI-1-2-r16` **`INTEGER (1..3)`**: optional, common **or** dedicated

  If **present**, it indicates the **bitlength** of the "***Transmission configuration information***" (click 6.9.1) field in **DCI messages Format 1_2**. If **absent**, **DCI messages Format 1_2** do **not** contain the said **field**. Refer to clauses 7.3.1 of TS 38.212 and 5.1.5 of TS 38.214.

  In case of **cross-carrier scheduling** (click 10.2.3), this field is present in the `ControlResourceSet` instance that corresponds to **cross-carrier scheduling** in **DCI Format 1_2** in the **scheduling cell,** if `enableDefaultBeamForCCS` is **absent** (refer to clause 5.1.5 of TS 38.214).

For more detail on these parameters, refer to the table "*ControlResourceSet field descriptions*" of TS 38.331.

### 6.12.6.2 `ControlResourceSet` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
ControlResourceSet ::= SEQUENCE {
  controlResourceSetId ControlResourceSetId,
  frequencyDomainResources BIT STRING (SIZE (45)),
  duration INTEGER (1..maxCoReSetDuration),
  cce-REG-MappingType CHOICE {
    interleaved SEQUENCE {
      reg-BundleSize ENUMERATED {n2, n3, n6},
      interleaverSize ENUMERATED {n2, n3, n6},
      shiftIndex INTEGER(0..maxNrofPhysicalResourceBlocks-1) OPTIONAL
                                                              -- Need S
    },
    nonInterleaved NULL
  },
  precoderGranularity ENUMERATED {sameAsREG-bundle, allContiguousRBs},
  tci-StatesPDCCH-ToAddList SEQUENCE(SIZE (1..maxNrofTCI-StatesPDCCH))
                    OF TCI-StateId OPTIONAL,       -- Cond NotSIB-initialBWP
  tci-StatesPDCCH-ToReleaseList SEQUENCE(SIZE (1..maxNrofTCI-StatesPDCCH))
                    OF TCI-StateId OPTIONAL,       -- Cond NotSIB-initialBWP
  tci-PresentInDCI ENUMERATED {enabled} OPTIONAL,                 -- Need S
  pdcch-DMRS-ScramblingID INTEGER (0..65535) OPTIONAL,            -- Need S
  ...,
  [[
  rb-Offset-r16 INTEGER (0..5) OPTIONAL,                          -- Need S
  tci-PresentDCI-1-2-r16 INTEGER (1..3) OPTIONAL,                 -- Need S
  coresetPoolIndex-r16 INTEGER (0..1) OPTIONAL,                   -- Need S
  controlResourceSetId-v1610 ControlResourceSetId-v1610 OPTIONAL  -- Need S
   ]],
  [[
  followUnifiedTCI-State-r17 ENUMERATED {enabled} OPTIONAL        -- Need R
  ]]
}
```

where `maxCoReSetDuration` is `3`, which is the **maximum quantity** of **consecutive symbol intervals** of **each** of the **CORESET**'s **monitoring occasions**, and `maxNrofPhysicalResourceBlocks-1` is `274`, which is the **maximum quantity**, minus **1**, of **PRBs** in **each** of the **UE**'s **bandwidth parts**.

## 6.12.7 `ControlResourceSetId`, `ControlResourceSetId-r16` and `ControlResourceSetId-v1610` data types

They can be **both broadcast** (or **unicast**) within `SIB1` (click 5.3.4) and **unicast** within the **Non-SI parameter group** (click 5.3.6).

For their **ancestor tree up to** `SIB1`, click 6.12.2).

For their **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 6.12.3.

They can be **both broadcast** (or **unicast**) within `SIB1` and **unicast** within the **Non-SI parameter group**.

**Each** of their **instances** contains **either all common parameters or all dedicated parameters** (click 5.1.3), depending on where it is **instantiated**.

They are a **descendant** of `PDCCH-ConfigCommon` (click 6.12.4) and of `PDCCH-Config` (click 6.12.5).

**Each** of their **instances** is an **integer** that **uniquely** identifies **each** of the **UE**'s **CORESET**s (click 6.8.7), within **each** of the **UE**'s **serving cells**.

The range of `ControlResourceSetId` is `0` to `11` and the one `ControlResourceSetId-r16` is `0` to `1`. Therefore, the **maximum quantity** of **different CORESETs** that can be configured within **each** of the **UE**'s **serving cells** is `12` in **R15** and `16` in **R16** and **R17**.

Value `0` **always** corresponds to **CORESET#0**. Values `1` to `15` correspond to **non-zero CORESETs**.

Their **complete ASN.1 specification**s are:

```
ControlResourceSetId ::= INTEGER (0..maxNrofControlResourceSets-1)

ControlResourceSetId-r16 ::= INTEGER (0..maxNrofControlResourceSets-r16-1)

ControlResourceSetId-v1610 ::= INTEGER
            (maxNrofControlResourceSets..maxNrofControlResourceSets-r16-1)
```

where `maxNrofControlResourceSets-1` is `11` and `maxNrofControlResourceSets-r16-1` is `15`. Both of them are the **maximum quantity**, minus **1**, of **CORESETs configurable** within **each** of the **UE**'s **serving cells**, with the **only difference** that the **second one** only applies to **R16** or later.

### 6.12.8 `SearchSpace` data type

It can be **both broadcast** (or **unicast**) within `SIB1` (click 5.3.4) and **unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `SIB1`, click 6.12.2.

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 6.12.3.

It is a **descendant** of `PDCCH-ConfigCommon` (click 6.12.4) and of `PDCCH-Config` (click 6.12.5).

**Each** of its **instances** contains **either all common parameters or all dedicated parameters** (click 5.1.3), depending on where it is **instantiated**.

**Each** of its **instances** configures **1 search space**.

Notice that there is an extension of `SearchSpace` for **R16** (`SearchSpaceExt-r16`) and another one for **R17** (`SearchSpaceExt-v1700`), none of which is described in detail in this book.

#### 6.12.8.1 `SearchSpace` parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `controlResourceSetId ControlResourceSetId` (click 6.12.7): optional, common **or** dedicated

  It is the **identifier** of the **CORESET** (click 6.8.7) to which **this search space** applies.

  Its data type `ControlResourceSetId` is an `INTEGER` from `0` to `11`.

  The **non-zero CORESETs** are configured in the **same bandwidth part** as this **search space**, **except** `commonControlResourceSetExt-r17` from `CFR-ConfigMCCH-MTCH-r17`, within `SIB20-r17` (click 5.5.3).

- `duration INTEGER (2..2559)`: optional, common **or** dedicated

  It is the **quantity** "$T_s$" of **consecutive slots** that contain **monitoring occasions** within **each monitoring period** of "$k_s$" **slots**. See `monitoringSlotPeriodicityAndOffset` in the previous bullet.

  Although its **range** is **very large**, only values from 1 to "$k_s$-1" may be configured..

  If `duration` is **absent**:

  - The **UE** applies the value `1` **slot**, **except** for **DCI Format 2_0**, where the **UE** ignores this parameter.
  - The **IAB-MT** applies the value **1 slot**, **except** for **DCI Format 2_0** and **DCI Format 2_5**. The **IAB-MT** ignores this parameter for **DCI Format 2_0** and **DCI Format 2_5**.

  For **subcarrier spacing** of **480** and **960 kHz**, `duration-r17` from `SearchSpaceExt-v1700` is used, and its value is restricted to be an **integer multiple** of "$L$" **slots** and smaller than **periodicity**, where "$L$" is the configured **length** of the **bitmap** `monitoringSlotsWithinSlotGroup-r17` from `SearchSpaceExt-v1700`. If `duration-r17` is absent, the **UE** assumes the **duration** is "$L$" **slots**. The **maximum** valid `duration` is **periodicity-L**.

- `monitoringSlotPeriodicityAndOffset CHOICE`: optional, common **or** dedicated

  It indicates the **slots** for **PDCCH monitoring**, configured as **slot periodicity** "$k_s$" and **slot offset** "$o_s$".

  Its **data type** is a `CHOICE` between **15** parameters: `sl1` to `sl2560`.

  The **numeric suffix** of the **chosen parameter** is the **monitoring slot periodicity** "$k_s$". It is therefore possible to configure **one** among the following **15** slot periodicities: `1`, `2`, `4`, `5`, `8`, `10`, `16`, `20`, `40`, `80`, `160`, `320`, `640`, `1,280` and `2,560`. The **first monitoring period** starts in **slot 0** of **frame 0**. Notice that **subsequent monitoring periods** might not be **aligned** to a frame start (see the second example below).

  The data type of **each** parameter in the `CHOICE` is an `INTEGER` with a range from `0` to the **periodicity minus 1** (i.e., from `0` to "$k_s$-1").

The **value** (of the **chosen parameter**) is the **monitoring slot offset** "$o_s$". The **offset** "$o_s$" is referred to the **first slot** in **each monitoring period**. Therefore, if "$o_s$" is `0`, the **monitoring slot** is the **first slot** in each **monitoring period**.

For **subcarrier spacings 15**, **30**, **60**, and **120 kHz** and if the **UE** is configured to monitor:

- **DCI Format 2_1**, **only** the parameters `sl1`, `sl2` or `sl4` are **applicable**.
- **DCI Format 2_0**, **only** the parameters `sl1`, `sl2`, `sl4`, `sl5`, `sl8`, `sl10`, `sl16` or `sl20` are **applicable** (refer to clause 10 of TS 38.213).
- **DCI Format 2_4**, **only** the parameters `sl1`, `sl2`, `sl4`, `sl5`, `sl8` or `sl10` are **applicable**.

For **subcarrier spacing 480 kHz** and if the **UE** is configured to monitor:

- **DCI Format 2_0**, **only** the parameters `sl4`, `sl8`, `sl16`, `sl20`, `sl32`, `sl40`, `sl64` or `sl80` are **applicable**.
- **DCI Format 2_1**, **only** the parameters `sl4`, `sl8` or `sl16` are **applicable**.
- **DCI Format 2_4**, **only** the parameters `sl4`, `sl8`, `sl16`, `sl20`, `sl32` or `sl40` are **applicable**.

For **subcarrier spacing 960 kHz** and if the **UE** is configured to monitor:

- **DCI Format 2_0**, **only** the parameters `sl8`, `sl16`, `sl32`, `sl40`, `sl64`, `sl80`, `sl128` or `sl160` are **applicable**.
- **DCI Format 2_1**, **only** the parameters `sl8`, `sl16`, or `sl32` are **applicable**.
- **DCI Format 2_4**, **only** the parameters `sl8`, `sl16`, `sl32`, `sl40`, `sl64` or `sl80` are **applicable**.

For **subcarrier spacings** of **480** and **960 kHz**, the configured **periodicity** and **offset** are **restricted** to be an **integer multiple** of "***L***" **slots**, where "***L***" is the **configured length** of the **bitmap** provided by `monitoringSlotsWithinSlotGroup-r17` from `SearchSpaceExt-v1700`. Consequently, for a given **periodicity**, the **offset** may take the following values:

$\{0, L, 2 \bullet L, \ldots, L \bullet floor(1/L \bullet (periodicity\text{-}1))\}$.

For **IAB-MT**:

- If the **IAB-MT** is configured to monitor **DCI Format 2_1**, **only** the parameters `sl1`, `sl2` or `sl4` are **applicable**.
- If the **IAB-MT** is configured to monitor **DCI Format 2_0** or **DCI Format 2_5**, **only** the parameters `sl1`, `sl2`, `sl4`, `sl5`, `sl8`, `sl10`, `sl16` or `sl20` are **applicable** (refer to clause 10 of TS 38.213).

If `monitoringSlotPeriodicityAndOffset-r17` from `SearchSpaceExt-v1700` is **present**, any **previously configured** `monitoringSlotPeriodicityAndOffset` is **released**, and if `monitoringSlotPeriodicityAndOffset` is **present**, any **previously configured** `monitoringSlotPeriodicityAndOffset-r17` is **released**.

- `monitoringSymbolsWithinSlot BIT STRING (SIZE (14))`: optional, common **or** dedicated

  It indicates the **first symbol interval(s)** of **each PDCCH monitoring** occasion within each **slot** that is configured for **PDCCH monitoring** (see above `monitoringSlotPeriodicityAndOffset` and `duration`). The **most significant (leftmost) bit** represents the **first symbol interval** in the **slot**, and **successive bits** are **successive symbol intervals**. **Each bit** set to `1` identifies the **first symbol interval** of one **PDCCH monitoring occasion** within the **slot**. If this bandwidth part has **extended cyclic prefix** (click 2.3.2), the **last two bits** are **ignored** by the **UE** or the **IAB-MT**.

  For **DCI Format 2_0**, and for **IAB-MT** with **DCI Format 2_5**, the **quantity** of **initial symbol intervals** that apply depend on the **duration** of the **CORESET** identified by `controlResourceSetId` (see above): the **first symbol interval** for **duration** `3`, the **two first** for **duration** `2` and the **three first** for **duration** `1`.

  For more detail, refer to clause 10 of TS 38.213.

- `nrofCandidates SEQUENCE`: optional, common **or** dedicated

  It is the **quantity** of **PDCCH candidates** at **each aggregation level** within **this search space**.

  Its **data type** is a `SEQUENCE` that contains **five** parameters `aggregationLevelX`, where `X` is an **aggregation level** `1`, `2`, `4`, `8` or `16`. **Each** of the said **five** parameters has the data type `ENUMERATED {n0, n1, n2, n3, n4, n5, n6, n8}`.

  The value "`n0`" indicates that the corresponding **aggregation level** is **not used** in **this search space**. **Each** of the other values indicates the **quantity PDCCH candidates** with the corresponding **aggregation level** may contain in **this search space**.

  The configuration indicated by `nrofCandidates` applies to **all** the **DCI Formats** in **this search space**, with one exception. This exception is a **common search space** and the **DCI message Format 2_0**, where the configuration stated by `searchSpaceType` (see below) applies.

  For more detail, refer to clause 10 of TS 38.213.

- `searchSpaceId SearchSpaceId` (click 6.12.9): mandatory, common **or** dedicated)

  It is the **identifier** of this **search space**. This **identifier** is **unique** within **each** of the **UE**'s **serving cells**. Its **data type** is an `INTEGER` from `0` to `39`. Therefore, **each UE** may be configured **up to** `40` **search spaces** within **each** of its **serving cells**.

  Value `0` **cannot** be used, because **search space** `0` is **not** configured with data type `SearchSpace`, and it is **rather** configured with parameter `searchSpaceZero` either from `MIB` (click 8.12.3) or from `ServingCellConfigCommon` (click 3.9.6).

  In case of **cross-carrier scheduling** (click 10.2.3), **search spaces** with the **same identifier** in the **scheduled cell** and in the **scheduling cell** are **linked** to **each other**. The **UE** applies the **search space** for the **scheduled cell only** if the

**Downlink bandwidth parts** in which the **linked search spaces** are configured in **scheduling cell** and **scheduled cell** are **both active**.

For an **IAB-MT**, the **search space** defines **how**/**where** to search for **PDCCH candidates** for an **IAB-MT** where each **search space** is associated with one **CORESET** and for a **scheduled cell** in the case of **cross-carrier scheduling**, except for `nrofCandidates` (see below), **all** the **optional parameters** are **absent**.

- `searchSpaceType` **`CHOICE`**: mandatory, common **or** dedicated

  It indicates whether this is a **common search space** (choice `common`) or a **UE specific search space** (choice `ue-Specific`) and the **DCI message Format(s)** that **this search space** may contain.

  Its **data type** is a **`CHOICE`** between parameters `common` and `ue-Specific`. **Each** of these two parameters is a **`SEQUENCE`** that contains **optional** parameters (click 6.12.8.2), **each** of which indicates a **DCI message Format** (or a combination of **Formats**) that **this search space** may contain.

  For the specific case of a **common search space** and for the **DCI message Format 2_0**, the parameter `nrofCandidates-SFI` (Slot Format Indication) indicates the **aggregation levels** that may be used, and the **quantity** of **PDCCH transmissions** or **PDCCH candidates** for **each aggregation level**. Notice that in this case, the **quantity** may only take the values `n1` or `n2`.

For more detail on these parameters, refer to the table "*SearchSpace field descriptions*" of TS 38.331.

### 6.12.8.2 `SearchSpace` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
SearchSpace ::= SEQUENCE {
  searchSpaceId SearchSpaceId,
  controlResourceSetId ControlResourceSetId OPTIONAL,      -- Cond SetupOnly
  monitoringSlotPeriodicityAndOffset CHOICE {
    sl1 NULL, sl2 INTEGER (0..1), sl4 INTEGER (0..3), sl5 INTEGER (0..4),
    sl8 INTEGER (0..7), sl10 INTEGER (0..9), sl16 INTEGER (0..15),
    sl20 INTEGER (0..19), sl40 INTEGER (0..39), sl80 INTEGER (0..79),
    sl160 INTEGER (0..159), sl320 INTEGER (0..319), sl640 INTEGER (0..639),
    sl1280 INTEGER (0..1279), sl2560 INTEGER (0..2559)
  } OPTIONAL,                                                -- Cond Setup4
  duration INTEGER (2..2559) OPTIONAL,                          -- Need S
  monitoringSymbolsWithinSlot BIT STRING (SIZE (14)) OPTIONAL, -- Cond Setup
  nrofCandidates SEQUENCE {
    aggregationLevel1  ENUMERATED {n0, n1, n2, n3, n4, n5, n6, n8},
    aggregationLevel2  ENUMERATED {n0, n1, n2, n3, n4, n5, n6, n8},
    aggregationLevel4  ENUMERATED {n0, n1, n2, n3, n4, n5, n6, n8},
    aggregationLevel8  ENUMERATED {n0, n1, n2, n3, n4, n5, n6, n8},
    aggregationLevel16 ENUMERATED {n0, n1, n2, n3, n4, n5, n6, n8},
  } OPTIONAL,                                                -- Cond Setup
```

```
    searchSpaceType CHOICE {
      common SEQUENCE {
        dci-Format0-0-AndFormat1-0 SEQUENCE {...} OPTIONAL,           -- Need R
        dci-Format2-0 SEQUENCE {
          nrofCandidates-SFI SEQUENCE {
            aggregationLevel1  ENUMERATED {n1, n2} OPTIONAL,          -- Need R
            aggregationLevel2  ENUMERATED {n1, n2} OPTIONAL,          -- Need R
            aggregationLevel4  ENUMERATED {n1, n2} OPTIONAL,          -- Need R
            aggregationLevel8  ENUMERATED {n1, n2} OPTIONAL,          -- Need R
            aggregationLevel16 ENUMERATED {n1, n2} OPTIONAL           -- Need R
          },
          ...
        } OPTIONAL,                                                   -- Need R
        dci-Format2-1 SEQUENCE {...} OPTIONAL,                        -- Need R
        dci-Format2-2 SEQUENCE {...} OPTIONAL,                        -- Need R
        dci-Format2-3 SEQUENCE {
          dummy1 ENUMERATED {sl1, sl2, sl4, sl5, sl8, sl10, sl16, sl20
          } OPTIONAL,                                              -- Cond Setup
          dummy2 ENUMERATED {n1, n2},
          ...
        } OPTIONAL                                                    -- Need R
      },
      ue-Specific SEQUENCE {
        dci-Formats ENUMERATED {formats0-0-And-1-0, formats0-1-And-1-1},
        ...,
        [[
        dci-Formats-MT-r16 ENUMERATED {formats2-5} OPTIONAL,          -- Need R
        dci-FormatsSL-r16 ENUMERATED {formats0-0-And-1-0, formats0-1-And-1-1,
              formats3-0, formats3-1, formats3-0-And-3-1} OPTIONAL,   -- Need R
        dci-FormatsExt-r16 ENUMERATED {formats0-2-And-1-2,
              formats0-1-And-1-1And-0-2-And-1-2} OPTIONAL             -- Need R
        ]]
      }
    }
}
```

Source web pages:

- https://www.nrexplained.com/rrc#ControlResourceSetId

### 6.12.9 `SearchSpaceId` data type

It can be **both broadcast** (or **unicast**) within `SIB1` (click 5.3.4) and **unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `SIB1`, click 6.12.2.

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 6.12.3.

It is a **descendant** of `PDCCH-ConfigCommon` (click 6.12.4) and of `PDCCH-Config` (click 6.12.5).

**Each** of its **instances** is an **integer** that **uniquely** identifies **each common** or **UE-specific search space** within **each** of the **UE**'s **serving cells**.

Its **complete ASN.1** (click 5.9) specification is:

```
SearchSpaceId ::= INTEGER (0..maxNrofSearchSpaces-1)
```

where `maxNrofSearchSpaces-1` is `39`, which is the **maximum quantity**, minus **1**, of **search spaces** within **each** of the **UE**'s **serving cells**.

### 6.12.10 `PDCCH-ServingCellConfig` data type

Its **only parent** is the ***`DownlinkConfig`*** **parameter set** (click 2.2.8.3), which is enclosed in `ServingCellConfig` (click 3.9.7).

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 6.12.3.

**All** its parameters are **dedicated** (click 5.1.3).

**Each** of its **instances** contains **PDCCH dedicated parameters** for a **given UE**, that apply the **same** for **all** the **bandwidth parts** of the corresponding **serving cell**.

### 6.12.10.1 `PDCCH-ServingCellConfig` parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `slotFormatIndicator SlotFormatIndicator` (click 2.3.5.7): optional, dedicated

  It configures a **UE** with a **group SFI-RNTI** (that the **UE** needs to get the **slot formats combinations** (click 2.3.4.2.3) for its cells) and with **monitoring** a **PDCCH** for **Availability Indicators** (**AI**).

For more detail on these parameters, refer to the table "*PDCCH-ServingCellConfig field descriptions*" of TS 38.331.

### 6.12.10.2 `PDCCH-ServingCellConfig` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
PDCCH-ServingCellConfig ::= SEQUENCE {
  slotFormatIndicator SetupRelease {SlotFormatIndicator}
                      OPTIONAL,                                   -- Need M
  ...,
  [[
  availabilityIndicator-r16 SetupRelease {AvailabilityIndicator-r16}
                            OPTIONAL,                             -- Need M
  searchSpaceSwitchTimer-r16 INTEGER (1..80) OPTIONAL             -- Need R
  ]],
  [[
  searchSpaceSwitchTimer-v1710 INTEGER (81..1280) OPTIONAL        -- Need R
  ]]
}
```

## 6.13 PDSCH overview

The **Physical Downlink Shared Channel** (**PDSCH**) is used by the **NR node** to send **user** data and **higher-layer control** data to **1 UE** (or **group** of **UEs**).

The most relevant **functions** of **PDSCH** are:

- Send **unicast Downlink user** data to **1 UE.**
- Send **unicast Downlink higher-layer control** data to **1 UE.**
- Send **multicast/broadcast Downlink user** data to **1 group** of **UEs**.
- Send **multicast/broadcast Downlink higher-layer control** data to **1 group** of **UEs**.

**Only 1 transmission scheme** is defined for **PDSCH**, and it is used for **all PDSCH transmissions**. It consists of **up to 8 layers** on **antenna ports** from **1000** to **1001** as defined in clause 7.3.1.4 of TS 38.211. The **DM-RS reception** procedures are defined in clause 5.1.6.2 of TS 38.214.

**Each PDSCH transmission** takes place in **1** of the **transmission occasions determined** by the **PDSCH scheduling** (click 6.14). **Each slot** can contain **0**, **1** or **several PDSCH transmissions** (click 6.14.6).

**Each PDSCH transmission** contains (click 6.16):

- **Unicast** and **multicast**: **1** or **2 codeword(s)** (click 4.12.1.2) in **1** to **8 layers**.
- **Broadcast**: **1 codeword** in **1 layer**.

**Each transport block** (click 4.12.1.1) or **codeword** is subject to a **specific processing** (click 6.16) for its **transmission** and for its **reception** through **PDSCH**.

For the **contents** of **PDSCH DM-RS** and of **PDSCH PT-RS**, click 6.17.

**PDCCH** is **configured** (click 6.18) with several **parameters** and **data types**.

The **PDSCH description** in this book covers almost nothing about **PDSCH** for **shared spectrum**.

For more detail on **PDSCH**, refer to:

- Clauses 4, 5 and 7.3.1 (**Phys.** and **modulation**) of TS 38.211.
- Clauses 4.2, 5 and 7.2 (**mux**. and **coding**) of TS 38.212.
- Clause 5 (**procedures**) of TS 38.214.
- Clauses 5.2.1 and 5.2.2 (**overall Tx**) and 10.2 (**scheduling**) of TS 38.300.

## 6.14 PDSCH scheduling

**PDSCH Tx scheduling**

The **NR node decides**, for **each** of its **PDSCH transmissions**, its **frequency-domain resources** (click 6.14.3), its **time-domain resources** (click 6.14.4), its **space resources**, its **transmission characteristics** and its **destination UE** (or **group** of **UEs**). The **NR node must** select the values of the said **elements** within a **set** of **Rx resources** and **characteristics** that the **NR node** has **previously configured** to the **destination UE** (or **group** of **UEs**).

The said **set** of **Rx resources** and **characteristics** is **previously configured** to **each UE either** with **PDSCH Rx semi-persistent scheduling** (click 6.14.1) or with **PDSCH Rx dynamic scheduling** (click 6.14.2).

**PDSCH Rx scheduling**

**Each UE must listen** on **all** the **PDSCH reception occasions** that the **NR node** has **scheduled** to it, **either** as **Rx configured grants** or as **Rx assignments**.

Within its **scheduled Rx PDSCH PREs**, **each UE** will **only receive data** in the **payload PDSCH PREs** because a **few** of the **scheduled PDSCH PREs** will be used for **physical signals** (click 4.11.3.2) **instead** of for **data**.

For an **overall** description of **scheduling**, **transmission occasions** and **transmissions**, click 4.11.4.

For more detail on **PDSCH scheduling**, refer to clause 10.2 of TS 38.300.

### 6.14.1 PDSCH Rx semi-persistent scheduling

**Semi-persistent scheduling** (click 4.11.4.3) provides **each UE** with a **PDSCH Rx configured grant**, which is a **set** of **periodical reception occasions**. **Configured grants** are provided to **each UE** using the **RRC** (click 5) **protocol**.

**PDSCH configured grants** can be **either** for **unicast** or for **multicast**. After the **UE** receives the **RRC configured grant**, the **grant** has to be **activated** with a **DCI message**, and it can be **deactivated** with another **DCI message**.

- **Unicast PDSCH configured grant**

  It is **activated** or **deactivated** with a **DCI message** (click 6.9) of **Format 1_0**, **1_1** or **1_2** with the **last 16 bits** of its **CRC masked** with **CS-RNTI** (click 6.9.10.2).

- **Multicast PDSCH configured grant**

  It is **activated** or **deactivated** with a **DCI message** of **Format 4_1** or **4_2** with the **last 16 bits** of its **CRC masked** with **G-CS-RNTI** (click 6.9.10.2).

**PDSCH semi-persistent scheduling** (i.e., **configured grants**) for **unicast** can be configured with `sps-Config` and with `sps-ConfigToAddModList-r16` **both** of them from `BWP-DownlinkDedicated` (click 2.2.8.72.2.8.10). **PDSCH semi-persistent scheduling** for **multicast** can be configured with `sps-ConfigMulticastToAddModList-r17` from `CFR-ConfigMulticast-r17`. The **data type** of the **first** parameter is `SPS-Config` and the one of the **second** and **third** parameters is a `SEQUENCE OF SPS-Config`.

**PDSCH configured grants** for **unicast** can be **released** with `sps-ConfigToReleaseList-r16`. **PDSCH configured grants** can be **deactivated** with `sps-ConfigDeactivationStateList-r16`. **Both** of them also from `BWP-DownlinkDedicated`. **PDSCH configured grants** for **multicast** can be **released** with `sps-ConfigMulticastToReleaseList-r17` from `CFR-ConfigMulticast-r17`. The **data type** of these **3** parameters is a `SEQUENCE OF SPS-ConfigIndex-r16`.

**Each UE** may be configured with **up to 8 active PDSCH configured grants** for **each bandwidth part** of **each serving cell**. When the **UE** has been **configured** with more than 1 **PDSCH configured grants**, the **2** following statements **apply**:

- The **NR node** decides **which** of these **PDSCH configured grants** (including **all** of them) are **active** at any **given time**.
- **Each PDSCH configured grant** is **individually activated** using **1 DCI message**. However, **1 DCI message** can be used to **deactivate multiple PDSCH configured grants**.

For an **overall** description of **scheduling**, **transmission occasions** and **transmissions**, click 4.11.4.

For more detail on **PDSCH scheduling**, refer to clause 10.2 of TS 38.300.

### 6.14.2 PDSCH Rx dynamic scheduling

**PDSCH dynamic scheduling** (click 4.11.4.3) provides the **UE** with a **one-shot set** of **PDSCH reception assignments**, where **each reception assignment** corresponds to **1 reception occasion**.

**Each** such **one-shot set** of **reception assignments** is provided to the **UE** with **1 Downlink Control Information** (**DCI**) **message** (click 6.9).

**Each** such **DCI message** :

- Has **Formats 1_x** for **unicast** delivery and **Formats 4_x** for **Multicast/Broadcast Services** (**MBS**).
- Signals from **1** to **8 PDSCH reception assignments** to the recipient **UE** (or group of **UEs**).
- Has the **last 16 bits** of its **CRC masked** with one of the **RNTIs** (individual or group) that the destination **UE** can manage.
- **Each resource type** is **assigned** with a **specific DCI field**: **a)** the **frequency resources** with "***Frequency domain resource assignment***" (click 6.14.3); **b)** the **time resources** with "***Time domain resource assignment***" (click 6.14.4); and **c)** the **space resources** with "***Antenna ports***".

A **PDSCH reception assignment overrides every PDSCH configured grant** in the **same serving cell** in case they **overlap** in **time**.

**Resource element groups** (**REGs**)

**PDSCH frequency-time resources** scheduled to **each UE** in **quanta** called **Resource Element Groups** (**REGs**). A **REG** is **1 symbol interval** over **1 Resource Block**. One **REG** therefore contains **12** Physical **Resource Elements** (**PREs**).

Within the **PREs** assigned for **each PDSCH reception**, some of them are **payload PREs** while others contain **DM-RS**, **PT-RS** or **empty PREs** (click 6.14.5). The **PDSCH information** is **conveyed** in the **payload PREs**, while the other **PREs** are used for **demodulation purposes**.

For an **overall** description of **scheduling**, **transmission occasions** and **transmissions**, click 4.11.4.

For more detail on **PDSCH scheduling**, refer to clause 10.2 of TS 38.300.

For more detail on **DCI messages**, click 6.9 and refer to clause 7.3 of TS 38.212.

### 6.14.3 PDSCH frequency-domain resources

The **dynamic scheduling** of **PDSCH frequency-domain resources** is encoded in the **DCI message field** "***Frequency domain resource assignment***".

The **PDSCH frequency-domain resource assignment** is the **same** for **all** the **1** to **8 PDSCH assignments** that may be done with **1 DCI message**.

The following table shows, for **PDSCH assignments**, the **DCI message Formats** where the said **field** is **present**, what is **assigned** by **each** of these **DCI Formats** and what is the **field**'s **bitlength** in **each** of these **DCI Formats**:

| **DCI message field "*Frequency domain resource assignment*"** | | |
|---|---|---|
| **Assigns** | **DCI Format** | **Bitlength** |
| **PDSCH U/B** | Format 1_0 (6.9.5) | Variable |
| **PDSCH Unicast** | Format 1_1 (6.9.6) | Variable |
| | Format 1_2 (6.9.7) | Variable |
| **PDSCH MBS** | Format 4_0 (6.9.9.1) | Variable |
| | Format 4_1 (6.9.9.2) | Variable |
| | Format 4_2 (6.9.9.3) | Variable |

The text "**U/B**" stands for **unicast** or **broadcast**.

The "***Frequency domain resource assignment***" contains **1 optional bit** and a **subfield**.

This **1 optional bit** is **only** present when `dynamicSwitch` (click 6.14.3.1) has been configured. When this **bit** is **present**, its value **0** indicates **DFA-Type 0** (see below) and its value **1** indicates **DFA-Type 1**. When this **bit** is **present**, it is the **most significant bit** of the "***Frequency domain resource assignment***" field.

The **bitlength** and **interpretation** of the **subfield** depends on the **DCI message Format**, on the **search space** and on the **type** of **frequency-domain resource allocation**.

There are **2 types** of **Downlink frequency-domain resource allocation**, which are **DFA-Type 0** and **DFA-Type 1**:

- **DFA-Type 0** (**bitmap based**, click 6.14.3.2)

  The **subfield** within the "***Frequency domain resource assignment***" **is** a **bitmap** that indicates a **set** of individual **Resource Block Groups** (**RBGs**, click 2.2.9).

- **DFA-Type 1** (**start-and-length**, click 6.14.3.3)

  The **subfield** within the "***Frequency domain resource assignment***" **is** a **resource indicator value** (**RIV**) that identifies a **block** of **contiguous Virtual Resource Blocks** (**VRBs**).

  The mapping from **VRBs** to **PRBs** may be **non-interleaved** or **interleaved** (click 4.12.15).

For the **selection** of the **DFA-Type**, click 6.14.3.1.

**Resource Block indexing**

The **Resource Block indexing** used in **PDSCH frequency-domain resource allocation** corresponds to **either** of the following cases:

- **RB indexing from the CORESET in which the scheduling DCI message was received**

  This case is for **DFA-Type 1** when the **scheduling DCI message** is **Format 1_0**, it is received within **any type** of **PDCCH common search space and CORESET**

- **RB indexing from CORESET#0**

  This case is for **DFA-Type 1** when the **scheduling DCI message** is **Format 1_0**, it is received within **any type** of **PDCCH common search space** and **CORESET#0** is **configured** for the **cell**.

- **RB indexing from the initial bandwidth part**

  This case is for **DFA-Type 1** when the **scheduling DCI message** is **Format 1_0**, it is received within **any type** of **PDCCH common search space** and **CORESET#0** is **not configured** for the **cell**.

- **RB indexing from the currently active Downlink bandwidth part**

  This case is when **either** [the **scheduling DCI message** [does **not** include the "***Bandwidth part indicator***" field] and [it is **not** a **Format 1_0 message** received in **any type** of **PDCCH common search space**]] or [the **UE** does **not support active bandwidth part change** via **DCI messages**].

- **RB indexing from the new active Downlink bandwidth part**

  This case is when [the **scheduling DCI message includes** the "***Bandwidth part indicator***" field] and [it is **not** a **Format 1_0 message** received in **any type** of **PDCCH common search space**] **and** [the **UE supports active bandwidth part change** via **DCI messages**]. The **new active Downlink bandwidth part** is the one indicated in the "***Bandwidth part indicator***" field.

**Regardless** of the case, **RB indexing** starts from the **lowest frequency RB**.

The following table shows, for **PDSCH assignments**, the **DCI message Formats** where the said **field** is **present**, what is **assigned** by **each** of these **DCI Formats** and what is the **field**'s **bitlength** in **each** of these **DCI Formats**:

| **DCI message field "*Bandwidth part indicator*"** | | |
|---|---|---|
| **Assigns** | **DCI Format** | **Bitlength** |
| **PDSCH Unicast** | Format 1_1 (6.9.6) | 0 to 2 |
| | Format 1_2 (6.9.7) | 0 to 2 |

The **UE** shall upon detection of **PDCCH** intended for the **UE** determine **first** the corresponding **Downlink bandwidth part** (or **CORESET**) and **then** the **resource allocation** within the **bandwidth part**.

**PDCCH reception with 2 PDCCH candidates**

When the **PDCCH** reception includes **2 PDCCH candidates** from **2 r**espective **search spaces**, as described in clause 10.1 of TS 38.213, for the purpose of determining the **Downlink RB set** of a **PDSCH** when scheduled by **DCI Format 1_0**, the **CORESET** with **lower ID** among the **2 CORESETs** that are associated with the **2 PDCCH candidates is used**.

For more detail on **DCI message fields**, click 6.9.1.

For more detail on **DCI messages**, click 6.9 and refer to clause 7.3 of TS 38.212.

For more detail on **Downlink frequency resource allocation**, refer to clause 5.1.2.2 of TS 38.214.

#### 6.14.3.1 Selection of the type of PDSCH frequency-domain resource allocation

The **UE** selects the **type** of **PDSCH frequency-domain resource allocation** (i.e., the **DFA-Type**) according to the following rules.

**DFA-Type 0 is used in the following cases:**

- When [the **DCI scheduling message** is **Format 1_1**]

  and [ [`resourceAllocation` from the `PDSCH-Config` (click 6.18.6) **instance** associated to `pdsch-Config` is set to `resourceAllocationType0`]

  or [ [`resourceAllocation` from the `PDSCH-Config` (click 6.18.6) **instance** associated to `pdsch-Config` is set to `dynamicSwitch`]

  and [the **most significant bit** of the "***Frequency domain resource***" **field** from the **scheduling DCI message** is **0**]]].

- When [the **DCI scheduling message** is **Format 1_2**]

  and [ [`resourceAllocationDCI-1-2-r16` from the `PDSCH-Config` (click 6.18.6) **instance** associated to `pdsch-Config` is set to `resourceAllocationType0`]

  or [ [`resourceAllocation-r16` from the `PDSCH-Config` (click 6.18.6) **instance** associated to `pdsch-Config` is set to `dynamicSwitch`]

  and [the **most significant bit** of the "***Frequency domain resource***" **field** from the **scheduling DCI message** is **0**]]].

- When [the **DCI scheduling message** is **Format 4_2**]

  and [ [`resourceAllocation` from the `PDSCH-Config` (click 6.18.6) **instance** associated to `pdsch-ConfigMulsticast-r17` is set to `resourceAllocationType0`]

  or [ [`resourceAllocation` from the `PDSCH-Config` (click 6.18.6) **instance** associated to `pdsch-ConfigMulticast-r17` is set to `dynamicSwitch`]

  and [the **most significant bit** of the "***Frequency domain resource***" **field** from the **scheduling DCI message** is **0**]]].

**DFA-Type 1 is used in the following cases:**

- When [the **DCI scheduling message** is **Format 1_0**, **4_0** or **4_1**]

- When [the **DCI scheduling message** is **Format 1_1**]

  and [ [`resourceAllocation` from the `PDSCH-Config` (click 6.18.6) **instance** associated to `pdsch-Config` is set to `resourceAllocationType1`]

  or [ [`resourceAllocation` from the `PDSCH-Config` (click 6.18.6) **instance** associated to `pdsch-Config` is set to `dynamicSwitch`]

  and [the **most significant bit** of the "***Frequency domain resource***" **field** from the **scheduling DCI message** is **1**]]].

- When [the **DCI scheduling message** is **Format 1_2**]

  and [ [`resourceAllocationDCI-1-2-r16` from the `PDSCH-Config` (click 6.18.6) **instance** associated to `pdsch-Config` is set to `resourceAllocationType1`]

  or [ [`resourceAllocation-r16` from the `PDSCH-Config` (click 6.18.6) **instance** associated to `pdsch-Config` is set to `dynamicSwitch`]

  and [the **most significant bit** of the "***Frequency domain resource***" **field** from the **scheduling DCI message** is **1**]]].

- When [the **DCI scheduling message** is **Format 4_2**]

  and [ [`resourceAllocation` from the `PDSCH-Config` (click 6.18.6) **instance** associated to `pdsch-ConfigMulsticast-r17` is set to `resourceAllocationType1`]

  or [ [`resourceAllocation` from the `PDSCH-Config` (click 6.18.6) **instance** associated to `pdsch-ConfigMulticast-r17` is set to `dynamicSwitch`]

  and [the **most significant bit** of the "***Frequency domain resource***" **field** from the **scheduling DCI message** is **1**]]].

Summarizing the above description

- **Each PDSCH reception** that is **scheduled** with a **DCI message** with **Format 1_1**, **1_2** or **4_2** uses the **DFA-Type** that is **set** in the **parameter** "`resourceAllocation...`" that corresponds to the **DCI Format**. ***If*** this corresponding parameter is set to `dynamicSwitch`, ***then***, the **PDSCH reception** uses the **DFA-Type** that is indicated in the "***Frequency domain resource***" **field**.
- **Each PDSCH reception** that is **scheduled** with a **DCI message** with **Format 1_0**, **4_0** or **4_1** uses **DFA-Type 1**.

It is **assumed** that when `pusch-Config` is set to `dynamicSwitch` the **UE supports dynamic switch**.

For more detail on **DCI message fields**, click 6.9.1.

For more detail on **DCI messages**, click 6.9 and refer to clause 7.3 of TS 38.212.

For more detail on **Downlink frequency resource allocation**, refer to clause 5.1.2.2 of TS 38.214.

### 6.14.3.2 PDSCH DFA-Type 0 frequency-domain resource allocation

**DFA-Type 0** is **bitmap based**. In **DFA-Type 0**, the **subfield** (click 6.14.3) within the **DCI field** "***Frequency domain resource assignment***" is a **bitmap** that indicates which **Resource Block Groups** (**RBGs**, click 2.2.9) are **assigned** to the **UE** for the **1** to **8** **PDSCH reception**(**s**). The **bitmap** contains **1 bit** for **each** and **every RBG** in the **bandwidth part**, from **RBG0** (**lowest frequency**) to **RBGn** (**highest frequency**). **RBG0** is mapped to the **most significant bit** of the **bitmap**. If a given bit value is **1**, it implies that the corresponding **RBG** is **assigned** to the **UE**.

**Different UEs** can **share** the **same RBGs** as long as **either** their **PUSCH time-domain resources** are different or the **RBGs correspond to MBS services**.

The **RBs** in the **RBGs** are **both virtual resource blocks** (**VRBs**) and **physical resource blocks** (**PRBs**), because in **DFA-Type 0**, **non-interleaved VRB-to-PRB mapping** (click 4.12.15) is used, which implies that the **VRBs** are **the same** as **PRBs**.

For the specific case of **Downlink frequency-domain** allocation **DFA-Type 0** that is scheduled using a **DCI** with **CRC** masked by **G-RNTI** for **multicast** or by **GCS-RNTI**, the following **changes** are applied to the **determination** of the **RBGs** that is described in 2.2.9:

- "$N_{BWP}^{start}$" is the **CRB index** of the **lowest frequency PRB** of the corresponding **common frequency resource** (**CFR**), instead of the corresponding **bandwidth part**.
- "$N_{RB}^{BWP}$" is the **RB size** of the corresponding **common frequency resource** (**CFR**), instead of the corresponding **bandwidth part**.
- The value of `rbg-Size` from `PDSCH-Config` (click 6.18.6) is taken from the `PDSCH-Config` **instance** corresponding to `pdsch-ConfigMulticast-r17` from `CFR-ConfigMulticast-r17`, instead of from the `PDSCH-Config` **instance** corresponding to `pdsch-Config` from `BWP-DownlinkDedicated` (click 2.2.8.7).

For more detail on **DCI message fields**, click 6.9.1.

For more detail on **DCI messages**, click 6.9 and refer to clause 7.3 of TS 38.212.

For more detail on **Downlink frequency resource allocation DFA-Type 0**, refer to clause 5.1.2.2.1 of TS 38.214.

### 6.14.3.3 PDSCH DFA-Type 1 frequency-domain resource allocation

**DFA-Type 1** is **start-and-length based**.

**DFA-Type 1** can use **both non-interleaved** or **interleaved VRB-to-PRB mapping**. In the **non-interleaved VRB-to-PRB mapping**, the **VRBs** are **the same** as the **PRBs**. In the **interleaved VRB-to-PRB mapping**, there is a **mapping procedure** from **VRBs** to **PRBs** (click 4.12.15).

In **DFA-Type 1**, the **subfield** (click 6.14.3) within the **DCI field** "***Frequency domain resource assignment***" is a **resource indicator value** (**RIV**). This **RIV** indicates the **lowest frequency VRB** "$RB_{start}$", and the **quantity** "$L_{RBs}$" of **contiguous VRBs** that are **assigned** to the **UE**.

The **RIV** value is defined by:

***If*** $(L_{RBs} - 1) \leq floor(N_{RB}^{BWP} / 2)$

***then*** **RIV** = $N_{RB}^{BWP} (L_{RBs} - 1) + RB_{start}$

***else*** **RIV** = $N_{RB}^{BWP} (N_{RB}^{BWP} - L_{RBs} + 1) + (N_{RB}^{BWP} - 1 - RB_{start})$

where "$1 \leq L_{RBs} \leq (N_{RB}^{BWP} - RB_{start})$" and "$N_{RB}^{BWP}$" is the **quantity** of **resource blocks** of the **corresponding** (click 6.14.3) **bandwidth part**, **CORESET#n** or **CORESET#0**.

The **bitlength** of the **RIV** is the **smallest one** that can contain the **required quantity** of **values**.

**Case 1: derived DCI Format 1_0**

***If*** the **DCI size** for a **scheduling DCI message Format 1_0** received in a **UE-specific search space** (**USS**) is **derived** from the **size** of a **scheduling message DCI Format 1_0** received in a **common search space** (**CSS**) but it is **applied** to an **active bandwidth part with RB size** "$N_{RB}^{BWP,active}$", ***then***, the **DFA-Type 1** field consists of a **resource indicator value** (**RIV**) corresponding to a **starting resource block** "$RB_{start}$":

$RB_{start} = 0, K, 2K, \ldots, (N_{RB}^{BWP,initial} - 1) \bullet K$

and a **length** "$L_{RBs}$" expressed in **contiguous virtual resource blocks** (**VRBs**):

$L_{RBs} = K, 2K, \ldots, N_{RB}^{BWP,initial} \bullet K$

where the value of "$N_{RB}^{BWP,initial}$" is given by:

***If*** **CORESET#0** is **configured** for the **cell**,

***then*** "$N_{RB}^{BWP,initial}$" is the **RB size** of **CORESET#0**.

***else*** "$N_{RB}^{BWP,initial}$" is the **RB size** "$N_{RB}^{BWP,0}$" of the **Downlink initial bandwidth part**.

This **RIV** value is defined by:

***If*** $(L'_{RBs} - 1) \leq floor(N_{RB}^{BWP,initial} / 2)$

***then*** **RIV** = $N_{RB}^{BWP,initial} (L'_{RBs} - 1) + RB'_{start}$

***else*** **RIV** = $N_{RB}^{BWP,initial} (N_{RB}^{BWP,initial} - L'_{RBs} + 1) + (N_{RB}^{BWP,initial} - 1 - RB'_{start})$

where "$L'_{RBs} = L_{RBs} / K$", "$RB'_{start} = RB_{start} / K$", "$L'_{RBs} \leq (N_{RB}^{BWP,initial} - RB'_{start})$" and the value of "$K$" is given by:

***If*** $N_{RB}^{BWP,active} \leq N_{RB}^{BWP,initial}$

***then*** $K = 1$

***else*** $K = max(1, 2, 4, 8)$ such that "$K \leq floor(N_{RB}^{BWP,active} / N_{RB}^{BWP,initial})$"

**Case 2: DCI Format 1_2 in RGBs**

***If*** the **scheduling DCI message** is **Format 1_2**, ***then***, the **DFA-Type 1** field consists of a **resource indicator value** (**RIV**) corresponding to a **starting resource block group** (**RBG**, click 2.2.9) "$RBG_{start}$":

$RBG_{start} = 0, 1, \ldots, (N_{RBG}^{BWP} - 1)$

and a **length** "$L_{RBGs}$" expressed in **contiguous virtual resource block groups** (**VRBGs**):

$L_{RBGs} = 0, 1, \ldots, N_{RBG}^{BWP}$

where the **RBGs** are defined as described in 2.2.9, with the **only difference** that in this case "***P***" takes the value of **resourceAllocationType1GranularityDCI-1-2-r16** (from **PDSCH-Config**, click 6.18.6) if **present**, and if **absent** "***P***=1".

This **RIV** value is defined by:

***If*** $(L_{RBGs} - 1) \leq floor(N_{RBG}^{BWP} / 2)$

***then*** **RIV** = $N_{RBG}^{BWP} (L_{RBGs} - 1) + RBG_{start}$

***else*** **RIV** = $N_{RBG}^{BWP} (N_{RBG}^{BWP} - L_{RBGs} + 1) + (N_{RBG}^{BWP} - 1 - RBG_{start})$

where "$1 \leq L_{RBGs} \leq (N_{RBG}^{BWP} - RBG_{start})$".

For more detail on **DCI message fields**, click 6.9.1.

For more detail on **DCI messages**, click 6.9 and refer to clause 7.3 of TS 38.212.

For more detail on **Downlink frequency resource allocation DFA-Type 1**, refer to clause 5.1.2.2.2 of TS 38.214.

In **DFA-Type 1**, the field "***Frequency domain resource assignment***" of **DCI Format 4_x messages** is used **for multicast/broadcast** and has some **differences** from the case of **DCI Format 1_x messages**, that has been described in this section.

For the description, of **DFA-Type 1 for multicast/broadcast**, refer to clause 5.1.2.2.3 of TS 38.214.

Source web pages:

- http://howltestuffworks.blogspot.com/2020/04/5g-nr-pdsch-resource-allocation-in.html

## 6.14.4 PDSCH time-domain resources

The **dynamic scheduling** of **PDSCH time-domain resources** is encoded in the **DCI message field** "***Time domain resource assignment***" (click 7.15.5) of **DCI Formats 1_x** and **4_x**.

**Adding 1** to the **value** "***m***" contained in "***Time domain resource assignment***" results in an **index**. This **index** "***m***+1" **identifies 1 entry** from a **PDSCH time-domain resource allocation list**. The **list** to use is **1 among several ones**.

The **UE first** determines **which PDSCH time-domain resource allocation list** to **use**, as described in 6.14.4.1.

The **UE then** extracts the corresponding **entry** from the **list** and **applies** it, as described in 6.14.4.2. The resulting **list entry** contains from **1** to **8 PDSCH time-domain resources**. **Each PDSCH time-domain resource** corresponds to **1** of the **1** to **8 PDSCH assignments** that can be performed with the **scheduling DCI message**. **Each PDSCH time-domain resource** indicates the **time** between the **reception** of the **scheduling DCI message** and the **actual reception** of the corresponding **PDSCH transmission**. The **list entry** contains **other** configuration **information**, such as the **PDSCH DM-RS mapping type**.

The following table shows, for **PDSCH scheduling**, the **DCI message Formats** where the said **field** is **present**, what is **assigned** by **each** of these **DCI Formats** and what is the **field**'s **bitlength** in **each** of these **DCI Formats**:

| | **DCI message field "*Time domain resource assignment*"** | |
|---|---|---|
| **Assigns** | **DCI Format** | **Bitlength** |
| **PDSCH U/B** | Format 1_0 (6.9.5) | 4 |
| **PDSCH Unicast** | Format 1_1 (6.9.6) | 0 to 6 |
| | Format 1_2 (6.9.7) | 0 to 4 |
| **PDSCH MBS** | Format 4_0 (6.9.9.1) | 4 |
| | Format 4_1 (6.9.9.2) | 4 |
| | Format 4_2 (6.9.9.3) | 0 to 4 |

The text "**U/B**" stands for **unicast** or **broadcast**.

The **field**'s **bitlength** in **each case** is sufficient to **index all** the **existing entries** in the corresponding **PDSCH time-domain resource allocation list**.

**Each DCI Format 1_1 message** can **schedule** from **1** to **8 PDSCH receptions** (i.e., from **1** to **8 PDSCH assignments**).

For more detail on **DCI messages**, click 6.9 and refer to clause 7.3 of TS 38.212.

For more detail on **PDSCH resource allocation** in the **time domain**, refer to clause 5.1.2.1 of TS 38.214.

Source web pages:

- http://howltestuffworks.blogspot.com/2020/04/5g-nr-pdsch-resource-allocation-in.html
- http://howltestuffworks.blogspot.com/2019/12/5g-nr-pdsch-resource-allocation-in-time.html

### 6.14.4.1 PDSCH time-domain resource allocation lists

For **each incoming DCI Format 1_x** or **4_y message**, the **UE** selects **1 PDSCH time-domain resource allocation list** among the following ones:

- **Default A** for **normal cyclic prefix** (table 5.1.2.1.1-2 of TS 38.214). The **list** in this table has **16 entries**.
- **Default A** for **extended cyclic prefix** (table 5.1.2.1.1-3 of TS 38.214). The **list** in this table has **16 entries**.
- **Default B** (table 5.1.2.1.1-4 of TS 38.214). The **list** in this table has **16 entries**.
- **Default C** (table 5.1.2.1.1-5 of TS 38.214). The **list** in this table has **16 entries**.
- The **list** contained in `pdsch-TimeDomainAllocationList` from `PDSCH-ConfigCommon` (click 6.18.57.15.4), with **data type** . This **list** can have **up to** `16` **entries**.
- The **list** contained in `pdsch-TimeDomainAllocationList` from `PDSCH-Config` (click 6.18.6). This **list** can have **up to** `16` **entries**.

- The **list** contained in `pdsch-TimeDomainAllocationList-r16` also from `PDSCH-Config`. This **list** can have **up to** `16` **entries**.
- The **list** contained in `pdsch-TimeDomainAllocationListDCI-1-2-r16`, also from `PDSCH-Config`. This **list** can have **up to** `16` **entries**.
- The **list** contained in `pdsch-TimeDomainAllocationList-r17`, from `PDSCH-ConfigBroadcast-r17`. This **list** can have **up to** `16` **entries**.
- The **list** contained in `pdsch-TimeDomainAllocationListForMultiPDSCH-r17` also from `PDSCH-Config`. This **list** can have **up to** `64` **multi-entries**, where **each multi-entry** corresponds to a **multiple PDSCH assignment** that contains `1` to `8` **entries**.

The **first** and **second parameters** above have **data type** `PDSCH-TimeDomainResourceAllocationList` (see within 6.18.5 and 6.18.6.1) which is a `SEQUENCE OF 1` to `16` **instances** of `PDSCH-TimeDomainResourceAllocation` (click 6.18.8). The **third**, **fourth** and **fifth parameters** above have **data type** `PDSCH-TimeDomainResourceAllocationList-r16` (see within 6.18.6.1), which is a `SEQUENCE OF 1` to `16` `PDSCH-TimeDomainResourceAllocation-r16`. The **sixth parameter** above has **data type** `MultiPDSCH-TDRA-List-r17` (see within 6.18.6.1), which is a `SEQUENCE OF 1` to `64` **instances** of `MultiPDSCH-TDRA-r17` (see within 6.18.6.1). This last **data type** is a `SEQUENCE OF 1` to `8` **instances** of `PDSCH-TimeDomainResourceAllocation-r16` (click 6.18.9).

The specific **PDSCH time-domain resource allocation list** to use is determined by **either** of the following **2 tables** of TS 38.214. Which **table** to use among **these 2** depends on the **DCI Format** of the **scheduling message**, as indicated below:

For **DCI message Formats 1_0**, **1_1**, **4_0**, **4_1** and **4_2**, the **PDSCH time-domain resource allocation list** is determined by table 5.1.2.1.1-1:

| RNTI Type | Search space type | SSB and CORESET multiplexing pattern | pdsch-TimeDomainAllocationList from PDSCH-ConfigCommon (6.18.5) | pdsch-TimeDomainAllocationList from PDSCH-Config (6.18.6) | [pdsch-TimeDomainAllocationList-r17 from PDSCH-ConfigBroadcast-r17 either within CFR-ConfigMCCH-MTCH-r17 or within MBSBroadcastConfiguration-r17-IEs] or [pdsch-TimeDomainAllocationList-r16 from PDSCH-Config within CFR-ConfigMulticast-r17] | pdsch-TimeDomainAllocationListForMultiPDSCH-r17 from PDSCH-Config (6.18.6) | PDSCH time-domain resource allocation list to be used |
|---|---|---|---|---|---|---|---|
| SI-RNTI | CSS-Type 0 | 1 | - | - | - | - | Default A for normal CP |
| | | 2 | - | - | - | - | Default B |
| | | 3 | - | - | - | - | Default C |

| RNTI Type | Search space type | SSB and CORESET multiplexing pattern | pdsch-TimeDomainAllocationList from PDSCH-ConfigCommon (6.18.5) | pdsch-TimeDomainAllocationList from PDSCH-Config (6.18.6) | [pdsch-TimeDomainAllocationList-r17 from PDSCH-ConfigBroadcast-r17 either within CFR-ConfigMCCH-MTCH-r17 or within MBSBroadcastConfiguration-r17-IEs] or [pdsch-TimeDomainAllocationList-r16 from PDSCH-Config within CFR-ConfigMulticast-r17] | pdsch-TimeDomainAllocationListForMultiPDSCH-r17 from PDSCH-Config (6.18.6) | PDSCH time-domain resource allocation list to be used |
|---|---|---|---|---|---|---|---|
| SI-RNTI | CSS-Type 0A | 1 | Absent | - | - | - | Default A |
| | | 2 | Absent | - | - | - | Default B |
| | | 3 | Absent | - | - | - | Default C |
| | | 1,2,3 | Present | - | - | - | pdsch-TimeDomainAllocationList from PDSCH-ConfigCommon |
| RA-RNTI, MSGB-RNTI, TC-RNTI | CSS-Type 1 | 1,2,3 | Absent | - | - | - | Default A |
| | | 1,2,3 | Present | - | - | - | pdsch-TimeDomainAllocationList from PDSCH-ConfigCommon |
| P-RNTI | CSS-Type 2 | 1 | Absent | - | - | - | Default A |
| | | 2 | Absent | - | - | - | Default B |
| | | 3 | Absent | - | - | - | Default C |
| | | 1,2,3 | Present | - | Absent | - | pdsch-TimeDomainAllocationList from PDSCH-ConfigCommon |
| MCCH-RNTI | CSS-Type 0/0B for broadcast | 1 | Absent | - | Absent | - | Default A |
| | | 2 | Absent | - | Absent | - | Default B |
| | | 3 | Absent | - | Absent | - | Default C |
| | | 1,2,3 | Present | - | Absent | - | pdsch-TimeDomainAllocationList from PDSCH-ConfigCommon |
| | | 1,2,3 | Absent or Present | - | Present | - | pdsch-TimeDomainAllocationList-r17 from PDSCH-ConfigBroadcast-r17 within CFR-ConfigMCCH-MTCH-r17 |

| RNTI Type | Search space type | SSB and CORESET multiplexing pattern | pdsch-TimeDomainAllocationList from PDSCH-ConfigCommon (6.18.5) | pdsch-TimeDomainAllocationList from PDSCH-Config (6.18.6) | [pdsch-TimeDomainAllocationList-r17 from PDSCH-ConfigBroadcast-r17 either within CFR-ConfigMCCH-MTCH-r17 or within MBSBroadcastConfiguration-r17-IEs] or [pdsch-TimeDomainAllocationList-r16 from PDSCH-Config within CFR-ConfigMulticast-r17] | pdsch-TimeDomainAllocationListForMultiPDSCH-r17 from PDSCH-Config (6.18.6) | PDSCH time-domain resource allocation list to be used |
|---|---|---|---|---|---|---|---|
| G-RNTI for broadcast | CSS-Type 0/0B for broadcast | 1 | Absent | - | Absent | - | Default A |
| | | 2 | Absent | - | Absent | - | Default B |
| | | 3 | Absent | - | Absent | - | Default C |
| | | 1,2,3 | Present | - | Absent | - | pdsch-TimeDomainAllocationList from PDSCH-ConfigCommon |
| | | 1,2,3 | Absent or Present | - | Present | - | pdsch-TimeDomainAllocationList-r17 from PDSCH-ConfigBroadcast-r17, if present within MBSBroadcastConfiguration-r17-IEs, otherwise within CFR-ConfigMCCH-MCCH-r17 |
| C-RNTI, MCS-C-RNTI, CS-RNTI | Any common search space associated with CORESET#0 | 1, 2, 3 | Absent | - | - | - | Default A |
| | | 1, 2, 3 | Present | - | - | - | pdsch-TimeDomainAllocationList from PDSCH-ConfigCommon |
| C-RNTI, MCS-C-RNTI, CS-RNTI | Any common search space not associated with CORESET#0 UE specific search space | 1,2,3 | Absent | Absent | - | - | Default A |
| | | 1,2,3 | Present | Absent | - | - | pdsch-TimeDomainAllocationList from PDSCH-ConfigCommon |
| | | 1,2,3 | Absent or Present | Present | - | - | pdsch-TimeDomainAllocationList from PDSCH-Config |
| | | 1,2,3 | Absent or Present | - | - | Present *but only for Format 1_1* | pdsch-TimeDomainAllocationListForMultiPDSCH-r17 from PDSCH-Config |

| RNTI Type | Search space type | SSB and CORESET multiplexing pattern | pdsch-TimeDomainAllocationList from PDSCH-ConfigCommon (6.18.5) | pdsch-TimeDomainAllocationList from PDSCH-Config (6.18.6) | [pdsch-TimeDomainAllocationList-r17 from PDSCH-ConfigBroadcast-r17 either within CFR-ConfigMCCH-MTCH-r17 or within MBSBroadcastConfiguration-r17-IEs] or [pdsch-TimeDomainAllocationList-r16 from PDSCH-Config within CFR-ConfigMulticast-r17] | pdsch-TimeDomainAllocationListForMultiPDSCH-r17 from PDSCH-Config (6.18.6) | PDSCH time-domain resource allocation list to be used |
|---|---|---|---|---|---|---|---|
| G-RNTI for multicast, G-CS-RNTI | CSS-Type 3 common search space for multicast | 1,2,3 | | - | Absent | - | Default A |
| | | 1,2,3 | Present | - | Absent | - | pdsch-TimeDomainAllocationList from PDSCH-ConfigCommon (Note 1) |
| | | 1,2,3 | Absent or Present | - | Present | - | pdsch-TimeDomainAllocationList-r16 from PDSCH-Config, within CFR-ConfigMulticast-r17 (Note 1) |
| Note 1: For a **UE** that supports **multicast**, the same **TDRA** table applies to **all G-RNTIs** and **G-CS-RNTIs** (configured for **multicast**) if configured on a given serving cell. | | | | | | | |

For **DCI message Format 1_2**, the **PDSCH time-domain resource allocation list** is determined by table 5.1.2.1.1-1A:

| pdsch-TimeDomainAllocationList from PDSCH-ConfigCommon (6.18.5) | pdsch-TimeDomainAllocationList from PDSCH-Config (6.18.6) | pdsch-TimeDomainAllocationListDCI-1-2 from PDSCH-Config (6.18.6) | PDSCH time-domain resource allocation list to be used |
|---|---|---|---|
| Absent | Absent | Absent | Default A |
| Yes | Absent | Absent | pdsch-TimeDomainAllocationList from PDSCH-ConfigCommon |
| Present or Absent | Present | Absent | pdsch-TimeDomainAllocationList from PDSCH-Config |
| Present or Absent | Present or Absent | Present | pdsch-TimeDomainAllocationListDCI-1-2 from PDSCH-Config |

### 6.14.4.2 PDSCH time-domain resource entries

The **UE** has already obtained **1 PDSCH time-domain resource allocation list** as it has been detailed in 6.14.4.1.

The **UE** then selects the "***m***+1" **list entry**, where "***m***" is the value from the field "***Time domain resource assignment***" (click 7.15.5) from the **scheduling DCI message**.

The **selected** "***m***+1" **list entry** provides the **UE** with the **time-domain resource assignment(s) either** for **1 PDSCH transmission** or for **1** to **8 PDSCH transmissions**

(for the case of a **multi-PDSCH assignment**). A **multi-PDSCH assignment** is **only** performed with a **PDSCH time-domain resource allocation list** that is **contained** in an **instance** of the **data type** `MultiPDSCH-TDRA-List-r17` (see within 6.18.6.1).

**Each PDSCH time-domain resource assignment** (corresponding to **1 PDSCH transmission**) provides the following information:

- The **slot offset** "$K_0$" between the **slot** of the **scheduling DCI message** and the **slot** of the **PDSCH reception**.
- **Either** [a **start-and-length indicator** (**SLIV**)]

  or [[the **first symbol interval** "***S***"] and [the **allocated quantity** of **symbol intervals** "***L***"]]

  that identifies the **symbol intervals** of the **PDSCH reception**.
- The **PDSCH DM-RS mapping type**: `typeA` or `typeB`.
- If `repetitionNumber-r16` and/or `repetitionNumber-v1730` (**both** from `PDSCH-TimeDomainResourceAllocation-r16`, click 6.18.9) is **present**, the **quantity** of **repetitions** (click 7.8.2.3.2) to be **applied** to the **PDSCH transmission**.

**Obtaining the values from a list entry**

For the case of a **PDSCH time-domain resource allocation list** (click 6.14.4.1) contained in a **RRC** parameter, the **selected** "***m***+1" **list entry** provides the corresponding values (e.g., `k0`, `mappingType`, or `startSymbolAndLength`).

However, for the case of a **PUSCH time-domain resource allocation list** (click 6.14.4.1) obtained from a **default table**, **selected** "***m***+1" **list entry** contains, on some cases, **2 subentries**. In such cases, the **applicable subentry** depends on the value of `dmrs-TypeA-Position`, **either** from `MIB` (click 8.12.3) or from `ServingCellConfigCommon` (click 3.9.6).

As one example of a **PDSCH time-domain resource allocation list** obtained from a **default table**, the following is the table **Default A** for **normal cyclic prefix** (which is table 5.1.2.1.1-2 of TS 38.214):

| Row index | dmrs-TypeA-Position | PDSCH DM-RS mapping type | $K_0$ | S | L |
|---|---|---|---|---|---|
| 1 | 2 | typeA | 0 | 2 | 12 |
| | 3 | typeA | 0 | 3 | 11 |
| 2 | 2 | typeA | 0 | 2 | 10 |
| | 3 | typeA | 0 | 3 | 9 |
| 3 | 2 | typeA | 0 | 2 | 9 |
| | 3 | typeA | 0 | 3 | 8 |
| 4 | 2 | typeA | 0 | 2 | 7 |
| | 3 | typeA | 0 | 3 | 6 |
| 5 | 2 | typeA | 0 | 2 | 5 |
| | 3 | typeA | 0 | 3 | 4 |
| 6 | 2 | typeB | 0 | 9 | 4 |
| | 3 | typeB | 0 | 10 | 4 |
| 7 | 2 | typeB | 0 | 4 | 4 |
| | 3 | typeB | 0 | 6 | 4 |
| 8 | 2 or 3 | typeB | 0 | 5 | 7 |

| Row index | dmrs-TypeA-Position | PDSCH DM-RS mapping type | $K_0$ | S | L |
|---|---|---|---|---|---|
| 9 | 2 or 3 | typeB | 0 | 5 | 2 |
| 10 | 2 or 3 | typeB | 0 | 9 | 2 |
| 11 | 2 or 3 | typeB | 0 | 12 | 2 |
| 12 | 2 or 3 | typeA | 0 | 1 | 13 |
| 13 | 2 or 3 | typeA | 0 | 1 | 6 |
| 14 | 2 or 3 | typeA | 0 | 2 | 4 |
| 15 | 2 or 3 | typeB | 0 | 4 | 7 |
| 16 | 2 or 3 | typeB | 0 | 8 | 4 |

**Each entry** (or **subentry**) in this **table** contains **1 PDSCH time-domain resource**.

**Symbol intervals of the PDSCH reception**

The **UE** determines the **specific symbol intervals** of the **PDSCH reception** from the following variables:

- **Slots offset "$K_0$":**

  **Offset** from **subframe slot index** "$S_{DCI}$" of the **slot** that delivered the **DCI message** to the **subframe slot index** "$S_{PDSCH}$" of the **slot** of the **PDSCH reception**. The value of "$S_{PDSCH}$" is computed as:

  $$S_{PDSCH} = floor(S_{DCI} \cdot \mu_{PDSCH} / \mu_{PDCCH}) + K_0$$

  The value of "$K_0$" in the **Default lists** can be **0** or **1**. ***If* both** values of "**μ**" are the **same**, if "$K_0$=**0**", the **PDSCH reception** is in the **current** slot (called a "**self-contained**" slot), which is a quite frequent case. ***Else***, if "$K_0$=**1**", the **PDSCH reception** is in the **next** slot, this is, the **slot right after** the **slot** that delivered the **DCI message**.

  The value of "$K_0$" in the **explicit lists** configured using **RRC** may be from `0` to `32`.

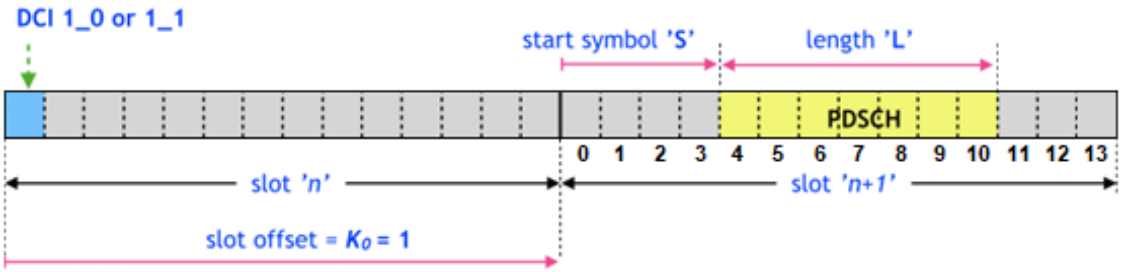
- **Start symbol "*S*"**:

  **Index** of the **first symbol interval** of **PDSCH** within the "$S_{PDSCH}$" slot (see the previous bullet).

- **Symbol length "*L*"**:

  **Quantity of symbol intervals** of **PDSCH** within the "$S_{PDSCH}$".

The following figure shows an example for the following values: "$K_0$ = **1**", "***S*** = **4**" and "***L*** = **7**":




In this example, **both** the **slot** of the **DCI message** and the **reception slot** have the **same numerology**. In case that they have a **different numerology**, the expression that

calculates the **PDSCH reception slot** "$S_{PDSCH}$" contains an **adjustment quotient**, as indicated above.

For **shared spectrum** in **FR1**, the **UE** reinterprets "***S***" and "***L***" in row 9 of Table 5.1.2.1.1-2 as "***S***=6" and "***L***=7" as described in TS 37.213.

It is **important to highlight** that there are several **specific cases** that imply **modifications** to the **procedures** described in this section. For more detail on these specific cases, refer to clause 5.1.2.1 of TS 38.214.

If a **UE** is configured with `pdsch-TimeDomainAllocationListForMultiPDSCH-r17` in which one or more rows contain multiple **PDSCH time-domain resources**, the **UE** does **not** expect to be configured with **neither** `repetitionNumber-r16` **neither** `repetitionNumber-r17` within the **instances** of `PDSCH-TimeDomainResourceAllocation-r16` (click 6.18.9) that correspond to `pdsch-TimeDomainAllocationListForMultiPDSCH-r17` from `PDSCH-Config` (click 6.18.6): refer to clause 5.1.2.1 of TS 38.214).

## 6.14.5 PDSCH scheduling of payload PREs, DM-RS PREs, PT-RS PREs and empty PREs

The **PDSCH scheduled Physical Resource Elements** (**PREs**) defined in the previous subsections can contain **PDSCH payload PREs**, **PDSCH Demodulation Reference Signals** (**DM-RS**) **PREs** (click 6.14.5.1, 6.14.5.2 and 6.14.5.3), **PDSCH Phase-Tracking Reference Signals** (**PT-RS**) **PREs** and **PDSCH empty** (no signal) **PREs** (click 6.14.5.4).

The **PDSCH payload PREs** are the **PDSCH scheduled PREs not** used for **any** of the other purposes just indicated. In more detail, the **PDSCH payload PREs** are the **PDSCH scheduled PREs** that satisfy all of the following requirements:

- The **Physical Resource Blocks** where they are contained are declared as available for **PDSCH** according to clause 5.1.4 of TS 38.214.
- They are **not** declared as "**not available** for **PDSCH**" (refer to clause 5.1.4 of TS 38.214).
- They are **not** used for:
  - **PDSCH DM-RS**, nor for **DM-RS** intended for other co-scheduled **UEs** (refer to clause 7.4.1.1.2 of TS 38.211).
  - **PDSCH PT-RS** (refer to clause 7.4.1.2.2 of TS 38.211).
  - Non-zero power **CSI-RS** (refer to clause 7.4.1.5.3 of TS 38.211).

**All** the **UEs scheduled** to receive a **PDSCH transmission must** receive the **PDSCH DM-RS PREs** that it contains, to support the demodulation of the **PDSCH payload PREs**.

The **scheduling** of **PDSCH DM-RS PREs**, **within** the **PDSCH** scheduled **PREs**, is **independently** specified for the **time domain** (click 6.14.5.1), for the **frequency domain** (click 6.14.5.2) and for the **space domain** (click 6.14.5.3).

For more detail on the **scheduling** of **PDSCH DM-RS**, refer to clauses 7.4.1.1.1 and 7.4.1.1.2 of TS 38.211 and to clause 5.1.6.2 of TS 38.214.

For more detail on the **scheduling** of **PDSCH PT-RS**, refer to clauses 7.4.1.2.1 and 7.4.1.2.2 of TS 38.211 and 5.1.6.3 of TS 38.214.

### 6.14.5.1 PDSCH DM-RS time-domain scheduling

The **symbol intervals** that contain **DM-RS** are identified with variable "*si*", that can contain **several values**. The **time reference** for "*si*" depends on the **PDSCH DM-RS mapping type**, as follows:

- For **PDSCH DM-RS mapping** `typeA`:

  "*si*" is indicated **relative** to the **first symbol interval** of the **slot**.

- For **PDSCH DM-RS mapping** `typeB`:

  "*si*" is indicated **relative** to the **first symbol interval** of the **PDSCH transmission**.

The **PDSCH DM-RS mapping type** is determined by:

- `mappingType` from `PDSCH-TimeDomainResourceAllocation` (click 6.18.8). This is a **mandatory** parameter.
- `mappingType-r16` from `PDSCH-TimeDomainResourceAllocation-r16` (click 6.18.9). This is a **mandatory** parameter.

The **data type** of these **2** parameters is an `ENUMERATED{typeA, typeB}`.

The **value(s)** of the **symbol interval index** "*si*" of the **DM-RS PREs** is/are obtained as:

$$si = \bar{l} + si'$$

Notice that TS 38.211 uses the notation "$\bar{l}$" instead of "*si*" and "$\bar{l}'$" instead of "*si'*".

The **value(s)** of "*si'*" represent **single** or **double symbol DM-RS**:

- **Single-symbol DM-RS** <=> "$si'=0$"

  **Each value** of "$\bar{l}$" implies **1 symbol interval** for **DM-RS**.

  **Single-symbol DM-RS** is **used** when [`maxLength` from `DMRS-DownlinkConfig` (click 6.18.11) is **absent**] or when [[it is **present**] and [**single-symbol** is indicated by the "***Antenna ports***" (click 6.9.1) **field** of the **scheduling DCI message** (click 6.9)]].

- **Double-symbol DM-RS** <=> "$si' \in \{0, 1\}$"

  **Each value** of "$\bar{l}$" implies **2 consecutive symbol intervals** for **DM-RS**.

  **Double-symbol DM-RS** are **used** when [`maxLength` from `DMRS-DownlinkConfig` (click 6.18.11) is **present**] and [**double-symbol** is indicated by the "***Antenna ports***" **field** (click 6.9.1) of the **scheduling DCI message** (click 6.9)].

The following table shows, for **PDSCH scheduling**, the **DCI message Formats** where the said **field** is **present**, what is **assigned** by **each** of these **DCI Formats** and what is the **field**'s **bitlength** in **each** of these **DCI Formats**:

| DCI message field "*Antenna ports*" | | |
|---|---|---|
| **Assigns** | **DCI Format** | **Bitlength** |
| **PDSCH unicast** | Format 1_1 (6.9.6) | 4 to 6 |
| | Format 1_2 (6.9.7) | 0 to 3 |
| **PDSCH MBS** | Format 4_2 (6.9.9.3) | 4 to 6 |

The **semantics** of the "***Antenna ports***" **field** for **PDSCH** has **many cases**, and it is detailed in tables 7.3.1.2.2-**1** to **-4** and **-1A** to **-4A** of TS 38.212.

The **1** or **several value(s)** contained in "$\bar{l}$" are obtained from the following tables from TS 38.212:

- **For single-symbol DM-RS**

  Table 7.4.1.1.2-3 (**1** to **4** values).

- **For double-symbol DM-RS**

  Table 7.4.1.1.2-4 (**1** or **2** values).

The following is table 7.4.1.1.2-3 of TS 38.211:

| $l_d$ [symbol interv.] | Value(s) of "$\bar{l}$" for single-symbol DMRS | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | `mappingType=typeA` | | | | `mappingType=typeB` | | | |
| | `dmrs-AdditionalPosition` | | | | `dmrs-AdditionalPosition` | | | |
| | `pos0` | `pos1` | `pos2` | `pos3` | `pos0` | `pos1` | `pos2` | `pos3` |
| **2** | - | - | - | - | 0 | 0 | 0 | 0 |
| **3** | 2 | 2 | 2 | 2 | 0 | 0 | 0 | 0 |
| **4** | 2 | 2 | 2 | 2 | 0 | 0 | 0 | 0 |
| **5** | $l_0$ | $l_0$ | $l_0$ | 2 | 0 | 0, 4 | 0, 4 | 0, 4 |
| **6** | $l_0$ | $l_0$ | $l_0$ | 2 | 0 | 0, 4 | 0, 4 | 0, 4 |
| **7** | $l_0$ | $l_0$ | $l_0$ | 2 | 0 | 0, 4 | 0, 4 | 0, 4 |
| **8** | $l_0$ | $l_0$, 7 | $l_0$, 7 | 2, 7 | 0 | 0, 6 | 0, 3, 6 | 0, 3, 6 |
| **9** | $l_0$ | $l_0$, 7 | $l_0$, 7 | 2, 7 | 0 | 0, 7 | 0, 4, 7 | 0, 4, 7 |
| **10** | $l_0$ | $l_0$, 9 | $l_0$, 6, 9 | 2, 6, 9 | 0 | 0, 7 | 0, 4, 7 | 0, 4, 7 |
| **11** | $l_0$ | $l_0$, 9 | $l_0$, 6, 9 | 2, 6, 9 | 0 | 0, 8 | 0, 4, 8 | 0, 3, 6, 9 |
| **12** | $l_0$ | $l_0$, 9 | $l_0$, 6, 9 | 2, 5, 8, 11 | 0 | 0, 9 | 0, 5, 9 | 0, 3, 6, 9 |
| **13** | $l_0$ | $l_0$, $l_1$ | $l_0$, 7, 11 | 2, 5, 8, 11 | 0 | 0, 9 | 0, 5, 9 | 0, 3, 6, 9 |
| **14** | $l_0$ | $l_0$, $l_1$ | $l_0$, 7, 11 | 2, 5, 8, 11 | - | - | - | - |

In this table, "$l_1$" is **11**, except if **all** the following conditions **hold**, in which case it is **12**:

- **Either** `lte-CRS-ToMatchAround`, `lte-CRS-PatternList1`, or `lte-CRS-PatternList2` is **present**.
- `dmrs-AdditionalPosition=pos1` and "$l_0 = 3$".
- The **UE** has indicated it is capable of `additionalDMRS-DL-Alt`.

The following is table 7.4.1.1.2-4 of TS 38.211:

| $l_d$ [symbol interv.] | Value(s) of "$\bar{l}$" for double-symbol DMRS | | | | | |
|---|---|---|---|---|---|---|
| | mappingType=typeA | | | mappingType=typeB | | |
| | dmrs-AdditionalPosition | | | dmrs-AdditionalPosition | | |
| | pos0 | pos1 | | pos0 | pos1 | |
| <4 | - | - | | - | - | |
| 4 | 2 | 2 | | - | - | |
| 5 | $l_0$ | $l_0$ | | 0 | 0 | |
| 6 | $l_0$ | $l_0$ | | 0 | 0 | |
| 7 | $l_0$ | $l_0$ | | 0 | 0 | |
| 8 | $l_0$ | $l_0$ | | 0 | 0, 5 | |
| 9 | $l_0$ | $l_0$ | | 0 | 0, 5 | |
| 10 | $l_0$ | $l_0$, 8 | | 0 | 0, 7 | |
| 11 | $l_0$ | $l_0$, 8 | | 0 | 0, 7 | |
| 12 | $l_0$ | $l_0$, 8 | | 0 | 0, 8 | |
| 13 | $l_0$ | $l_0$, 10 | | 0 | 0, 8 | |
| 14 | $l_0$ | $l_0$, 10 | | - | - | |

Some **relevant remarks** on **both** tables are:

- The **selected row** "$l_d$" is as follows:
  - For **PDSCH DM-RS mapping** `typeA`, it is the **slot symbol index** of the **last symbol interval** of the **PDSCH transmission**, **plus 1**.
  - For **PDSCH DM-RS mapping** `typeB`, it is the **duration**, in **symbol intervals**, of the **PDSCH transmission** (i.e., of the **PDSCH assignment**).
- The **selected column** is **jointly obtained** with `mappingType` from `PDSCH-TimeDomainResourceAllocation` (click 6.18.8) and `dmrs-AdditionalPosition` from `DMRS-DownlinkConfig` (click 6.18.11). The possible values of `mappingType` are `typeA` or `typeB`. The possible values of `dmrs-AdditionalPosition` are `pos0`, `pos1`, `pos2`, and `pos3`.
- The **selected cell** contains the **1** to **4 value(s)** of "$\bar{l}$", that on some cases are referred to "$l_0$" (see next bullet) and/or to "$l_1$" (see above).
- "$l_0$" is **2** or **3**, as indicated by `dmrs-TypeA-Position`, **either** from `MIB` (click 8.12.3) or from `ServingCellConfigCommon` (click 3.9.6).

**PDSCH DM-RS mapping** `typeB` is subject to **several conditions** that are **detailed** in clause 7.4.1.1.2 of TS 38.211.

For more detail on the **time-domain scheduling** of **PDSCH DM-RS**, refer to clauses 7.4.1.1.1 and 7.4.1.1.2 of TS 38.211 and to clause 5.1.6.2 of TS 38.214.

### 6.14.5.2 PDSCH DM-RS frequency-domain scheduling

The **subcarriers** that contain **DM-RS** are identified with variable "$sc$", that contains **multiple values**. The **expression** for "$sc$" depends on the **DM-RS frequency configuration type**, as follows:

- For **DM-RS frequency configuration DDM-Type 1**:

$$sc = 4n + 2sc' + \Delta$$

- For **DM-RS frequency configuration DDM-Type 2**:

  $sc = 6n + sc' + \Delta$

where "$sc' \in \{0, 1\}$" and "$n \in \{0, 1, 2, \ldots\}$". For the **subcarrier offset** "**Δ**" see below.

Notice that TS 38.211 uses the notation "***k***" instead of "*sc*" and "***k'***" instead of "*sc'*".

The **DM-RS frequency configuration type** is determined by `dmrs-Type` from `DMRS-DownlinkConfig` (click 6.18.11). If it is **absent**, the **DM-RS frequency configuration** is **DDM-Type 1**. If it is **present**, the **configuration** is **DDM-Type 2**.

The **reference point** for **subcarrier** "*sc*=0" is:

- ***If*** the **scheduling DCI message** was received in **CORESET#0** within a **common search space CSS-Type 0** and its **CRC** was masked with **SI-RNTI**, ***then***,

  *sc*=0 is **subcarrier 0** of the **lowest-frequency resource block** of **CORESET#0**.

  This happens for example in **initial access** (click 8).

- ***Else***,

  *sc*=0 is **subcarrier 0** of the **Common Resource Block 0** (click 2.2.4).

The value of **subcarrier offset** "**Δ**" depends on the assignment of **antenna ports** (click 4.11.5) and is obtained from table 7.4.1.1.2-1 for **PDSCH DM-RS** configuration **DDM-Type 1** and 7.4.1.1.2-2 for **DDM-Type 2** (**both** tables from TS 38.211). These tables are copied below:

Table 7.4.1.1.2-1: variables for **PDSCH DM-RS** configuration **DDM-Type 1**

| *p* | CDM group λ | Δ | $w_f(sc')$ | | $w_t(si')$ | |
|---|---|---|---|---|---|---|
| | | | *sc' = 0* | *sc' = 1* | *sc' = 0* | *sc' = 1* |
| **1000** | 0 | 0 | +1 | +1 | +1 | +1 |
| **1001** | 0 | 0 | +1 | -1 | +1 | +1 |
| **1002** | 1 | 1 | +1 | +1 | +1 | +1 |
| **1003** | 1 | 1 | +1 | -1 | +1 | +1 |
| **1004** | 0 | 0 | +1 | +1 | +1 | -1 |
| **1005** | 0 | 0 | +1 | -1 | +1 | -1 |
| **1006** | 1 | 1 | +1 | +1 | +1 | -1 |
| **1007** | 1 | 1 | +1 | -1 | +1 | -1 |

Table 7.4.1.1.2-2: variables for **PDSCH DM-RS** configuration **DDM-Type 2**

| *p* | CDM group λ | Δ | $w_f(sc')$ | | $w_t(si')$ | |
|---|---|---|---|---|---|---|
| | | | *sc' = 0* | *sc' = 1* | *sc' = 0* | *sc' = 1* |
| **1000** | 0 | 0 | +1 | +1 | +1 | +1 |
| **1001** | 0 | 0 | +1 | -1 | +1 | +1 |
| **1002** | 1 | 2 | +1 | +1 | +1 | +1 |
| **1003** | 1 | 2 | +1 | -1 | +1 | +1 |
| **1004** | 2 | 4 | +1 | +1 | +1 | +1 |
| **1005** | 2 | 4 | +1 | -1 | +1 | +1 |
| **1006** | 0 | 0 | +1 | +1 | +1 | -1 |
| **1007** | 0 | 0 | +1 | -1 | +1 | -1 |
| **1008** | 1 | 2 | +1 | +1 | +1 | -1 |
| **1009** | 1 | 2 | +1 | -1 | +1 | -1 |
| **1010** | 2 | 4 | +1 | +1 | +1 | -1 |
| **1011** | 2 | 4 | +1 | -1 | +1 | -1 |

The assigned **antenna port number(s)** "*p*" (click 4.11.5) is/are entered into the corresponding table above to obtain **subcarrier offset** "**Δ**". The **subcarrier indexes** "*sc*" (where **PDSCH DM-RS** are located) are **shifted upwards** in the value of "**Δ**". **Each value** of "**Δ**" therefore indicates a different **PDSCH DM-RS group**.

The result of the **DM-RS frequency scheduling procedure** (as explained up to now) is the following:

- **DM-RS frequency scheduling DDM-Type 1**

  **1 PDSCH DM-RS symbol** is placed in **even subcarrier indexes** (i.e., {**0**, **2**, **4**, …}), plus **offset** "**Δ**". The value of "**Δ**" is **0** or **1** depending on the **antenna port number**, as indicated in the table above.

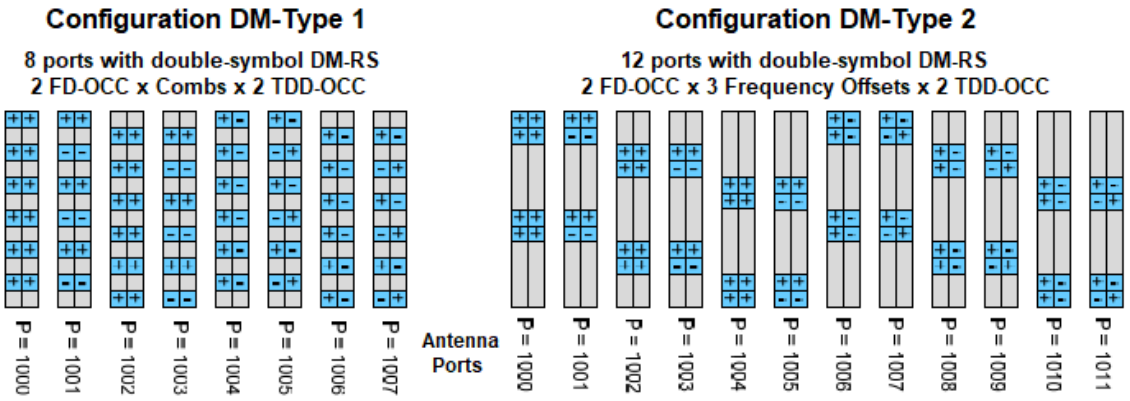
  **DDM-Type 1** with **single-symbol DM-RS** supports **4 DM-RS ports** (ports **1000** to **1003**) and with **double-symbol DM-RS** supports **8 DM-RS ports** (ports **1000** to **1007**).

- **DM-RS frequency scheduling DDM-Type 2**

  **1 PDSCH DM-RS symbol** is placed in **subcarrier indexes** {**0**, **1**, **6**, **7**, **11**, **12**, …} plus, an offset "**Δ**". The value of "**Δ**" is **0**, **2** or **4** depending on the **antenna port number**, as indicated in the table above.

  **DDM-Type 2** with **single-symbol DM-RS** supports **8 DM-RS ports** (ports **1000** to **1007**) and with **double-symbol DM-RS** supports **12 DM-RS ports** (ports **1000** to **1011**).

The following figure shows the difference between frequency configurations **DDM-Type 1** and **DDM-Type 2** (**both** with **double-symbol**) and the **frequency location impact** of "**Δ**", whose value results from the corresponding **antenna port number**:




In this figure, **subcarriers** are represented in **bottom-up ascending order**, from **0** to **11**. **Symbol intervals** are represented in **left-to-right ascending order**. For the purpose of the **PDCSCH DM-RS RE location**, you may ignore the "+" and the "-" in the figure. For further detail on the "+" and the "-", click 6.17.1 and refer to tables 7.4.1.1.2-**1** to **-5** of TS 38.211.

Notice that clauses 7.4.1.1 of TS 38.211 and 5.1.6.2 of TS 38.214 contain **clarifications** and **particular** cases that **modify** the **overall procedure** presented in this section.

For more detail on the **frequency-domain scheduling** of **PDSCH DM-RS**, refer to clauses 7.4.1.1.1 and 7.4.1.1.2 of TS 38.211 and to clause 5.1.6.2 of TS 38.214.

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_PDSCH_DM-RS.html
- https://www.nrexplained.com/dmrs_pdsch
- https://www.linkedin.com/pulse/5g-nr-pdsch-dmrs-figured-out-andrew-kolomatski/

### 6.14.5.3 PDSCH DM-RS space-domain scheduling

The usage of **single-symbol PDSCH DM-RS** or **double-symbol PDSCH DM-RS** has an impact on the **supported antenna ports**, as indicated in table 7.4.1.1.2-5 of TS 38.211:

| | *si'* | Supported antenna ports "*p*" | |
|---|---|---|---|
| | | Configuration DDM-Type 1 | Configuration DDM-Type 2 |
| **Single-symbol DM-RS** | 0 | 1000 to 1003 | 1000 to 1005 |
| **Double-symbol DM-RS** | 0, 1 | 1000 to 1007 | 1000 to 1011 |

This table shows which **DM-RS antenna ports** can be used for **each combination** of **DM-RS frequency configuration type** and **single/double DM-RS symbol**. For **single-symbol DM-RS**, **up to 4 DM-RS antenna ports** can be used for configuration **DDM-Type 1** and a **up to 6 DM-RS antenna ports** can be used for configuration **DDM-Type 2**. For **double-symbol DM-RS**, **up to 8 DM-RS antenna ports** can be used for configuration **DDM-Type 1** and a **up to 12 DM-RS antenna ports** can be used for configuration **DDM-Type 2**.

When **maxLength** from **DMRS-DownlinkConfig** (click 6.18.11) is **absent**, **single-symbol DM-RS** are used. If it is **present**, the usage of **single-symbol** or **double-symbol DM-RS** is determined by the scheduling **DCI message**.

This information is used to **select** which **antenna port table** is associated with **DCI Format 1_1** for setting up **SISO/MIMO.**

For more detail on the **space-domain scheduling** of **PDSCH DM-RS**, refer to clauses 7.4.1.1.1 and 7.4.1.1.2 of TS 38.211 and to clause 5.1.6.2 of TS 38.214.

### 6.14.5.4 PDSCH empty REs scheduling

The **PDSCH empty REs must be** present when the **quantity** of **PDSCH DM-RS groups** requires it. This happens in configuration **DDM-Type 1** when the quantity of **ports** is **larger than 2**, and in configuration **DDM-Type 2** when the quantity of **ports** is **larger than 4**. In the figure right above, you can see that when the quantity of **ports** is as indicated, the combination of the existing **PDSCH DM-RS groups occupy all** the subcarriers in the Resource Block, making **mandatory** to leave **PDSCH empty REs**. You may also see that when the quantity of **ports** is less than or equal the indicated value, you may have **PDSCH DM-RS empty REs**, but you could also use them as **PDSCH payload REs**. Whenever it is possible **both** to have **PDSCH empty RE groups** or to use that **REs** for **PDSCH payload**, the corresponding configuration is determined by **field** "***Antenna ports***" from **DCI message Formats 1_1** and **1_2** (click 6.9.6 and 6.9.7). This **field** is introduced in tables 7.3.1.2.2-1 to -4A (of TS 38.212), that provide the value of "**Number of DM-RS CDM group(s) without data**". The values of "**Number of DM-RS CDM group(s) without data**" can be **1** or **2** in **DDM-Type 1**, and **1**, **2** or **3** in **DDM-Type 2**. The actual value of "**Number of DM-RS CDM group(s) without data**" **minus 1** indicates the quantity of **PDSCH empty RE groups**.

In configuration **DDM-Type 1** you may have **0** or **1** groups of **PDSCH empty REs**. In configuration **DDM-Type 2** you may have **0**, **1** or **2** groups of **PDSCH empty REs**.

### 6.14.5.5 Examples of PDSCH DM-RS scheduling

The sections after this one contain several examples of **PDSCH DM-RS** configuration corresponding to various combinations of parameter values.

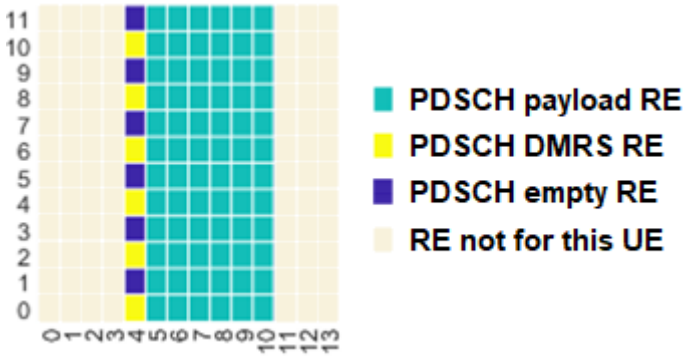
**All** the **examples** are for **PDSCH subcarrier spacing** of **30 kHz**, "$N_{ID}^{nSCID}$ = 10" and "$n_{SCID}$ = 0".

**All** the **examples** with **PDSCH DM-RS time-domain mapping** **typeA** are for a **PDSCH allocation** of **all** the **symbols** (**14 symbols**) in **1 Resource Block**. These examples are the ones in all the example sections, except the last one.

**All** the **examples** with **PDSCH DM-RS time-domain mapping** **typeB** are for a **PDSCH** allocation of **symbols 4** to **10** (**both** included) in **1 Resource Block**. These examples are the ones in the last example section.

**Each example** is composed of one set of **parameter values** followed by a figure representing the **location** of the **different REs**. The figure identifies with different **colors** the **PDSCH payload REs**, the **PDSCH DM-RS REs**, the **PDSCH empty** (no signal) **REs** and the **REs not assigned to this UE**. The following figure illustrates the format

used in the figures of the different examples. The rectangle represents **Resource Block 0** (with RB subcarriers 0 to 11) over **one slot** (slot symbols 0 to 13):




All the examples in the following subsections have been taken (having peformed some corrections and adjustments) from this web page:

https://www.sharetechnote.com/html/5G/5G_PDSCH_DM-RS.html

### 6.14.5.6 Examples of base parameter values, changing the quantity of layers

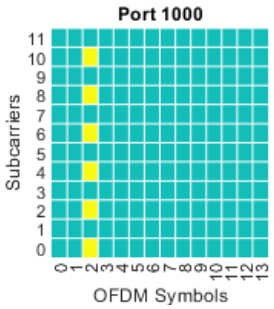
The example configuration in this **first subsection** is a **base configuration**. In **subsequent subsections** some of the configuration **parameters** will be **changed**, so you can see their comparative effect in the resulting layout of **PDSCH REs**.

**All** the **examples** in this section **share** the following **parameter** values and **DCI message field** values:

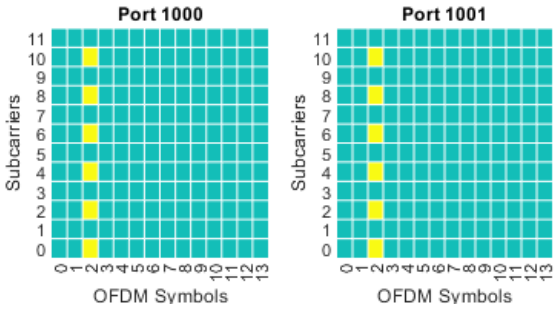
`mappingType` = `typeA`
`dmrs-Type` = `type1`
`maxLength` **is absent, which implies "single-symbol"**
`dmrs-AdditionalPosition` = `pos0`
`dmrs-TypeA-Position` = `pos2`
**DCI message field "*Antenna ports*" => "Number of DM-RS CDM group(s) without data" = 1**

The following **4 examples** correspond to a different **quantity** of **PDSCH layers**, from **1** to **4**.

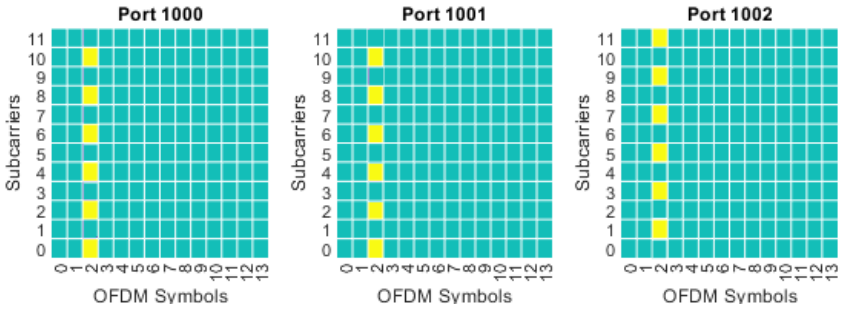
**Example** with **1** **PDSCH layer**:



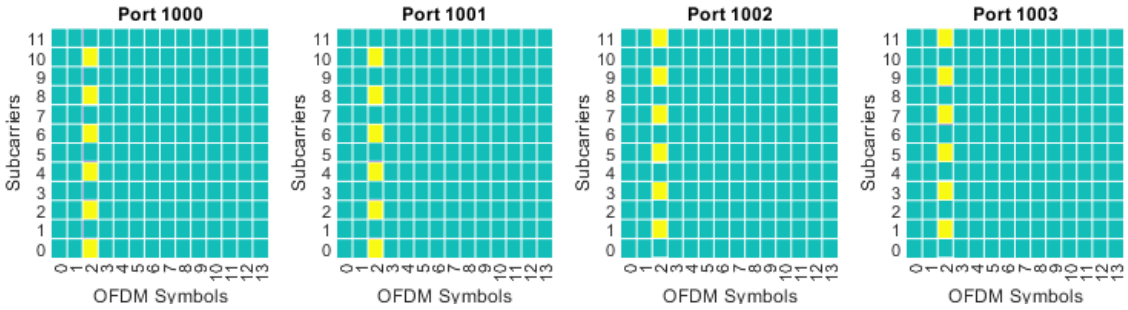
**Example** with **2** **PDSCH** **layers**:




**Example** with **3** **PDSCH** **layers**:




**Example** with **4** **PDSCH** **layers**:



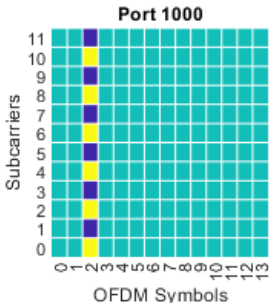

### 6.14.5.7 Examples of 2 CDM groups without data, changing the quantity of layers

**All** the **examples** in this section **share** the following **parameter** values and **DCI message field** values. Notice that, in respect to the **base configuration** from 6.14.5.6, the value of "**Number of DM-RS CDM group(s) without data**" has been changed from **1** to **2**:

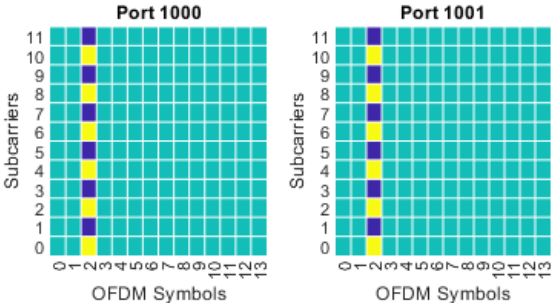
**mappingType** = **typeA**
**dmrs-Type** = **type1**
**maxLength** **is absent, which implies "single-symbol"**
**dmrs-AdditionalPosition** = **pos0**
**dmrs-TypeA-Position** = **pos2**
**DCI message field "*Antenna ports*" => "Number of DM-RS CDM group(s) without data" = 2**

The **following 4** examples correspond to a different **quantity** of **PDSCH layers**, from **1** to **4**.

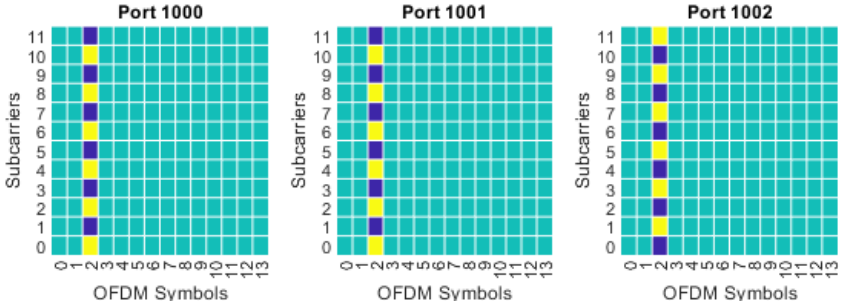
**Example** with **1 PDSCH layer**:



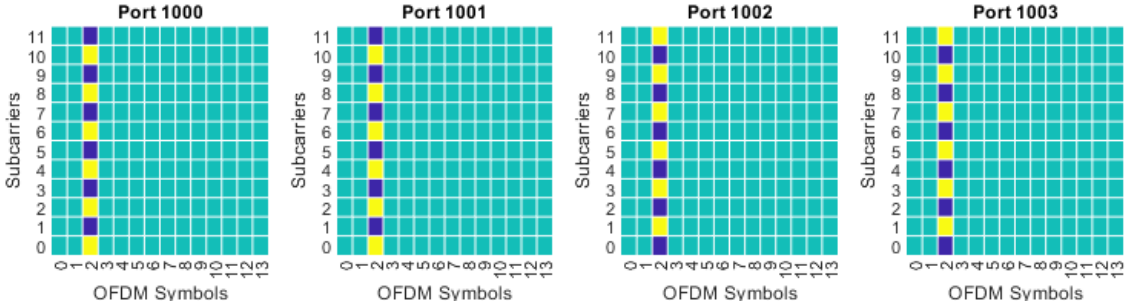

**Example** with **2 PDSCH layers**:




**Example** with **3 PDSCH layers**:




**Example** with **4 PDSCH layers**:




### 6.14.5.8 Examples of frequency configuration (`dmrs-Type`) `type2`, changing the quantity of layers and CDM groups without data

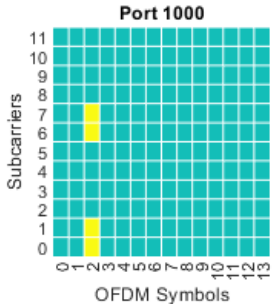
**All** the **examples** in this section **share** the following **parameter** values. Notice that, in respect to the **base configuration** from 6.14.5.6, the **frequency configuration** (`dmrs-`

**Type**) has been **changed** from **DDM-Type 1** to **DDM-Type 1**, and the "**Number of DM-RS CDM group(s) without data**" is taking **several values**:

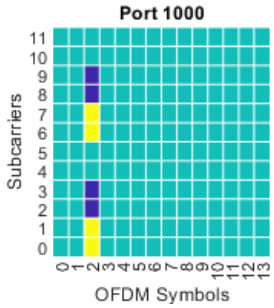
**mappingType** = **typeA**
**dmrs-Type** = **DDM-Type 2**
**maxLength** **is absent, which implies "single-symbol"**
**dmrs-AdditionalPosition** = **pos0**
**dmrs-TypeA-Position** = **pos2**

The following **12 examples** correspond to **all** the combinations between the **quantity** of **PDSCH layers** (from **1** to **4**) and the value of "**Number of DM-RS CDM group(s) without data**" (from **1** to **3**).

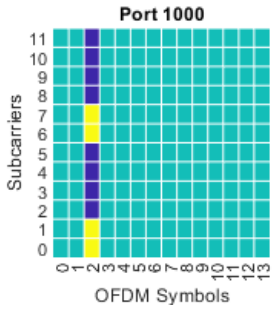
**Example** with **1** **PDSCH layer** and "**Number of DM-RS CDM group(s) without data**"**=1:**



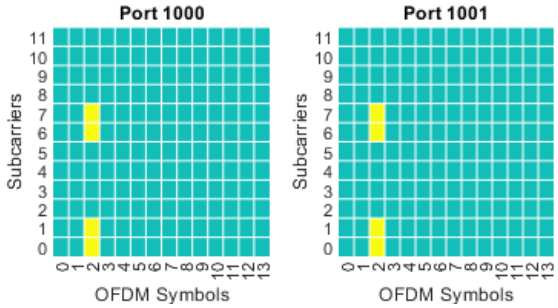

**Example** with **1** **PDSCH layer** and "**Number of DM-RS CDM group(s) without data**"**=2:**



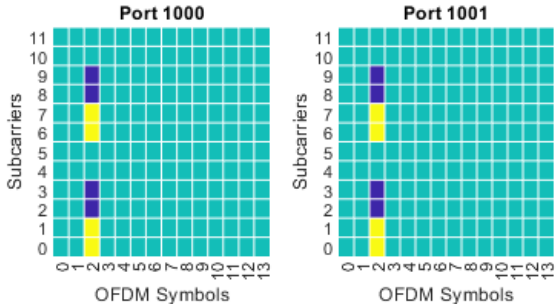

**Example** with **1** **PDSCH layer** and "**Number of DM-RS CDM group(s) without data**"**=3:**



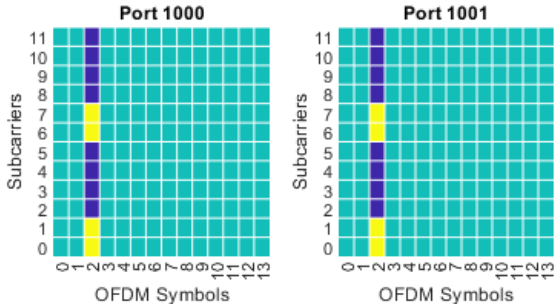
**Example** with **2** **PDSCH** **layers** and "**Number of DM-RS CDM group(s) without data**"**=1:**



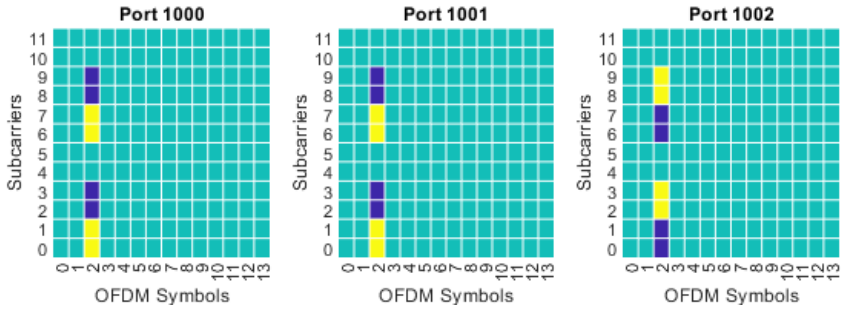

**Example** with **2** **PDSCH** **layers** and "**Number of DM-RS CDM group(s) without data**"**=2:**



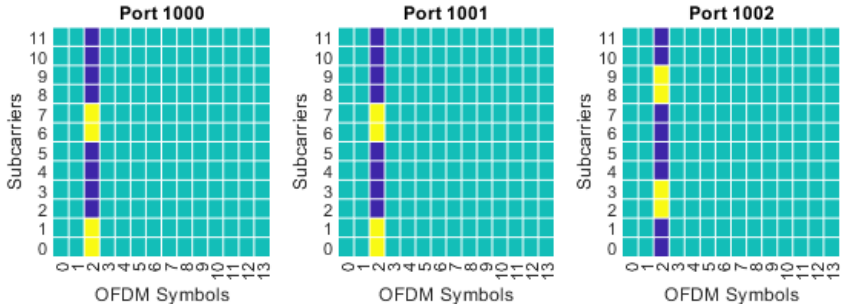

**Example** with **2** **PDSCH** **layers** and "**Number of DM-RS CDM group(s) without data**"**=3:**



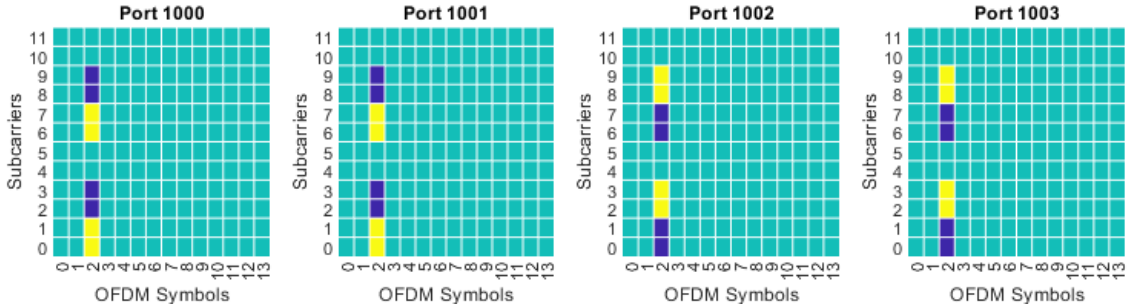

**Example** with **3** **PDSCH** **layers** and "**Number of DM-RS CDM group(s) without data**"**=2:**



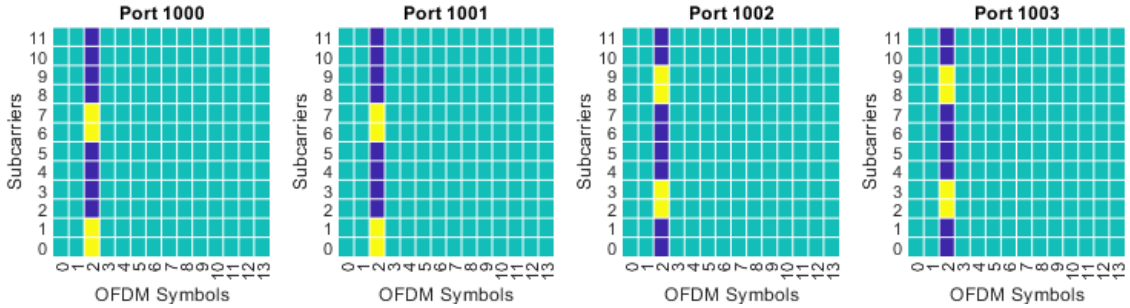
**Example** with **3** **PDSCH** **layers** and "**Number of DM-RS CDM group(s) without data**"**=3:**




**Example** with **4** **PDSCH** **layers** and "**Number of DM-RS CDM group(s) without data**"**=2:**




**Example** with **4** **PDSCH** **layers** and "**Number of DM-RS CDM group(s) without data**"**=3:**



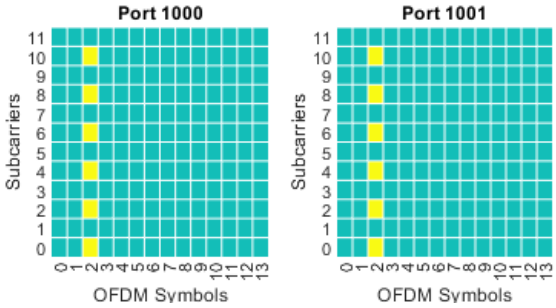

### 6.14.5.9 Examples of base parameter values, changing the quantity of layers, the frequency configuration (`dmrs-Type`) and CDM groups without data

**All** the **examples** in this section **share** the following **parameter** values, which are **the same** as the **base configuration** from 6.14.5.6, while the **frequency configuration** (`dmrs-Type`) and the "**Number of DM-RS CDM group(s) without data**" are taking **several values**:

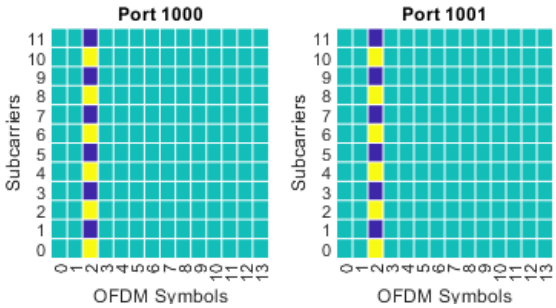
`mappingType` = `typeA`
`maxLength` **is absent, which implies "single-symbol"**
`dmrs-AdditionalPosition` = `pos0`
`dmrs-TypeA-Position` = `pos2`

The following **8 examples** correspond to **all** the combinations between the **quantity** of **PDSCH layers** (**2** and **4**), the **DM-RS** frequency configuration (**DDM-Type 1** and **DDM-Type 2**), and the value of "**Number of DM-RS CDM group(s) without data**" (**1** and **2**).

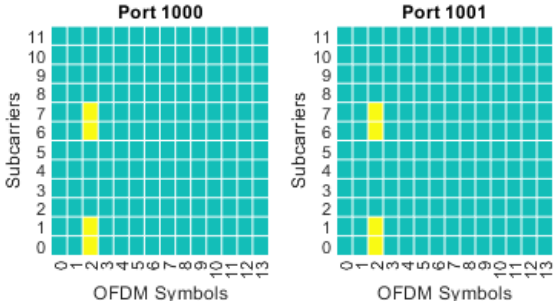
**Example** with **2 PDSCH layers**, **frequency configuration** (`dmrs-Type`) **DDM-Type 1** and "**Number of DM-RS CDM group(s) without data**"**=1:**



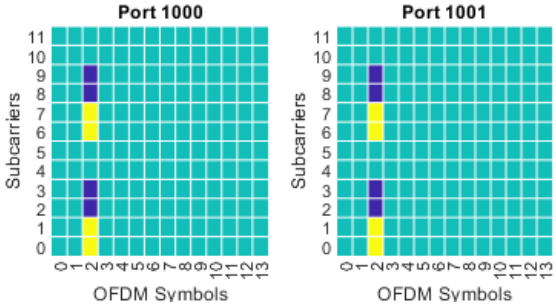

**Example** with **2 PDSCH layers**, **frequency configuration** (`dmrs-Type`) **DDM-Type 1** and "**Number of DM-RS CDM group(s) without data**"**=2:**



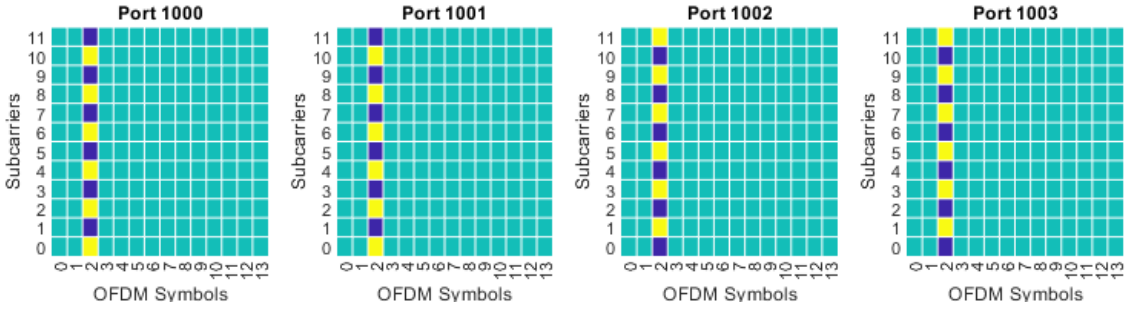

**Example** with **2 PDSCH layers**, **frequency configuration** (`dmrs-Type`) **DDM-Type 2** and "**Number of DM-RS CDM group(s) without data**"**=1:**



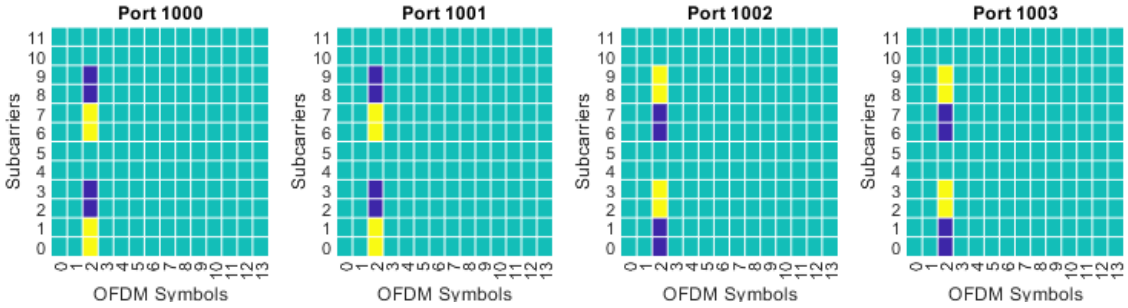

**Example** with **2 PDSCH layers**, **frequency configuration** (`dmrs-Type`) **DDM-Type 2** and "**Number of DM-RS CDM group(s) without data**"**=2:**



**Example** with **4 PDSCH layers**, **frequency configuration** (`dmrs-Type`) **DDM-Type 1** and "**Number of DM-RS CDM group(s) without data**"**=2:**




**Example** with **4 PDSCH layers**, **frequency configuration** (`dmrs-Type`) **DDM-Type 2** and "**Number of DM-RS CDM group(s) without data**"**=2:**



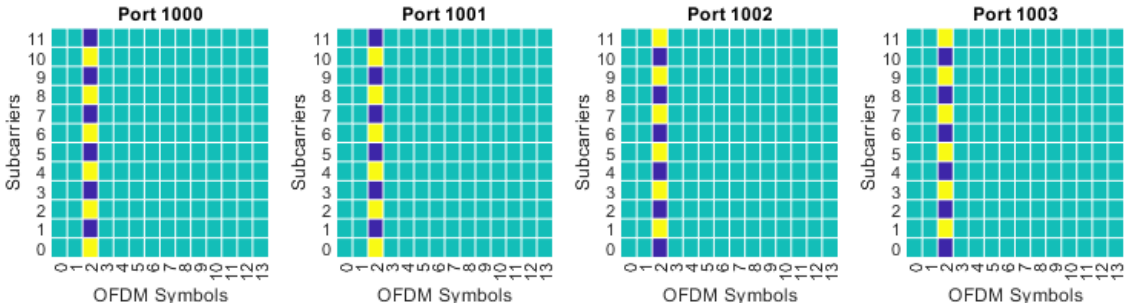

### 6.14.5.10 Examples of 4 layers and 2 CDM groups without data, changing `dmrs-AdditionalPosition`

**All** the **examples** in this section **share** the following **parameters**. Notice that, in respect to the base configuration from 6.14.5.6, the **quantity** of **layers** is now **fixed** to **4** and the value of "**Number of DM-RS CDM group(s) without data**" has been changed from **1** to **2**:

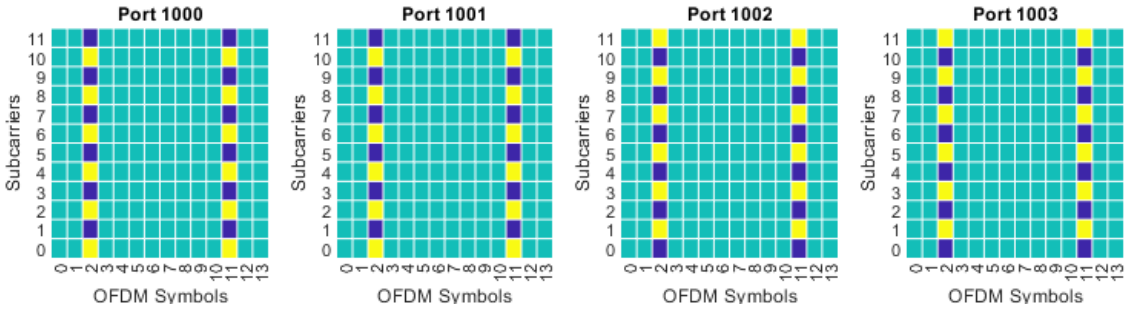
**Quantity of PDSCH layers = 4**
`mappingType` **=** `typeA`
`dmrs-Type` **=** `type1`
`maxLength` **is absent, which implies "single-symbol"**
`dmrs-AdditionalPosition` **=** `pos0`
`dmrs-TypeA-Position` **=** `pos2`
**DCI message field "*Antenna ports*" => "Number of DM-RS CDM group(s) without data" = 2**

The following **4 examples** correspond to different values of `dmrs-AdditionalPosition` (from `pos0` to `pos3`).

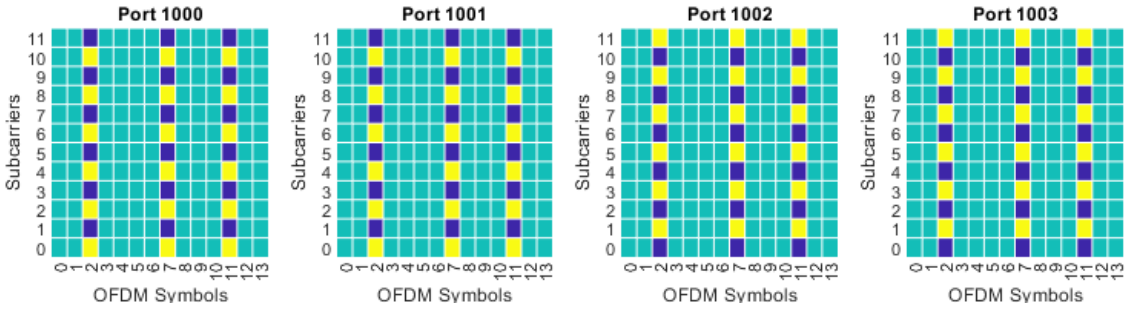
**Example** with `dmr-AdditionalPosition` **=** `pos0`



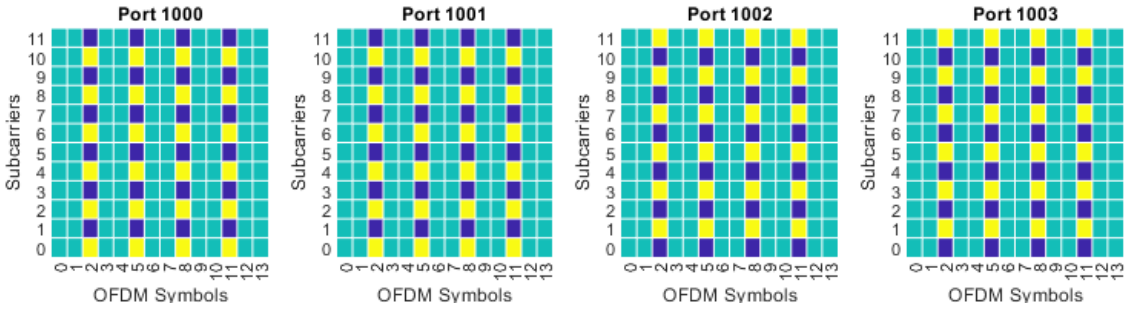
**Example** with `dmr-AdditionalPosition` = `pos1`




**Example** with `dmr-AdditionalPosition` = `pos2`




**Example** with `dmr-AdditionalPosition` = `pos3`



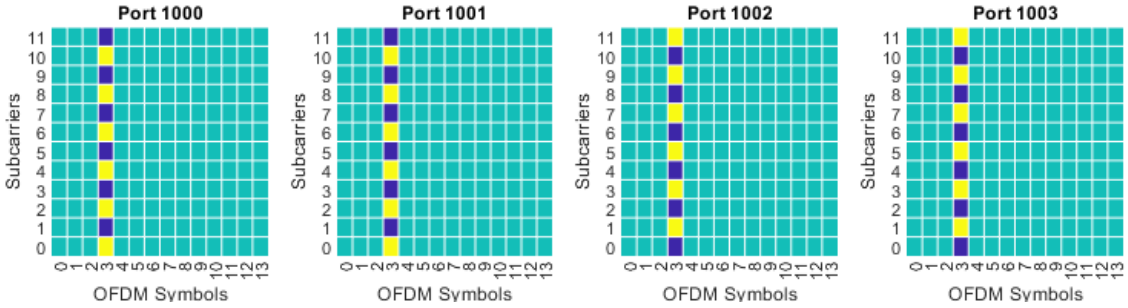

### 6.14.5.11 Examples of 4 layers and `dmrs-TypeA-position`=`pos3`, changing `dmrs-AdditionalPosition`

**All** the **examples** in this section **share** the following **parameters**. Notice that, in respect to the base configuration from 6.14.5.6, the **quantity** of **layers** is now **fixed** to **4**, the `dmrs-TypeA-Position` has been **changed** from `pos2` to `pos3`, and `dmrs-AdditionalPosition` is taking **several values**:

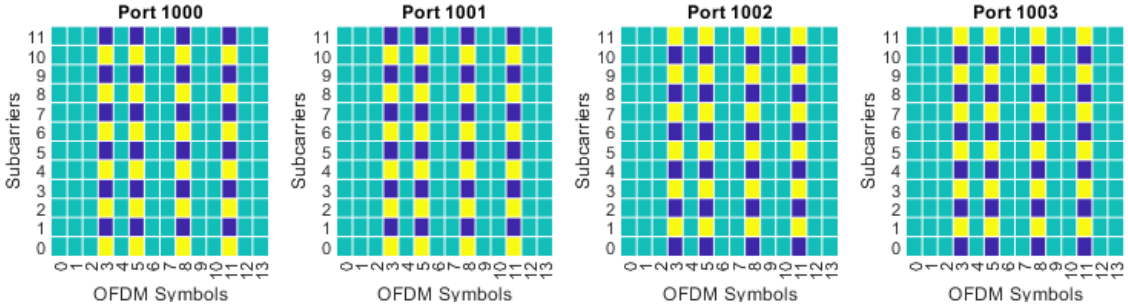
**Quantity of PDSCH layers = 4**
`mappingType` = `typeA`
`dmrs-Type` = `type1`
`maxLength` **is absent, which implies "single-symbol"**
`dmrs-TypeA-Position` = `pos3`
**DCI message field "*Antenna ports*" => "Number of DM-RS CDM group(s) without data" = 2**

The following **2 examples** correspond to different values of `dmrs-AdditionalPosition` (`pos0` and `pos3`).

**Example** with `dmr-AdditionalPosition = pos0`




**Example** with `dmr-AdditionalPosition = pos3`



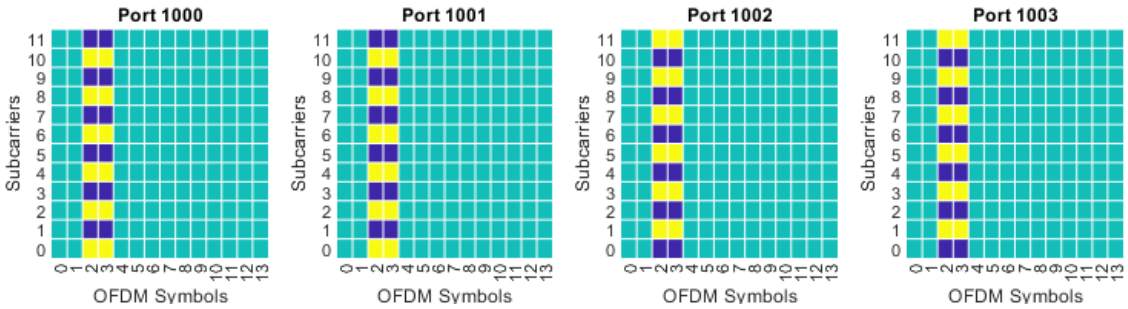

### 6.14.5.12 Examples of 4 layers and double-symbol, changing `dmrs-AdditionalPosition`

**All** the **examples** in this section **share** the following **parameters**. Notice that, in respect to the examples with base parameter values from 6.14.5.6, the **quantity** of **layers** is now **fixed** to **4**, **single-symbol** has been changed to **double-symbol** and `dmrs-AdditionalPosition` is taking **several values**:

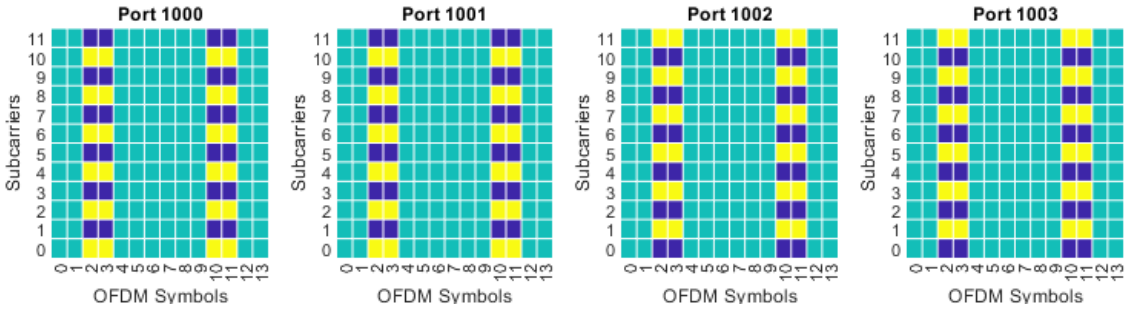
**Quantity of PDSCH layers = 4**
`mappingType = typeA`
`dmrs-Type = type1`
`maxLength = len2`, and the **scheduling DCI message** indicates "**double-symbol**"
`dmrs-TypeA-Position = pos2`
**DCI message field "*Antenna ports*" => "Number of DM-RS CDM group(s) without data" = 2**

The following **2 examples** correspond to a different value of `dmrs-AdditionalPosition` (`pos0` and `pos1`).

**Example** with `dmr-AdditionalPosition = pos0`



**Example** with `dmr-AdditionalPosition = pos1`




### 6.14.5.13 Examples for `mappingType=typeB`

**All** the **examples** in this section **share** the following **parameter** values:

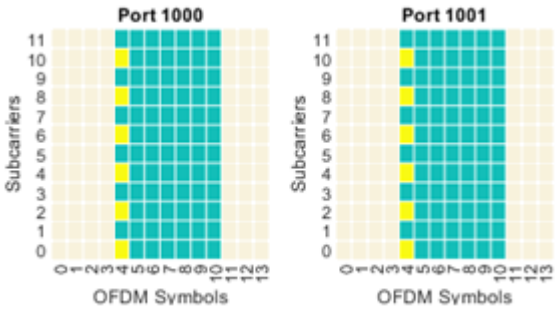
`mappingType = typeB`
`maxLength` **is absent, which implies "single-symbol"**
`dmrs-TypeA-Position = pos2` **(Notice that due to the other values, the value of this parameter is irrelevant)**

Notice that in respect to the examples with base parameter values from 6.14.5.6, `mappingType` has been **changed** from `typeA` to `typeB`. In the different examples in this section, we are **variating** the value of the **frequency configuration** (`dmrs-Type`), the value of "**Number of DM-RS CDM group(s) without data**" and the value of `dmrs-AdditionalPosition`.

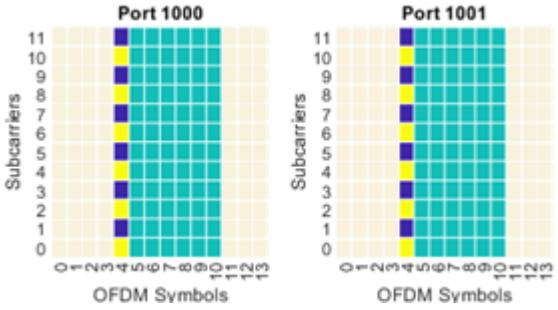
Notice also that the **PDSCH allocation** in the examples in this section is **different** to **all** the other sections, and spans **slot symbol indexes 4** to **7** (both included).

The following **7 examples** correspond to **some** combinations of the quantity of **PDSCH layers** (**2** and **4**), the **frequency configuration** (`dmrs-Type`) (**DDM-Type 1** and **DDM-Type 2**), the value of "**Number of DM-RS CDM group(s) without data**" (**1** and **2**) and the `dmrs-AdditionalPosition` (`pos0` and `pos1`).

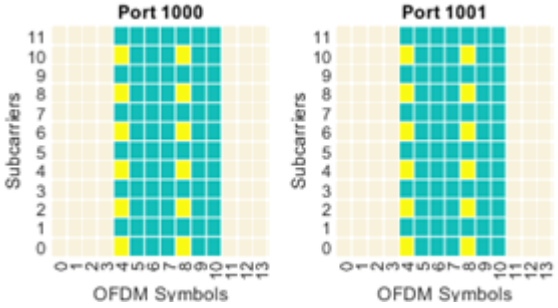
**Example** with **2 PDSCH layers**, **frequency configuration** (`dmrs-Type`) **DDM-Type 1**, `dmr-AdditionalPosition = pos0` and "**Number of DM-RS CDM group(s) without data**"**=1**



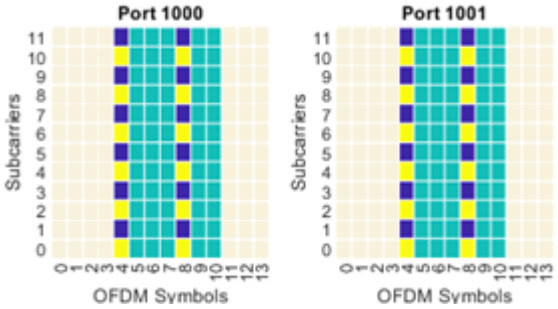
**Example** with **2 PDSCH layers**, **frequency configuration** (`dmrs-Type`) **DDM-Type 1**, `dmr-AdditionalPosition` = `pos0` and "**Number of DM-RS CDM group(s) without data**"=**2**



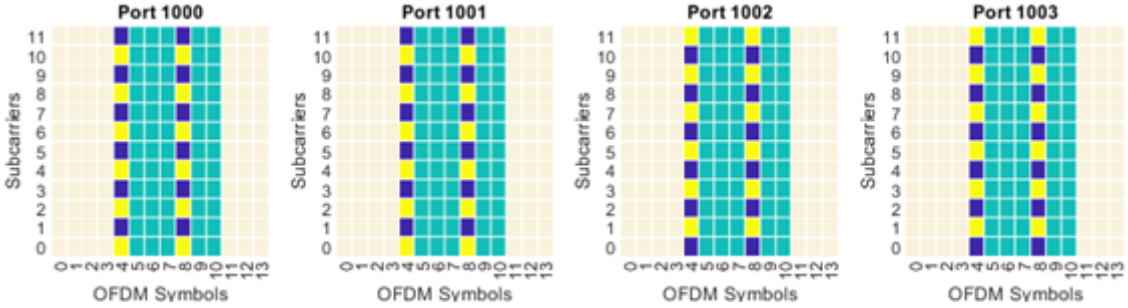

**Example** with **2 PDSCH layers**, **frequency configuration** (`dmrs-Type`) **DDM-Type 1**, `dmr-AdditionalPosition` = `pos1` and "**Number of DM-RS CDM group(s) without data**"=**1**



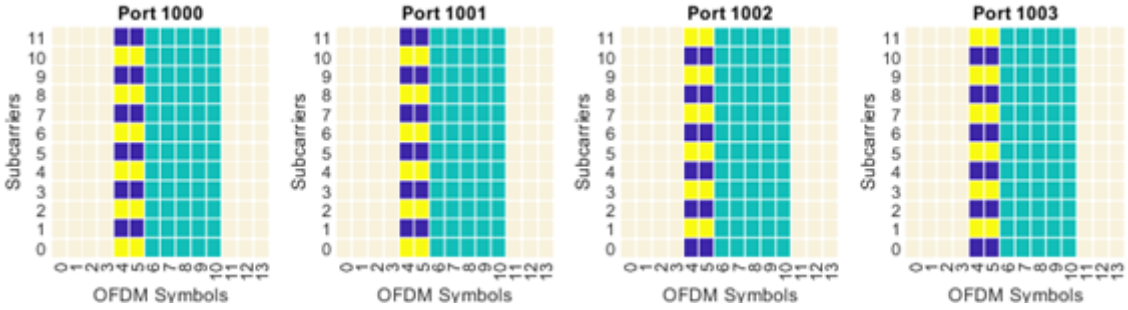

**Example** with **2 PDSCH layers**, **frequency configuration** (`dmrs-Type`) **DDM-Type 1**, `dmr-AdditionalPosition` = `pos1` and "**Number of DM-RS CDM group(s) without data**"=**2**



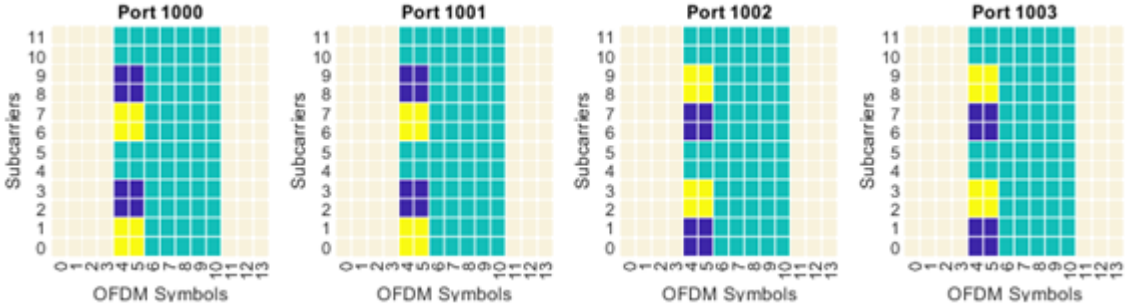

**Example** with **4 PDSCH layers**, **frequency configuration** (`dmrs-Type`) **DDM-Type 1**, `dmr-AdditionalPosition` = `pos1` and "**Number of DM-RS CDM group(s) without data**"=**2**



**Example** with **4 PDSCH layers**, **frequency configuration** (`dmrs-Type`) **DDM-Type 1**, `dmr-AdditionalPosition` = `pos0` and "**Number of DM-RS CDM group(s) without data**"=**2**




**Example** with **4 PDSCH layers**, **frequency configuration** (`dmrs-Type`) **DDM-Type 1**, `dmr-AdditionalPosition` = `pos0` and "**Number of DM-RS CDM group(s) without data**"=**2**




## 6.14.6 PDSCH cell-wide scheduling

The **NR node** makes a **cell-wide decision** for **each Downlink slot** (click 2.3.1) in **each cell** (click 3.1), based on the **state** and **characteristics** of **each** and **every active UE** in the **cell**. This **state** and **characteristics** of **each** and **every active UE** consist of: **HARQ retransmissions** (click 4.5.6.6) **required**, **available new data** to be **transmitted**, **QoE**, its **priority**, **quantity** of **Rx antennas**, **UE capabilities** and **channel state information**, among **others**.

As a result of this **NR node Downlink cell-wide decision**, the following **variables** are determined in the corresponding **cell** for the **processing** of **data** in **each PDSCH transmission**, that is **signaled** by **1 PDSCH assignment(s)**, among the **1** to **8** contained in **each DCI message 1_x** or **4_x**.

**Both** for **new data** and for **retransmitted data**:

- **Assigned PDSCH Virtual Resource Elements** (**VREs**, click 2.2.5.7). This **assignment** also determines the **quantity** "$N_{layer}$" of **space-division layers** (click 2.4.3.5), which is the **same** as the **quantity** of **Tx antenna ports** (click 4.11.5) in the **assigned VREs**.
- **Assigned transmission bitlength** "$N_{bit}^{Tx(q)}$" for **each sequence** "**q**" of **rate-matched block**(**s**).

  For the variable "**q**$\in\{$**0**, **1**$\}$", click 4.12.1. The calculation of "$N_{bit}^{Tx(q)}$" is detailed further below.
- **Redundancy version** "$rv_{id}$" used for **each**, or **both**, of the **1** or **2 sequences** of **rate-matched block**(**s**) (click 4.12.6).

- **Modulation** (click 11.2.1) used for **each** of the **1** or **2** **scrambled codeword(s)** "**q**" (click 4.12.8). **Each modulation** has **1 modulation order** "$Q_m$".

**Only** for **new data**:

- **Bitlength** "$N_{bit}^{TB(q)}$" of **each** of the **1** or **2** **new transport block(s)** "**q**". The **bitlength** "$N_{bit}^{TB(q)}$" can **only** take some **allowed values**. For more detail, click 6.16.2 and refer to clause 5.1.3.2 of TS 38.214.
- **LDPC base graph selection 1** or **2** of **each** of the **1** or **2** **new transport block(s)** "**q**". The **LDPC base graph selection** is **identical** for **PUSCH** and for **PDSCH**. For more detail, refer to clauses 6.2.2 (for **PUSCH**) and 7.2.2 (for **PSDCH**).
- **Toggle** the "***New data indicator***" **bit** (click 4.5.6.3) of **each** of the **1** or **2** **new transport block(s)** "**q**".

**Only** for **HARQ retransmitted data**:

- What **encoded block(s)** will be **retransmitted**. They are the **unacknowledged** (**NACK**) **encoded block(s)** that fit in the **assigned PREs**.
- Do **not toggle** the "***New data indicator***" **bit** (click 4.5.6.3) of **each** of the **1** or **2** **retransmitted** sequences of **encoded block(s)** "**q**".

The **assigned transmission bitlength** "$N_{bit}^{Tx(q)}$" for **symbol sequence** "**q**" is:

$$N_{bit}^{Tx(0)} = N_{bit}^{layer(0)} \bullet floor(N_{layer}/2)$$

$$N_{bit}^{Tx(1)} = N_{bit}^{layer(1)} \bullet ceiling(N_{layer}/2)$$

where the two previous formulas arise from **layer mapping** (click 4.12.11), and the value of "$N_{bit}^{layer(q)}$" is:

$$N_{bit}^{layer(q)} = N_{symb}^{layer} \bullet Q_m^{(q)}$$

where $Q_m^{(q)}$ is the **modulation order** used for **symbol sequence** "**q**", and the value of "$N_{symb}^{layer}$" is:

$$N_{symb}^{layer} = 12 \bullet N_{PRB}^{A} \bullet N_{symb}^{A} - N_{PPS}^{A}$$

Where "$N_{PRB}^{A}$" is the quantity of **PRBs**, "$N_{symb}^{A}$" is the quantity of **symbol intervals** and "$N_{PPS}^{A}$" is the quantity of **PREs** occupied by **Physical Signals** (i.e., **PREs** that are **not** a **PDSCH payload PRE**), **all** of them **within** the **transmission assignment**.

**Some** of the **above variables** are **interrelated**, because their **values must comply** with a given **cross-relationship**. A few **relevant cases** of these **cross-relationships** are:

- For **each transport block** "**q**", **only some combinations** of its **bitlength** "$N_{bit}^{TB(q)}$" and its **LDPC base graph** are **allowed**. For more detail click 6.16.2 and refer to clause 7.2.2 of TS 38.212.
- The **bitlength** "$N_{bit}^{TB(q)}$", the **LDPC base graph 1** or **2**, the **modulation order** "$Q_m$" and the **assigned transmission bitlength** "$N_{bit}^{Tx(q)}$" of **transport block** "**q**" are **interrelated** because, after the **encoding** and **modulation** of the **transport block(s)**, the resulting **total quantity** of **symbols** should be the **same** as the **quantity** of **assigned PDSCH payload VREs**, and **rate-matching** (click 4.12.6) **cannot** cope with **any** amount of **mismatch**.
- For **HARQ retransmitted data**, the **retransmitted** **encoded block(s)** determine the **quantity** of **rate-matched block sequence(s)**. In turn, the **quantity** of **rate-**

**matched block sequence(s) determines** the **range** of **layers** that can be used: **1** to **4 layer(s)** for **1 sequence** and **5** to **8 layers** for **2 sequences**.

## 6.15 PDSCH payload contents

**Each PDSCH transmission contains** (click 6.9) **1** or **2 codeword(s)** (click 4.12.1.2).

For a **new transmission**, **each codeword** contains **1 encoded transport block** (click 4.12.1.1).

For a **HARQ retransmission** (click 4.5.6.6), **each codeword** contains **either 1 encoded fragment** of a **transport block** or **1 encoded transport block**.

For an overview of the **data flow** across the **user-plane protocol stack**, click 4.10.4.

## 6.16 PDSCH payload processing

The **Tx processing** of the **PDSCH payload** by the **NR node** is **jointly described** with the **Tx processing** of the **PUSCH payload** by the **UE** because **both processes** are **quite similar**. Whenever there is a **difference** between the **Tx processing** of the **PDSCH payload** and the one of the **PUSCH payload** it is **explicitly indicated**. For this joint description **PDSCH/PUSCH payload Tx processing**, click 4.12.2.

For the **Rx processing** of the **PDSCH payload** by the **UE**, the **UE first** has to determine the following variables:

- **Modulation** and **target code rate** (click 6.16.1).
- **Redundancy version** and **quantity** of **received sequence(s)** (click 6.16.2).
- **Quantity of transport block(s)** and its/their **size** (click 6.16.3).
- **LDPC base graph** (click 6.16.4).

**After** having **determined** the above **variables**, the **Rx processing** of the **PDSCH payload** by the **UE** is **jointly described** with the **Rx processing** of the **PUSCH payload** by the **NR node** because **both processes** are **quite similar**. Whenever there is a **difference** between the **Rx processing** of the **PDSCH payload** and the one of the **PUSCH payload** it is **explicitly indicated**. For this joint description **PDSCH/PUSCH payload Rx processing**, click 4.12.16.

### 6.16.1 UE determination of PDSCH modulation order and target code rate

The **UE** determines the **modulation order** "$Q_m^{(q)}$" (click 11.2.1) and the **target code rate** "***R***" used to demodulate **each layer symbol sequence** "**q**" based on the respective **1** or **2** "***modulation and coding scheme***" field(**s**) contained in the corresponding **DCI message 1_x** or **4_x**. **Each** "***Modulation and coding scheme***" field has **5 bits** and is denoted by "$I_{MCS}$". The **UE** inputs **each** received $I_{MCS}$ into **one** of the tables 5.1.3.1-**1** to **-4** of TS 38.214. The **table** to use depends on the **DCI message Format**, on the **RNTI** used to mask its **CRC** and on other factors (refer to clause 5.1.3.2 of TS 38.214). The following is table 5.1.3.1-1 to show its structure and values:

| MCS Index $I_{MCS}^{(q)}$ | Modulation Order $Q_m^{(q)}$ | Target code rate [$R \times 1024$] | Spectral efficiency |
|---|---|---|---|
| **0** | 2 | 120 | 0.2344 |
| **1** | 2 | 157 | 0.3066 |
| **2** | 2 | 193 | 0.3770 |

| MCS Index $I_{MCS}^{(q)}$ | Modulation Order $Q_m^{(q)}$ | Target code rate [$R \times 1024$] | Spectral efficiency |
|---|---|---|---|
| **3** | 2 | 251 | 0.4902 |
| **4** | 2 | 308 | 0.6016 |
| **5** | 2 | 379 | 0.7402 |
| **6** | 2 | 449 | 0.8770 |
| **7** | 2 | 526 | 1.0273 |
| **8** | 2 | 602 | 1.1758 |
| **9** | 2 | 679 | 1.3262 |
| **10** | 4 | 340 | 1.3281 |
| **11** | 4 | 378 | 1.4766 |
| **12** | 4 | 434 | 1.6953 |
| **13** | 4 | 490 | 1.9141 |
| **14** | 4 | 553 | 2.1602 |
| **15** | 4 | 616 | 2.4063 |
| **16** | 4 | 658 | 2.5703 |
| **17** | 6 | 438 | 2.5664 |
| **18** | 6 | 466 | 2.7305 |
| **19** | 6 | 517 | 3.0293 |
| **20** | 6 | 567 | 3.3223 |
| **21** | 6 | 616 | 3.6094 |
| **22** | 6 | 666 | 3.9023 |
| **23** | 6 | 719 | 4.2129 |
| **24** | 6 | 772 | 4.5234 |
| **25** | 6 | 822 | 4.8164 |
| **26** | 6 | 873 | 5.1152 |
| **27** | 6 | 910 | 5.3320 |
| **28** | 6 | 948 | 5.5547 |
| **29** | 2 | reserved | |
| **30** | 4 | reserved | |
| **31** | 6 | reserved | |

The modulation order $Q_m$ is the **quantity of bits** for **each** modulation **symbol** and allows the **UE** to determine the **modulation** from the table in 11.2.1.

For more detail on **modulation** and **code rate determination**, refer to clause 5.1.3.1 of TS 38.214)

## 6.16.2 UE determination of PDSCH redundancy version and quantity of received sequence(s)

The **UE** determines the **PDSCH redundancy version** "$rv_{id}$" from the "***Redundancy version***" field (or **fields**) contained in the scheduling **DCI message**.

The **UE** determines if **1** or **2 sequences** of received **rate-matched block**(s) are contained in the **scheduled Physical Resource Elements** (**PREs**) by the **quantity** "$N_{layer}$" of **space-multiplexing layers** (click 2.4.3.5) and on configuration **parameters**:

- If $1 \leq N_{layer} \leq 4$, then **1 sequence** of **rate-matched block**(s) are received.
- If $5 \leq N_{layer} \leq 8$, and **configuration allows** for it, then **2 sequences** of **rate-matched block**(s) are received.

For more detail on the **UE** determination of **PDSCH** "***Redundancy version***", refer to clause 5.1.3.2 of TS 38.214.

### 6.16.3 UE determination of PDSCH quantity and size of transport block(s)

**Transmission of 1 or 2 transport block(s)**

Transmission of **1** or **2 PDSCH transport block(s)** (click 4.12.1) in **1 DCI PDSCH assignment** depends on the **value** of `maxNrofCodeWordsScheduledByDCI` from `PDSCH-config` (click 6.18.6):

- **1 PDSCH transport block** if the value is "`n1`".
- **2 PDSCH transport blocks** if the value is "`n2`".

Notice that **data type** `PDSCH-config` has an instantiation for **unicast** and another one for **multicast**.

When **2 PDSCH transport blocks** are **enabled**, a certain combination of "***Modulation and coding scheme index***" "$I_{MCS}$", of the "***Redundancy version***" "$rv_{id}$", of the **DCI message type** and of parameter `pdsch-TimeDomainAllocationListForMultiPDSCH-r17`, may **inhibit** the **transmission** of **either** the **first** or the **second transport block**.

**Transport block size determination**

There are several cases. A quite frequent case is shown here.

In the case that:

- **PDSCH** assigned with **either** of **DCI Formats 1_0**, **1_1**, **4_0**, **4_1**, **4_2** or **1_2**,
- where its **CRC** is scrambled with **either** of **CRNTI**, **MCS-C-RNTI**, **TC-RNTI**, **CS-RNTI**, **G-RNTI**, **G-CS-RNTI**, **MCCH-RNTI** or **SI-RNTI**,
- and **either** of the following holds:
  - Table 5.1.3.1-**2** of TS 38.214 is used and $\mathbf{0} \leq I_{MCS} \leq \mathbf{27}$
  - Table 5.1.3.1-**4** of TS 38.214 is used and $\mathbf{0} \leq I_{MCS} \leq \mathbf{26}$
  - A table other than 5.1.3.1-**2** and -**4** of TS 38.214 is used and $\mathbf{0} \leq I_{MCS} \leq \mathbf{28}$

then, the **UE** determines the **transport block size** "$N_{bit}^{TB}$" with the **following procedure**:

1. **Obtain quantity** "$N_{RE}$" of **assigned PDSCH payload PREs** within the **slot** for this **transport block**

   The **quantity** "$N_{RE}^{PRB}$" of **PREs** allocated for **PDSCH** within a **PRB** is first computed as:

   $$N_{RE}^{PRB} = 12\ N_{symb}^{sh} - N_{DMRS}^{PRB} - N_{oh}^{PRB}$$

   where "$N_{symb}^{sh}$" is the **quantity** of **symbols** of the **PDSCH assignment** within the **slot**, "$N_{DMRS}^{PRB}$" is the **quantity** of **PREs** for **DM-RS** in **each PRB** (including the **overhead** of the "**Number of DM-RS CDM group(s) without data**") and "$N_{oh}^{PRB}$" is an additional **overhead**. The value of "$N_{oh}^{PRB}$" is **0** by **default**, ant it takes the value of `xOverhead` (if **present**) from `PDSCH-ServingCellConfig` (click 6.18.10) under some circumstances. The value of `xOverhead` may be `6`, `12` or `18`.

   The **quantity** "$N_{RE}^{PRB}$" should **not** be **larger** than "12•13=**156**" which is **13 symbols** over the **12 subcarriers** of the **PRB**.

Then, the value of "$N_{RE}$" is:

$$N_{RE} = n_{PRB} \bullet N_{RE}^{PRB}$$

where "$n_{PRB}$" is the **total quantity** of **PRBs allocated** for the **UE**.

2. **Obtain quantity** "$N_{info}$" of **pre-FEC** useful **bits** within the **assigned PDSCH payload PREs** within the **slot** for this **transport block**

$$N_{info} = N_{RE} \bullet R \bullet Q_m \bullet N_{Layer}$$

where "$R$" is the **base code rate** (useful/total bits) "$Q_m$" is the **modulation order** (bits/symbol) and "$N_{Layer}$" is the **quantity** of **layers**.

If "$N_{info} \leq$ **3,824**", go to point **3.1**. Otherwise, go to point 3.2

**3.a Obtain quantity** "$N'_{info}$" by **quantizing** "$N_{info}$" to the **rounded down integer quantity** of "$2^z$", as follows:

$$N'_{info} = 2^z \, \mathbf{floor}(N_{info} / 2^z)$$

where the value of "**z**" is:

$$z = \mathbf{max}(3, \mathbf{floor}(\log_2(N_{info}))-6)$$

**Finally**, the **transport block bitlength** "$N_{bit}^{TB}$" is the **minimum** value from table 5.1.3.2-**1** of TS 38.214 that is **larger** than "$N'_{info}$". **All** the **92 values** in the said table are a **multiple** of **8 bits** and **range** from **24** to **3,824 bits** (i.e., from **3** to **478 bytes**).

**4.a Obtain quantity** "$N'_{info}$" by **quantizing** "$N_{info}$-24" to the **rounded up integer quantity** of "$2^w$", subject to a **minimum** value of **3,480**, as follows:

$$N'_{info} = \mathbf{max}(3480, 2^w \, \mathbf{ceiling}((N_{info} -24)/2^w))$$

where the value of "**w**" is:

$$w = \mathbf{floor}(\log_2(N_{info}-24))-5$$

**Finally**, the **transport block bitlength** "$N_{bit}^{TB}$" is **computed** as:

$$N_{bit}^{TB} = 8C \bullet \mathbf{ceiling}((N'_{info}+24)/8C)-24$$

where the value of C is:

- If (**code rate** $R \leq 1/4$), then

  $C=\mathbf{ceiling}((N'_{info}+24)/3816)$

- If (**code rate** $R > 1/4$) and ($N'_{info} > 8424$), then:

  $C=\mathbf{ceiling}((N'_{info}+24)/8424)$

- If (**code rate** $R > 1/4$) and ($N'_{info} \leq 8424$), then:

  $C=1$

For more detail on the **UE** determination of **PDSCH transport block(s) size**, refer to clause 5.1.3.2 of TS 38.214.

Source web pages:

- https://www.techplayon.com/5g-nr-transport-block-size-tbs-calculation/
- https://5g-tools.com/5g-nr-tbs-transport-block-size-calculator

- https://www.intechopen.com/chapters/71476

### 6.16.4 UE determination of PDSCH LDPC base graph

For the **initial PDSCH transmission** of an **encoded block** "$eb_m^{(q)}(i_{eb})$" and for its **successive retransmissions**, its **encoding** remains the **same**, and it uses **either** **LDPC base graph 1** or **2** according to:

If ($N_{bit}^{TB(q)} \leq 292$) or (($N_{bit}^{TB(q)} \leq 3{,}824$) and ($R \leq 0.67$)) or ($R \leq 0.25$)

then **LDPC base graph** "**q**" = **2**

else **LDPC base graph** "**q**" = **1**

where "$R$" is the **code rate** (click 6.16.1).

For more detail on **LDPC base graph selection**, refer to clause 7.2.2 of TS 38.212.

## 6.17 PDSCH DM-RS and PT-RS contents

### 6.17.1 PDSCH DM-RS contents

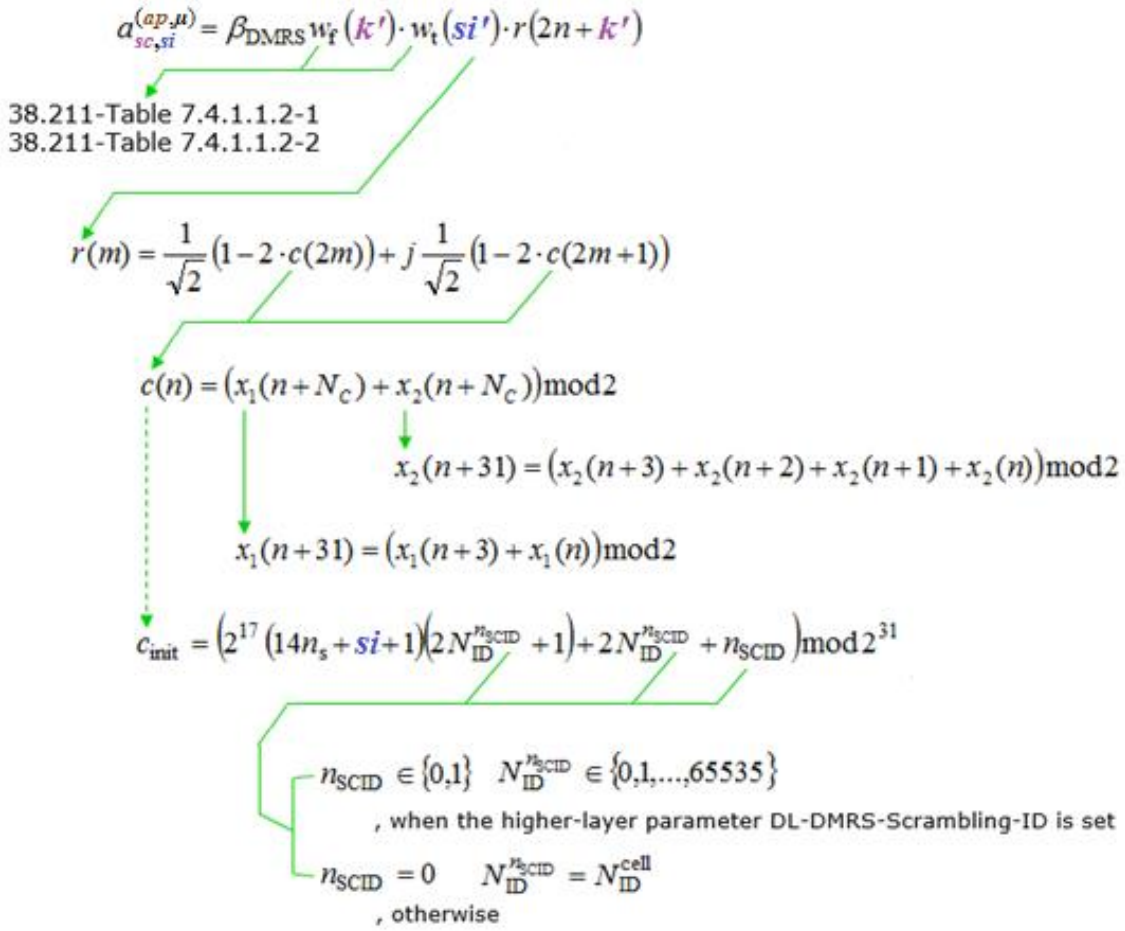
The **PDSCH demodulation reference signals** (**DM-RS**) (click 4.11.3.2.1) within **each PDSCH transmission**. These **signals** are characterized by their **scheduling** (click 6.14) within the **PDSCH transmission** and by their **contents**.

The **contents** of **PDSCH DM-RS** are defined by equation:

$$a_{sc,si}^{(ap,\mu)} = \beta_{\text{DMRS}}\, w_f(sc')\, \bullet\, w_t(si')\, \bullet\, r(2n+sc')$$

where "$sc$" is the **subcarrier index** (click 6.14.3), "$si$" is the **symbol interval index** (click 6.14.4), "$ap$" is the **antenna port number** and "**μ**" is the **subcarrier spacing**.

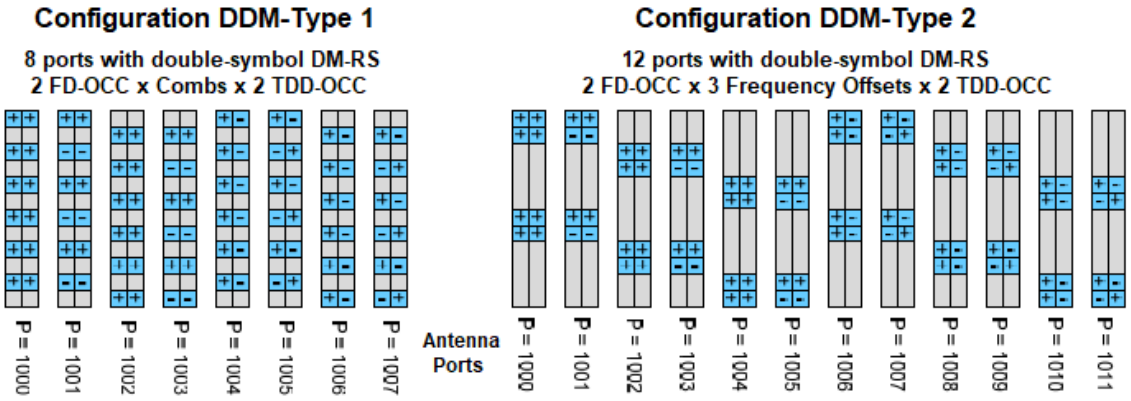
The **different variables** of the **above equation** are **expanded** in the following **equations**:




Some **relevant remarks** on these equations are:

- "***r*(*m*)**" is **QPSK modulation**.
- "$w_f(sc')$" implements the **Orthogonal Covering Coding** (**OCC**) within **each Code Division Multiplexing** (**CDM**) **group**.
- "$w_t(0)$" applies to **even elements** {0, 2, 4, ...} of the **input sequence** "*r*()", while "$w_t(1)$" applies to **odd elements** {1, 3, 5, ...}.
- The values of "*sc*", "*sc'*", "*si*", "*si'*" and "**Δ**" have already been explained in 6.14.5.1.

The following figure shows an example of **double-symbol PDSCH DM-RS frequency configuration DDM-Type 1** and **DDM-Type 2**:




In this figure, **subcarriers** are represented in **bottom-up ascending order**, from **0** to **11**. **Symbol intervals** are represented in **left-to-right ascending order**. The "+" and the "**-**" in the figure indicate the values of $w_f(sc')$, which are "+1" and "-1" (respectively).

The following table shows the **parameters** (and their respective value ranges) that are used for the **sequence generation** of **DM-RS**.

| 38.211-7.4.1.1 | RRC parameter | Range |
|---|---|---|
| $n_{SCID}$ | Value of **DCI field** "***DMRS sequence initialization***", if **present**. Otherwise: **0** | **0**, **1** |
| $N_{ID}^0$ | `scramblingID0`, if **present**. Otherwise: `physCellId` | `0..65535` |
| $N_{ID}^1$ | `scramblingID1` if **present**. Otherwise: `physCellId` | `0..65535` |

Some **relevant remarks** on this table are:

- **Both** `scramblingID0` and `scramblingID1` are from `DMRS-DownlinkConfig` (click 6.18.11).
- Recall that `physCellId` is the **Physical cell identifier** of **this cell**.

For more detail on **PDSCH DM-RS contents**, refer to clause 7.4.1.1.1 of TS 38.211.

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_PDSCH_DM-RS.html
- https://www.nrexplained.com/dmrs_pdsch

## 6.17.2 PDSCH PT-RS contents

The **PDSCH phase-tracking reference signals** (**PT-RS**) are the **PT-RS** (click 4.11.3.2) within **each PDSCH transmission**. These **signals** are characterized by their **scheduling** within the **PDSCH transmission** and by their **contents**.

The **PDSCH PT-RS** are configured with parameter `phaseTrackingRS` from `DMRS-DownlinkConfig` (click 6.18.11). If it is **absent**, the **UE** assumes that there are **no PDSCH PT-RS**.

The **UE** shall assume that **PDSCH PT-RS** are present only in the resource blocks in **each PDSCH transmission**, and only if the procedure in TS 38.214 indicates phase-tracking reference signals being used.

For more detail on **PDSCH PT-RS contents**, refer to clause 7.4.1.2 of TS 38.211.

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_PTRS_DL.html

## 6.18 PDSCH parameters

The **UE**'s **Physical Downlink Shared Channel** (**PDSCH**, click 6.13) is configured with:

- **Common parameters** that are **broadcast** within **MIB**, as described in section 6.18.1.
- **Common** parameters that are **broadcast** (or **unicast**) within **SIB1**, as described in section 6.18.2.
- **Common** parameters that are **broadcast** (or **unicast**) within **other SIBS and posSIBs**, as described in section 6.18.3.
- **Common** and **dedicated parameters** that are **unicast** within the **Non-SI parameter group**, as described in section □.

Notice that there are other related data types, such as, **PDSCH-HARQ-ACK-CodebookList-r16**, **PDSCH-HARQ-ACK-EnhType3Index-r17** or **PDSCH-HARQ-ACK-EnhType3-r17**, that are not described in this book.

Possibly existing **PDSCH parameters** within **other SIBs** and **posSIBs** (click 5.3.5) are **not addressed** in this book.

For an overall explanation on **UE configuration parameters** click 5. For a brief overview of **ASN.1**, click 5.9.

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_PDSCH_DM-RS.html

### 6.18.1 PDSCH parameters within MIB

**PDSCH** is configured with **common parameters** that are **broadcast** in **MIB** (click 5.3.3), which is sent within **SSB** (click 6.1), in the following case:

- In the **UE**'s **primary cell**, during **cell selection**.

The corresponding parameter is **dmrs-TypeA-Position** from **MIB** (click 8.12.3).

Notice that **ServingCellConfigCommon** (click 3.9.6) has a **same-name** parameter with the **same data type** and the **same semantics**.

The **ancestor tree** (click 5.9.8) of this parameter **up to** the **top** is:

```
                          BCCH-BCH-Message (click 5.8.5.1)
message                     BCCH-BCH-MessageType (click 5.8.5.1)
  mib                         MIB (click 8.12.35.3.3)
    dmrs-TypeA-Position         ENUMERATED {pos2, pos3}
```

### 6.18.2 PDSCH parameters within SIB1

**PDSCH** is configured with **common parameters** that are **broadcast** (or **unicast**) within SIB1 (click 5.3.4) in the following case:

- In **UE**'s **primary cell**, for **PDSCH** in the **Downlink initial bandwidth part**, **during initial access**.

For **PDSCH** parameters within SIB1, the **most relevant related data type** is:

- PDSCH-ConfigCommon (click 6.18.5): common parameters

The following **data types**, that are **descendants** of PDSCH-ConfigCommon (and also of PDSCH-Config, click 6.18.6), are also relevant:

- PDSCH-TimeDomainResourceAllocationList (see within 6.18.5 and 6.18.6.1): instantiated as common parameters
- PDSCH-TimeDomainResourceAllocation (click 6.18.8): instantiated as common parameters

The **ancestor tree** (click 5.9.8) of this **most relevant related data type** up to SIB1 is:

```
                                SIB1 (click 5.7.4)
servingCellConfigCommon           ServingCellConfigCommonSIB (0,1) (click 3.9.3)
  downlinkConfigCommon              DownlinkConfigCommonSIB (click 2.2.6.4)
    initialDownlinkBWP                BWP-DownlinkCommon (click 2.2.8.6)
      pdsch-ConfigCommon                PDSCH-ConfigCommon (0,1) (click 6.18.5)
```

For the **ancestor trees** of SIB1 **up to** the **top**, click 5.3.4.

### 6.18.3 PDSCH parameters within other SIBs and posSIBs

**PDSCH** is configured with parameters **within** the **Other SIBs and posSIBs parameter group** (click 5.3.5) in the following cases:

- For **semi-persistent scheduling** of **broadcast services** delivered through **MCCH** and **MTCH**.

For **PDSCH** parameters within **other SIBs and posSIBs**, the **most relevant related data type** is:

- PDSCH-ConfigBroadcast-r17 (click 6.18.7): instantiated as common parameters

The following **data types**, that are **descendants** of PDSCH-ConfigBroadcast-r17 (and also of PDSCH-Config, click 6.18.6), are also relevant:

- PDSCH-TimeDomainResourceAllocationList (see within 6.18.5 and 6.18.6.1): instantiated as common parameters
- PDSCH-TimeDomainResourceAllocation-r16 (click 6.18.8): instantiated as common parameters

The **ancestor tree** (click 5.9.8) of this **most relevant related data type**, up to SystemInformation is:

```
systemInformation          SystemInformation (click 5.7.5)
  systemInformation          SystemInformation-IEs (click 5.7.6)
    sib20-v1700                SIB20-r17
      cfr-ConfigMCCH-MTCH-r17    CFR-ConfigMCCH-MTCH-r17
        pdsch-ConfigMCCH-r17       PDSCH-ConfigBroadcast-r17 (click 6.18.7)
```

Some **relevant remarks** on this **ancestor tree** are:

- For the **ancestor trees** of `SystemInformation` **up to** the **top**, click 5.3.5.
- The `PDSCH-ConfigBroadcast-r17` **data type** is devoted to **MBS broadcast**.

## 6.18.4 PDSCH parameters within the Non-SI parameter group

**PDSCH** is configured with parameters **within** the **Non-SI parameter group** (click 5.3.6) in the following cases:

- In **UE**'s **primary cell**, **PDSCH** for the **additional bandwidth parts** (i.e., **other** than the **initial bandwidth part**).
- In the **primary-secondary cell** and in **secondary cells**, **PDSCH** for **all bandwidth parts**.
- In **UE**'s **primary cell**, **PDSCH** for the **initial bandwidth part**, if configuration changes are done **after initial access**.

For **PDSCH** parameters **within** the **Non-SI parameter group**, the **most relevant related data types** are:

- `PDSCH-ConfigCommon` (click 6.18.5): common parameters
- `PDSCH-Config` (click 6.18.6): dedicated parameters
- `PDSCH-ConfigBroadcast-r17` (click 6.18.7): instantiated as dedicated parameters
- `PDSCH-ServingCellConfig` (click 6.18.10): dedicated parameters

The following **data types**, **descendants** of `PDSCH-ConfigCommon` (click 6.18.5) and of `PDSCH-Config` (click 6.18.6) are also relevant:

- `PDSCH-TimeDomainResourceAllocationList` (see within 6.18.5 and 6.18.6.1): instantiated as dedicated parameters
- `PDSCH-TimeDomainResourceAllocation` (click 6.18.8): instantiated as dedicated parameters

The following **data types**, **descendants** of `PDSCH-ConfigBroadcast-r17` (click 6.18.7) and of `PDSCH-Config` (click 6.18.6) and are also relevant:

- `PDSCH-TimeDomainResourceAllocationList-r16` (see within 6.18.6.1): instantiated as dedicated parameters
- `PDSCH-TimeDomainResourceAllocation-r16` (click 6.18.9): instantiated as dedicated parameters

The following **data types**, **descendants** of `PDSCH-Config` (click 6.18.6) are also relevant:

- `Dummy-TDRA-List` (see within 6.18.6.1): dedicated parameters
- `MultiPDSCH-TDRA-List-r17` (see within 6.18.6.1): dedicated parameters
- `MultiPDSCH-TDRA-r17` (see within 6.18.6.1): dedicated parameters

For **PDSCH** parameters within the **Non-SI parameter group**, we show the following **3 ancestor trees**.

**First**, the **ancestor tree** (click 5.9.8) of `PDSCH-ConfigCommon`, of `PDSCH-Config` and of `PDSCH-ServingCellConfig` up to `CellGroupConfig` is:

```
                                CellGroupConfig (click 3.7.3)
spCellConfig                      SpCellConfig (click 3.9.5)
  reconfigurationWithSync           ReconfigurationWithSync (0,1) (click 3.9.8)
    spCellConfigCommon                ServingCellConfigCommon (click 3.9.6)
      downlinkConfigCommon              DownlinkConfigCommon (click 2.2.6.8)
        initialDownlinkBWP                BWP-DownlinkCommon (click 2.2.8.6)
          pdsch-ConfigCommon                PDSCH-ConfigCommon (click 6.18.5)
  spCellConfigDedicated             ServingCellConfig (click 3.9.7)
    begin of DownlinkConfig parameter set (click 2.2.8.3)
    initialDownlinkBWP                BWP-DownlinkDedicated (click 2.2.8.7)
      pdsch-Config                      PDSCH-Config (0,1) (click 6.18.6)
      cfr-ConfigMulticast-r17           CFR-ConfigMulticast-r17 (0,1)
        pdsch-ConfigMulticast-r17         PDSCH-Config (0,1) (click 6.18.6)
    downlinkBWP-ToReleaseList         BWP-Id (0..4) (click 2.2.8.13)
    downlinkBWP-ToAddModList          BWP-Downlink (0..4) (click 2.2.8.5)
      bwp-Common                        BWP-DownlinkCommon (click 2.2.8.6)
        pdsch-ConfigCommon                PDSCH-ConfigCommon (click 6.18.5)
      bwp-Dedicated                     BWP-DownlinkDedicated (click 2.2.8.7)
        pdsch-Config                      PDSCH-Config (0,1) (click 6.18.6)
        cfr-ConfigMulticast-r17           CFR-ConfigMulticast-r17 (0,1)
          pdsch-ConfigMulticast-r17         PDSCH-Config (0,1) (click 6.18.6)
    pdsch-ServingCellConfig           PDSCH-ServingCellConfig (0,1) (click 6.18.10)
    end of DownlinkConfig parameter set
sCellToAddModList                 SCellConfig (0..maxNrofSCells) (click 3.9.4)
  sCellConfigCommon                 ServingCellConfigCommon (0,1) (click 3.9.6)
    downlinkConfigCommon              DownlinkConfigCommon (click 2.2.6.8)
      initialDownlinkBWP                BWP-DownlinkCommon (click 2.2.8.6)
        pdsch-ConfigCommon                PDSCH-ConfigCommon (click 6.18.5)
  sCellConfigDedicated              ServingCellConfig (click 3.9.7)
    begin of DownlinkConfig parameter set (click 2.2.8.3)
    initialDownlinkBWP                BWP-DownlinkDedicated (click 2.2.8.7)
      pdsch-Config                      PDSCH-Config (0,1) (click 6.18.6)
      cfr-ConfigMulticast-r17           CFR-ConfigMulticast-r17 (0,1)
        pdsch-ConfigMulticast-r17         PDSCH-Config (0,1) (click 6.18.6)
    downlinkBWP-ToAddModList          BWP-Downlink (0..maxNrofBWPs) (click 2.2.8.5)
      bwp-Common                        BWP-DownlinkCommon (click 2.2.8.6)
        pdsch-ConfigCommon                PDSCH-ConfigCommon (click 6.18.5)
      bwp-Dedicated                     BWP-DownlinkDedicated (click 2.2.8.7)
        pdsch-Config                      PDSCH-Config (0,1) (click 6.18.6)
        cfr-ConfigMulticast-r17           CFR-ConfigMulticast-r17 (0,1)
          pdsch-ConfigMulticast-r17         PDSCH-Config (0,1) (click 6.18.6)
    pdsch-ServingCellConfig           PDSCH-ServingCellConfig (0,1) (click 6.18.10)
    end of DownlinkConfig parameter set
```

Some **relevant remarks** on this **first ancestor tree** are:

- For the **ancestor trees** of `CellGroupConfig` (click 3.7.3) **up to** some **RRC NR messages** click 3.7.1 and **up to** some **RRC inter-node messages** click 3.7.2.
- The constant `maxNrofSCells` is `31`, which is the **maximum quantity** of **UE**'s **secondary cells** (i.e., **other** than the **special cells**) in **each cell group**
- The constant `maxNrofBWPs` is `4`, which is the **maximum quantity** of **UE**'s **additional bandwidth parts** (i.e., **other** than the **initial bandwidth part**).

**Second**, the **additional ancestor tree** (click 5.9.8) of **PDSCH-Config**, **up to** the **top**, is:

```
                                    DL-DCCH-Message (click 5.8.5.4)
message                               DL-DCCH-MessageType (click 5.8.5.4)
  c1                                    CHOICE
    rrcRelease                            RRCRelease (click 5.8.5.4)
      rrcRelease                            RRCRelease-IEs
        suspendConfig                         SuspendConfig (0,1)
          sdt-Config-r17                        SDT-Config-r17 (0,1)
            sdt-MAC-PHY-CG-Config-r17             SDT-CG-Config-r17 (0,1)
                                                    OCTET STRING (CONTAINING
                                                    SDT-MAC-PHY-CG-Config-r17)
              cg-SDT-ConfigInitialBWP-DL-r17      BWP-DownlinkDedicatedSDT-r17
                pdsch-Config-r17                    PDSCH-Config (0,1) (6.18.6)
```

Some **relevant remarks** on this **second ancestor tree** are:

- This **PDSCH-Config** instance is contained in **RRCRelease message**, in addition to the ones contained in **RRCReconfiguration**, **RRCResume** and **RRCSetup** (click 5.8.5.11) **messages** from the **first ancestor tree**.
- This **PDSCH-Config** instance is devoted to **small data transmission** (**SDT**).

**Third**, the **ancestor tree** (click 5.9.8) of **PDSCH-ConfigBroadcast-r17**, **up to** the **top**, is:

```
                                  MCCH-Message-r17
message                             MCCH-MessageType-r17
  c1                                  CHOICE
    mbsBroadcastConfiguration-r17   MBSBroadcastConfiguration-r17
      criticalExtensions              CHOICE
        mbsBroadcastConfiguration-r17   MBSBroadcastConfiguration-r17-IEs
          pdsch-ConfigMTCH-r17            PDSCH-ConfigBroadcast-r17 (0,1)
                                                                    (6.18.7)
```

Some **relevant remarks** on this **third ancestor tree** are:

- The **PDSCH-ConfigBroadcast-r17** data type is devoted to **MBS broadcast**.

Recall that **Sidelink** channels (and consequently **Sidelink** data types and parameters) are **not** covered in this book.

## 6.18.5 PDSCH-ConfigCommon data type

It can be **both broadcast** (or **unicast**) within **SIB1** (click 5.3.4) and **unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to SIB1**, click 6.18.2.

For its **ancestor tree up to CellGroupConfig** (click 3.7.3), click 6.18.4.

Its **only parent data type** is **BWP-DownlinkCommon** (click 2.2.8.6).

Its **only parameter** is **common** (click 5.1.3).

**Each** of its **instances** defines the **list** of **time-domain configurations** for **timing** from **Downlink assignment** to **Downlink data**, of its corresponding **bandwidth part** (**initial** or **additional**), of **common configuration**, of its corresponding **cell** (**special** or **secondary**), of its corresponding **cell group** (**master** or **secondary**).

Its **only** parameter (with its data type and semantics) is:

- **`pdsch-TimeDomainAllocationList PDSCH-TimeDomainResourceAllocationList`**: optional, common

  It is the **list** of **time-domain configurations** for **timing** from **Downlink assignment** to **Downlink data** (refer to table 5.1.2.1.1-1 of TS 38.214).

  Its **data type** is a **`SEQUENCE OF PDSCH-TimeDomainResourceAllocation`** (click 6.18.8), defined as follows:

```
PDSCH-TimeDomainResourceAllocationList ::= SEQUENCE
      (SIZE(1..maxNrofDL-Allocations)) OF
      PDSCH-TimeDomainResourceAllocation
```

  where **`maxNrofDL-Allocations`** is **`16`**, which is the **maximum quantity** of **PDSCH time-domain resource allocations** in **each** of the **UE**'s **Downlink bandwidth parts**.

For more detail on this parameter, refer to the table "*PDSCH-ConfigCommon field descriptions*" of TS 38.331.

Its **complete ASN.1** (click 5.9) specification is:

```
PDSCH-ConfigCommon ::= SEQUENCE {
  pdsch-TimeDomainAllocationList PDSCH-TimeDomainResourceAllocationList
                                 OPTIONAL,                    -- Need R
  ...
}
```

## 6.18.6 `PDSCH-Config` data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** **`CellGroupConfig`** (click 3.7.3), click 6.18.4.

Its **3 parent data types** are **`BWP-Downlink`** (click 2.2.8.5), **`BWP-DownlinkDedicatedSDT-r17`** and **`CFR-ConfigMulticast-r17`**.

**All** its parameters are **dedicated** (click 5.1.3).

**Each** of its **instances** defines: a) several parameters for **all PDSCH resources** (click 7.6.1); b) a certain amount of **individual PDSCH resources** (click 7.6.1); and c) several **PDSCH resource sets** (click 7.6.7).

**Each** of its **instances** configures the **UE** with **most PDSCH variables**, of its corresponding **bandwidth part** (**initial** or **additional**), of **dedicated configuration**, of its corresponding **cell** (**special** or **secondary**), of its corresponding **cell group** (**master** or **secondary**). The "**corresponding**" variables are determined by the **specific place**, within the overall **ASN.1 specification**, where this **data type** has been **instantiated**.

### 6.18.6.1 PDSCH-Config parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- **dmrs-DownlinkForPDSCH-MappingTypeA DMRS-DownlinkConfig** (click 6.18.11): optional, dedicated
- **dmrs-DownlinkForPDSCH-MappingTypeB DMRS-DownlinkConfig** (click 6.18.11): optional, dedicated
- **dmrs-DownlinkForPDSCH-MappingTypeA-DCI-1-2-r16 DMRS-DownlinkConfig** (click 6.18.11): optional, dedicated
- **dmrs-DownlinkForPDSCH-MappingTypeB-DCI-1-2-r16 DMRS-DownlinkConfig** (click 6.18.11): optional, dedicated

  **Each** of these **4 parameters** provides the **PDSCH DM-RS time-domain resource allocation** (click 7.11.5.1) using (respectively) **mapping** **typeA** or **typeB**. The **mapping type** may be chosen **dynamically** with parameter **mappingType** from **PDSCH-TimeDomainResourceAllocation** (click 6.18.8).

  The **first two** parameters apply to **DCI Format 1_1**, while the **last two** parameters (ending in "**-DCI-1-2-r16**") apply to **DCI Format 1_2** (refer to clause 7.3.1 of TS 38.212).

- **dummy SetupRelease{Dummy-TDRA-List}**: optional, dedicated

  This parameter is **not used**. If **received** it shall be **ignored** by the **UE**.

- **maxNrofCodeWordsScheduledByDCI ENUMERATED {n1, n2}**: optional, dedicated

  It is the **quantity** of **codeword(s)** that may be **transmitted** in **1 DCI PDSCH assignment**.

- **pdsch-AggregationFactor ENUMERATED {n2, n4, n8}**: optional, dedicated

  It is the **quantity** of **repetitions** for **data** (refer to clause 5.1.2.1 of TS 38.214). If it is **absent** in a **PDSCH-Config** **instance** which is **not** used for **MBS CFR**, the **UE** applies the value **1**.

- **pdsch-TimeDomainAllocationList PDSCH-TimeDomainResourceAllocationList**: optional, dedicated

  It is a **list** of **time-domain configurations** for **timing** from **Downlink assignment** to **Downlink data** (refer to table 5.1.2.1.1-1 of TS 38.214).

  Its **data type** is a **SEQUENCE OF PDSCH-TimeDomainResourceAllocation** (click 6.18.8), defined as follows:

```
PDSCH-TimeDomainResourceAllocationList ::= SEQUENCE
      (SIZE(1..maxNrofDL-Allocations)) OF
      PDSCH-TimeDomainResourceAllocation
```

where **maxNrofDL-Allocations** is **16**, which is the **maximum quantity** of **PDSCH time-domain resource allocations** in this **list**, that can be configured to **each** of the **UE**'s **Downlink bandwidth parts**.

This parameter applies to **PDSCH transmissions scheduled** with a **DCI message** with **Formats 1_0** or **1_1** (refer to table 5.1.2.1.1-1 of TS 38.214). In case that **pdsch-TimeDomainAllocationListDCI-1-2-r16** from this **same data type** is **absent**, this **parameter** also applies to **DCI messages** with **Format 1_2**.

Notice that there is a **version-different parameter** **pdsch-TimeDomainAllocationList-r16** (see the next bullet) within this same **data type**. The **NR node** does **not configure both** of them in the **same instance** of **PDSCH-Config**.

Notice that there is a **same-name** (**common**) **parameter**, with the **same data type** and the **same semantics**, from **PDSCH-ConfigCommon** (click 6.18.57.15.5).

- **pdsch-TimeDomainAllocationList-r16** **PDSCH-TimeDomainResourceAllocationList-r16**: optional, dedicated

  It is a **list** of **time-domain configurations** for **timing** from **Downlink assignment** to **Downlink data** (refer to table 5.1.2.1.1-1 of TS 38.214).

  Its **data type** is a **SEQUENCE OF** **PDSCH-TimeDomainResourceAllocation-r16** (click 6.18.8), defined as follows:

```
PDSCH-TimeDomainResourceAllocationList-r16 ::= SEQUENCE
      (SIZE(1..maxNrofDL-Allocations)) OF
      PDSCH-TimeDomainResourceAllocation-r16
```

  where **maxNrofDL-Allocations** is **16**, which is the **maximum quantity** of **PDSCH time-domain resource allocations** in this **list**, that can be configured to **each** of the **UE**'s **Downlink bandwidth parts**.

  This parameter applies to **PDSCH transmissions scheduled** with a **DCI message** with **Formats 1_0** or **1_1** (refer to table 5.1.2.1.1-1 of TS 38.214). In case that **pdsch-TimeDomainAllocationListDCI-1-2-r16** from this **same data type** is **absent**, this **parameter** also applies to **DCI messages** with **Format 1_2**.

  Notice that there is a **version-different parameter** **pdsch-TimeDomainAllocationList-r16** (see the previous bullet) within this same **data type**. The **NR node** does **not configure both** of them in the **same instance** of **PDSCH-Config**.

- **pdsch-TimeDomainAllocationListDCI-1-2-r16** **PDSCH-TimeDomainResourceAllocationList-r16**: optional, dedicated

  It is a **list** of **PDSCH time-domain resource assignments** (click 6.14.4 and refer to clause 5.1.2.1 of TS 38.214).

  Its **data type** is a **SEQUENCE OF** **PDSCH-TimeDomainResourceAllocation-r16** (click 6.18.8), defined as follows:

```
PDSCH-TimeDomainResourceAllocationList-r16 ::= SEQUENCE
      (SIZE(1..maxNrofDL-Allocations)) OF
      PDSCH-TimeDomainResourceAllocation-r16
```

  where **maxNrofDL-Allocations** is **16**, which is the **maximum quantity** of **entries** in the **list**.

This **parameter** can be configured for **each Downlink bandwidth part**.

If **present**, this parameter applies to **PDSCH transmissions scheduled** with a **DCI message** with **Format 1_2** (refer to table 5.1.2.1.1-1A of TS 38.214).

- **pdsch-TimeDomainAllocationListForMultiPDSCH-r17 SetupRelease{MultiPDSCH-TDRA-List-r17}**: optional, dedicated

  It is a **list** of 1 to 16 **multi-entries** of **PDSCH time-domain resource assignments**. **Each** such **multi-entry corresponds** to a **multiple PDSCH time-domain resource assignment** (click 6.14.4 and refer to clause 5.1.2.1 of TS 38.214).

  Its **data type** **MultiPDSCH-TDRA-List-r17** is a **SEQUENCE OF** **MultiPDSCH-TDRA-r17**, defined as follows:

```
MultiPDSCH-TDRA-List-r17 ::= SEQUENCE
      (SIZE(1..maxNrofDL-AllocationsExt-r17)) OF
      MultiPDSCH-TDRA-r17
```

  where **maxNrofDL-AllocationsExt-r17** is **64**, which is the **maximum quantity** of **PDSCH time-domain resource allocations** in this **list**, that can be configured to **each** of the **UE**'s **Downlink bandwidth parts**.

  **Each instance** of the **data type** **MultiPDSCH-TDRA-r17** represents a **multiple PDSCH time-domain resource assignment**. **Each instance** of **MultiPDSCH-TDRA-r17** contains a **list** of 1 to 8 **PDSCH time-domain resource assignments**. The **data type** **MultiPDSCH-TDRA-r17** is a **SEQUENCE OF** **PDSCH-TimeDomainResourceAllocation-r16**, defined as follows:

```
MultiPDSCH-TDRA-r17 ::= SEQUENCE {
  pdsch-TDRA-List-r17 SEQUENCE (SIZE(1..maxNrofMultiplePDSCHs-r17))
                      OF PDSCH-TimeDomainResourceAllocation-r16,
...
}
```

  where **maxNrofMultiplePDSCHs-r17** is **8**, which is the **maximum quantity** of **PDSCH time-domain resource allocations** in **each multiple-PDSCH assignment**.

  This parameter **only applies PDSCH transmissions scheduled** with a **DCI message** with **Format 1_1**, because it is the **only Format** that allows for **multiple PDSCH assignments**.

  Notice that there is **data type** **PUSCH-TimeDomainResourceAllocation-r16** (see within 7.15.5.1) that although has a **different name**, it has **reciprocal semantics** (in respect to **PUSCH**) to the ones of **MultiPDSCH-TDRA-r17** (see within 6.18.6.1).

- **resourceAllocation ENUMERATED {resourceAllocationType0, resourceAllocationType1, dynamicSwitch}**: mandatory, dedicated

- **resourceAllocationDCI-1-2-r16 ENUMERATED {resourceAllocationType0, resourceAllocationType1, dynamicSwitch}**: optional, dedicated

  **Each** of these **2 parameters** indicates the **PDSCH frequency-domain resource allocation** of **either DFA-Type 0** or **DFA-Type 1** (click 6.14.3) for **non-fallback DCI**. Parameter **resourceAllocation** applies to **DCI Format 1_1**, while

parameter **resourceAllocationDCI-1-2-r16** applies to **DCI Format 1_2**. For more detail, refer to clause 5.1.2.2 of TS 38.214.

For more detail on these parameters, refer to the table "*PDSCH-Config field descriptions*" of TS 38.331.

### 6.18.6.2 PDSCH-Config ASN.1 specification

**Part** of its **ASN.1** (click 5.9) specification is:

```
PDSCH-Config ::= SEQUENCE {
  dataScramblingIdentityPDSCH INTEGER (0..1023) OPTIONAL,           -- Need S
  dmrs-DownlinkForPDSCH-MappingTypeA SetupRelease {DMRS-DownlinkConfig}
                                     OPTIONAL,                      -- Need M
  dmrs-DownlinkForPDSCH-MappingTypeB SetupRelease {DMRS-DownlinkConfig}
                                     OPTIONAL,                      -- Need M
  Other parameters (not shown)
  vrb-ToPRB-Interleaver ENUMERATED {n2, n4} OPTIONAL,               -- Need S
  resourceAllocation ENUMERATED {resourceAllocationType0,
                                 resourceAllocationType1, dynamicSwitch},
  pdsch-TimeDomainAllocationList SetupRelease {
             PDSCH-TimeDomainResourceAllocationList} OPTIONAL,      -- Need M
  pdsch-AggregationFactor ENUMERATED {n2, n4, n8} OPTIONAL,         -- Need S
  Other parameters (not shown)
  maxNrofCodeWordsScheduledByDCI ENUMERATED {n1, n2} OPTIONAL,      -- Need R
  Other parameters (not shown)
  maxMIMO-Layers-r16 SetupRelease{MaxMIMO-LayersDL-r16} OPTIONAL,   -- Need M
  Other parameters (not shown)
  pdsch-TimeDomainAllocationListDCI-1-2-r16 SetupRelease {
             PDSCH-TimeDomainResourceAllocationList-r16} OPTIONAL, -- Need M
  Other parameters (not shown)
  resourceAllocationDCI-1-2-r16 ENUMERATED {resourceAllocationType0,
        resourceAllocationType1, dynamicSwitch} OPTIONAL,           -- Need M
  priorityIndicatorDCI-1-1-r16 ENUMERATED {enabled} OPTIONAL,       -- Need S
  dataScramblingIdentityPDSCH2-r16 INTEGER (0..1023) OPTIONAL,      -- Need R
  pdsch-TimeDomainAllocationList-r16 SetupRelease {
             PDSCH-TimeDomainResourceAllocationList-r16} OPTIONAL, -- Need M
  Other parameters (not shown)
  dummy SetupRelease {Dummy-TDRA-List} OPTIONAL,                    -- Need M
  Other parameters (not shown)
  pdsch-TimeDomainAllocationListForMultiPDSCH-r17 SetupRelease {
             MultiPDSCH-TDRA-List-r17} OPTIONAL                     -- Need M
  Other parameters (not shown)
}
```

## 6.18.7 PDSCH-ConfigBroadcast-r17 data type

It is **both broadcast** (or **unicast**) within **other SIBs** and **posSIBs** (click 5.3.5) and **unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** the **top** (within **other SIBs and posSIBs**), click 6.18.3.

For its **ancestor tree up to** **CellGroupConfig** (click 3.7.3) (within the **Non-SI parameter group**), click 6.18.4.

Its **2 parent data types** are **MBSBroadcastConfiguration-r17-IEs** and **CFR-ConfigMCCH-MTCH-r17**.

**All** its **parameters** are **common** (click 5.1.3).

**Each** of its **instances** configures the **UE** with **most variables** for **PDSCH broadcast**.

### 6.18.7.1 PDSCH-ConfigBroadcast-r17 parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- **mcs-Table-r17** **ENUMERATED** {qam256, qam64LowSE}: optional, common

  It indicates which **MCS** table the **UE** shall use for **broadcast PDSCH**. If it is **absent** the **UE** applies the value **64QAM**. This parameter applies to **DCI Format 4_0 messages** with the **last 16 bits** of their **CRC** masked with **MCCH-RNTI** or with **G-RNTI** (refer to clause 5.1.3.1 of TS 38.214).

- **pdsch-TimeDomainAllocationList-r17** **PDSCH-TimeDomainResourceAllocationList-r16**: optional, instantiated as common

  It is a **list** of **PDSCH time-domain resource assignments** (click 6.14.4 and refer to clause 5.1.2.1 of TS 38.214).

  Its **data type** is a **SEQUENCE OF** **PDSCH-TimeDomainResourceAllocation-r16** (click 6.18.9), defined as follows:

  ```
  PDSCH-TimeDomainResourceAllocationList-r16 ::= SEQUENCE
        (SIZE(1..maxNrofDL-Allocations)) OF
        PDSCH-TimeDomainResourceAllocation-r16
  ```

  where **maxNrofDL-Allocations** is **16**, which is the **maximum quantity** of **entries** in the **list**.

  If it is **absent**, the **UE** follows the **PDSCH time domain resource allocation procedure** that is specified in clause 5.1.2.1.1 of TS 38.214.

  This **parameter** applies to **DCI Format 4_0 messages** (refer to table 5.1.2.1.1-1 of TS 38.214).

  This **parameter** can be configured for **each Downlink bandwidth part**.

  It has **version-different parameters** from **PDSCH-ConfigCommon** (click 6.18.5) and from **PDSCH-Config** (click 6.18.6).

- **pdschConfigList-r17** **SEQUENCE**: mandatory, instantiated as common

  It is the **list** of **PDSCH parameters** which can be **configured** per **G-RNTI**. **Only one entity** is allowed to be **configured** for an **instance** that is included in **SIB20**.

  Its **data type** is a **SEQUENCE (SIZE (1..maxNrofPDSCH-ConfigPTM-r17)) OF** **PDSCH-ConfigPTM-r17**.

For more detail on these parameters, refer to the table "*PDSCH-ConfigBroadcast field descriptions*" of TS 38.331.

### 6.18.7.2 PDSCH-ConfigBroadcast-r17 ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
PDSCH-ConfigBroadcast-r17 ::= SEQUENCE {
  pdschConfigList-r17 SEQUENCE (SIZE (1..maxNrofPDSCH-ConfigPTM-r17))
                      OF PDSCH-ConfigPTM-r17,
  pdsch-TimeDomainAllocationList-r17
              PDSCH-TimeDomainResourceAllocationList-r16 OPTIONAL, -- Need R
  rateMatchPatternToAddModList-r17 SEQUENCE (SIZE
      (1..maxNrofRateMatchPatterns)) OF RateMatchPattern OPTIONAL, -- Need R
  lte-CRS-ToMatchAround-r17 RateMatchPatternLTE-CRS OPTIONAL,      -- Need R
  mcs-Table-r17 ENUMERATED {qam256, qam64LowSE} OPTIONAL,          -- Need S
  xOverhead-r17 ENUMERATED {xOh6, xOh12, xOh18} OPTIONAL           -- Need S
}
```

## 6.18.8 PDSCH-TimeDomainResourceAllocation data type

It can be **both broadcast** (or **unicast**) within SIB1 (click 5.3.4) and **unicast** within the **Non-SI parameter group** (click 5.3.6).

Its **only parent data type** is PDSCH-TimeDomainResourceAllocationList (see within 6.18.5 and 6.18.6.1). The **only parent data types** of PDSCH-TimeDomainResourceAllocationList are PDSCH-ConfigCommon (click 6.18.5) and PDSCH-Config (click 6.18.6).

**Each** of its **instances** contains **either all common parameters** or **all dedicated parameters** (click 5.1.3), depending on where it is **instantiated**.

**Each** of its **instances** configures **1** case of **PDSCH time domain allocation** that can be **selected** for **1 PDSCH assignment** within to the corresponding **UE**'s **Downlink bandwidth part**.

Its **ancestor tree**, within SIB1 **up to** PDSCH-ConfigCommon (click 6.18.5) is:

```
                  PDSCH-ConfigCommon (click 6.18.5)
pdsch-TimeDomainAllocationList
                    PDSCH-TimeDomainResourceAllocationList (0,1) (click 6.18.5)
                      PDSCH-TimeDomainResourceAllocation
                      (1..maxNrofDL-Allocations) (click 6.18.8)
```

Some **relevant remarks** on this **ancestor tree** are:

- The constant maxNrofDL-Allocations is 16, which is the maximum **quantity** of **Downlink time-domain resource allocations** in the corresponding **list**.
- For the **ancestor tree** of PDSCH-ConfigCommon (click 6.18.5) **up to** the **top** click 6.18.2.

Its **ancestor tree**, within the **Non-SI parameter group**, **up to** PDSCH-Config (click 6.18.6) is:

```
                  PDSCH-Config (click 6.18.6)
pdsch-TimeDomainAllocationList
                    PDSCH-TimeDomainResourceAllocationList (0,1) (click 6.18.6)
                      PDSCH-TimeDomainResourceAllocation
                      (1..maxNrofDL-Allocations) (click 6.18.8)
```

Some **relevant remarks** on this **ancestor tree** are:

- The constant maxNrofDL-Allocations is 16, which is the maximum **quantity** of **Downlink time-domain resource allocations** in the corresponding **list**.

- For the **ancestor tree** of `PDSCH-Config` (click 6.18.6) **up to** `CellGroupConfig` (click 3.7.3), click 6.18.4.

Notice that there is a **version-different data type** `PDSCH-TimeDomainResourceAllocation-r16` (click 6.18.9) that has **extended semantics**.

Notice that there is the **data type** `PUSCH-TimeDomainresourceAllocation` (click 7.15.6) that has a **reciprocal name** and **reciprocal semantics**

#### 6.18.8.1 `PDSCH-TimeDomainResourceAllocation` parameters

This **data type** shares its table of **field descriptions** in TS 38.331, which is the table "*PDSCH-TimeDomainResourceAllocation field descriptions*", with its **version-different data type** `PDSCH-TimeDomainResourceAllocation-r16` (click 6.18.9).

Its **three parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `k0 INTEGER(0..32)`: optional, common **or** dedicated

  It is the **slot offset** between the **slot** of the **scheduling DCI message** and **its scheduled PDSCH reception** (refer to clause 5.1.2.1 of TS 38.214). If it is **absent**, the **UE** applies the value `0`.

  Notice that there are 2 **version-different** parameters, with a **similar data type** and **exactly** the **same field description**, from `PDSCH-TimeDomainResourceAllocation-r16` (click 6.18.9)

- `mappingType ENUMERATED {typeA, typeB}`: mandatory, common **or** dedicated

  It is the **time-domain resource allocation mapping type** (click 6.14.5.1) for **PDSCH DM-RS**.

  Notice that there is a **same-name** parameter, with the **same data type** and **exactly** the **same field description**, from `PDSCH-TimeDomainResourceAllocation-r16` (click 6.18.9)

  Notice that there is a **same-name** parameter, with the **same data type** and **reciprocal semantics**, from `PUSCH-TimeDomainResourceAllocation` (click 7.15.6).

  Notice that there is a **version-different** parameter, with the **same data type** and **reciprocal semantics**, from `PUSCH-Allocation-r16` (click 7.15.7).

- `startSymbolAndLength INTEGER (0..127)`: mandatory, common **or** dedicated

  It indicates the **start symbol interval** and the **quantity** of **allocated symbol intervals** of the **PDSCH transmission**, encoded as a **start-and-length indicator value** (**SLIV**). The network configures this parameter so that the allocation does **not cross** the **slot boundary** (refer to clause 5.1.2.1 of TS 38.214).

  Notice that there is a **version-different** parameter, with the **same data type** and **exactly** the **same field description**, from `PDSCH-TimeDomainResourceAllocation-r16` (click 6.18.9)

Notice that there is a **same-name** parameter, with the **same data type** and **reciprocal semantics**, from `PUSCH-TimeDomainResourceAllocation` (click 7.15.6).

For more detail on these parameters, refer to the table "*PDSCH-TimeDomainResourceAllocation field descriptions*" of TS 38.331.

### 6.18.8.2 `PDSCH-TimeDomainResourceAllocation` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
PDSCH-TimeDomainResourceAllocation ::= SEQUENCE {
  k0 INTEGER(0..32) OPTIONAL,                                    -- Need S
  mappingType ENUMERATED {typeA, typeB},
  startSymbolAndLength INTEGER (0..127)
}
```

## 6.18.9 `PDSCH-TimeDomainResourceAllocation-r16` data type

It can be **both broadcast** (or **unicast**) within **other SIBs and posSIBS** (click 5.3.5) and **unicast** within the **Non-SI parameter group** (click 5.3.6).

Its **only parent data types** are `PDSCH-TimeDomainResourceAllocationList-r16` (see within 6.18.5 and 6.18.6.1) and `MultiPDSCH-TDRA-r17` (see within 6.18.6.1).

**Each** of its **instances** contains **either all common parameters** or **all dedicated parameters** (click 5.1.3), depending on where it is **instantiated**.

**Each** of its **instances** configures **1** case of **PDSCH time domain allocation** that can be **selected** for **1 PDSCH assignment** within a **multiple PDSCH assignment**, within the corresponding **UE**'s **Downlink bandwidth part**.

Its **ancestor tree**, within the **other SIBs and posSIBs**, **up to** `PDSCH-ConfigBroadcast-r17` (click 6.18.7) is:

```
                  PDSCH-ConfigBroadcast-r17 (click 6.18.7)
pdsch-TimeDomainAllocationList-r17
                    PDSCH-TimeDomainResourceAllocationList-r16 (0,1) (click 6.18.6.1)
                      PDSCH-TimeDomainResourceAllocation-r16
                      (1..maxNrofDL-Allocations) (click 6.18.9)
```

Some **relevant remarks** on this **ancestor tree** are:

- The constant `maxNrofDL-Allocations` is `16`, which is the maximum **quantity** of **Downlink time-domain resource allocations** in the corresponding **list**.
- For the **ancestor tree** of `PDSCH-ConfigBroadcast-r17` **up to** the **top** click 6.18.3.

Its **ancestor tree**, within the **Non-SI parameter group**, **up to** `PDSCH-Config` (click 6.18.6) is:

```
                    PDSCH-Config (click 6.18.6)
pdsch-TimeDomainAllocationList-r16
                      PDSCH-TimeDomainResourceAllocationList-r16 (0,1) (6.18.6.1)
                        PDSCH-TimeDomainResourceAllocation-r16
                        (1..maxNrofDL-Allocations) (click 6.18.9)
pdsch-TimeDomainAllocationListDCI-1-2-r16
                      PDSCH-TimeDomainResourceAllocationList-r16 (0,1) (6.18.6.1)
                        PDSCH-TimeDomainResourceAllocation-r16
                        (1..maxNrofDL-Allocations) (click 6.18.9)
```

```
dummy                  Dummy-TDRA-List (0,1) (click 6.18.6.1)
                         MultiPDSCH-TDRA-r17
                         (1..maxNrofDL-Allocations) (click 6.18.6.1)
    pdsch-TDRA-List-r17 PDSCH-TimeDomainResourceAllocation-r16
                          (1..maxNrofMultiplePDSCHs-r17) (click 6.18.9)
pdsch-TimeDomainAllocationListForMultiPDSCH-r17
                       MultiPDSCH-TDRA-List-r17 (0,1) (click 6.18.6.1)
                         MultiPDSCH-TDRA-r17
                         (1..maxNrofDL-AllocationsExt-r17) (click 6.18.6.1)
    pdsch-TDRA-List-r17 PDSCH-TimeDomainResourceAllocation-r16
                            (1..maxNrofMultiplePDSCHs-r17) (click 6.18.9)
```

Some **relevant remarks** on this **ancestor tree** are:

- The constant `maxNrofDL-Allocations` is `16`, which is the maximum **quantity** of **Downlink time-domain resource allocations** in the corresponding **list**.
- The constant `maxNrofMultiplePDSCHs-r17` is `8`, which is the maximum **quantity** of **Downlink time-domain resource allocations** in the corresponding **list**.
- The constant `maxNrofDL-AllocationsExt-r17` is `64`, which is the maximum **quantity** of **sets** of **Downlink time-domain resource allocations** in the corresponding **list**.
- For the **ancestor tree** of `PDSCH-Config` (click 6.18.6) up to `CellGroupConfig` (click 3.7.3), click 6.18.4.

Notice that there is a **version-different data type** `PDSCH-TimeDomainResourceAllocation` (click 6.18.8), with somehow similar semantics.

Notice that there is the **data type** `PUSCH-Allocation-r16` (click 7.15.7) that has a **somehow reciprocal name** and **reciprocal semantics**.

### 6.18.9.1 `PDSCH-TimeDomainResourceAllocation-r16` parameters

This **data type** shares its table of **field descriptions** in TS 38.331, which is the table "*PDSCH-TimeDomainResourceAllocation field descriptions*", with its **version-different data type** `PDSCH-TimeDomainResourceAllocation` (click 6.18.8).

Its **six parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `k0-r16 INTEGER(0..32)`: optional, common **or** dedicated
- `k0-v1710 INTEGER(33..128)`: optional, common **or** dedicated

  **Each** of their field descriptions is **exactly** the **same** as the one of their **version-different parameter** `k0` from `PDSCH-TimeDomainResourceAllocation` (click 6.18.8). The **data type** of `k0` is exactly the same as the one of `k0-r16`.

  Parameter `k0-v1710` is **only** applicable for **PDSCH subcarrier spacing** of **480 kHz** or **960 kHz**.

- `mappingType ENUMERATED {typeA, typeB}`: mandatory, common **or** dedicated

  Its **data type** and **field description** are **exactly** the **same** as the ones of its **same-name parameter** from `PDSCH-TimeDomainResourceAllocation` click 6.18.8).

Notice that there is a **same-name** parameter, with the **same data type** and **reciprocal semantics**, from `PUSCH-TimeDomainResourceAllocation` (click 7.15.6).

Notice that there is a **version-different** parameter, with the **same data type** and **reciprocal semantics**, from `PUSCH-Allocation-r16` (click 7.15.7).

- `startSymbolAndLength-r16 INTEGER (0..127)`: mandatory, common **or** dedicated

  Its **data type** and **field description** are **exactly** the **same** as the ones of its **version-different parameter** from `PDSCH-TimeDomainResourceAllocation` (click 6.18.8).

  Notice that there is a **version-different** parameter, with the **same data type** and **reciprocal semantics**, from `PUSCH-TimeDomainResourceAllocation` (click 7.15.6).

  Notice that there is a **same-name** parameter, with the **same data type** and **reciprocal semantics**, from `PUSCH-Allocation-r16` (click 7.15.7).

- `repetitionNumber-r16 ENUMERATED {n2, n3, n4, n5, n6, n7, n8, n16}`: optional, common **or** dedicated

- `repetitionNumber-v1730 ENUMERATED {n2, n3, n4, n5, n6, n7, n8, n16}`: optional, common **or** dedicated

  **Each** of them indicates the **quantity** of **PDSCH transmission occasions** for **slot-based repetition** scheme in the corresponding **data type instance** of `RepetitionSchemeConfig`. It is used as specified in TS 38.214.

  Parameter `repetitionNumber-r16` is conditioned to **DCI message Formats 1_0** and **1_0**. Parameter `repetitionNumber-v1730` is conditioned to **DCI Format 1_2**.

  Notice that there are **2 parameters** named `numberOfRepetitions-r16` and `numberOfRepetitionsExt-r17`, with a **similar data type** and **reciprocal semantics**, from `PUSCH-Allocation-r16` (click 7.15.7).

For more detail on these parameters, refer to the table "*PDSCH-TimeDomainResourceAllocation field descriptions*" of TS 38.331.

### 6.18.9.2 `PDSCH-TimeDomainResourceAllocation-r16` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
PDSCH-TimeDomainResourceAllocation-r16 ::= SEQUENCE {
  k0-r16 INTEGER(0..32) OPTIONAL,                                    -- Need S
  mappingType-r16 ENUMERATED {typeA, typeB},
  startSymbolAndLength-r16 INTEGER (0..127),
  repetitionNumber-r16 ENUMERATED {n2, n3, n4, n5, n6, n7, n8, n16}
                                        OPTIONAL, -- Cond Formats1-0and1-1
  ...,
  [[
  k0-v1710 INTEGER(33..128) OPTIONAL                                 -- Need S
  ]],
  [[
```

```
    repetitionNumber-v1730 ENUMERATED {n2, n3, n4, n5, n6, n7, n8, n16}
                           OPTIONAL                              -- Cond Format1-2
  ]]
}
```

### 6.18.10 PDSCH-ServingCellConfig data type

Its **only parent data type** is `ServingCellConfig` (click 3.9.7).

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 6.18.4.

**All** its parameters are **dedicated** (click 5.1.3).

**Each** of its **instances** contains **PDSCH dedicated parameters** for a **given UE**, that apply the **same** for **all** the **bandwidth parts** of the corresponding **serving cell**.

#### 6.18.10.1 PDSCH-ServingCellConfig parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

**HARQ-related parameters:**

- `codeBlockGroupTransmission SetupRelease {PDSCH-CodeBlockGroupTransmission}`: optional, dedicated

  If it is **present**, **PDSCH HARQ** works in **code block group** (**CBG**) **mode** (click 4.5.6.4).

  If it is **absent**, **PDSCH HARQ** works in **transport block mode**.

  It contains `maxCodeBlockGroupsPerTransportBlock`, which indicates the (**maximum**) **quantity** of **CBGs** for **each PDSCH transport block**. Its **data type** is `ENUMERATED {n2, n4, n6, n8}`, that stand for {2, 4, 6, 8}. In case of **2 PDSCH transport blocks**, the **maximum value** that can be configured is `4`.

- `nrofHARQ-ProcessesForPDSCH ENUMERATED {n2, n4, n6, n10, n12, n16}`: optional, dedicated
  `nrofHARQ-ProcessesForPDSCH-v1700 ENUMERATED {n32}`: optional, dedicated

  **Either** of these **2 parameters** indicates the **quantity** of **PDSCH HARQ processes** for this **UE**. The **second parameter** was introduced in **R17** to extend the possible values to **32**. If **neither** of them is **present** the default value is **8**.

Notice that there **2 reciprocal parameters** (to the **first** and **last**) from `PUSCH-ServingCellConfig` (click 7.15.8).

For more detail on **5G HARQ**, click 4.5.6.

For more detail on these parameters, refer to the table "*PDSCH-ServingCellConfig field descriptions*" of TS 38.331.

### 6.18.10.2 PDSCH-ServingCellConfig ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
PDSCH-ServingCellConfig ::= SEQUENCE {
  codeBlockGroupTransmission SetupRelease {PDSCH-CodeBlockGroupTransmission}
                             OPTIONAL,                                  -- Need M
  xOverhead ENUMERATED {xOh6, xOh12, xOh18} OPTIONAL,                   -- Need S
  nrofHARQ-ProcessesForPDSCH ENUMERATED {n2, n4, n6, n10, n12, n16}
                             OPTIONAL,                                  -- Need S
  pucch-Cell ServCellIndex OPTIONAL,                        -- Cond SCellAddOnly
  ...,
  [[
  maxMIMO-Layers INTEGER (1..8) OPTIONAL,                               -- Need M
  processingType2Enabled BOOLEAN OPTIONAL                               -- Need M
  ]],
  [[
  pdsch-CodeBlockGroupTransmissionList-r16 SetupRelease {
             PDSCH-CodeBlockGroupTransmissionList-r16} OPTIONAL     -- Need M
  ]],
  [[
  downlinkHARQ-FeedbackDisabled-r17 SetupRelease {
             DownlinkHARQ-FeedbackDisabled-r17} OPTIONAL,           -- Need M
  nrofHARQ-ProcessesForPDSCH-v1700 ENUMERATED {n32} OPTIONAL        -- Need R
  ]]
}
```

## 6.18.11 DMRS-DownlinkConfig data type

Its **only parent data type** is (click 6.18.6). For the **ancestor tree** of **PDSCH-Config** **up to** **CellGroupConfig** (click 3.7.3), click 6.18.4.

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

**All** its parameters are **dedicated** (click 5.1.3).

**Each** of its **instances** configures a **UE** with the configuration for **either** **PDSCH DM-RS mapping type** **typeA** or **typeB** (see **PDSCH-Config** **parameters**, click 6.18.6.1). However, **only** the value of its parameters **dmrs-Type**, **dmrs-AdditionalPosition** and **maxLength** may be **different** between the configuration of **typeA** and of **typeB**.

### 6.18.11.1 DMRS-DownlinkConfig parameters

**All** its **parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- **dmrs-AdditionalPosition** **ENUMERATED** {**pos0**, **pos1**, **pos3**}: optional, dedicated

  Its value is used to determine the **Downlink symbol intervals** that **contain PDSCH DM-RS**. If it is **absent**, the **UE** applies the value **pos2**. For more detail, click 6.14.5.1 and refer to 7.4.1.1.2 of TS 38.211.

- **dmrs-Downlink-r16** **ENUMERATED** {**enabled**}: optional, dedicated

  It indicates whether **low PAPR PDSCH DM-RS** is used (refer to 7.4.1.1.1 of TS 38.211).

- **dmrs-Type** **ENUMERATED** {**type2**}: optional, dedicated

  It is the **PDSCH DM-RS frequency-domain configuration type** to be used (refer to clause 7.4.1.1.1 of TS 38.211). If it is **present**, the **UE** uses **DDM-Type 2**. If it is **absent**, the **UE** uses **DDM-Type 1**.

- `maxLength ENUMERATED {len2}`: optional, dedicated

  It indicates the **maximum quantity** of **front-load PDSCH DM-RS symbols**. In other words, it indicates the **quantity** of **consecutive DM-RS symbols** for **each DM-RS position**. If it is **absent**, **single-symbol DM-RS** is used. If it is **present**, **single-symbol DM-RS** or **double symbol DM-RS** is used as indicated by the "***Antenna ports***" field of the **scheduling DCI message** (click 6.9). For more detail, click 6.14.5.1 and refer to 7.4.1.1.2 of TS 38.211.

- `phaseTrackingRS PTRS-DownlinkConfig`: optional, dedicated

  It configures **PDSCH phase tracking reference signals** (**PTRS**). If it is **absent**, the **UE** assumes that **Downlink PTRS** are **absent** (refer to clause 5.1.6.3 of TS 38.214).

- `scramblingID0 INTEGER (0..65535)`: optional, dedicated

- `scramblingID1 INTEGER (0..65535)`: optional, dedicated

  These two parameters are the **PDSCH DM-RS scrambling initialization 0** and **1** (respectively). If they are **absent**, the **UE** uses the **physical cell identity** (**PCI**, click 3.5.3.2) of the cell. For more detail on these scrambling ids, refer to clause 7.4.1.1.1 of TS 38.211.

For more detail on these parameters, refer to the table "*DMRS-DownlinkConfig field descriptions*" of TS 38.331.

### 6.18.11.2 `DMRS-DownlinkConfig` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
DMRS-DownlinkConfig ::= SEQUENCE {
  dmrs-Type ENUMERATED {type2} OPTIONAL,                          -- Need S
  dmrs-AdditionalPosition ENUMERATED {pos0, pos1, pos3} OPTIONAL, -- Need S
  maxLength ENUMERATED {len2} OPTIONAL,                           -- Need S
  scramblingID0 INTEGER (0..65535) OPTIONAL,                      -- Need S
  scramblingID1 INTEGER (0..65535) OPTIONAL,                      -- Need S
  phaseTrackingRS SetupRelease {PTRS-DownlinkConfig} OPTIONAL,    -- Need M
  ...,
  [[
  dmrs-Downlink-r16 ENUMERATED {enabled} OPTIONAL                 -- Need R
  ]]
}
```

## 6.19 PSS and SSS

The **Primary Synchronization Signal** (**PSS**) and the **Secondary Synchronization Signal** (**SSS**) are **sent**, together with the **Physical Broadcast Channel** (**PBCH**, click 6.2), as the **Synchronization Signal and PBCH Block** (**SSB**, click 6.1).

For **PSS** and **SSS scheduling**, click 6.1.2.

Both the **PSS** and the **SSS** consist of sending a **127-bit complex number sequence across** the **frequency domain**, occupying **only 1 symbol interval**, and **127 subcarriers**. This is somehow similar to the way the **RA preamble** (click 7.2.1) works.

For more detail on the **contents** of **PSS**, click 6.19.2.

For more detail on the **contents** of **SSS**, click 6.19.3.

Source web pages:

- https://es.mathworks.com/videos/5g-explained-synchronization-signal-blocks-in-5g-nr-1577445332091.html?s_tid=srchtitle
- https://www.linkedin.com/pulse/5g-cell-selection-process-bikas-singh/
- https://www.sharetechnote.com/html/5G/5G_PSS.html#:~:text=5G%2FNR%20%2D%20PSS,to%20the%20structure%20of%20SSB)
- http://howltestuffworks.blogspot.com/2019/10/5g-nr-primary-synchronization-signal-pss.html
- https://howltestuffworks.blogspot.com/2019/10/5g-nr-secondary-synchronization-signal.html

### 6.19.1 PSS and SSS scheduling

**PSS** and **SSS** are **jointly broadcast** with **PBCH** (click 6.2) as the "**Synchronization signal** and **PBCH block**" (**SSB**, click 6.1). The **PSS** and **SSS scheduling** is therefore the **same** as the one of **SSB**. The **SSB** is **broadcast** in **periodical bursts**, with one (or more) **SSB instances** in each **burst**. **One SSB burst** is **usually** sent every **20 ms**, but other values can be configured. For more detail on **SSB scheduling**, click 6.1.2.

### 6.19.2 PSS contents and processing

The **subcarrier spacing** and the **cyclic prefix** of **PSS** are the ones of **SSB** (click 6.1.4).

**Each primary synchronization signal** (**PSS**) is **1** out of **3** **cyclic shifts** of a **reg-7 M-sequence** (click 11.20.6) whose length is **127** **bits**.

The **PSS** provides the **UE** with **Downlink frequency** and **symbol synchronization**. The **PSS also** provides the **UE** with the **PCI identity** "$N_{ID}^{(2)}$".

The **PCI identity** "$N_{ID}^{(2)}$" is **1** of the **2** **parts** of the **Physical Cell Identity** (**PCI**, click 3.5.3.2). The **other part** of the **PCI** is the **PCI group** "$N_{ID}^{(1)}$", that is conveyed in **SSS** (click 6.19.3). Do **not confuse** the **PCI identity** (click 3.5.3.2), the **PCI group**, the **physical cell identity** (**PCI**), and the **network level cell identities** (click 3.5.3) among themselves, because all of them are **different identifiers**.

The **PCI identity** "$N_{ID}^{(2)}$" can take the values {**0..2**}, **each** of which is mapped to **1** of the **3** **cyclic shifts** of the **reg-7 M-sequence**. The resulting **127**-bit **sequence** is modulated using **BPSK** and transmitted as the **PSS**.

The **PSS** is formally defined with the following "$d_{PSS}()$" equation:

$$d_{PSS}(n, N_{ID}^{(2)}) = 1 - 2 \cdot sx(n, N_{ID}^{(2)}), \text{ where } n \in \{0..126\}$$

which converts the **shifted M-sequence** "$sx()$" of "**0**" and "**1**" into a **sequence** of "**1**" and "**-1**". **Each** such **value** is used as **multiplier** of the base **BPSK symbol**, and the **resulting symbol** is placed in **1 PSS RE**. The **first symbol** corresponding to "$d_{PSS}(0)$" is placed in the **lowest frequency subcarrier** of **PSS**, and **each successive "$d_{PSS}(n)$" symbol** is placed in the **next PSS subcarrier upwards**.

The definition of the **shifted M-sequence** "$sx()$" is:

$$sx(n, N_{ID}^{(2)}) = x(n + 43 \bullet N_{ID}^{(2)})$$

which produces a **cyclic shift** of the **M-sequence** "$x()$" by either **0**, **43** or **86** **bits**, corresponding to the "$N_{ID}^{(2)}$" values **0**, **1** or **2** (respectively). Notice that "+" represents addition modulus **127**.

Some **relevant remarks** on this equation are:

- The **shift spacing** value of **43** corresponds to the **maximum** for **three shifts** over a total of **127 possible shifts**, which is ***ceiling***(**127**/**3**)=**43**.
- Since **127**/**3** is **not** an **integer**, the **3** resulting **shift spacings** are **not** the **same**. The resulting **circular shift spacings** are **43**, **43** and **41**.

The definition of the **M-sequence** "$x()$" is:

$$x(n) = x(n-1) \oplus x(n-7)$$

with $[x(6)\ x(5)\ x(4)\ x(3)\ x(2)\ x(1)\ x(0)] = [1\ 1\ 1\ 0\ 1\ 1\ 0]$

For more detail on **PSS**, refer to clauses 7.4.2 and 7.4.2.2 of TS 38.211.

In **LTE**, the **PSS** is generated using a **Zadoff-Chu sequence** (click 11.20.4).

### 6.19.3 SSS contents and processing

The **subcarrier spacing** and the **cyclic prefix** of **SSS** are the ones of **SSB** (click 6.1.4).

**Each secondary synchronization signal** (**SSS**) is **1** out of **3** **cyclic shifts** of **112** **reg-7** **Gold sequences** (click 11.20.5) from the **same family**, and whose length is **127** **bits**. There are **3** **sets**, **each** containing **3** **cyclic shifts** of **112** **Gold sequences**. The **set** to use depends on the value of the **PCI identity** "$N_{ID}^{(2)}$". The "$N_{ID}^{(2)}$" is already known to the **UE** because the **PSS** is acquired **before** acquiring the **SSS**.

The **SSS** provides the **UE** with measurements of **reference signal received power** (**RSRP**) and of **reference signal received quality** (**RSRQ**). The **SSS** **also** provides the **UE** with the **PCI group** "$N_{ID}^{(1)}$".

The **PCI group** "$N_{ID}^{(1)}$" is **1** of the **2** **parts** of the **Physical Cell Identity** (**PCI**, click 3.5.3.2). The **other** **part** of the **PCI** is the **PCI identity** "$N_{ID}^{(2)}$", that is conveyed in **PSS** (click 6.19.1). Do **not confuse** the **PCI group** (click 3.5.3.2), the **PCI identity**, the **physical cell identity** (**PCI**) and the **network level cell identities** (click 3.5.3) among themselves, because all of them are **different identifiers**.

The **PCI group** "$N_{ID}^{(1)}$" can take the values {**0**..**335**}, **each** of which is mapped to **1** out of **3** **cyclic shifts**, of **1** out of **112** **reg-7** **Gold sequences**, from the **set** corresponding to the value of "$N_{ID}^{(2)}$". The corresponding **127**-bit **shifted** **reg-7** **Gold sequence** is modulated using **BPSK** and transmitted as the **SSS**.

The **SSS** is formally defined with the following "$d_{SSS}()$" equation:

$$d_{SSS}(n, N_{ID}^{(1)}, N_{ID}^{(2)}) = [1 - 2 \bullet sx_1(n, N_{ID}^{(1)}, N_{ID}^{(2)})] \bullet [1 - 2 \bullet sx_2(n, N_{ID}^{(1)})]$$

where $n \in \{0..126\}$

which converts the **shifted M-sequences** "$sx_1()$" and "$sx_2()$" of "**0**" and "**1**" into one **sequence** of "**1**" and "**-1**", and their **product** is equivalent to an **exclusive-or**, therefore resulting in a **reg-7** **Gold sequence**. **Each** such **value** is used as **multiplier** of the base **BPSK** **symbol**, and the **resulting** **symbol** is placed in **one** **SSS** **RE**. The **first** **symbol**

corresponding to "$d_{SSS}(0)$" is placed in the **lowest frequency subcarrier** of **SSS**, and **each successive** "$d_{SSS}(n)$" **symbol** is placed in the **next SSS subcarrier upwards**.

The definition of the **two shifted sequences** "$sx_1()$" and "$sx_2()$" of the **preferred pair** is:

$$sx_1(n, N_{ID}^{(1)}, N_{ID}^{(2)}) = x_1\,(n + S_1(N_{ID}^{(1)}, N_{ID}^{(2)}))$$

$$sx_2(n, N_{ID}^{(1)}) = x_2(n + S_2(N_{ID}^{(1)}))$$

where "+" represents addition modulus **127** and **both** "$x_1()$" and "$x_2()$" use the **same seed** (click 11.20.6.3):

$$[x_1(6)\ x_1(5)\ x_1(4)\ x_1(3)\ x_1(2)\ x_1(1)\ x_1(0)] = [0\ 0\ 0\ 0\ 0\ 0\ 1]$$

$$[x_2(6)\ x_2(5)\ x_2(4)\ x_2(3)\ x_2(2)\ x_2(1)\ x_2(0)] = [0\ 0\ 0\ 0\ 0\ 0\ 1]$$

From (click 11.20.5.1), these two equations are a **Gold sequence** defined as:

$$c(n+S_1(), S_2() - S_1())$$

where the **index** "$i$" of the **Gold sequence** is "$S_2$ - $S_1$", its **shift** is "$S_1$" and "+" and "–" represent addition and subtraction **modulus** **127**. Notice that we have omitted the arguments of "$S_1()$" and "$S_2()$" for clarity's sake.

The **shift** of **each Gold sequence** is:

$$S_1(N_{ID}^{(1)}, N_{ID}^{(2)}) = 15 \bullet floor(N_{ID}^{(1)}/112) + 5\ N_{ID}^{(2)})$$

$$S_1(N_{ID}^{(1)}, N_{ID}^{(2)}) = 15 \bullet floor(N_{ID}^{(1)}/112)$$

$$= 15+5\ N_{ID}^{(2)} \text{ for } N_{ID}^{(1)} \in \{0..111\},$$

$$= 30+5\ N_{ID}^{(2)} \text{ for } N_{ID}^{(1)} \in \{112..223\},$$

$$= 45+5\ N_{ID}^{(2)} \text{ for } N_{ID}^{(1)} \in \{224..335\}$$

$$= \{15, 20 \text{ or } 25\} \text{ for } N_{ID}^{(1)} \in \{0..111\},$$

$$= \{30, 35 \text{ or } 40\} \text{ for } N_{ID}^{(1)} \in \{112..223\},$$

$$= \{45, 50 \text{ or } 55\} \text{ for } N_{ID}^{(1)} \in \{224..335\}$$

where **each** of the **3** **shifts** between **curly braces** corresponds to "$N_{ID}^{(2)} \in \{0, 1 \text{ or } 2\}$", respectively.

The **index** of **each Gold sequence** is:

$$i(N_{ID}^{(1)}, N_{ID}^{(2)}) = S_2(N_{ID}^{(1)}) - S_1(N_{ID}^{(1)}, N_{ID}^{(2)})$$

$$= (N_{ID}^{(1)} \bmod 112) - S_1(N_{ID}^{(1)}, N_{ID}^{(2)})$$

$$= \{0..111\} - 15 - 5\ N_{ID}^{(2)} \text{ for } N_{ID}^{(1)} \in \{0..111\},$$

$$= \{0..111\} - 30 - 5\ N_{ID}^{(2)} \text{ for } N_{ID}^{(1)} \in \{112..223\},$$

$$= \{0..111\} - 45 - 5\ N_{ID}^{(2)} \text{ for } N_{ID}^{(1)} \in \{224..335\}$$

$$= \{-15..96\},\ \{-20..91\} \text{ or } \{-25..86\} \text{ for } N_{ID}^{(1)} \in \{0..111\},$$

$$= \{-30..81\},\ \{-35..76\} \text{ or } \{-40..71\} \text{ for } N_{ID}^{(1)} \in \{112..223\},$$

$$= \{-45..66\},\ \{-50..61\} \text{ or } \{-55..56\} \text{ for } N_{ID}^{(1)} \in \{224..335\}$$

$= \{111..96\}, \{106..91\}$ or $\{101..86\}$ for $N_{ID}^{(1)} \in \{0..111\}$,

$= \{96..81\}, \{91..76\}$ or $\{86..71\}$ for $N_{ID}^{(1)} \in \{112..223\}$,

$= \{81..66\}, \{76..61\}$ or $\{71..56\}$ for $N_{ID}^{(1)} \in \{224..335\}$,

where **each** of the **3 ranges** of "***i*** " between curly braces corresponds to "$N_{ID}^{(2)} \in \{0, 1$ or $2\}$".

Notice that the **ranges** of the **index values** are **modulus 127**. As you may see, **all** the **127** available **Gold sequences** are used.

Notice that the **UE already knows** the value of the **PCI identity** "$N_{ID}^{(2)}$" from **PSS**, and it is now **receiving** the value of the **PCI group** "$N_{ID}^{(1)}$".

Notice also that the **two M-sequences** "$x_1()$" and "$x_2()$" that **also** belong to the **Gold sequence family** are **not** used. The reason for **not** using the **M-sequence** "$x_1()$" is that **PSS** and **SSS** should have a **low cross-correlation**, to reduce their mutual cross interference. By selecting the **PSS M-sequence** as one of the generating **M-sequences** of the **SSS Gold sequence family**, and **not** using it for **SSS**, the following **2 advantages** are obtained:

- It is **guaranteed** that there is **low cross-correlation** between **any** of the **3 PSS M-sequence shifts** and any of the **127 SSS Gold sequences**, because **all** these sequences belong to the same **Gold sequence family**, and therefore, they have a **low pairwise cross-correlation**.
- A **few transistors** are **saved** in the implementation by reusing the **same circuitry** for **both tasks**, because the **M-sequence** used for **PSS** and for generating **SSS** is **the same**.

For more detail on **SSS**, refer to clauses 7.4.2 and 7.4.2.3 of TS 38.211

Source web page:

- https://ieeexplore.ieee.org/document/8422145

## 6.20 CSI-RS

**Channel state information reference signals** (**CSI-RS**) consist of the transmission by the **NR node** of **known signals** that allow **each receiving UE** to determine the **current channel state** (i.e., its **current characteristics**). **CSI** are used for **detailed channel sounding** and for tracking **channel characteristics** that may vary **rapidly** in time/frequency.

The **SSB** (click 6.1) **cannot** provide the **required channel state information** because it is transmitted over a **limited bandwidth** and a **large periodicity interval**. **SSB** can be used for power measurements to estimate *Free-space path-loss* and channel quality but due to limited bandwidth and low duty cycle, **SSB** is not very suitable for more detailed channel sounding.

The **NR node** can configure the **UE** to use **CSI-RS** for:

- **Beam management** (**CQI**, **RI**, **PMI Measurements**): Measurements should be sent by **UE** to **NR nodes** in order to understand and estimate the correct direction of beams.

- **Connected-state mobility**: For calculating **Reference Signal Received Power** (**RSRP**), **Reference Signal Received Quality** (**RSRQ**) and **Signal and Interference to Noise Ratio** (**SINR**).
- **Radio link failure detection**: To check if **channel** is **out-of-sync** or **in-sync**.
- **Beam failure detection and Recovery**: Based on **estimation** of **such signals**, the **UE** can be forced to perform a **contention-free random access** (**CFRA**, click 9.1.2) **procedure**.
- **Time** and **Frequency Synchronization** (**Tracking Reference Signals**).
- **Coordinated MultiPoint** transmission (**CoMP**, click 10.11).

The **UE reports back** (click 4.5.7.6) to the **NR node** relevant channel parameters determined using **CSI-RS**.

**Differences between 5G CSI-RS** and **LTE CRS**

The advantages of using **CSI-RS** over **LTE CRS** are:

- **CSI-RSs** consume fewer air interface resources to perform channel measurement/feedback.
- Different sets of **CSI-RSs** can be allocated to different **UEs** to form directional beams.

**CSI-RS** is configured **per device** rather than **per cell**, as it was in **LTE**. In contrast to the **Cell-specific reference signal** (**CRS**) from **LTE R8**, **LTE R10** introduced this concept of **CSI-RS** with the **addition** of **up to 8-layer spatial multiplexing** where the **Reference Signals** are **not transmitted continuously**, which led to requirement of **8-layer channel estimation** while **only 4-layer spatial multiplexing** was used in **LTE R8**. Extending **CRS** to **8 layers** would have added more **signaling overhead** than desired, resulting in the introduction of **CSI-RS**. Also, **CSI-RS** is transmitted on different **antenna ports** (click 4.11.5) than **CRS** and **CSI-RS** uses **code-domain orthogonality combined** with **time/frequency orthogonality** unlike **CRS** which used **only time/frequency orthogonality**.

As **5G NR** minimizes always on strategy (except for **SSB**), there are no **CRS-like** signals in **NR**, but the same concept of **CSI-RS** is reused and extended in **NR** to provide support for **beam management** and connected-state mobility.

For more detail on **CSI-RS**, refer to clause 7.4.1.5 of TS 38.331.

### 6.20.1 CSI-RS scheduling

The **time-domain locations** of **channel-state information reference signal** (**CSI-RS**) are provided by `firstOFDMSymbolInTimeDomain` and `firstOFDMSymbolInTimeDomain2`, **both** from `CSI-RSResourceMapping` and by `firstOFDMSymbolInTimeDomain` from `CSI-RS-ResourceConfigMobility`, defined relative to the **start** of a **slot**.

For the **specific case** of **Non-Zero Power** (**NZP**) **CSI-RS**, the **time-domain locations** are provided by `firstOFDMSymbolInTimeDomain-r17` from `TRS-ResourceSet-r17`, possibly **adding 4**.

The **frequency-domain locations** are given by **either**: a) a **bitmap** provided by `frequencyDomainAllocation` from `CSI-RS-ResourceMapping`; or b) a **start** and

**range** of **RBs** within either `CSI-RS-ResourceConfigMobility` or `TRS-ResourceSet`. These parameters are used jointly with table 7.4.1.5.3-1 of TS 38.211.

Consequently, the said **CSI-RS** within the carrier can be flexibly configured, but **CSI-RS** it **cannot** be located in any of the following places:

- Within any **CORESET**.
- Within any **SSB**.
- Within any **PDSCH DM-RS**.

One possible configuration for **CSI-RS** is sending (in **one slot**) **1 Resource Element** for **each antenna port** (click 4.11.5) up to **32 antenna ports**. **Resource Elements** corresponding to **CSI-RS antenna ports** in the **frequency domain** can be in **non-consecutive subcarriers**. **Resource Elements** corresponding to **CSI-RS antenna ports** in the **time** domain can be in **non-consecutive symbols**.

Depending on the **quantity** of **antenna ports**, there can be **2** possible **arrangements**:

- **Single-Port**: A **single port CSI-RS** occupies a **single Resource element** (**RE**) within a block corresponding to **1 slot** in the **time domain** and **1 Resource Block** in the **frequency domain**.
- **Multi-Port**: **multiple orthogonally transmitted per-antenna-port CSI-RS**, **sharing** the **overall set** of **REs** assigned for the configured **multi-port CSI-RS**. **Sharing** can be based on **combinations** of: **Code domain sharing** (**CDM**: different **orthogonal patterns**), **Frequency Domain Sharing** (**FDM**: different **subcarriers** within an **OFDM symbol**) or **Time Domain Sharing** (**TDM**: different **OFDM symbols** within a **slot**).

## 6.21 PRS

The **Positioning Reference Signals** (**PRS**) are **Downlink reference signals** transmitted by **NR nodes** to support:

- **UE-based positioning**

  The **UE measures** the received **PRS** and **computes** its **position.**

- **UE-assisted positioning**

  The **UE** sends **raw data Uplink**, and the UE's **position** is **calculated outside** of the **UE** (e.g., in the network).

Some **relevant characteristics** of **PRS** are:

- **Unidirectional**

  They are **Downlink-only**.

- **Specific configuration**

  They are configured with **specific parameters**, independent of the **PDSCH** and the **PDCCH regular resources**.

- **Non-SI configuration**

  Their configuration **parameters** belong to the **Non-SI parameter group** (click 5.3.6).

- **Flexible frequency scheduling**

  Their **frequency scheduling** can be **flexibly configured**.

- **Flexible time scheduling**

  Their **time scheduling** can be **flexibly configured**, as **periodic**, **semi-persistent** or **on-demand**.

- **Muted symbols**

  They support **muted symbols** to **reduce interference** between **neighboring cells**.

- **Multi-beam support**

  They can be **transmitted** via **multiple beams** to support **multi-panel NR nodes**.

- **PRS-specific symbols**

  They use **specific physical-layer symbols**, that are **not** used **other physical signals**, **nor** in the **6 physical binary channels**.

- **High-frequency support**

  They are designed to be **used** in **FR2 bands**, in addition to **FR1 bands**.

- **Massive MIMO**

  Their combination with **massive MIMO** results in **high-precision positioning**.

Some relevant **PRS-based positioning techniques** are:

- **Downlink Time-Difference of Arrival** (**DL-TDOA**)

  The **UE measures** the **relative time difference** of **PRS** that are received from **different cells**.

- **Downlink Angle of Departure** (**DL-AoD**)

  The **UE estimates** the **PRS Angle of Departure** (**AoD**) of **PRS**, based on **beamformed** transmission.

- **Downlink Time of Arrival** (**DL-ToA**)

  The **UE measures** the **Time of Arrival** (**ToA**) of **PRS** from **one or more cells**.

- **Multi-RTT**

  The **UE measures** the **round-trip time** (**RTT**) to **multiple NR nodes**, using **both Downlink** and **Uplink** transmission.

**PRS** are particularly important since **R16**, in which **5G positioning** capabilities were **significantly expanded**.

For more detail on **CSI-RS**, refer to clause 7.4.1.7 of TS 38.331.

### 6.21.1 PRS scheduling

The **time-domain locations** of the **positioning reference signal** (**RIM-RS**) are provided by:

- The **first symbol interval** of **PRS** is given by `dl-PRS-ResourceSymbolOffset-r17` and `dl-PRS-ResourceSlotOffset-r17`, **both** from `NR-DL-PRS-Resource-r17`.

- The **duration** of **PRS** within a **slot** can possibly be given by `numSymbols-r17` from `NR-DL-PRS-PDC-ResourceSet-r17`. Notice that clause 7.4.1.7.3 of TS 38.211 indicates that the **duration** of **PRS** within a **slot** is given by "`dl-PRS-NumSymbols`", but TS 38.331 does **not contain** such a parameter.

The **frequency-domain locations** are given by `dl-PRS-PointA-r16` from `NR-PRS-MeasurementInfo-r16` and `dl-PRS-CombSizeN-AndReOffset-r17` from `NR-DL-PRS-Resource-r17`. These parameters are used jointly with table 7.4.1.7.3-1 of TS 38.211. Notice that clause 7.4.1.7.3 of TS 38.211 indicates that on some cases "$K_{comb}^{PRS}$" is given by **duration** of **PRS** within a **slot** is given by "`dl-PRS-CombSizeN`", but TS 38.331 does **not contain** such a parameter. Parameter dl-PRS-ResourceBandwidth-r17 from `NR-DL-PRS-PDC-ResourceSet-r17` specifies the number of **PRBs** allocated for **all** the **DL-PRS Resource** (allocated **DL-PRS bandwidth**) in multiples of **4 PRBs** in this **resource set**. **All DL-PRS Resources** of the **DL-PRS-PDC Resource Set** have the **same bandwidth**. Integer value `1` corresponds to **24 PRBs**, value `2` corresponds to **28 PRBs**, value `3` corresponds to **32 PRBs** and so on. The **range** of values is therefore from **24 PRBs** to **272 PRBs** in **steps** of **4 PRBs**.

## 6.22 RIM-RS

The **Remote Interference Management Reference Signals** (**RIM-RS**) allow a **NR node** to **detect** the **presence** and **activity** of **neighbor** (click 3.1.2) **NR nodes** that may **interfere** with **its UEs**, **without** requiring a **signaling interface** (e.g., **Xn**) between them. The **NR node** that is performing the **RIM-RS detection** is called the **RIM-reporting NR node**. **Each NR node** that is **transmitting RIM-RS** is called the **RIM-sending NR node**.

An **overview** of the **interference mitigation** that is **enabled** by **RIM-RS** is:

1. **RIM-RS transmission**

   An **RIM-sending NR node** is configured to **transmit RIM-RS** on **specified time–frequency** resources. The **transmitted RIM-RS** are **not** meant for **UEs**, but **rather** for **neighbor NR nodes**.

2. **RIM-RS Reception**

   A **neighboring RIM-reporting NR node** monitors for **RIM-RS** within **specific measurement windows**. It measures the **RIM-RS Received Power** (**RRP**) per **symbol interval** and per **subcarrier** (or per **PRB**).

3. **Detection and Reporting**

   ***If*** the **RRP exceeds** a configured **threshold**, ***then***, the **RIM-reporting NR node**:

   - **Identifies** the **time-frequency interference region**.
   - **Optionally** determines the **Physical Cell ID** (**PCI**) of the **interfering cell**.

   Afterwards, the **RIM-reporting NR node** generates a **report**. This **RIM report** may be used **locally** for **autonomous scheduling decisions**, and/or, **sent upstream** (to **SON/ML functions**) in case that a **control architecture** is **present**

4. **Interference Mitigation**

   Afterwards, the **RIM-reporting NR node** can:

   - **Avoid scheduling UEs** on **REs** that are **heavily used** by **neighbor NR nodes**.

- **Coordinate muting patterns**, in the case that **higher-level coordination** is available.
- Configure **coordinated multipoint** (**CoMP**, click 10.13) and/or **frequency-domain inter-cell interference coordination** (**ICIC**).

**RIM-RS** were **introduced** in **R16** to **enhance inter-cell interference coordination** (**ICIC**), especially for **uncoordinated deployments** like:

- **Distributed deployments** with **minimal Xn/N2/N3 interface usage**.
- **High-density urban networks**.
- **Private** or **enterprise** 5G **networks**.
- **Multi-operator** or **multi-vendor** environments.

For more detail on **RIM-RS**, refer to clause 7.4.1.6 of TS 38.331.

# 7. NR Uplink physical channels and signals

## 7.1 PRACH overview

The **physical random access channel** (**PRACH**) is used to **convey** the **random access** (**RA**) **preamble** (click 7.3.1) for the **random access procedure** (click 9)

It is important to **highlight** that **PRACH** is **different** to the **other 5 physical binary channels** in the following **relevant** aspects:

- Its **symbols** are **PRACH-specific** and do **not** belong to the constellation of a given **digital modulation**. The **PRACH-specific symbols** belong to a **Zadoff-Chu** (**ZC**, click 11.20.4) **complex number sequence**. Conversely, the **payload symbols** of **every other physical binary channel** belong to the constellation of a given **digital modulation**.
- Its **symbol duration** can be **very different** to the **symbol durations** that are used in **every other physical binary channel**.
- It does **not** have **symbol intervals**, **each** containing **1 symbol** and its **cyclic prefix**, as it is the case of **every other physical binary channel**.
- It **only** has **1 cyclic prefix** for **each sequence** of **several consecutive repetitions** of the **same PRACH-specific symbol** that is **transmitted** in **each subcarrier** of the **RA preamble**. Conversely, in **every other 5 physical binary channel**, **each** and **every symbol** has its **own cyclic prefix**.
- It may use **two PRACH-specific subcarrier spacings** (**1.25 kHz** and **5 kHz**), in **addition** to the **7 numerology subcarrier spacings** that are used in **every other physical channel**.
- Its **subcarriers** are **not structured** in terms of **resource blocks**. Conversely, **every other physical binary channel** has its **subcarriers structured** in **resource blocks**.
- Its **subcarriers** are **not necessarily aligned** to the ones of the **CRB range** of the **same subcarrier spacing**. Conversely, **every other physical binary channel** has its **subcarriers aligned** to the ones of the **CRB range** of the **same subcarrier spacing**.
- It does **not** have **Demodulation Reference Signals** (**DM-RS**), because it does **not** convey a **modulated binary sequence**. Conversely, **every other physical binary channel** has **DM-RS** (with the **only exception** of `format0` of **PDCCH**).

**Each PRACH transmission** takes place in **1** of the **transmission occasions determined** by the **PRACH scheduling** (click 7.2).

**Each PRACH transmission** contains **1 random access preamble** (click 7.3.1).

**Each RA preamble** is the result of a **specific processing** (click 7.3) for its **transmission** through **PRACH**.

**PRACH** is **configured** (click 7.4) with several **parameters** and **data types**.

Among other **techniques** to **improve reliability**, **PRACH** uses **Zadoff-Chu complex number sequences** (click 11.20.4) and **symbol repetition**. For more detail, click 7.3.2 and 7.3.4.

The **PRACH description** in this book covers almost nothing about **PRACH** in **shared spectrum**.

Because of **half-duplex UEs** (click 2.3.4.5) there is a **fixed PRACH timing advance** (click 2.3.6.1):

- In **FR1** it is "25,600 $T_c$" or **13 µs**, which is **10 µs** for the **NR node Rx to Tx switch** (click 2.3.4.3) plus **3 µs** for **BTS** to **BTS** synchronization.
- In **FR2** it is "13,792 $T_c$" or **7 µs**, which is **3 µs** for the **NR node Rx to Tx switch** (click 2.3.4.3) plus **3 µs** for **BTS** to **BTS** synchronization, plus **1 µs** extra margin.

For more detail on **PRACH**, refer to:

- Clause 6.3.3 (**Phys.** and **modulation**) of TS 38.211.
- Clause 8.1 (**RA preamble**) of TS 38.213.
- Clause 5.3.4 of TS 38.300.

## 7.2 PRACH scheduling

The **frequency resources** and **time resources** of **PRACH** are used for the **transmission** of the **RA preamble** (click 7.3.1). The **RA preamble** may be transmitted as **Msg1** (click 9.3.2) or within **MsgA** (click 9.3.6) of a **random access procedure** (click 9).

**PRACH** is **different** to **every other physical binary channel** in several **relevant** aspects (click 7.1).

**PRACH Tx scheduling**

It is **semi-persistent scheduling**. The **UE** is configured by the **NR node** with a **configured grant** of **transmission occasions**.

**PRACH Rx scheduling**

The **NR node listens** in **PRACH** for a **reception** of a **RA preamble** in the **corresponding transmission occasions** configured to **each UE**.

For an **overall** description of **scheduling**, **transmission occasions** and **transmissions**, click 4.11.4.

### 7.2.1 PRACH frequency-domain resources

The **PRACH frequency-domain resources** define the **RA preamble transmission occasions** in the **frequency domain**.

The **UE** has "$N_{fTx}^{PRACH}$" **RA preamble frequency transmission occasions** in the **same symbol intervals**, but in **different frequency locations**. **Each** of these "$NF_{fTx}^{PRACH}$" **frequency transmission occasions** is **located within** a **range** of **contiguous Common Resource Blocks** (**CRB$_\mu$**) of **PUSCH** in the **active bandwidth part** (**BWP**), which is called "**PRACH RB$_\mu$ set**". The **quantity** "$N_{RB}^{RA}$" of **CRB$_\mu$** in **every** such **PRACH RB$_\mu$ set** is **fixed** for **every allowed combination** of **RA preamble length**, **PRACH subcarrier spacing** and **active BWP PUSCH subcarrier spacing**. This **quantity** "$N_{RB}^{RA}$" of **CRBs** is obtained from table 6.3.3.2-1 of TS 38.211 (see further below).

The **numerology** and the **CRB index** of **PUSCH** of the **active BWP** are denoted "**µ**", and "**CRB$_\mu$ index**" (respectively).

Within **each** of the said "$NF_{fTx}^{PRACH}$" **PRACH RB$_\mu$ sets**, the **lowest frequency subcarrier** of its enclosed **RA preamble** is **centered** a **constant quantity** "$\bar{s}$" of **PRACH subcarrier spacings** above the **center** of the **lowest frequency subcarrier** of the **PRACH RB$_\mu$ set**. Same as the case of "$N_{RB}^{RA}$" indicated above, the **quantity** "$\bar{k}$" of **PRACH subcarrier spacings** is **fixed** for **every allowed combination** of **RA preamble length**, **PRACH subcarrier spacing** and **active BWP PUSCH subcarrier spacing**. Same as "$N_{RB}^{RA}$", this **quantity** "$\bar{k}$" of **PRACH subcarrier spacings** is also obtained from table 6.3.3.2-1 of TS 38.211.

All the above is formally defined in 7.2.2.3.

The **allowed combinations** of **PRACH sequence length** "$L_{RA}$", **PRACH subcarrier spacing** "$\Delta f_{RA}$" and **active BWP PUSCH subcarrier spacing** "$\Delta f$", plus the **corresponding values** of "$N_{RB}^{RA}$" and "$\bar{k}$", are obtained from table 6.3.3.2-1 of TS 38.211:

| $L_{RA}$ [PSym] | $\Delta f_{RA}$ (PRACH SCS) [kHz] | $\Delta f$ (active BWP PUSCH SCS) [kHz] | $N_{RB}^{RA}$ [active BWP PUSCH PRBs] | $\bar{k}$ [$\Delta f_{RA}$] |
|---|---|---|---|---|
| **839** | **1.25** | **15** | 6 | ***7*** |
| **839** | **1.25** | **30** | 3 | ***1*** |
| **839** | **1.25** | **60** | 2 | ***133*** |
| **839** | **5** | **15** | 24 | ***12*** |
| **839** | **5** | **30** | 12 | ***10*** |
| **839** | **5** | **60** | 6 | ***7*** |
| **139** | **15** | **15** | 12 | 2 |
| **139** | **15** | **30** | 6 | 2 |
| **139** | **15** | **60** | 3 | 2 |
| **139** | **30** | **15** | 24 | 2 |
| **139** | **30** | **30** | 12 | 2 |
| **139** | **30** | **60** | 6 | 2 |
| **139** | **60** | **60** | 12 | 2 |
| **139** | **60** | **120** | 6 | 2 |
| **139** | **120** | **60** | 24 | 2 |
| **139** | **120** | **120** | 12 | 2 |
| **139** | **120** | **480** | 3 | ***1*** |
| **139** | **120** | **960** | 2 | ***23*** |
| **139** | **480** | **120** | 48 | 2 |
| **139** | **480** | **480** | 12 | 2 |
| **139** | **480** | **960** | 6 | 2 |
| **139** | **960** | **120** | 96 | 2 |
| **139** | **960** | **480** | 24 | 2 |
| **139** | **960** | **960** | 12 | 2 |
| **571** | **30** | **15** | 96 | 2 |
| **571** | **30** | **30** | 48 | 2 |
| **571** | **30** | **60** | 24 | 2 |
| **571** | **120** | **120** | 48 | 2 |
| **571** | **120** | **480** | 12 | ***1*** |
| **571** | **120** | **960** | 7 | ***47*** |
| **571** | **480** | **120** | 192 | 2 |
| **571** | **480** | **480** | 48 | 2 |
| **571** | **480** | **960** | 24 | 2 |
| **1,151** | **15** | **15** | 96 | 1 |
| **1,151** | **15** | **30** | 48 | 1 |
| **1,151** | **15** | **60** | 24 | 1 |
| **1,151** | **120** | **120** | 97 | 6 |

| $L_{RA}$ [PSym] | $\Delta f_{RA}$ (PRACH SCS) [kHz] | $\Delta f$ (active BWP PUSCH SCS) [kHz] | $N_{RB}^{RA}$ [active BWP PUSCH PRBs] | $\bar{k}$ [$\Delta f_{RA}$] |
|---|---|---|---|---|
| **1,151** | **120** | **480** | 25 | 23 |
| **1,151** | **120** | **960** | 13 | 45 |

Some **relevant remarks** on this table are:

- The **first three columns** indicate the **allowed combinations** of **PRACH sequence length** "$L_{RA}$", **PRACH subcarrier spacing** "$\Delta f_{RA}$" and **active BWP subcarrier spacing** "$\Delta f$".
- The **last two columns** indicate the corresponding value of "$N_{RB}^{RA}$" and "$\bar{k}$" for **each allowed combination** of **PRACH sequence length** "$L_{RA}$", **PRACH subcarrier spacing** "$\Delta f_{RA}$" and **active BWP subcarrier spacing** "$\Delta f$".
- The **units** of the **frequency transmission occasion size** "$N_{RB}^{RA}$" are **RB$_\mu$** (i.e., "$\Delta f$"), this is, **RBs** with the **active BWP PUSCH subcarrier spacing**.
- The **units** of the **RA preamble offset** "$\bar{k}$" are **PRACH subcarrier spacings** (i.e., "$\Delta f_{RA}$").
- The **frequency transmission occasion size** "$N_{RB}^{RA}$" is the **minimum quantity** of **PUSCH PRBs** that can fit the **RA preamble** "$L_{RA} \cdot \Delta f_{RA}$" plus **twice** its offset "$\bar{k}$".
- The **offset** "$\bar{k}$" (approximately) **centers** the **RA preamble** within its corresponding **PRACH CRB$_\mu$ range** (that has "$N_{RB}^{RA}$" **RB$_\mu$**) with some exceptions. The exceptions are **all** the **long sequence RA Formats** (click 7.3.2), plus **4 short sequence RA Formas**. These exceptions of "$\bar{k}$" have been identified with ***bold italics***. The **RA preamble cannot** be **exactly centered** within its corresponding **PRACH CRB$_\mu$ range** because the **RA preamble** is **always** an **odd** (**prime**) **quantity** of **PRACH subcarriers**, while the **allocated active BWP subcarriers** are **always** an **even** (multiple of 12) **quantity**.

**Location of the "$NF_{fTx}^{PRACH}$" PRACH RB$_\mu$ sets**

The "$NF_{fTx}^{PRACH}$" **PRACH RB$_\mu$ sets** indicated at the beginning of this section are indexed from **0** to "$NF_{fTx}^{PRACH}$-1", ordered from **lowest** to **highest** frequency. "$NF_{fTx}^{PRACH}$" can take the values **1**, **2**, **4** or **8**. The value of "$NF_{fTx}^{PRACH}$" is obtained as indicated further below.

The "$NF_{fTx}^{PRACH}$" **lowest CRB$_\mu$ indexes** of the "$NF_{fTx}^{PRACH}$" **PRACH RB$_\mu$ sets** are denoted "$k_1$/12". The "$NF_{fTx}^{PRACH}$" values of "$k_1$/12" are determined by the equation:

$$\frac{k_1}{12} = N_{\text{BWP},i}^{\text{start}} + n_{\text{RA}}^{\text{start}} - 2^{\mu_0-\mu}\left(N_{\text{grid}}^{\text{start},\mu_0} + N_{\text{grid}}^{\text{size},\mu_0}/2\right) + \begin{cases} n_{RA}\, N_{\text{RB}}^{\text{RA}} & \begin{cases} \text{if } L_{\text{RA}} \in \{839, 139\} \\ \text{or if } L_{\text{RA}} \in \{571, 1151\} \text{ in FR2-2} \end{cases} \\ N_{\text{RB,UL},n_0+n_{RA}}^{\text{start},\mu} - N_{\text{RB,UL},n_0}^{\text{start},\mu} & \text{if } L_{\text{RA}} \in \{571, 1151\} \text{ in FR1} \end{cases}$$

Notice that this equation is the **same** as the equation in clause 5.3.2 of TS 38.211, after replacing "$N_{sc}^{RB}$" by 12, replacing "$k_0^{\mu}$" by its expression, and compacting the result. The variable "$n_{RA}$" in this equation takes **all** the values from **0** to "$NF_{fTx}^{PRACH}$-1".

**Both** "$k_1$" and "$k_1/12$" identify the **location** of the **center frequency** of the **lowest frequency subcarrier$_{\mu}$** of **each PRACH RB$_{\mu}$ set**. The **only difference** between "$k_1$" and "$k_1/12$" is their **units**. The **equation** for "$k_1$" identifies the **location** as the **center frequency** of the **subcarrier$_{\mu}$** that has the **common subcarrier index (CSC$_{\mu}$)** "$k_1$". However, the **equation** for "$k_1/12$" identifies the **location** as the **center frequency** of the **lowest frequency subcarrier** of the **CRB$_{\mu}$** that has the **CRB$_{\mu}$ index** "$k_1/12$".

The **first part** of the "$k_1/12$" equation:

$$N_{\mathrm{BWP},i}^{\mathrm{start}} + n_{\mathrm{RA}}^{\mathrm{start}} - 2^{\mu_0-\mu}\left(N_{\mathrm{grid}}^{\mathrm{start},\mu_0} + N_{\mathrm{grid}}^{\mathrm{size},\mu_0}/2\right)$$

is the **CRB$_{\mu}$ index** of the **base value** of "$k_1/12$", which independent of "$n_{RA}$".

The **second part** of the "$k_1/12$" equation:

$$+\begin{cases} n_{RA}\, N_{\mathrm{RB}}^{\mathrm{RA}} & \begin{cases} \text{if } L_{\mathrm{RA}} \in \{839, 139\} \\ \text{or if } L_{\mathrm{RA}} \in \{571, 1151\} \text{ in FR2-2} \end{cases} \\ N_{\mathrm{RB,UL},n_0+n_{RA}}^{\mathrm{start},\mu} - N_{\mathrm{RB,UL},n_0}^{\mathrm{start},\mu} & \text{if } L_{\mathrm{RA}} \in \{571, 1151\} \text{ in FR1} \end{cases}$$

results in the "$NF_{fTx}^{PRACH}$" **offsets** (expressed as a quantity of **CRB$_{\mu}$**) in respect the said **base value**, which correspond to "$n_{RA}$" equal **0** to "$NF_{fTx}^{PRACH}$-1". These "$NF_{fTx}^{PRACH}$" **offsets** produce the "$NF_{fTx}^{PRACH}$" values of "$k_1/12$".

Returning to the **first part** of the "$k_1/12$" equation:

$$N_{\mathrm{BWP},i}^{\mathrm{start}} + n_{\mathrm{RA}}^{\mathrm{start}} - 2^{\mu_0-\mu}\left(N_{\mathrm{grid}}^{\mathrm{start},\mu_0} + N_{\mathrm{grid}}^{\mathrm{size},\mu_0}/2\right)$$

it is composed of **three terms**:

- The **first term** "$N_{BWP,i}^{start}$" is the **lowest CRB$_{\mu}$ index**. This is also the **CRB$_{\mu}$ index** of the **physical resource block 0** (**PRB0**) of **PUSCH** in the **active BWP**.
- The **second term** "$n_{RA}^{start}$" is an **offset** (expressed as a **quantity** of **CRB$_{\mu}$**) above the **first term** which determines the **lowest CRB$_{\mu}$ index** value produced by "$k_1/12$" (**possibly** adjusted by the **third term**). This **second term** may also be interpreted as the lowest **PRB index** value produced by "$k_1/12$" (**possibly** adjusted by the **third term**).
- The **third term** $-2^{\mu_0-\mu}\left(N_{\mathrm{grid}}^{\mathrm{start},\mu_0} + N_{\mathrm{grid}}^{\mathrm{size},\mu_0}/2\right)$ is a **negative** adjustment on the result of the **two first terms**. The absolute value of this adjustment is the **CRB$_{\mu 0}$ index** value of the **central CRB$_{\mu 0}$** of the **CRB$_{\mu 0}$ grid** (all of them with **numerology** "$\mu_0$"), expressed as a **CRB$_{\mu}$ index** value. Notice that "$2^{\mu 0-\mu}$" is a scale factor to convert a **CRB$_{\mu 0}$ index** value into a **CRB$_{\mu}$ index** value. The constant "$\mu_0$" is the **numerology** of the **largest subcarrier spacing** in `scs-SpecificCarrierList` from `FrequencyInfoUL-SIB` (click 2.2.6.7). Consequently, "$\mu_0 \geq \mu$".

Returning to the **second part** of the "$k_1/12$" **equation**:

$$+\begin{cases} n_{RA}\, N_{RB}^{RA} & \begin{cases} \text{if } L_{RA} \in \{839, 139\} \\ \text{or if } L_{RA} \in \{571, 1151\} \text{ in FR2-2} \end{cases} \\ N_{RB,UL,n_0+n_{RA}}^{start,\mu} - N_{RB,UL,n_0}^{start,\mu} & \text{if } L_{RA} \in \{571, 1151\} \text{ in FR1} \end{cases}$$

it has the following **three cases**, that are determined by the **band frequency range** and by the **RA preamble sequence length** "$L_{RA}$":

- For "$L_{RA}$" equal **839** (**long** preamble **Formats**) or **139**, and also

  for **FR2-2** with "$L_{RA}$" equal **571** or **1,151**

  The **offset** of **each** of the "$NF_{fTx}^{PRACH}$-1" **RA preamble frequency occasions**, in respect to the **base value**, is "$n_{RA}\, N_{RB}^{RA}$" (expressed as a **quantity** of **RB$_\mu$**). Consequently, the "$NF_{fTx}^{PRACH}$" **PRACH RB$_\mu$ sets** are **contiguous** in frequency.

- For **FR1** with "$L_{RA}$" equal **571** or **1,151**

  The **offset** of **each** of the **RA preamble frequency occasions**, in respect to the **base value**, is "$N_{RB,UL,n0+nRA}^{start,\mu}$ - $N_{RB,UL,n0}^{start,\mu}$". The value of "$N_{RB,UL,n0}^{start,\mu}$" is the **same** as the **first part** of the "$k_1/12$" **equation**. The value of "$n_0$" is 0. Consequently, in this case, the result of "$k_1/12$" is "$N_{RB,UL,nRA}^{start,\mu}$". Its "$n_{RA}$" values may have the following two cases:

  - **Channel bandwidth** is **neither 40**, **60**, **80 nor 100 MHz**

    In this case, there are **no intra-cell guardbands**. Refer to table 5.3.3 of TS 38.101-**1**.

    Consequently, there is **only 1 PRACH RB$_\mu$ set** ("$NF_{fTx}^{PRACH}$" is **1**). Refer to sentence "*contains no intra-cell guard bands, the number of RB sets for the carrier is $N_{RB\text{-}set,x}$=1.*" from clause 7 of TS 38.214.

    Consequently, the **only PRACH RB$_\mu$ set** occupies **all** the **RBs** of the **resource grid$_\mu$**. Refer to the equations for "**RB$_{s,x}^{start,\mu}$**", and "**RB$_{s,x}^{end,\mu}$**" from clause 7 of TS 38.214, for ***s*=0** (i.e., for only **1 PRACH RB$_\mu$ set**).

  - **Channel bandwidth** is **either 40**, **60**, **80 or 100 MHz**

    The **number**, **location** and **bandwidth** of the **PRACH RB$_\mu$ sets** are stated in table 5.3.3 of TS 38.101-**1**. The said table (presented in a different way) is:

| | | Channel Bandwidth [MHz] | | | |
|---|---|---|---|---|---|
| **SCS [kHz]** | **Element** | **40** | **60** | **80** | **100** |
| **15** | $N_{RB}^{RA}$ | 105 | N/A | N/A | N/A |
| | $NF_{fTx}^{PRACH}$ | 2 | | | |
| | $k_1/12 - N_{grid,x}^{start,\mu}$ | 0 and 111 | | | |
| **30** | $N_{RB}^{RA}$ | 50 | | | 50, 50, 49, 50, 50 |
| | $NF_{fTx}^{PRACH}$ | 2 | 3 | 4 | 5 |
| | $k_1/12 - N_{grid,x}^{start,\mu}$ | 0 and 56 | 0, 56 and 112 | 0, 56, 112 and 168 | 0, 56, 112, 167 and 223 |

| SCS [kHz] | Element | Channel Bandwidth [MHz] | | | |
|---|---|---|---|---|---|
| | | 40 | 60 | 80 | 100 |
| 60 | $N_{RB}^{RA}$ | 23 | | | |
| | $NF_{fTx}^{PRACH}$ | 2 | 3 | 4 | 5 |
| | $k_1/12 - N_{grid,x}^{start,\mu}$ | 0 and 28 | 0, 28 and 56 | 0, 28, 56 and 84 | 0, 28, 56, 84 and 112 |

Some **relevant remarks** on this table are:

- The units of "$N_{RB}^{RA}$" are **$RB_\mu$**.
- The units of "$NF_{fTx}^{PRACH}$" are **quantity** of **RA preamble frequency occasions**.
- The units of "$k_1/12 - N_{grid,x}^{start,\mu}$" are **$RB_\mu$ index**.

Notice that in **this case** (i.e., **FR1** with "$L_{RA}$" equal **571** or **1,151**), the value of "$N_{RB}^{RA}$" indicated in table 6.3.3.2-1 of TS 38.211 (see above) is ignored.

For more detail on **this case**, refer to clauses 5.3.2 of TS 38.211, which refers to clause 7 of TS 38.214, which refers to clause 5.3.3 of TS 38.101-**1**.

- For **FR2-1** with "$L_{RA}$" equal **571** or **1,151**

  This case is **not defined**, and therefore **no value** is **added** to the **first part** of the "$k_1/12$" **equation**. Consequently, this case consists of **only 1 RA preamble frequency occasion**, that is **located** in the **base value**.

From the above explanation we may conclude that:

- The **RA preamble** is **not structured** in terms of **RBs**. Therefore, it **cannot** be aligned to the **CRBs** of any **CRB range**.
- The **PRACH subcarriers** with **PRACH-specific subcarrier spacing** are **not aligned** with the **subcarriers** of a **CRB range**. Actually, there are **no CRB ranges** for the **PRACH-specific subcarrier spacings**. The **lowest frequency PRACH subcarrier** from **each** of the "$NF_{fTx}^{PRACH}$" **RA preamble frequency occasions** is **centered** in a **specific location** "$k_1/12+\bar{k}$" of the **active bandwidth part**.
- The **PRACH subcarriers** with **numerology subcarrier spacing** are **aligned** with the **subcarriers** of the corresponding **CRB range only** in case that the **lowest $CRB_\mu$ index** of **all** the "$NF_{fTx}^{PRACH}$" **RA preamble frequency occasions** is a **multiple** of the **ratio** between the **PRACH subcarrier spacing** and the **PUSCH subcarrier spacing**.

The **UE** obtains the **values** required for the "$k_1$" **equation** in the following way:

- "$n_{RA}^{start}$"

  The way the **UE** gets its value depends on whether **either** `msgA-RO-FrequencyStart-r16` from `RACH-ConfigGenericTwoStepRA-r16` (click 9.5.9) is configured, **and**, a **type-2 random access procedure** is initiated as described in clause 8.1 of TS 38.213:

  - If the said condition holds, "$n_{RA}^{start}$" is the value of the configured parameter (among the two said ones).

- Otherwise, "$n_{RA}^{start}$" is the value of `msg1-FrequencyStart` from `RACH-ConfigGeneric` (click 9.5.4), as described in clause 8.1 of TS 38.213.

- "$NF_{fTx}^{PRACH}$"

  For **FR1** with "$L_{RA}$" equal **571** or **1,151**, the **UE** knows "$NF_{fTx}^{PRACH}$" from table 5.3.3 of TS 38.101-**1** (see the resulting table above).

  Otherwise, the **UE** gets "$NF_{fTx}^{PRACH}$" from either:

  - `msg1-FDM` from `RACH-ConfigGeneric` (click 9.5.4), within `SIB1`.
  - `msgA-RO-FDM-r16` from `RACH-ConfigGenericTwoStepRA-r16` (click 9.5.9), if **present**.

For more detail on **RA preamble frequency occasions**, refer to clauses 5.3.2 and 6.3.3 of TS 38.211.

Source web page:

- https://www.nrexplained.com/ra_msg1

## 7.2.2 PRACH time-domain resources

The **PRACH time -domain resources** define the **RA preamble transmission occasions** in the **time domain**.

This section contains an **overview** (click 7.2.2.1), an **example** (click 7.2.2.2) and a **formal definition** (click 7.2.2.3).

For more detail on **PRACH time scheduling**, refer to clauses 5.3.2 and 6.3.3.2 of TS 38.211.

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_RACH.html
- https://www.prodevelopertutorial.com/5g-nr-rach-process-msg1-generation-prach-preamble/
- https://www.rfwireless-world.com/5G/5G-NR-PRACH.html

### 7.2.2.1 Overview of PRACH time-domain resources

The **RA preamble** can be transmitted in **1 frame** every "$x$" **frames**. The specific **system frame numbers** "$n_f$" where the **RA preamble** may be transmitted are the ones that fulfil:

$$n_f \bmod x = y$$

The values of "$x$" and "$y$" are found in columns "$x$" and "$y$" after entering the **8-bit PRACH configuration index** into the corresponding table from 6.3.3.2-2 to 6.3.3.2-4 of TS 38.211. The **UE** obtains the **PRACH configuration index** (click 7.4) out of `prach-ConfigurationIndex` from `RACH-ConfigGeneric` (click 9.5.4) within `SIB1` (click 5.3.4), or if **configured**, out of `msgA-PRACH-ConfigurationIndex-r16` from `RACH-ConfigGenericTwoStepRA-r16` (click 9.5.9).

Within **each frame**, the **RA preamble can** be transmitted within **one**, **or more**, **subframes** (i.e., **15 kHz slots**) or **60 kHz slots**. The **subframes** are used when the **PRACH subcarrier spacing** "$\Delta f_{RA}$" is **30 kHz or less**. The **60 kHz slots** are used when the **PRACH subcarrier spacing** "$\Delta f_{RA}$" is **60 kHz or more**. The **index(es)**, within the **frame**, corresponding to the **subframe(s)**, or **60kHz slot(s)**, where **PRACH can** be

transmitted are found in the **column** "*Subframe number*" or "*Slot number*" (respectively), after entering the **8-bit PRACH configuration index** into the corresponding **table 6.3.3.2-2** to **6.3.3.2-4**. When checking the values in the said tables, recall that a **frame** contains **10 subframes** (i.e., **10 15 kHz slots**) or **40 60 kHz slots**. A fragment of two of the said tables is copied further below.

Within **each corresponding subframe**, or **60 kHz slot**, the **RA preamble can** be transmitted in **1** or **2 PRACH slots**. The quantity of **PRACH slots** is indicated in the column "*Number of PRACH slots ...*", after entering the **8-bit PRACH configuration index** into the corresponding table from 6.3.3.2-2 to 6.3.3.2-4. For the **long** preamble **Formats** (**0** to **3**), it is always only **1 PRACH slot**.

Within **each corresponding subframe**, or **60 kHz slot**, the **1**, or **2**, **PRACH slots**(**s**) are **located** in the **PRACH slot index**(**es**) "$n_{slot}^{RA}$" according to the following table:

| PRACH subcarrier spacing $\Delta f_{RA}$ [kHz] | #*PRACH Slots* = 1 | #*PRACH Slots* = 2 |
|---|---|---|
| **1.25, 5, 15, 60** | $n_{slot}^{RA}=0$ | |
| **30, 120** | $n_{slot}^{RA}=1$ | $n_{slot}^{RA}\in\{0, 1\}$ |
| **480** | $n_{slot}^{RA}=7$ | $n_{slot}^{RA}\in\{3, 7\}$ |
| **960** | $n_{slot}^{RA}=15$ | $n_{slot}^{RA}\in\{7, 15\}$ |

For **each PRACH slot**, the **quantity** of **RA preamble time occasions** is indicated in the column "$N_t^{RA,slot}$", after entering the **8-bit PRACH configuration index** into the corresponding **table 6.3.3.2-2** to **6.3.3.2-4**. For the **long** preamble **Formats** (**0** to **3**), there is **only 1 RA preamble time occasion**. This is why their entries in the tables always contain "**-**" in the column for **transmission occasions**, and "**0**" in the column "$N_{dur}^{RA}$" that indicates the **RA preamble total duration**.

For **each PRACH slot**, the **first RA preamble time occasion** is located in **PRACH slot symbol index** $l_0$. The value of $l_0$ is indicated in the column "$l_0$ *Starting symbol*", after entering the **8-bit PRACH configuration index** into the corresponding table from 6.3.3.2-2 to 6.3.3.2-4. For the **long** preamble **Formats** (**0** to **3**) and for **all Formats** in **paired spectrum** bands, the value of $l_0$ is always **0**.

In case there are **several RA preamble time occasions**, they are spaced the **total duration** of the **RA preamble** (including its **guard time**). The said **total duration** is denoted "$N_{dur}^{RA}$" and may be found in the column "$N_{dur}^{RA}$", after entering the **8-bit PRACH configuration index** into the corresponding table from 6.3.3.2-2 to 6.3.3.2-4. The value of "$N_{dur}^{RA}$" is expressed as an **integer quantity** of **PRACH subcarrier spacing symbol intervals**.

Notice that **quantity** of **RA preamble time occasions** multiplied by the **total RA preamble duration** never exceeds the separation between any **two consecutive PRACH slots**, and therefore, the **RA preambles** that could be **transmitted according** to the **RA preamble time occasions** do **not overlap** in **time**

For the **time location** of the **RA preamble**, a timing advance between **Uplink** and **Downlink** transmission "$N_{TA}$=**0**" (click 2.3.6.1) shall be assumed. Notice that the **UE** has not transmitted anything yet, and therefore there is **no reference** timing information to set a proper value for "$N_{TA}$". This implies that the **RA preamble** will **not** be **synchronized** with the **Uplink** frame structure.

A **random access occasion** (**RO**) is a **grouping** of **PRACH occasions** that is **periodically repeated** and **scheduled** in time.

The following is a **fragment** from table 6.3.3.2-2 of TS 38.211, corresponding to **FR1 paired spectrum** (**FDD** or **SUL**) bands:

| PRACH Config. Index | Preamble format | $n_f$ mod $x$ = y | | Subframe quantity | $l_0$ Starting symbol | Quantity of PRACH slots within a subframe | $N_t^{RA,slot}$, quantity of PRACH occasions within a PRACH slot | $N_{dur}^{RA}$, PRACH duration |
|---|---|---|---|---|---|---|---|---|
| | | $x$ | $y$ | | | | | |
| **0** | 0 | 16 | 1 | 1 | 0 | - | - | 0 |
| **1** | 0 | 16 | 1 | 4 | 0 | - | - | 0 |
| **…** | … | … | … | … | 0 | … | … | … |
| **25** | 0 | 1 | 0 | 0,2,4,6,8 | 0 | - | - | 0 |
| **26** | 0 | 1 | 0 | 1,3,5,7,9 | 0 | - | - | 0 |
| **27** | 0 | 1 | 0 | 0,1,2,3,4,5,6,7,8,9 | 0 | - | - | 0 |
| **28** | 1 | 16 | 1 | 1 | 0 | - | - | 0 |
| **…** | … | … | … | … | 0 | … | … | … |
| **50** | 1 | 1 | 0 | 1,4,7 | 0 | - | - | 0 |
| **51** | 1 | 1 | 0 | 2,5,8 | 0 | - | - | 0 |
| **52** | 1 | 1 | 0 | 3,6,9 | 0 | - | - | 0 |
| **53** | 2 | 16 | 1 | 1 | 0 | - | - | 0 |
| **…** | … | … | … | … | 0 | … | … | … |
| **59** | 2 | 1 | 0 | 5 | 0 | - | - | 0 |
| **60** | 3 | 16 | 1 | 1 | 0 | - | - | 0 |
| **…** | … | … | … | … | 0 | … | … | … |
| **85** | 3 | 1 | 0 | 1,3,5,7,9 | 0 | - | - | 0 |
| **86** | 3 | 1 | 0 | 0,1,2,3,4,5,6,7,8,9 | 0 | - | - | 0 |
| **87** | A1 | 16 | 0 | 4,9 | 0 | 1 | 6 | 2 |
| **…** | … | … | … | … | 0 | … | … | … |
| **106** | A1 | 1 | 0 | 0,1,2,3,4,5,6,7,8,9 | 0 | 2 | 6 | 2 |
| **107** | A1 | 1 | 0 | 1,3,5,7,9 | 0 | 2 | 6 | 2 |
| **108** | A1/B1 | 2 | 0 | 4,9 | 0 | 1 | 7 | 2 |
| **…** | … | … | … | … | 0 | … | … | … |
| **116** | A1/B1 | 1 | 0 | 0,2,4,6,8 | 0 | 2 | 7 | 2 |
| **117** | A2 | 16 | 1 | 2,6,9 | 0 | 1 | 3 | 4 |
| **…** | … | … | … | … | 0 | … | … | … |
| **135** | A2 | 1 | 0 | 0,1,2,3,4,5,6,7,8,9 | 0 | 2 | 3 | 4 |
| **136** | A2 | 1 | 0 | 1,3,5,7,9 | 0 | 2 | 3 | 4 |
| **137** | A2/B2 | 2 | 1 | 2,6,9 | 0 | 1 | 3 | 4 |
| **…** | … | … | … | … | 0 | … | … | … |
| **145** | A2/B2 | 1 | 0 | 0,2,4,6,8 | 0 | 2 | 3 | 4 |
| **146** | A2/B2 | 1 | 0 | 0,1,2,3,4,5,6,7,8,9 | 0 | 2 | 3 | 4 |
| **147** | A3 | 16 | 1 | 4,9 | 0 | 1 | 2 | 6 |
| **…** | … | … | … | … | 0 | … | … | … |
| **165** | A3 | 1 | 0 | 0,1,2,3,4,5,6,7,8,9 | 0 | 2 | 2 | 6 |
| **166** | A3 | 1 | 0 | 1,3,5,7,9 | 0 | 2 | 2 | 6 |
| **167** | A3/B3 | 2 | 1 | 2,6,9 | 0 | 2 | 2 | 6 |
| **…** | … | … | … | … | 0 | … | … | … |
| **175** | A3/B3 | 1 | 0 | 0,2,4,6,8 | 0 | 2 | 2 | 6 |
| **176** | A3/B3 | 1 | 0 | 0,1,2,3,4,5,6,7,8,9 | 0 | 2 | 2 | 6 |
| **177** | B1 | 16 | 0 | 4,9 | 0 | 1 | 7 | 2 |
| **…** | … | … | … | … | 0 | … | … | … |
| **196** | B1 | 1 | 0 | 0,1,2,3,4,5,6,7,8,9 | 0 | 2 | 7 | 2 |
| **197** | B1 | 1 | 0 | 1,3,5,7,9 | 0 | 2 | 7 | 2 |
| **198** | B4 | 16 | 0 | 4,9 | 0 | 2 | 1 | 12 |
| **…** | … | … | … | … | 0 | … | … | … |
| **217** | B4 | 1 | 0 | 0,1,2,3,4,5,6,7,8,9 | 0 | 2 | 1 | 12 |
| **218** | B4 | 1 | 0 | 1,3,5,7,9 | 0 | 2 | 1 | 12 |
| **219** | C0 | 8 | 1 | 4 | 0 | 2 | 7 | 2 |
| **…** | … | … | … | … | 0 | … | … | … |
| **234** | C0 | 1 | 0 | 0,1,2,3,4,5,6,7,8,9 | 0 | 2 | 7 | 2 |

| PRACH Config. Index | Preamble format | $n_f$ mod $x$ = y | | Subframe quantity | $l_0$ Starting symbol | Quantity of PRACH slots within a subframe | $N_t^{RA,slot}$, quantity of PRACH occasions within a PRACH slot | $N_{dur}^{RA}$, PRACH duration |
|---|---|---|---|---|---|---|---|---|
| | | $x$ | $y$ | | | | | |
| **235** | C0 | 1 | 0 | 1,3,5,7,9 | 0 | 2 | 7 | 2 |
| **236** | C2 | 16 | 1 | 4,9 | 0 | 1 | 2 | 6 |
| **…** | … | … | … | … | 0 | … | … | … |
| **254** | C2 | 1 | 0 | 0,1,2,3,4,5,6,7,8,9 | 0 | 2 | 2 | 6 |
| **255** | C2 | 1 | 0 | 1,3,5,7,9 | 0 | 2 | 2 | 6 |

The following is a **fragment** from **table 6.3.3.2-4** of TS 38.211, corresponding to **FR2** bands:

| PRACH Config. Index | Preamble format | $n_f$ mod $x$ = $y$ | | Slot number | $l_0$ Starting symbol | Quantity of PRACH slots within a 60 kHz slot | $N_t^{RA,slot}$, quantity of PRACH occasions within a PRACH slot | $N_{dur}^{RA}$, PRACH duration |
|---|---|---|---|---|---|---|---|---|
| | | $x$ | $y$ | | | | | |
| **0** | A1 | 16 | 1 | 4,9,14,19,24,29,34,39 | 0 | 2 | 6 | 2 |
| **…** | … | … | … | … | … | … | … | … |
| **28** | A1 | 1 | 0 | 0,1,2,…,39 | 7 | 1 | 3 | 2 |
| **29** | A2 | 16 | 1 | 4,9,14,19,24,29,34,39 | 0 | 2 | 3 | 4 |
| **…** | … | … | … | … | … | … | … | … |
| **58** | A2 | 1 | 0 | 0,1,2,…,39 | 5 | 1 | 2 | 4 |
| **59** | A3 | 16 | 1 | 4,9,14,19,24,29,34,39 | 0 | 2 | 2 | 6 |
| **…** | … | … | … | … | … | … | … | … |
| **88** | A3 | 1 | 0 | 0,1,2,…,39 | 7 | 1 | 1 | 6 |
| **89** | B1 | 16 | 1 | 4,9,14,19,24,29,34,39 | 2 | 2 | 6 | 2 |
| **…** | … | … | … | … | … | … | … | … |
| **111** | B1 | 1 | 0 | 0,1,2,…,39 | 8 | 1 | 3 | 2 |
| **112** | B4 | 16 | 1,2 | 4,9,14,19,24,29,34,39 | 0 | 2 | 1 | 12 |
| **…** | … | … | … | … | … | … | … | … |
| **143** | B4 | 1 | 0 | 0, 1, 2,…, 39 | 2 | 1 | 1 | 12 |
| **144** | C0 | 16 | 1 | 4,9,14,19,24,29,34,39 | 0 | 2 | 7 | 2 |
| **…** | … | … | … | … | … | … | … | … |
| **172** | C0 | 1 | 0 | 0,1,2,…,39 | 8 | 1 | 3 | 2 |
| **173** | C2 | 16 | 1 | 4,9,14,19,24,29,34,39 | 0 | 2 | 2 | 6 |
| **…** | … | … | … | … | … | … | … | … |
| **201** | C2 | 1 | 0 | 0,1,2,…,39 | 7 | 1 | 1 | 6 |
| **202** | A1/B1 | 16 | 1 | 4,9,14,19,24,29,34,39 | 2 | 1 | 6 | 2 |
| **…** | … | … | … | … | … | … | … | … |
| **219** | A1/B1 | 1 | 0 | 1,3,5,7,…,37,39 | 2 | 1 | 6 | 2 |
| **220** | A2/B2 | 16 | 1 | 4,9,14,19,24,29,34,39 | 2 | 1 | 3 | 4 |
| **…** | … | … | … | … | … | … | … | … |
| **237** | A2/B2 | 1 | 0 | 1,3,5,7,…,37,39 | 2 | 1 | 3 | 4 |
| **238** | A3/B3 | 16 | 1 | 4,9,14,19,24,29,34,39 | 2 | 1 | 2 | 6 |
| **…** | … | … | … | … | … | … | … | … |
| **255** | A3/B3 | 1 | 0 | 1,3,5,7,…,37,39 | 2 | 1 | 2 | 6 |

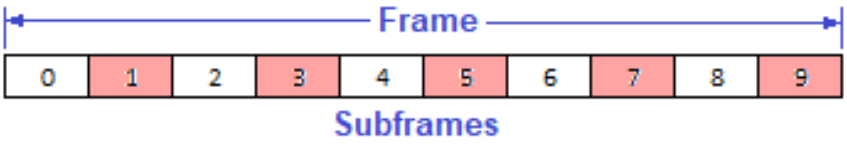
### 7.2.2.2 Example of PRACH time-domain resources

Let us do an example for a **FR1 TDD band**, with **PRACH subcarrier spacing** of **30 kHz** and `prach-ConfigurationIndex=131`. Since this is an **FR1 TDD band**, we need to refer to table 6.3.3.2-3. The entry corresponding to the index `131` is:

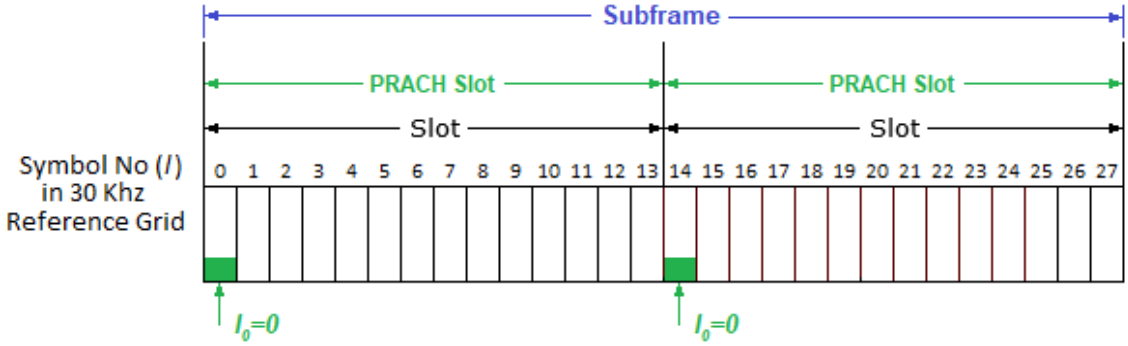
| PRACH Configuration Index | Preamble format | $n_{SFN} \bmod x = y$ | | Subframe number | Starting symbol | Number of PRACH slots within a subframe | $N_t^{RA,slot}$, number of time-domain PRACH occasions within a PRACH slot | $N_{dur}^{RA}$, PRACH duration |
|---|---|---|---|---|---|---|---|---|
| | | $x$ | $y$ | | | | | |
| 131 | A3 | 1 | 0 | 1, 3, 5, 7, 9 | 0 | 1 | 2 | 6 |

From columns "***x***" and "***y***", we see that the **RA preamble** may be sent in **any frame**, because all system frame numbers $n_{SFN}$ comply with $n_{SFN}$ ***mod*** 1 = 0.

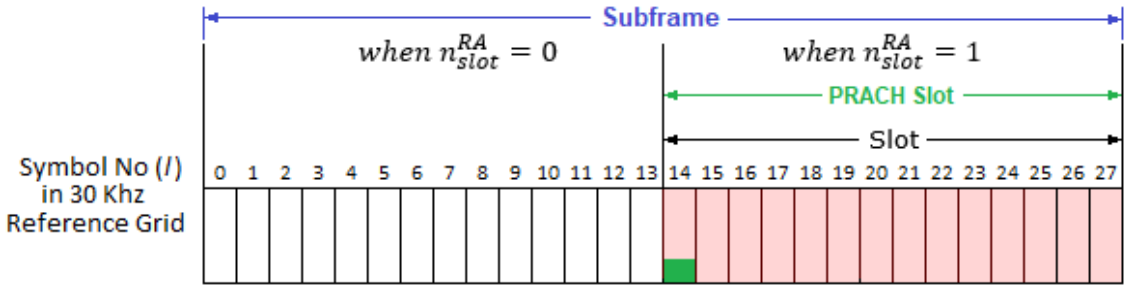
From column "*Subframe number*", we see that the **RA preamble** may be sent in **subframes** **1**, **3**, **5**, **7** and **9**.



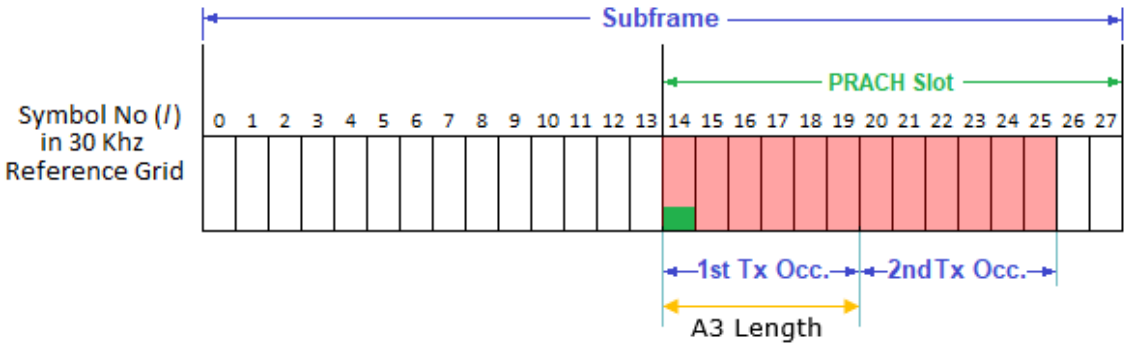

From column "*Starting symbol*" we that the symbol offset "$l_0$" is **0**. This means that offset of the **PRACH slot(s)** within each of these subframes, is **0** PRACH-subcarrier symbols, and therefore, in this case the **PRACH slots** are **aligned** with the "normal" **slots**.




From column "*Quantity of PRACH slots within a subframe*" we get value "**1**", which means that each subframe only has **1** **PRACH slot**. Inputting this value (**1**) and the PRACH subcarrier spacing (30 kHz) in the table for $n_{slot}^{RA}$ (see above), we get that $n_{slot}^{RA}$**=1**, meaning that the **PRACH slot** has index **1** in the subframe (with symbol offset **0**):

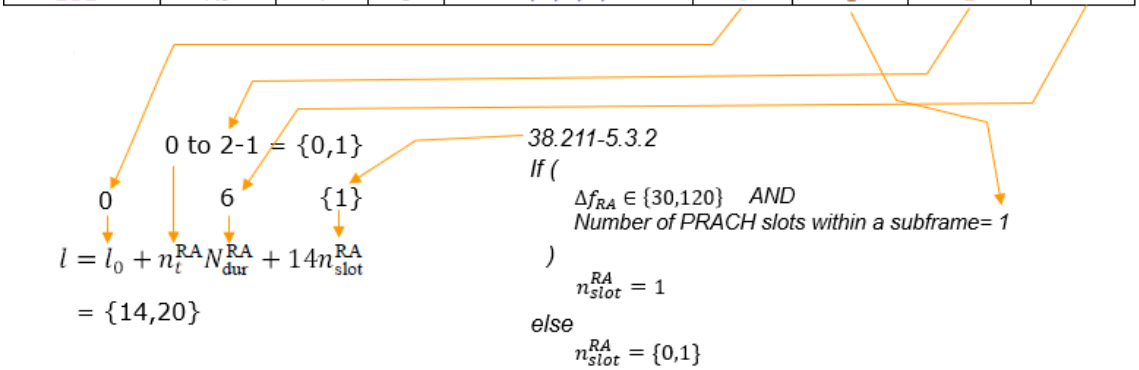


From column "*Quantity of time-domain PRACH occasions within a PRACH slot*" we get that there are **2** transmission occasions in each PRACH slot. From the column "*PRACH Duration*" we get that **each PRACH transmission occasion** lasts **6** PRACH-subcarrier **symbols**. Combining both, we get the following result, that would be repeated in subframes **1**, **3**, **5**, **7** and **9**:




The way to see this following the equations is as follows:

| PRACH Configuration Index | Preamble format | $n_{SFN} \bmod x = y$ | | Subframe number | Starting symbol | Number of PRACH slots within a subframe | $N_t^{RA,slot}$, number of time-domain PRACH occasions within a PRACH slot | $N_{dur}^{RA}$, PRACH duration |
|---|---|---|---|---|---|---|---|---|
| | | $x$ | $y$ | | | | | |
| 131 | A3 | 1 | 0 | 1, 3, 5, 7, 9 | 0 | 1 | 2 | 6 |




### 7.2.2.3 Formal definition of PRACH time-domain resources

The **text description** performed in the "**Overview of PRACH time scheduling**" (click 7.2.2.1) is now presented in the form of **equations** (as it is done in clause 5.3.2 of TS 38.211).

The **system frame numbers** "$n_f$" where the **RA preamble can** be transmitted are the ones that fulfil:

$$n_f \bmod x = y$$

where "$x$" and "$y$" are provided by columns "$x$" and "$y$" from the corresponding table 6.3.3.2-2 to 6.3.3.2-4 of TS 38.211.

The **symbol interval index** "$l$" (according to the **PRACH subcarrier spacing**) within the **subframe**, or **60 kHz slot**, where the **RA preamble** transmission may start is given by:

$$l = 14\, n_{slot}^{RA} + l_0 + n_t^{RA}\, N_{dur}^{RA}$$

where:

- Each value of "$n_{slot}^{RA}$" indicates the **location** of a **PRACH slot** within **each subframe** or **60 kHz slot**. The location is expressed as the **slot index** within the **subframe**, or the **60kHz slot**, where the **RA preamble** can be transmitted. The value of "$n_{slot}^{RA}$" is indicated in the following table:

| PRACH subcarrier spacing $\Delta f_{RA}$ [kHz] | *#PRACH Slots* = 1 | *#PRACH Slots* = 2 |
|---|---|---|
| 1.25, 5, 15, 60 | $n_{slot}^{RA}=0$ | |
| 30, 120 | $n_{slot}^{RA}=1$ | $n_{slot}^{RA}\in\{0, 1\}$ |
| 480 | $n_{slot}^{RA}=7$ | $n_{slot}^{RA}\in\{3, 7\}$ |
| 960 | $n_{slot}^{RA}=15$ | $n_{slot}^{RA}\in\{7, 15\}$ |

  Each of the two columns "*#PRACH Slots*" in this table corresponds to the **value** of the column "*Quantity of PRACH slots within a subframe*" or the column "*Quantity of PRACH slots within a 60 kHz slot*" (as corresponds) from the corresponding table 6.3.3.2-2 to 6.3.3.2-4 of TS 38.211.

- "$l_0$" is the **PRACH subcarrier symbol** index, **within** the **PRACH slot**, where the **first RA preamble time occasion** is located. Its value is obtained from column "$l_0$" from the corresponding table 6.3.3.2-2 to 6.3.3.2-4 in TS 38.211

- Each value of "$n_t^{RA}$" indicates the location of **one PRACH transmission occasion** within the **PRACH slot**. It ranges from **0** to $N_t^{RA,slot}$-1. The value of $N_t^{RA,slot}$ is obtained from the corresponding table 6.3.3.2-2 to 6.3.3.2-4 in TS 38.211.

- "$N_{dur}^{RA}$" is the **time separation** between **2 consecutive PRACH transmission occasions**, within a **PRACH slot**. It is equivalent to the **total duration** of the **RA preamble** (including its guard period), **expressed in PRACH-subcarrier symbol intervals**. Its value is obtained from the corresponding table 6.3.3.2-2 to 6.3.3.2-4 in TS 38.211.

For more detail on **RA preamble time transmission occasions**, refer to clauses 5.3.2 and 6.3.3.2 of TS 38.211.

## 7.3 PRACH contents and processing

**PRACH** is used by the **UE** to send its **RA preamble** as part of the **random access procedure** (click 9).

**PRACH** is **different** to **every other physical binary channel** in several **relevant** aspects (click 7.1).

**PRACH** carries data from the **RACH transport** channel (the **RA preamble sequence index**), but it **does not** carry data from **any logical channel**.

The **subcarrier spacing** of **PRACH** is the **one** of the **transmitted RA preamble** (click 7.3.1).

The **frequency-time location** of the **PRACH transmission conveys** to the **NR node** the **information** that it **needs** to **compute** (click 9.1.3.1) the **RA-RNTI** (click 9.1.7).

For the case of **Integrated Access** and **Backhaul Mobile Termination** (**IAB-MT**), the **PRACH** contents may have some variations, not detailed here.

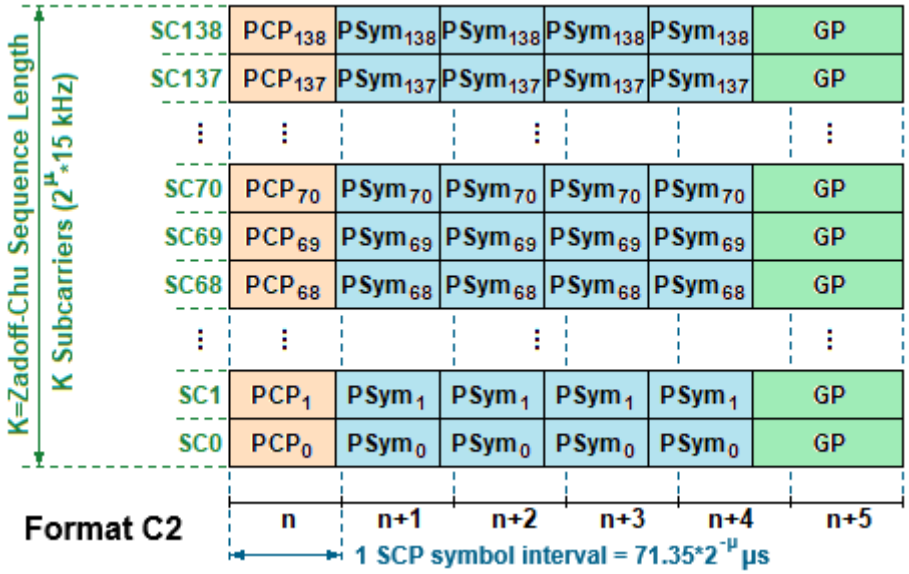
For more detail on **PRACH contents**, refer to clauses 5.3.2 and 6.3.3 of TS 38.211 and to TS 38.300.

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_RACH.html
- https://www.prodevelopertutorial.com/5g-nr-rach-process-msg1-generation-prach-preamble/
- https://www.rfwireless-world.com/5G/5G-NR-PRACH.html

### 7.3.1 RA preamble

Within **each PRACH subcarrier**, the **RA preamble** starts with a **PRACH-specific cyclic prefix** (**PCP**), followed by **one or more identical PRACH-specific symbols**. **Each RA preamble** is followed by a **PRACH-specific Guard Period**. **Each RA preamble sequence** is sent **across** the **frequency domain**, occupying **only 1 PRACH-specific symbol interval**, and as many **contiguous PRACH subcarriers** as the **RA sequence length**. **All** the **PRACH subcarriers** that compose the **RA preamble sequence** have **the same PRACH subcarrier spacing** (which can be a **PRACH-specific** value). This is **somehow** similar to the way **PSS** and **SSS** work (click 6.19), in the sense that a **sequence of symbols** is sent over **one symbol interval** and across **as many subcarriers** as the **sequence length**. The following figure shows **one example** of a **RA preamble**, for the **control/shared** (i.e., **not PRACH-specific**) values of **subcarrier spacing**:




Some **relevant remarks** on this figure are:

- **PCP$_x$** is the **PRACH-specific cyclic prefix** (**PCP$_x$**).
- **PSym$_x$** is the **PRACH-specific symbol**. Notice that in **each subcarrier** there are **four repetitions** of the **same** **PRACH-specific symbol**. The result is **four repetitions** of the **PRACH sequence**, **each** of them in **one Psym** over "$L_{RA}$" **subcarriers**.
- **Subindex** "**x**" in the **PRACH-specific symbols** (**PSym$_x$**) is the **PSymbol index** within the **Zadoff-Chu** (**ZC**, click 11.20.4) **sequence**. **Subindex** "**x**" in the **PRACH-specific cyclic prefixes** (**PCP$_x$**) indicates to what **Psymbol** corresponds

the **PCP**. The **subindex** "**x**" also indicates the **subcarrier index** within the **RA preamble** where **PSymbol$_x$** and **PCP$_x$** are located.

- **Each PRACH-specific symbol** (**PSym$_0$** to **PSym$_{138}$**), and its corresponding **PRACH-specific cyclic prefix** (**PCP$_0$** to **PCP$_{138}$**), has a **specific value** in **each subcarrier** (**SC0 to SC138**). **Each such value** is determined by the **ZC sequence**.
- **GP** is the trailing **PRACH Guard Period** (**GP**).
- The **total duration** of this example of **RA preamble** is **6 normal-short CP symbol intervals** with the same value of **subcarrier spacing** used in this **RA preamble**, as it is shown at the bottom of the figure.
- The **RA preamble** example shown in this figure corresponds to **RA preamble Format C2** (click 7.3.2) with a **sequence length** of **139 PRACH-specific symbols**.

The **duration** of the **PRACH-specific cyclic prefix** and of the **PRACH Guard Period**, plus the **quantity** of **repetitions**, depend on the **RA preamble Format** (click 7.3.2). **Each RA preamble Format** also indicates what is(are) its allowed **ZC sequence length**(**s**) and on its allowed **subcarrier spacing**(**s**).

The **RA preamble bandwidth** is:

**PRACH Bandwidth = PRACH sequence length • PRACH subcarrier spacing**

$$BW_{PRACH} = L_{RA} \cdot \Delta f_{RA}$$

The **allocated bandwidth** to send the **RA preamble** is stated in table 6.3.3.2-1 (click 7.2.1) of TS 38.211, indicated as a certain **quantity of RBs** with the **subcarrier spacing** of **PUSCH**. Notice that the **allocated bandwidth** is **larger** than the **RA preamble bandwidth**, which allows for some **guardband** at both ends of the **RA preamble**, to separate it from the **PUCCH/PUSCH subcarriers**.

The **RA preamble** is sometimes called the "**PRACH random access preamble**" or the "**Random Access preamble**".

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_RACH.html
- https://www.5g-networks.net/5g-nr-rach-preamble-types-long-and-short-preambles/
- https://www.prodevelopertutorial.com/5g-nr-rach-process-msg1-generation-prach-preamble/
- https://www.rfwireless-world.com/5G/5G-NR-PRACH.html

## 7.3.2 RA preamble Formats

**Each RA preamble** (click 7.3.1) **Format** defines the **duration** of the **PRACH-specific cyclic prefix** and of the **PRACH Guard Period**, plus the **quantity** of **repetitions**. **Each RA preamble Format** also indicates what is(are) its allowed **ZC sequence length**(**s**) and its allowed **subcarrier spacing**(**s**).

There are **13** different **RA preamble Formats**. **Nine** of these **preamble Formats** are combined with **8 subcarrier spacings** and **3 sequence lengths**. The result is a **considerably large** quantity of **RA preamble options**. The main reason for defining

many **different preamble options** is to provide a **suitable preamble** for each value of **cell reach**.

The **subcarrier spacings** allowed for **PRACH** ($\Delta f_{RA}$) depend on the **RA preamble Format** and on the **frequency band**. For **long RA preamble Formats** (**0** to **3**), "$\Delta f_{RA}$" may only take the **PRACH-specific** values **1.25 kHz** or **5 kHz** and it is determined by the **RA preamble Format**. For **short RA preamble Formats** (**A1** to **C2**), the allowed values for "$\Delta f_{RA}$" depend on the **Frequency Range**: for **FR1** bands, it is either **15 kHz** or **30 kHz**; for **FR2-1** bands, it is either **60 kHz** or **120 kHz**; for **FR2-2** bands, it is **120 kHz**, **480 kHz** or **960 kHz**. See more detail further below. Moreover, the **PRACH subcarrier spacing** depends on the **PUSCH subcarrier spacing**, because it is only allowed to use the **combinations** stated in table 6.3.3.2-1 (click 7.2.1) of TS 38.211.

The **13 RA preamble Formats** can be classified in **2 types**, according to the **time duration** of the **RA preamble** as: **long RA preamble Formats** (sometimes called "**long sequence preamble Formats**") and **short RA preamble Formats** (sometimes called "**short sequence preamble Formats**")

- **Long preamble Formats: 0**, **1**, **2** and **3**

  Their **RA preamble total duration** (including the **guard period**) is **1 ms** for **Formats 0** and **3**, it is **3 ms** for **Format 1** and it is **~4.3 ms** for **Format 2**. The **duration** of **one PRACH-specific symbol** is **200 µs** for format **3** and **800 µs** for **Formats 0**, **1** and **2**.

  Their **sequence length** "$L_{RA}$" is **839** **PRACH-specific** **symbols**, sent over the **same quantity** of **contiguous PRACH subcarriers**.

  Their **subcarrier spacing** is **1.25 kHz** for **Formats 0**, **1**, and **2** and it is **5 kHz** for **Format 3**. Notice that **these two subcarrier spacings** are **PRACH-specific** and **cannot** be used for **any other** physical layer element.

  They support **all 3 types** of **sequence sets** (see below): **unrestricted**, **restricted Type A** and **restricted Type B**.

  They are part of the cell random access configuration, and each cell is limited to **only one long** preamble **Format**.

  They can **only** be used in **FR1**, mainly for **large** and **very large cells** (up to **100 km reach**).

- **Short preamble Formats: A1**, **A2**, **A3**, **B1**, **B2**, **B3**, **B4**, **C0** and **C2**

  Their **RA preamble total duration** (including the guard period) ranges from **2.23 µs** (**A1**, **B1** or **C0** with **SCS=960 kHz**) to **857 µs** (**B4** with **SCS=15 kHz**).

  Their **sequence length** "$L_{RA}$" may be **139**, **571** or **1,151** **PRACH-specific** **symbols**, sent over the **same quantity** of **PRACH subcarriers**. The sequence lengths **571** and **1,151** were introduced in **5G-R16**.

  Their **subcarrier spacing** may be **15** or **30 kHz** in **FR1** and **60**, **120**, **480** or **960 kHz** in **FR2**. Sequence length **139** supports subcarrier spacings of **15**, **30**, **60**, **120**, **480**, and **960 kHz**. Sequence length **571** supports subcarrier spacings of **30**, **120**, and **480 kHz**. Sequence length **1,151** supports subcarrier spacings of **15** and **120 kHz**.

  **Sequence length** "$L_{RA}$" of **139** can be used **both** for **licensed spectrum** and for **shared spectrum**.

**Sequence lengths** "$L_{RA}$" of **571** and **1,151**: **a**) in **FR1** they can **only** be used for **shared spectrum**; and **b**) in **FR2**, they can be used for **licensed spectrum** and for **shared spectrum**.

They **only** support **unrestricted sequence sets** (see below).

They mainly target **small/normal cells** and **indoor** deployment scenarios.

Notice that the **four possible values** of the **sequence length** "$L_{RA}$" are **prime**, which results in **valuable properties** of the **ZC sequence**.

For the relationship between the **RA preamble Formats** and **cell size**, click 3.10.2.2.

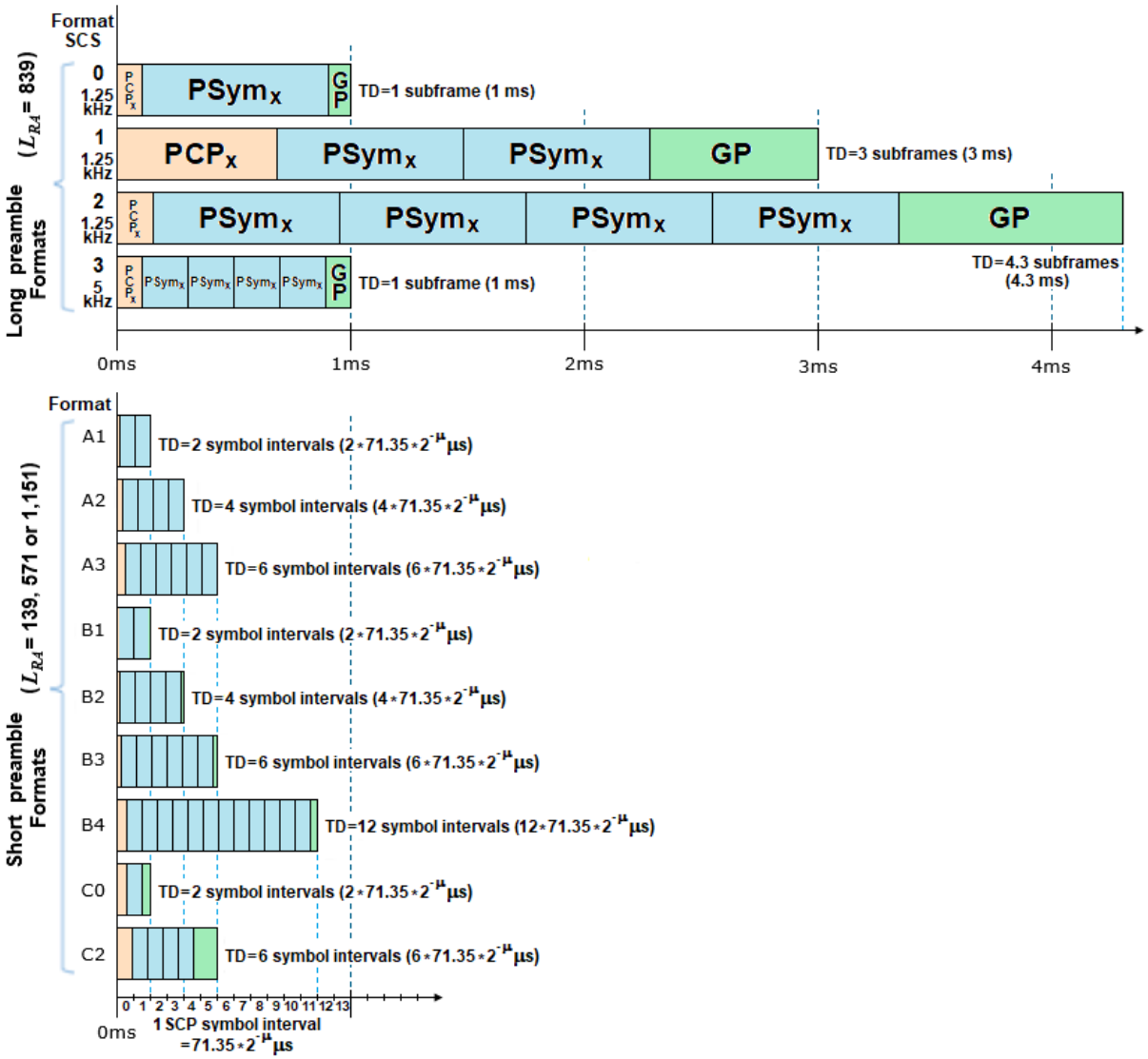
The following table summarizes the allowed combinations of **frequence range**, **subcarrier spacing**, **preamble Format** and **sequence length**:

| | | | **Sequence length $L_{RA}$ [Psymbols]** | | | |
|---|---|---|---|---|---|---|
| | | | **Long preambles** | **Short preambles** | | |
| **Frequency Range** | **SCS [kHz]** | **Formats** | **839** | **139** | **571** | **1,151** |
| **FR1** | **1.5** | **0, 1, 2** | Yes | No | No | No |
| | **5** | **3** | Yes | No | No | No |
| | **15** | **A1 to C2** | No | Yes | No | Yes |
| | **30** | **A1 to C2** | No | Yes | Yes | No |
| **FR2-1** | **60** | **A1 to C2** | No | Yes | No | No |
| | **120** | **A1 to C2** | No | Yes | Yes | Yes |
| **FR2-2** | **480** | **A1 to C2** | No | Yes | Yes | No |
| | **960** | **A1 to C2** | No | Yes | No | No |

The following figure summarizes the **13 preamble Formats**[1], grouped by **type** (**long** or **short**), showing for **each Format** the **duration** of the **PRACH-specific cyclic prefix** (mauve rectangle), of **each PRACH-specific symbol** (blue rectangle), of the **PRACH Guard Period** (green rectangle) and the **quantity** of **sequence repetitions** (quantity of

1 Initially there were also formats A0 and C1, but 3GPP later removed them.

blue rectangles), where **each sequence** is sent over **one PRACH-specific symbol** (**PSym**) and "$L_{RA}$" **subcarriers**:




Some **relevant remarks** on this figure are:

- For the **long** RA preamble **Formats**, the **Total Duration** (labelled "**TD**") of the **RA preamble** plus its **Guard Period** is indicated in **subframes** (i.e., in **ms**) because in this case the **Total Duration** is fixed, since each format **must** use **one specific value** of **subcarrier spacing**.
- For the **short** RA preamble **Formats**, the **Total Duration** (labelled "**TD**") of the **RA preamble** plus its **Guard Period** is indicated in **normal-short CP symbol intervals** (click 2.3.2), whose **duration** ($71.35 \cdot 2^{-\mu}$ µs) depends on the **subcarrier spacing** used. Recall that the **short** RA preamble **Formats** may be used with **all** the **5G subcarrier spacings** (except **240 kHz**).
- The **durations depicted** in the figure for the **short** RA preamble **Formats** correspond to a **subcarrier spacing** of **15 kHz**, because in this way you may **visually compare** their duration with the ones of the **long** RA preamble **Formats**. If a **larger subcarrier spacing** were used for the **short** RA preamble **Formats**, their **time duration** would be **proportionally reduced**.

- The last part of each **PRACH-specific symbol** acts as a **cyclic prefix** (click 11.13) for the next **PRACH-specific symbol**. The **total duration** (including the **Guard Period**) of a **RA preamble** with "**n**" **PRACH-specific symbols** is the same as the **duration** as the same quantity of **normal-short CP symbol intervals** with the same **subcarrier spacing**. There are some relevant benefits of this new design:
  - It allows the **gNB** to use the same **fast Fourier transform** (**FFT**) **computation both** for **data** and for **RA preamble detection**.
  - Due to the composition of **multiple shorter PRACH-specific symbols** per **RA preamble**, the new **short** preamble **Formats** are more robust against frequency errors and time varying channels.

**Short** RA preamble **Formats** have the following relevant characteristics:

- In most cases it is possible to have multiple preamble transmissions multiplexed in time within a single **PRACH slot**. In other words, for short preambles there may be multiple **PRACH transmission occasions** in the **frequency domain** (click 7.2.1) **as well as** in the **time domain** (click 7.2.2) within a single **PRACH slot**.
- Their **short time duration** enables the possibility of **NR node Rx beam sweeping** (click 2.4.1.5) during **PRACH** reception in such a way that the **same preamble** can be **received** from **different beams** at the **NR node**.
- The following **combinations** of **short** RA preamble **Formats** "**A**" and "**B**" are supported: **A1/B1**, **A2/B2**, and **A3/B3**. The way to "**combine**" an "**Ai**" and a "**Bi**" format is by sending the **Bi** format in the **last transmission occasion** within the **PRACH slot** (7.2.2) and sending the **Ai** format in **all the other ones**.
- **Short** RA preamble **Formats B2** and **B3** are **always** used **combined with** their corresponding **A2** and **A3 Formats**, as **A2/B2** and **A3/B3**.
- **Short** RA preamble **Formats A1** to **A3** and **B1** to **B3** are identical except for the **slightly** RA preamble **cyclic prefix** of the **B** formats.
- **Short** RA preamble **Formats** are mainly intended for **cells** which are unable to accommodate relatively long periods of **Uplink** transmission. In the case that the **NR node** uses **Tx beam sweeping** (click 6.1.2.3) in **FR2**, **only 1 beam** can be **active** at any given time so it is **not possible** to share **PRACH occasions** (click 7.2.2) across beams **nor frequency multiplex** the **PRACH occasions** associated with **each beam**.
- **Short** RA preamble **Formats** allow the set of **PRACH occasions** to be **time multiplexed** with **minimal latency**, i.e., the **time** taken to **cycle** through the **set** of **PRACH occasions** belonging to the **set** of **beams** is relatively **short**.

The definition of the **RA preamble Formats** and their characteristics is performed in tables 6.3.3.1-1 (**short** RA preamble **Formats**: **0** to **3**) and 6.3.3.1-2 (**long** RA preamble **Formats**: **A1** to **C2**) of TS 38.211. Both tables have been summarized as follows:

**Long preamble Formats: 0 to 3**

| Long preamble Format | $L_{RA}$ (sequence length) [PRACH symbols] | $\Delta f_{RA}$ (PRACH subcarrier spacing) [kHz] | $Nu$ (duration of all repetitions) | | $N_{PCP}^{RA}$ PCP duration [$T_c$] | Guard Period duration [$T_c$] | Total Duration [subframes] | Supported sets |
|---|---|---|---|---|---|---|---|---|
| | | | PSymbol repetitions | PSymbol duration [$T_c$] | | | | |
| **0** | 839 | 1.25 | 1 | 24,576$\kappa$ | 3,168$\kappa$ | 2,976$\kappa$ | 1 | Unrestricted plus restricted Type A or Type B |
| **1** | | | 2 | | 21,024$\kappa$ | 21,984$\kappa$ | 3 | |
| **2** | | | 4 | | 4,688$\kappa$ | 29,264$\kappa$ | 4.3 | |
| **3** | | 5 | 4 | 6,144$\kappa$ | 3,168$\kappa$ | 2,976$\kappa$ | 1 | |

**Short preamble Formats: A1 to C2**

| Short preamble Format | $L_{RA}$ (sequence length) [PRACH symbols] | | | $\Delta f_{RA}$ (PRACH subcarrier spacing) [kHz] | $Nu$ (duration of all repetitions) | | $N_{PCP}^{RA}$ PCP duration [$T_c$] | Guard Period duration [$T_c$] | Total Duration [SCP sym] | Supported sets |
|---|---|---|---|---|---|---|---|---|---|---|
| | $\mu \in \{0,1,2,3,5,6\}$ | $\mu \in \{0,3\}$ | $\mu \in \{1,3,5\}$ | | PSymbol repetitions | PSymbol duration [$T_c$] | | | | |
| **A1** | 139 | 1,151 | 571 | $15 \cdot 2^{\mu}$ | 2 | 2,048$\kappa \cdot 2^{-\mu}$ | 288$\kappa \cdot 2^{-\mu}$ | 0 | 2 | Unrestricted |
| **A2** | | | | | 4 | | 576$\kappa \cdot 2^{-\mu}$ | 0 | 4 | |
| **A3** | | | | | 6 | | 864$\kappa \cdot 2^{-\mu}$ | 0 | 6 | |
| **B1** | | | | | 2 | | 216$\kappa \cdot 2^{-\mu}$ | 72$\kappa$ $2^{-\mu}$ | 2 | |
| **B2** | | | | | 4 | | 360$\kappa \cdot 2^{-\mu}$ | 216$\kappa$ $2^{-\mu}$ | 4 | |
| **B3** | | | | | 6 | | 504$\kappa \cdot 2^{-\mu}$ | 360$\kappa$ $2^{-\mu}$ | 6 | |
| **B4** | | | | | 12 | | 936$\kappa \cdot 2^{-\mu}$ | 792$\kappa$ $2^{-\mu}$ | 12 | |
| **C0** | | | | | 1 | | 1,240$\kappa \cdot 2^{-\mu}$ | 1,096$\kappa$ $2^{-\mu}$ | 2 | |
| **C2** | | | | | 4 | | 2,048$\kappa \cdot 2^{-\mu}$ | 2,912$\kappa$ $2^{-\mu}$ | 6 | |

The meaning of the different symbols in the previous two tables are:

- "$L_{RA}$" is the **ZC root sequence length**

  It is the **length** of **each** preamble sequence, in **PRACH-specific symbols**. In the **long** preamble **Formats**, there is only **one value** of "$L_{RA}$", independently of the preamble format. In the **short** preamble **Formats**, the allowed values of "$L_{RA}$" **do not depend** on the preamble format and rather depend on the **PRACH subcarrier spacing**, as indicated by the "**μ**" values in each of the corresponding column headers.

- **$\Delta f_{RA}$** is the **PRACH subcarrier spacing**

  It indicates the allowed **subcarrier spacing**(**s**) for the **RA preamble** sequence. In the **short** preamble **Formats**, **not all** the **combinations** of **sequence length** and **subcarrier spacing** are allowed (see table further above).

- "$N_u$" is the **duration** of **all** the **repetitions** of the **PRACH-specific symbols**.

  Indicates the **duration** of **all** the **repetitions** of the **PRACH-specific symbols**. "$N_u$" is the **quantity** of **PRACH-specific symbol repetitions** ("PSymbol repetitions") multiplied by the **duration** of **one PRACH-specific symbol**

("PSymbol duration"), which is expressed as a quantity of basic **5G** time intervals "$T_c$".

- The quantity of **repetitions** depends **only** on the **preamble Format**.
- The **PSymbol duration** is expressed as the quantity of **5G** basic time intervals "$T_c$". Notice that the **PSymbol duration** is **totally independent** of the **sequence length** and of the **preamble format**. However, the **PSymbol duration** is **inversely proportional** to the **PRACH subcarrier spacing**.

- "$N_{PCP}^{RA}$" is the **PRACH-specific cyclic prefix duration**

  Indicates the **duration** of the **specific cyclic prefix** of the **RA preamble**. The value of "$N_{PCP}^{RA}$" is expressed as a quantity of **5G basic time intervals** "$T_c$" (see below). Notice that the **duration** of the **PRACH-specific cyclic prefix** is **independent** of the **sequence length**, of the **PSymbol duration** and of the **Psymbol repetitions**. The **PRACH-specific cyclic prefix duration** is **specific** of **each RA preamble format**. In the **short** preamble **Formats**, the **PRACH-specific cyclic prefix duration** is **inversely proportional** to the **PRACH subcarrier spacing**. Notice that we are using the symbol "$N_{PCP}^{RA}$", which is **slightly different** to the one used in TS documents, that is "$N_{CP}^{RA}$", in order to highlight that this **cyclic prefix** is the **PRACH-specific one**.

- **Guard Period**

  It is the **guard period** after the transmission of the **RA preamble**, where another **RA preamble cannot** be transmitted.

- **Total Duration**

  It is the duration of the **RA preamble** plus its **guard period**. For the **long** preamble **Formats**, it is expressed in **subframes** (**1 ms**). For the **short** preamble **Formats**, it is expressed in quantity of **normal-short CP symbol intervals** (click 2.3.2) of the corresponding **PRACH subcarriers spacing**.

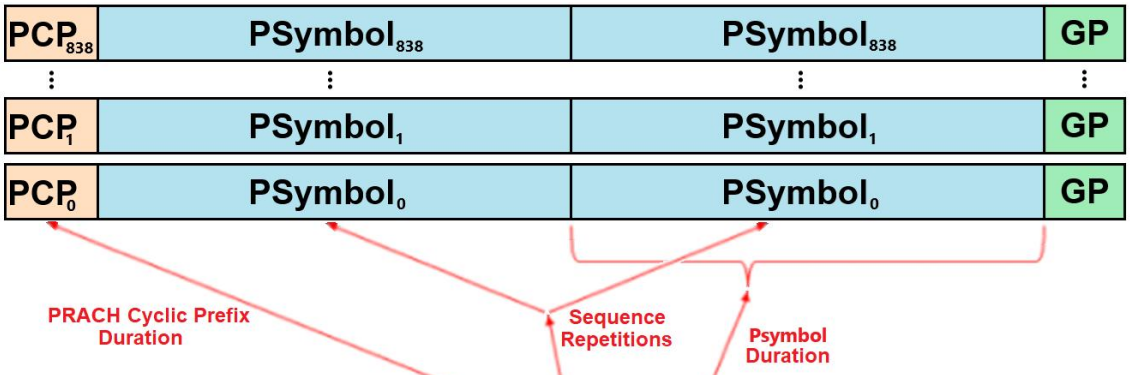
- **Supported sets**

  Indicates the type(s) of **set** of **RA preamble sequences** (**unrestricted**, **restricted Type A** and **restricted Type B**) that is supported by **each preamble Format**. Notice that in the first table **all types** are supported, while in the second table only the **unrestricted type** is supported.

- $T_c$ and "**κ**" (***Kappa***)

  ***Kappa*** is "$T_s/T_c$", and its value is **64**; $T_c$**~509 ps**; $T_s$**=κ**$T_c$~64•509**=32.6 ns**. For more detail on "$T_c$", "$T_s$" and "**κ**", click 2.3.2.

The following figure shows an example of how is a **RA preamble** defined from the data of these tables:




| Format | $L_{RA}$ | $\Delta f^{RA}$ | $N_u$ | $N_{PCP}^{RA}$ | Support for restricted sets |
|---|---|---|---|---|---|
| 0 | 839 | 1.25 kHz | $24576\kappa$ | $3168\kappa$ | Type A, Type B |
| 1 | 839 | 1.25 kHz | $2 \cdot 24576\kappa$ | $2 \cdot 21024\kappa$ | Type A, Type B |
| 2 | 839 | 1.25 kHz | $4 \cdot 24576\kappa$ | $4688\kappa$ | Type A, Type B |
| 3 | 839 | 5 kHz | $4 \cdot 6144\kappa$ | $3168\kappa$ | Type A, Type B |

**LTE** only has **long** preamble **Formats**, and its Formats **0** and **2** are identical to **5G** preamble **Formats 0** and **1**, respectively.

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_RACH.html
- https://www.5g-networks.net/5g-nr-rach-preamble-types-long-and-short-preambles/
- https://www.prodevelopertutorial.com/5g-nr-rach-process-msg1-generation-prach-preamble/
- https://www.rfwireless-world.com/5G/5G-NR-PRACH.html

### 7.3.3 RA preamble sequences per cell

**Each cell** (click 3.1.1) has a **set** of **64 random access (RA) preamble sequences**. **Each** of these **sequences** is called a "**signature**". The **specific RA preamble sequence** sent by a **UE** in its **RA preamble** is part of the **random access procedure** (click 9).

The **set** of **64 RA preamble sequences** is computed **departing** from the following **Zadoff-Chu** (**ZC**, click 11.20.4) **root sequence**:

$$x_u(n) = e^{-j\frac{\pi u n(n+1)}{L_{RA}}}, n = 0,1,\ldots,L_{RA}-1$$

The variables in this expression are:

- "***u***" is the **ZC root sequence index**

  It is an **integer** from **0** to "$L_{RA}$-2". Since in this case the **four choices** of "$L_{RA}$" **values** are **all prime**, **all** the possible values of "***u***" are **valid**. The **UE** gets "***u***" out of `prach-RootSequenceIndex` or `prach-RootSequenceIndex-r16` from `RACH-ConfigCommon` (click 9.5.3), or out of `msgA-PRACH-`

`RootSequenceIndex-r16` from `RACH-ConfigCommonTwoStepRA-r16` (click 9.5.8), **both data type** instances within `SIB1` (click 5.3.4).

- "$L_{RA}$" is the **ZC root sequence length**

  The **sequence length** is the **quantity** of **PRACH-specific symbols** in the **ZC root sequence**. The possible values of "$L_{RA}$" are: `839` for the **long** preamble **Formats** and `139`, `571` or `1151` for the **short** preamble **Formats**. The **UE** learns this from the **same parameters** indicated in the **previous bullet**.

The **set** of **RA preamble sequences** is defined by **cyclic shifting** the **ZC root sequence** indicated above with parameter "$C_v$", as follows:

$$x_{u,v}(n)=x_u((n+C_v) \bmod L_{RA})$$

The **set** of values for "$C_v$" is obtained by a lengthy expression arising from several cases and parameters ("$L_{RA}$", **PRACH subcarrier spacing**, the **type of set** of **RA preamble** sequences, `SIB1` parameters, various tables, etc). There are **3 types of set** of **RA preamble sequences**: **unrestricted**, **restricted Type A** and **restricted Type B**, and its value is obtained by the **UE** out of `restrictedSetConfig` from `RACH-ConfigCommon` (click 9.5.3), within `SIB1` (click 5.3.4).

For more detail on "$C_v$", refer to clause 6.3.3.1 of TS 38.211.

In case the **64** preamble sequences that are needed **cannot** be generated by the "$C_v$" **cyclic shifts** of the **ZC root sequence**, the **required additional ZC root sequence(s)** are generated using the **next value(s)** of the **sequence index** "$u$" (where the **sequence index** is cyclic and "$u$=**0**" follows "$u$=$L_{RA}$-2"). The **64 RA preamble sequences** thus produced are indexed in increasing order of **first** increasing cyclic shift "$C_v$", and then, in increasing order of the logical root sequence index starting from "$u$".

### 7.3.4 RA preamble formal definition

The **OFDM baseband signal** for the **RA preamble**, as the time-continuous signal "$s_l^{(p,\mu)}(t)$" on **antenna port** "$p$" (click 4.11.4.8.3) is defined by:

$$s_l^{(p,\mu)}(t) = \sum_{k=0}^{L_{\mathrm{RA}}-1} a_k^{(p,\mathrm{RA})}\, e^{j2\pi(k+Kk_1+\bar{k})\Delta f_{\mathrm{RA}}\left(t-N_{\mathrm{PCP},l}^{\mathrm{RA}}T_{\mathrm{c}}-t_{\mathrm{start}}^{\mathrm{RA}}\right)}$$

$$\text{where } t_{\mathrm{start}}^{\mathrm{RA}} \le t < t_{\mathrm{start}}^{\mathrm{RA}} + \left(N_{\mathrm{u}} + N_{\mathrm{PCP},l}^{\mathrm{RA}}\right)T_{\mathrm{c}} \text{ and } K = \Delta f / \Delta f_{\mathrm{RA}}$$

A brief explanation of this equation is:

- **Sequencing term "Σ"**

  The summation "**Σ**" represents the **sequencing** in frequency of **each instance** of the **PRACH-specific symbol** $a_k^{(p,RA)}$ of the **RA preamble**. The **summation index** "$k$" ranges from **0** to "$L_{RA}$-1", which is the **PRACH sequence length**. Therefore, index "$k$=**0**" represents the **first symbol** in the **PRACH sequence**, while **index** "$k$=$L_{RA}$-**1**" represents the **last symbol** in the sequence.

- **Time repetitions term "$a_k^{(p,RA)}$"**

  **Each** term "$a_k^{(p,RA)}$" is **one cyclic prefix** followed by the corresponding **repetitions** of the **PRACH-specific symbol** having index "$k$" in the **PRACH sequence** "$y_{u,v}(k)$". The value of **each** such symbol "$y_{u,v}(k)$" is produced by the

**ZC** sequence (click 7.3.3). The **quantity** of **repetitions** is determined by the **RA preamble format** (click 7.3.2). These **PRACH-specific symbol repetitions** are adjusted to the corresponding power level (**$\beta_{PRACH}$**). Therefore:

$$a_k^{(p,RA)} = \beta_{PRACH}\, y_{u,v}(k)$$

- **Frequency-spreading term "($k$+…) $\Delta f_{RA}$"**

  This represents the placement of **each PRACH-specific symbol** from the **PRACH sequence** in its corresponding **contiguous PRACH subcarrier**. This frequency-spreading term consists of the summation index "$k$" multiplying "$\Delta f_{RA}$", which is the **PRACH subcarrier spacing**. This implies that **each successive PRACH-specific symbol** from the **ZC sequence** is **frequency-placed** at **consecutive PRACH subcarriers**. The **first ZC symbol** is placed in the **lowest frequency PRACH subcarrier**, and **successive symbols** are placed in the **next consecutive upwards PRACH subcarrier**.

- **Frequency repetitions term** "(…+$K\, k_1$+…) $\Delta f_{RA}$"

  This represents the **frequency locations** of the **center** of the **lowest frequency subcarrier** of **each** of the **RA preambles sent**. The value "$K\, k_1 \Delta f_{RA} = k_1\, \Delta f$" indicates that **each such location** corresponds to the **center frequency** of a **subcarrier** having **index** "$k_1$" within the **active BWP**. For an explanation on the value of "$k_1$", click 7.2.1.

- **Frequency offset term "(…+$\bar{k}$) $\Delta f_{RA}$"**

  This represents the **frequency offset** of the **RA preamble subcarriers**, from the **frequency location** of the **RA preamble** indicated in the previous bullet point. The **offset** is "$\bar{k}$" times the **PRACH subcarrier spacing** "$\Delta f_{RA}$". The value of "$\bar{k}$" is provided in table 6.3.3.2-1 (click 7.2.1) of TS 38.211.

- **PRACH-specific cyclic prefix temporal start term "($t$ - … - $t_{start}^{RA}$)"**

  This represents the **starting instant** of the **RA preamble** (i.e., the starting instant of the **PRACH-specific cyclic prefix**), which is "$t = t_{start}^{RA}$".

- **PRACH-specific symbols temporal start term "($t$ - $N_{PCP,l}^{RA}\, T_c$ - …)"**

  This represents the **starting instant** of the **first PRACH-specific symbol repetition** (i.e., right after the **PRACH-specific cyclic prefix**), which is "$t = t_{start}^{RA} + N_{PCP,l}^{RA}\, T_c$".

- **Temporal range term "$t_{start}^{RA} \leq t < t_{start}^{RA} + (N_u + N_{PCP,l}^{RA})\, T_c$"**

  This represents that the equation is **only valid** from the **starting instant** of the **RA preamble** "$t_{start}^{RA}$" to the **end** of the **time duration** of the **RA preamble**, which is the **duration** "$N_{PCP,l}^{RA}$" of the **PRACH-specific cyclic prefix plus** the **duration** "$N_u$" of the corresponding **repetitions** of the **PRACH-specific symbol** (both expressed as a quantity of basic time intervals "$T_c$"). The values of "$N_{PCP,l}^{RA}$" and "$N_u$" can be found in the two tables defining the formats of the **RA preamble**.

Finally, the frequency-domain representation of **each RA preamble sequence** is obtained by the expression:

$$y_{u,v}(n) = \sum_{m=0}^{L_{RA}-1} x_{u,v}(m) \cdot e^{-j\frac{2\pi mn}{L_{RA}}}$$

For more detail on the **RA preamble**, refer to clause 5.3.2 of TS 38.211.

## 7.4 PRACH parameters

The **UE** obtains **most** of the **common parameters** to configure **PRACH** (click 7.1) in the **Uplink initial bandwidth part(s)** from `SIB1` parameter `rach-ConfigCommon` from `BWP-UplinkCommon` (click 2.2.8.9), whose **data type** is `RACH-ConfigCommon` (click 9.5.3).

**Most** of the **PRACH parameters** are closely interrelated with the **random access parameters** . Consequently, for the **PRACH parameters**, refer to the chapter on **random access parameters** (click 9.5).

For an overall explanation on **UE configuration parameters** click 5. For a brief overview of **ASN.1**, click 5.9.

## 7.5 PUCCH overview

The **Physical Uplink Control Channel** (**PUCCH**) is used to send **Uplink Control Information messages** (**UCI messages**, click 7.7) from **each UE** to the **NR node**. These **UCI messages** convey **control information** for the **Physical layer**.

However, ***if*** [[an **UCI message** does **not** carry **Scheduling Request** (**SR**, click 7.7.1.2) **content**] **and** [the **UE** has **PUSCH data** to send]], ***then***, the **UCI message** is sent through **PUSCH** (click 7.10), **instead** of through **PUCCH**.

The most relevant **functions** of **PUCCH** are:

- Send **Downlink HARQ** (**N**)**ACKs** (click 7.7.1.1) from **each UE** to the **NR node**.
- Send **Scheduling Requests** (**SR**, click 7.7.1.2) from **each UE** to the **NR node**.
- Send the **Channel State Information** (**CSI**, click 7.7.1.3) from **each UE** to the **NR node**.
- **Send** the **Last Resource Retransmission** (**LRR**, click 7.7.1.4) information from **each UE** to the **NR node**.

**Each PUCCH transmission** takes place in **1** of the **transmission occasions determined** by the **PUCCH scheduling** (click 7.5). **Each slot** can contain **0**, **1** or **2 PUCCH transmissions** (click 7.6.8).

**Each PUCCH transmission** contains **1 Uplink Control Information** (**UCI**) **message** (click 7.7).

**Each UCI message** is subject to a **specific processing** (click 7.7) for its **transmission** and **reception** through **PUCCH**.

**Each PUCCH transmission** in a **slot** conforms to **1** among **several PUCCH resources**. (click 7.6.3). **Each PUCCH resource** conforms to **1 among 5 PUCCH resource**

**formats** (click 7.6.5.1), **each** refined with **1 among 7 PUCCH resource format variants** (click 7.6.6).

**Each UE** may be configured up to **56 PUCCH resources** in the **PUCCH resource sets** (click 7.6.7) of **each** of its **Uplink** (or **SUL**) **bandwidth parts**. These **PUCCH resources** are grouped into **several PUCCH resource sets** (click 7.6.7).

The **UE** determines (click 7.6.1) which **PUCCH resource** should it **use** for **each PUCCH transmission** based on the **PUCCH scheduling** (click 7.5) and on the **bitlength** of the **PUCCH UCI message** (click 7.7.2) to be **transmitted**.

**PUCCH resources** are configured in the following way:

- Before **RRC connection establishment**, the **UE** gets its **PUCCH resource common configuration** from **SIB1** (click 5.3.4). For more detail on this, click 7.6.5.2.
- After **RRC connection establishment**, the **UE** gets its **PUCCH resource dedicated configuration** from its **dedicated RRC parameters**. For more detail on this, click 7.6.5.3.

**PUCCH** is **configured** (click 7.8.2) with several **parameters** and **data types**.

Among other **techniques** to **improve reliability**, **PUCCH** uses **scrambling**, **BPSK**, **QPSK**, **polar coding**, **CRC**, **frequency-spreading**, **frequency hopping** and **cyclic shift hopping**.

**PDCCH** and **PUCCH** are the **2 only physical channels** that **do not** carry data from **any transport** channel. They **only** convey **hard real-time control information** for the **physical layer**. Since the **information** they convey is **hard-real time**, they **cannot improve reliability** through **retransmission**. This is why they do **not** use the **HARQ protocol**. They **rather** improve **reliability** with **extremely robust modulation**, **coding** and **diversity**.

Among other **techniques** to **improve reliability**, **PUCCH** uses **scrambling**, **BPSK**, **QPSK**, **polar coding**, **CRC**, **frequency-spreading**, **frequency hopping**, **cyclic shift hopping** and **DFT spreading**. For more detail on **PDCCH reliability techniques**, click 7.7.

**PUCCH symbols in TS documents**

The following table **summarizes** some **relevant PUCCH symbols** used in **TS documents**:

| Symbol | Meaning |
|---|---|
| $r_{u,v}^{(\alpha,\delta,m)}(n)$ | **Low-PAPR type 1 complex number sequence**. |
| $N_{PUCCH}^{repeat}$ | **Quantity** of **slots** where a **PUCCH transmission** is **repeated** in the **inter-slot repeat variant** |
| $r_{PUCCH}$ | **PUCCH resource index** within a **PUCCH resource set**. |
| $O_{UCI}$ | **Bitlength** of the **UCI message**. |
| $O^{ACK}$ | **Bit sequence** of **ACK information** within a **UCI message**. |
| $O_{ACK}$ | **Bitlength** of **ACK information** within a **UCI message**. |
| $O^{SR}$ | **Bit sequence** of **SR information** within a **UCI message**. |
| $O_{SR}$ | **Bitlength** of **SR information** within a **UCI message**. |
| $O^{CSI}$ | **Bit sequence** of **CSI information** within a **UCI message**. |
| $O_{CSI}$ | **Bitlength** of **CSI information** within a **UCI message**. |
| $O_{UCI}$ or $A$ | **Bitlength** of the **complete UCI message**. |

**All** the **bitlengths** in **this** table are **before adding** the **CRC**.

The **PUCCH description** in this book covers almost nothing about **PUCCH** for **shared spectrum**.

For more detail on **PUCCH**, refer to:

- Clauses 4, 5 and 6.3.2 (**Phys**. and **modulation**) of TS 38.211.
- Clauses 4.1, 5 and 6.3 (**mux**., **coding** and **UCI**) of TS 38.212.
- Clause 9 (**UCI reporting**) of TS 38.213.
- Clauses 5.3.1 and 5.3.3 (**overall Tx**) and 10.3 (**scheduling**) of TS 38.300.

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_PUCCH.html
- https://www.sharetechnote.com/html/5G/5G_UCI.html
- https://www.rfwireless-world.com/5G/5G-NR-UCI.html
- https://www.sharetechnote.com/html/lte_toolbox/Matlab_LteToolbox_5G_FrameStructure_UL.html#Format_0_Ex01
- https://www.sharetechnote.com/html/lte_toolbox/Matlab_LteToolbox_5G_PUCCH0.html

## 7.6 PUCCH scheduling

**PUCCH Tx scheduling**

It is **semi-persistent scheduling** for **scheduling requests** (**SRs**, click 7.7.1.2) and for **channel state information** (**CSI**, click 7.7.1.3), and **dynamic scheduling** for **HARQ** (**N**)**ACKs** (click 7.7.1.1).

**PUCCH Tx scheduling** consists of the **NR node assigning each UE** a certain **set** of **transmission occasions** where the **UE may transmit PUCCH** (e.g., a **scheduling request**, click 7.7.1.2) and **assigning each UE some transmission occasions** where the **UE must transmit PUCCH** (e.g., a **HARQ** (**N**)**ACK**, click 4.5.6.7).

**Each PUCCH transmission** must conform to **1** of the **PUCCH resources** (click 7.6.5) that have been assigned by the **NR node** to **each UE**. However, the **PUCCH transmission occasions** do **not** state what **PUCCH resource** should be **used**.

When the **selected PUCCH resource** includes the **inter-slot repeat variant**, the **first PUCCH transmission** that took place in **1** of the **scheduled PUCCH transmission occasions** is **repeated** by the **UE** (with some adjustments) in **several consecutive slots**, according to the configuration of the **PUCCH resource**, and to the **scheduled transmission occasions**.

The following **3 symbols** will be used for the **5 PUCCH `formats`**:

- "$N_{syin}^{PUCCH,n}$" is the **total quantity** of **symbol intervals** of the **PUCCH transmission**, for the corresponding `format` "`n`".
- "$M_{sc}^{PUCCH,n}$" is the **total quantity** of **subcarriers** of the **PUCCH transmission**, for the corresponding `format` "`n`".

- "$N_{RE}^{PUCCH,n}$" is the **total quantity** of **resource elements** of the **PUCCH transmission**, for the corresponding `format` "`n`":

$$N_{RE}^{PUCCH,n} = N_{syin}^{PUCCH,n} \bullet M_{sc}^{PUCCH,n}$$

**PUCCH Rx scheduling**

**PUCCH Rx scheduling** consists of the **NR node listening** in **all** the **transmission occasions** that it has assigned to **all its UEs**.

For an **overall** description of **scheduling**, **transmission occasions** and **transmissions**, click 4.11.4.

### 7.6.1 PUCCH scheduling of HARQ-ACKs

**PUCCH scheduling** for the **UE** to **send back** the **HARQ-ACKs** may be **semi-persistent** and/or **dynamic**.

**Semi-persistent PUCCH scheduling** is frequently used when the **PDSCH scheduling** is performed with **semi-persistent scheduling** (e.g., for **voice** over **IP**). For more detail on **semi-persistent scheduling** of **UCI HARQ ACKs**, refer to clause 9.2.3 of TS 38.213.

The remaining part of this section describes **PUCCH dynamic scheduling** for the **UE** to **send back** the **PDSCH HARQ-ACKs** through **PUCCH**

**Dynamic PUCCH scheduling** for the **UE** to **send back** the **HARQ-ACKs** is performed with the same **DCI message** (click 6.9) that **assigns** the **UE** the **PDSCH resources** for the reception of the **1** or **2 PDSCH codeword(s)** whose corresponding **(N)ACK** has to be **sent back** by the **UE**.

**PUCCH slot dynamic scheduling**

Because **5G HARQ** is **asynchronous** (click 11.12), it is **required** to provide **explicit timing** information. In **FDD bands**, the **PDSCH HARQ (N)ACK** is **sent back** a **few slots** after the **reception** of the **1** or **2 PDSCH codeword(s)**. In **TDD bands**, the **PDSCH HARQ (N)ACK** is **sent back aligned** with **Uplink slots** (based on the **duplexing pattern**).

The **DCI message field** "***PDSCH-to-HARQ_feedback timing indicator***" indicates the **quantity** of **slots between** the **slot** where the **1** or **2 PDSCH codeword(s)** was **received** and the **slot** where **its (N)ACK** is **sent back** through **PUCCH**. The following table shows, for **PDSCH HARQ**, the **DCI message Formats** where the said **field** is **present**, what is **assigned** by **each** of these **DCI Formats** and what is the **field**'s **bitlength** in **each** of these **DCI Formats**:

| | DCI message field "*PDSCH-to-HARQ_feedback timing indicator*" | | |
|---|---|---|---|
| | **Assigns** | **DCI Format** | **Bitlength** |
| **PDSCH HARQ** | **PDSCH** | Format 1_0 (6.9.5) | 3 |
| | | Format 1_1 (6.9.6) | 0 to 3 |
| | | Format 1_2 (6.9.7) | 0 to 3 |
| | **PDSCH MBS** | Format 4_1 (6.9.9.2) | 3 |
| | | Format 4_2 (6.9.9.3) | 0 to 3 |

This **field**'s **semantics** is the **same** for **all** these **DCI Formats**. The value of this **field** is an **index** to a **list** of "**quantity** of **slots**". **Field value** "0" indicates the **first value** in the

**list**. **Field value** "1" indicates the **second value** in the **list**, and so on. For example, if the value is "11" or "011", the **quantity** of **slots** is the **fourth value** in the **list**.

In respect to the **list** of "**quantity** of **slots**", there are the following **4 cases**.

- For a **DCI Format 1_0 message**, the **list** of "**quantity** of **slots**" depends on the **numerology** of **PUCCH** as follows:

| μ | SCS [kHz] | List of "quantity of slots" |
|---|---|---|
| ≤3 | 30, 60, 120, 240 | {1, 2, 3, 4, 5, 6, 7, 8} |
| 5 | 480 | {7, 8, 12, 16, 20, 24, 28, 32} |
| 6 | 960 | {13, 16, 24, 32, 40, 48, 56, 64} |

- For (a **unicast DCI Format other** than **1_0**) or (requesting **HC-Type 3 HARQ-ACK codebook report without** scheduling a **PDSCH reception** as described in clause 9.1.4 of TS 38.213), the **list** of "**quantity** of **slots**" is provided by the corresponding **parameter**:

```
dl-DataToUL-ACK                SEQUENCE (SIZE (1..8)) OF INTEGER ( 0..15 )
dl-DataToUL-ACK-r16            SEQUENCE (SIZE (1..8)) OF INTEGER (-1..15 )
dl-DataToUL-ACK-DCI-1-2        SEQUENCE (SIZE (1..8)) OF INTEGER ( 0..15 )
dl-DataToUL-ACK-r17            SEQUENCE (SIZE (1..8)) OF INTEGER (-1..127)
dl-DataToUL-ACK-DCI-1-2-r17 SEQUENCE (SIZE (1..8)) OF INTEGER ( 0..127)
dl-DataToUL-ACK-v1700          SEQUENCE (SIZE (1..8)) OF INTEGER (16..31 )
```

  **All** these parameters are from `PUCCH-Config` (click 7.9.4). As you may see, **each** and **every** of these **parameters** allows to configure a `SEQUENCE OF 1 to 8` values of "**quantity** of **slots**", where **each value** may take a **range** that is specific to **each parameter**.

  If the **DCI message** indicates a **cell** for the **PUCCH transmission**, as described in clause 9.A of TS 38.213, the resulting **quantity** of **slots** corresponds to **slots** of the **active Uplink BWP** of the **cell**; **otherwise**, it corresponds to **slots** of the **active Uplink BWP** of the **PCell**.

- For a **DCI Format 4_1 message**, the **list** of "**quantity** of **slots**" is provided, if **present**, by **parameter**:

```
dl-DataToUL-ACK-MulticastDCI-Format4-1-r17
                               SEQUENCE (SIZE (1..8)) OF INTEGER ( 0..15 )
```

  from `PUCCH-Config` (click 7.9.4). If this **parameter** is **absent**, the **list** of "**quantity** of **slots**" is {1, 2, 3, 4, 5, 6, 7, 8}.

- For a **DCI Format 4_2 message**, the **list** of "**quantity** of **slots**" is provided by the corresponding **parameter**:

```
pucch-ConfigMulticast1-r17
pucch-ConfigurationListMulticast1-r17
pucch-ConfigMulticast2-r17
pucch-ConfigurationListMulticast2-r17
```

  **All** these parameters are from `PUCCH-Config` (click 7.9.4).

The **UE** is required to indicate to the **NR node** its **minimum HARQ processing time** capability, so that the **NR node** will **always signal** a **larger** or **equal value** of "***PDSCH-to-HARQ_feedback timing indicator***".

The **UE** determines a **quantity** of **HARQ-ACK** information bits "$O_{ACK}$" as described in clauses 9.1 through 9.1.5 of TS 38.213 and a corresponding **PUCCH resource set** as described in clause 9.2.1.

A **UE** does **not expect** to transmit **more than one PUCCH** with **HARQ-ACK information** in a **slot** per **priority index**, if the **UE** is **not** configured with `ackNackFeedbackMode` = `separate`.

For more detail on **HARQ timing**, refer to clause 9.2.3 of TS38.213.

For more detail on **PDSCH HARQ**, refer to clauses 5.3.2 of TS 38.321 and 5.1 of TS 38.214. For more detail on **PUSCH HARQ**, refer to clauses 5.4.2 of TS 38.321 and 6.1 of TS 38.214.

### 7.6.2 PUCCH semi-persistent scheduling of scheduling requests (SRs)

**Scheduling requests** may be transmitted on **slots** configured with **semi-persistent scheduling**, in the **slot REs** defined by the **corresponding PUCCH resource**.

**Each instance** of **semi-persistent scheduling** for **scheduling request** (**SRs**) **transmission occasions** is configured with a **periodicity** "$SR_{PERIODICITY}$" in **symbol intervals** or **slots** and with an **offset** "$SR_{OFFSET}$" in **slots**.

The **slots** for **SR transmission occasions** depend on "$SR_{PERIODICITY}$", "$SR_{OFFSET}$" and **PUCCH numerology** "**μ**" as follows:

- $SR_{PERIODICITY} < N_{syin}^{slot}$ (expressed in **symbol intervals**)

  The **SR transmission occasions** take place **every** "$SR_{PERIODICITY}$" **symbol intervals**, and the **first symbol interval** has the **absolute symbol interval index** "($l_0$ ***mod*** $SR_{PERIODICITY}$)", where "$l_0$" is the value of `startingSymbolIndex` from the corresponding **instance** of the corresponding **data type** `PUCCH-format0` to `PUCCH-format4` (click 7.9.8). In this case, "$SR_{OFFSET}$" **must** be **0**.

  Expressing this in a **formal** way, the **SR transmission occasions start** in **symbol intervals** with **absolute symbol index** "$l$" that satisfy:

  $$0 = (l - (l_0 \ mod \ SR_{PERIODICITY})) \ mod \ SR_{PERIODICITY}$$

  The variable "$N_{syin}^{slot}$" is the **quantity** of **symbol intervals** in **each slot** (i.e., **14** or **12**, depending on **CP**).

- $SR_{PERIODICITY} \geq$ **1 slot** (expressed in **slots**)

  The **SR transmission occasions** take place **every** "$SR_{PERIODICITY}$" **slots**, where the **first slot** has the **absolute slot index** "$SR_{OFFSET}$". Expressing this in a **formal** way, the **SR transmission occasions** are in **slots** with **frame slot index** "$n_{s,f}^{\mu}$" within **frames** with **absolute index** "$n_f$" that satisfy:

  $$0 = (n_f \bullet N_{slot}^{frame,\mu} + n_{s,f}^{\mu} - SR_{OFFSET}) \ mod \ ()$$

  where "$N_{slot}^{frame,\mu}$" is the **quantity** of **slots** in a **frame** for **numerology** "**μ**".

- $SR_{PERIODICITY}$ = **1 slot** (expressed in **slots**)

  The **SR transmission occasions** take place **every slot**. In this case, "$SR_{OFFSET}$" **must** be **0**.

**Every SR transmission occasion** in a **slot** where the **quantity** of **scheduled symbol intervals** (click 7.5) for the **PUCCH transmission** is **less** than what is **required** is **not**

**valid**, and the **UE** does **not** transmit **PUCCH** in the **slot**. The **scheduled symbol intervals** are the ones from the **first symbol interval** of the **PUCCH resource** to the **last symbol interval** in the **slot**, **both included**.

The **semi-persistent PUCCH transmissions occasions** configured to **different UEs** may **overlap**, because **PUCCH supports code-division multiple-access** (click 11.9) of **different UEs** over the **same REs**. The **code-division multiple-access** is implemented with **complex sequences** and with **orthogonal cover codes** (**OCC**). Notice that the **PUCCH sections** in the **TS documents** use the term "**multiplexing**" instead of "**multiple access**" but the mechanism is the same (this is only an academic terminology question).

The **frequency** of the **scheduled SR transmissions occasions** is a **balance** between **wasted resources** (unused transmission occasions) and **delay** (time between a **UE** willing to transmit and being able to send its **PUCCH scheduling request**). This is why **each SR transmission** must occupy **very few resources**.

**Each instance** of **semi-persistent scheduling** for **scheduling request** (**SRs**) is configured with **1 instance** of `SchedulingRequestResourceConfig` (click 7.9.12). **Each** of its **instances** contains:

- `periodicityAndOffset`, that provides "$SR_{PERIODICITY}$" and "$SR_{OFFSET}$". The said values can also be provided by the **extension parameter** `periodicityAndOffset-r17` from `SchedulingRequestResourceConfig-1700` (click 7.9.13).
- `resource`, that **identifies** the `PUCCH-Resource` (click 7.9.7) instance, with **either PUCCH** `format0` or `format1`, where the **scheduling request transmission** will take place.
- `schedulingRequestId`, that **identifies** the **instance** of `SchedulingRequestToAddMod` (click 7.9.15) that provides the parameters `sr-ProhibitTimer` and `sr-TransMax`.

**Each UE** may be configured with **up to** `8` **instances** of `SchedulingRequestResourceConfig` (click 7.9.12), within `PUCCH-Config` (click 7.9.4), in **each** of its **Uplink** (or **SUL**) **bandwidth parts**.

**Each UE** may be configured with **up to** `8` **instances** of `SchedulingRequestToAddMod` (click 7.9.15), within `MAC-CellGroupConfig` (click 3.7.4) in **each** of its **cell groups**. The **ancestor tree** (click 5.9.8) of `SchedulingRequestToAddMod` is:

```
                                  CellGroupConfig (click 3.7.3)
mac-CellGroupConfig                 MAC-CellGroupConfig (0..1) (click 3.7.4)
  schedulingRequestConfig             SchedulingRequestConfig (0..1) (click 7.9.14)
    schedulingRequestToAddModList   SchedulingRequestToAddMod (0..8) (7.9.15)
```

The **SR transmission occasions** in **PUCCH** are subject to the **limitations** for **UE transmissions** described in clauses 11.1, 11.1.1 and 17.2 of TS 38.213

For more detail on **semi-persistent scheduling** of **UCI scheduling requests**, refer to clause 9.2.4 of TS 38.213.

### 7.6.3 PUCCH scheduling of multiple UCI types

For a description of **multiplexing** [**HARQ-ACK** or **CSI**] and [**scheduling requests**] in **PUCCH**, refer to clause 9.2.5.1 of TS 38.213.

For a **transmission occasion** of a **single CSI report**, a **PUCCH resource** is provided by `pucch-CSI-ResourceList` from `CSI-ReportConfig`. For a **transmission occasion** of **multiple CSI reports**, **1** or **2 PUCCH resource identifiers** can be provided by `multi-CSI-PUCCH-ResourceList` from `PUCCH-Config` (click 7.9.4). ***If*** a **UE** is provided a **first** and a **second** `PUCCH-Config` **instance** in a given **bandwidth part**, ***then***, `multi-CSI-PUCCH-ResourceList` is provided by the **first** `PUCCH-Config` **instance**, and the `PUCCH-ResourceId` (click 7.9.7) **instance** in `pucch-CSI-ResourceList` or in `multi-CSI-PUCCH-ResourceList` indicates a corresponding **PUCCH resource** in the `PUCCH-Resource` (click 7.9.7) **instance** provided by the **first** `PUCCH-Config`.

For more detail on **multiplexing HARQ-ACK** and/or **CSI** and/or **SR** in **PUCCH**, refer to clause 9.2.5.2 of TS 38.213.

For a description of **reporting UCI** of **different priorities**, refer to clause 9.2.5.2 of TS 38.213.

For a description of **scheduling** for **multiple UCI types**, refer to clause 9.2.5 of TS 38.213.

## 7.6.4 UE determination of which PUCCH resource to use

This consists of **determining** which **PUCCH resource** the **UE** should **use** for **each PUCCH transmission**. This determination **depends** on whether the **PUCCH transmission** has been **scheduled** with **semi-persistent** or **dynamic scheduling**, and on other factors.

### 7.6.4.1 PUCCH resource set determination

The **PUCCH resource set** to be used depends on the **bitlength** "$O_{UCI}$" of the **UCI message** to be sent by the **UE**, as follows:

- $O_{UCI} \leq 2$

  The **UE** uses the **first PUCCH resource set** (i.e., the one with `pucch-ResourceSetId` = `0`).

- $2 < O_{UCI} \leq N_1$

  The **UE** uses the **second PUCCH resource set** (i.e., the one with `pucch-ResourceSetId` = `1`).

- $N_1 < O_{UCI} \leq N_2$

  The **UE** uses the **third PUCCH resource set** (i.e., the one with `pucch-ResourceSetId` = `2`).

- $N_2 < O_{UCI} \leq 1{,}706$

  The **UE** uses the **fourth PUCCH resource set** (i.e., the one with `pucch-ResourceSetId` = `3`).

The values $N_1$ and $N_2$ in the bullets above correspond respectively to `maxPayloadSize` of the `PUCCH-ResourceSet` **instances** `1` and `2`. If `maxPayloadSize` is **not** defined for **PUCCH resource set** ***i***, then $N_i$ takes the **default** value "`1,706`", and there can be **no other PUCCH resource set** above "***i***".

For more detail on **PUCCH resource set determination**, refer to clauses 9.2.1 of TS 38.213.

### 7.6.4.2 PUCCH resource format determination

The **UE determines** the **PUCCH format** (click 7.6.5.1) to be **used** based on:

- The **UCI message bitlength**.
- The **quantity** of **symbol intervals** of the **PUCCH transmission opportunity**.
- Whether the **interlace option** (click 7.6.5.3.8) is **configured**, or not, in the **bandwidth part** where this transmission takes place.
- Whether an **orthogonal cover code** (**OCC**) is **configured**, or **not**, in the **bandwidth part** where this transmission takes place.

The following table summarizes the selection of a **PUCCH resource format** in **each PUCCH semi-persistent** transmission:

| **Determination of the PUCCH resource** format | | **Quantity of PUCCH symbol intervals (plus Interlace and OCC)** | | |
|---|---|---|---|---|
| | | **≤ 2** | **≥ 4** | |
| | | | **(No OCC) or Interlace** | **OCC and no Interlace** |
| **UCI message bitlength [bits]** | **≤ 2** | format0 | format1 | |
| | **≥ 3** | format2 | format3 | format4 |

**Recall** from 7.6.5.1 that **PUCCH cannot** have **3 symbol intervals**.

For **semi-persistent scheduling**, the **UE determines** the **format variant** (click 7.6.6.8) to be **used** based on the **configuration** of the **bandwidth part** where this **UCI** transmission takes place.

For more detail on **PUCCH resource format determination**, refer to clause 9.2.2 of TS 38.213.

### 7.6.4.3 PUCCH resource determination for dynamic scheduling

For **dynamic PUCCH scheduling**, the **DCI message field** "***PUCCH resource indicator***" indicates what **PUCCH resource** (click 7.6.5) within the corresponding **PUCCH resource set** (click 7.6.7), should the **UE** use for the **PUCCH transmission** that is **triggered** by the **DCI message**. This **field** has **3 bits**, except in **DCI Format 1_2** where it has **0** to **3 bits**. The value of this **field** is an **index** to a **list** of **PUCCH resources** within the corresponding **PUCCH resource set**. **Field value** "**0**" indicates the **first PUCCH resource** in the **list**. **Field value** "**1**" indicates the **second PUCCH resource** in the **list**, and so on. For example, if the value is "11" or "011", the **fourth PUCCH resource** in the **list** is used by the **UE**. The following table shows, for **PUCCH assignments**, the **DCI message Formats** where the said **field** is **present**, what is **assigned** by **each** of these **DCI Formats** and what is the **field**'s **bitlength** in **each** of these **DCI Formats**:

| | **DCI message field "*PUCCH resource indicator*"** | | |
|---|---|---|---|
| | **Assigns** | **DCI Format** | **Bitlength** |
| **PUCCH scheduling** | **PDSCH Unicast** | Format 1_0 (6.9.5) | 3 |
| | | Format 1_1 (6.9.6) | 3 |
| | | Format 1_2 (6.9.7) | 0 to 3 |
| | **PDSCH Multicast** | Format 4_1 (6.9.9.2) | 3 |
| | | Format 4_2 (6.9.9.3) | 3 |

For the case of **DCI Format 1_2**, the **field bitlength** is indicated by `numberOfBitsForPUCCH-ResourceIndicatorDCI-1-2-r16` from `PUCCH-Config` (click 7.9.4).

For the purpose of determining a **PUCCH resource** for a **PUCCH transmission** in a **slot** using a "***PUCCH resource indicator***" field from **DCI message** that performs a **PDSCH assignment**, and for the purpose of determining the **slot** for the **PUCCH transmission**:

- A **UE** is assumed to **generate HARQ-ACK** (click 4.5.6.7) **information regardless** of **whether** or **not** the **PDSCH** reception provides a **codeword** for a **PDSCH HARQ process** with **disabled HARQ-ACK information** as indicated by `downlinkHARQ-FeedbackDisabled-r17` from `PDSCH-ServingCellConfig` (click 6.18.10) if **present**.
- A **UE** is assumed to **not** generate **HARQ-ACK** information associated with a **G-RNTI** for **multicast** or with a **G-CS-RNTI** with **disabled HARQ-ACK information** as described in clause 18 of TS 38.213.

For **shared spectrum** in **FR1**, **PUCCH** `format0` and `format1` are extended to use resource in **1 PRB interlace**, and **PUCCH** `format2` and `format3` are extended to use **up to 2 PRB interlaces**, in **both** cases within **1 RB set**. **PUCCH** `format2` and `format3` are also **enhanced** to support **multiplexing** of **up to 4 UEs** in the **same PRB interlace** when **1 interlace** is used.

For more detail on **PUCCH resource selection**, refer to clauses 9.2 and 9.2.2 of TS 38.213.

### 7.6.5 PUCCH resources

**Each PUCCH resource** is a **set** of **REs**, in **1** or **more slots**, where **PUCCH transmission(s) may** take place in a **specific way**.

**Each PUCCH resource** can be characterized by its `format` (click 7.6.5.1), by its `format` **variant**, (click 7.6.6.8) and by **some parameters** (e.g., `nrofSymbols`, click 7.6.5.3.3).

The **RE characteristics** of a **PUCCH resource** are:

- The **quantity** "$N_{PUCCH}^{repeat}$" of **consecutive slots** where the **PUCCH transmission** containing **1 UCI message** (click 7.7) is **repeated** (with a couple possible adjustments).
- The **quantity** of **rectangle(s)** of **REs** within **each** and **every** of the said **consecutive slots**.
- The **first symbol interval** of the **rectangle(s)** of **REs** within **each** and **every** of the said **consecutive slots**.
- The **quantity** "$N_{syin}^{PUCCH}$" of **symbol intervals** of the **rectangle(s)** of **REs** within **each** and **every** of the said **consecutive slots**.
- The **quantity** of **CRBs** of the **rectangle(s)** of **REs**, within **each** and **every** of the said **consecutive slots**.
- The **lowest frequency PRB** of **each rectangle of REs** within **each** and **every** of the **consecutive slots**.

- The **content type** (**payload** or **DM-RS**) of **each RE** of the **PUCCH resource**.

  This depends on the **PUCCH resource** `format` (click 7.6.5.1).

- For the case of "$N_{PUCCH}^{repeat}$>1" Whether the **DM-RS** are **recalculated** in **each** and **every repeated slot**, or **rather every few repeated slots**.

  The **second choice** corresponds to the **DM-RS bundling variant** (click 7.6.5.3.7).

The **transmission processing characteristics** of a **PUCCH resource** are:

- The **encoding** used for the **UCI message**.

  This depends on the **PUCCH resource** `format` (click 7.6.5.1) and on the **UCI message bitlength** (click 7.7.2).

- The **modulation** used in the **payload REs**.

  This depends on the **PUCCH resource** `format` (click 7.6.5.1) and on the **UCI message bitlength** (click 7.7.2).

- The **multiple access** technique used in the **REs**.

  This depends on the **PUCCH resource** `format` (click 7.6.5.1).

- Other characteristics (e.g., **orthogonal cover codes**, **spreading** or **multiplexing**).

**Each UE** may be configured up to `56` **PUCCH resources** in the **PUCCH resource sets** (click 7.6.7) of **each** of its **Uplink** (or **SUL**) **bandwidth parts**. These **PUCCH resources** are grouped into **several PUCCH resource sets** (click 7.6.7).

**Several UEs may** be configured the **same PUCCH resource**(**s**) because **PUCCH** supports **complex number sequences** and **orthogonal cover codes** to multiplex the **PUCCH transmissions** of **several UEs** over the **same REs**.

For how a **UE determines** what **PUCCH resource** to use for each **PUCCH transmission** click 7.6.4.

### 7.6.5.1 PUCCH resource `formats` and variants

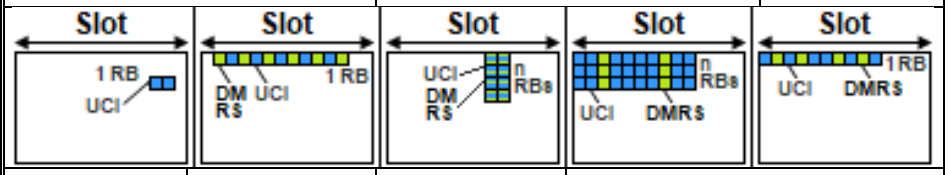
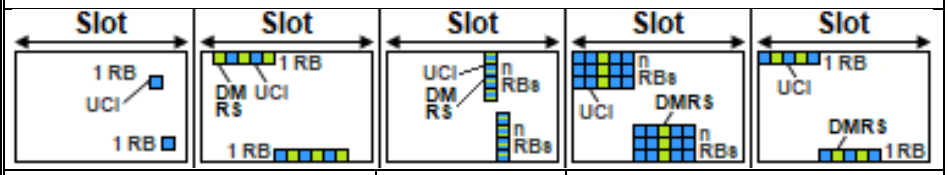
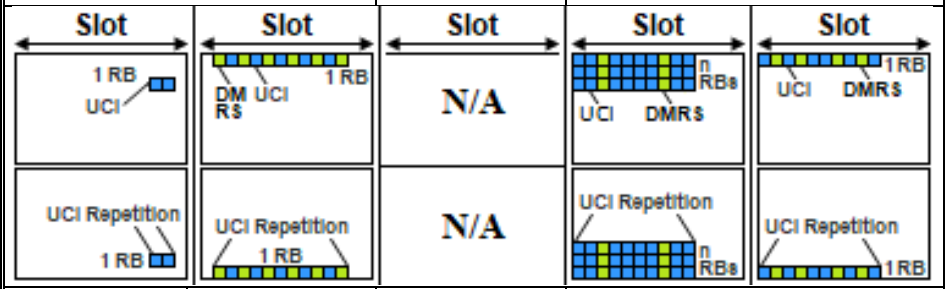
**Each PUCCH resource** `format` is a definition of the **REs** used for a **PUCCH resource** (click 7.6.3). There are **5 PUCCH resource** `formats`, identified as `format0` to `format4`.

**Each PUCCH resource** `format` can be characterized by **quantity** and **location** of the **PUCCH REs**, which of them are **PUCCH DM-RS REs**, the **modulation**(**s**) to use, the **channel coding** and other relevant characteristics.

**Each** of these **5 PUCCH resource** `formats` is **refined** with **1** of the **7 allowed** `format` **variants** (click 7.6.6) that is **applicable** to it. **Each** of the **7 variants** is **only applicable** to **some** of the **5** `formats`. The result is **32 different PUCCH resource types** (click 7.6.6), **each characterized** by a **pair** of (`format` and **allowed variant**). **Each PUCCH transmission conforms** to **1** among the **32 possible types of PUCCH resources**.

The **characteristics** and relevant **parameters** of the **PUCCH scheduling** of its **payload REs** and its **DM-RS REs** for the **5 PUCCH resource** `formats`, its **7 variants** and its **32 types** are **summarized** in the table below.

For a summary table of **characteristics** and relevant **parameters** of the **payload** (i.e., **UCI message**) **processing** for the **5 PUCCH formats**, click 7.8.1.

| | PUCCH resource REs | | | | |
|---|---|---|---|---|---|
| format (7.6.5.3.3) | format0 | format1 | format2 | format3 | format4 |
| format length | Short | Long | Short | Long | |
| Configurable in which PUCCH resource sets | Only in first resource set | | Only in non-first resource set | | |
| startingPRB (7.6.5.3.2) | PRB-Id ::= INTEGER (0..maxNrofPhysicalResourceBlocks-1) | | | | |
| startingSymbolIndex (7.6.5.3.2) | 0 to 13 | 0 to 10 | 0 to 13 | 0 to 10 | |
| nrofSymbols [symbs] (7.6.5.3.3) | 1 or 2 | 4 to 14 | 1 or 2 | 4 to 14 | |
| subslotLengthForPUCCH-r16 [symbs] (7.6.5.3.3) | N/A | NCP: 2 or 7<br>ECP: 2 or 6 | N/A | Normal CP (NCP): 2 or 7<br>Extended CP (ECP): 2 or 6 | |
| Quantity of PRBs | 1 | | N/A | 10, 20 (*) | 1 |
| nrofPRBs (7.6.5.3.3) | N/A | | 1 to 16 | 1 to 16 (**) | N/A |
| Only in FR2-2 nrofPRBs-r17 (7.6.5.3.3) | 1 to 16 | | N/A | | 1 to 16 (**) |
| Examples of PUCCH resources with the basic variant (7.6.6.1) |  | | | | |
| Variant intraSlotFrequencyHopping (7.6.5.3.4) | Only if 2 symbols | Yes | Only if 2 symbols | Yes | |
| secondHopPRB (7.6.5.3.4, 7.6.5.3.6) | PRB-Id ::= INTEGER (0..maxNrofPhysicalResourceBlocks-1) | | | | |
| Examples of PUCCH resources with the intra-slot frequency hopping variant (7.6.6.1) |  | | | | |
| nrofSlots [slots] (7.6.5.3.5) | 2, 4, 8 | | N/A | 2, 4, 8 | |
| pucch-RepetitionNrofSlots-r17 [slots] (7.6.5.3.5) | 1, 2, 4, 8 | | | | |
| Variant interslotFrequencyHopping variant (7.6.5.3.6) | Yes | | N/A | Yes | |
| Variant pucch-DMRS-Bundling-r17 (7.6.5.3.7) | Yes | | N/A | Yes | |
| Examples of PUCCH resources with the inter-slot frequency hopping variant: even slot followed by odd slot (7.6.6.5) |  | | | | |
| Multiplexing of payload and DM-RS | N/A | TDM | FDM | TDM | |

| | PUCCH resource REs | | | | |
|---|---|---|---|---|---|
| format (7.6.5.3.3) | format0 | format1 | format2 | format3 | format4 |
| **PUCCH total [REs]**<br>**(2nd row: only in FR2-2)** | 12 or 24<br>12 to 384 | 48 to 168<br>48 to 2,688 | 12 to 384 | 48 to 3,360<br>48 to 2,688 | 48 to 168<br>48 to 2,688 |
| **Option of additionalDMRS** | N/A | | | Yes | |
| **Payload REs / Total REs** | 1 | 2/5 to 1/2 | 2/3 | 1/2 to 6/7 | |
| **PUCCH payload [REs]**<br>**(2nd row: in FR2-2)** | 12 or 24<br>12 or 384 | 24 to 84<br>24 to 1,344 | 8 to 256 | 24 to 2,880<br>24 to 2,304 | 24 to 144<br>24 to 2,304 |
| **Notes** | No DM-RS | LTE PF1 | | LTE PF4 | LTE PF5 |

Some **relevant remarks** on this table are:

- Row "**Quantity of PRBs**" and column "format3": the "(*)" indicates that **these values** are **only** for **interlaced mapping**. Value **10** is for **1 interlace** and value **20** is for **2 interlaces**.
- Rows "nrofPRBs" and "**Only in FR2-2** nrofPRBs-r17": the "(**)" indicates that **these 2 cells only** apply to **non-interlaced mapping** and that **not all values** from 1 to 16 can be configured. **Only** the **following values** may be configured:

  $$M_{RB}^{PUCCH,3,4} = 2^{\alpha_2} \bullet 3^{\alpha_3} \bullet 5^{\alpha_5} = \{1, 2, 3, 4, 5, 6, 8, 9, 10, 12\}$$

  where "$\alpha_2$", "$\alpha_3$" and "$\alpha_5$" are **non-negative integers**, and the **assignment** is done according to clauses 9.2.3, 9.2.5.1 and 9.2.5.2 of TS 38.213. For the case of "$M_{RB}^{PUCCH,4}$", it is given by clause 9.2.1 of TS 38.213.
- Row "**Configurable in which PUCCH resource sets**" indicates, for **each PUCCH resource format**, in **which PUCCH resource sets** it **can** be **configured**.
- Three rows "**Examples of PUCCH resource…**":
  - For **common remarks** on these **PUCCH resource examples**, click 7.6.6.8.
  - For more detail on the examples of **PUCCH resource** with **basic variant** (i.e., **no intra-slot frequency hopping**, **no inter-slot repeat** (ergo **no inters-slot frequency hopping** (ergo **no DM-RS bundling**)) and **no interlace**), click 7.6.6.1. These **same patterns** are **repeated** in **several consecutive slots** if **inter-slot repeat** is additionally **set** (with **no inter-slot frequency hopping** (ergo **no DMRS bundling**)): for more detail, click 7.6.6.2.
  - For more detail on the examples of **PUCCH resource** with **intra-slot frequency hopping** only (i.e., **no inter-slot frequency hopping** (ergo **no DMRS bundling**)) and **no interlace**), click 7.6.6.2. These **same patterns** are **repeated** in **several consecutive slots** if **inter-slot repeat** is additionally **set**: for more detail, click 7.6.6.4.
  - For more detail on the examples of **PUCCH resource** with **inter-slot frequency hopping** (ergo **no intra-slot frequency hopping** and **inter-slot repeat**), **no DM-RS bundling** and **no interlace**, click 7.6.6.5.
  - The examples for format3 are **without additional DM-RS** (click 7.6.8), while the examples for format4 are **with additional DM-RS**.

- Row "**Payload and DM-RS Multiplexing**": in **PUCCH** `format1`, `format3` and `format4`, the **payload REs** and the **DM-RS REs** are **time multiplexed** to have a **low peak-to-average power-ratio** (**PAPR**).
- Row "**PUCCH total REs**": is the **range** of the **total quantity** of **REs** that is **scheduled** for **each PUCCH transmission**.
- Row "**PUCCH payload REs**": is the **range** of the **total quantity** of **payload REs** (i.e., **PUCCH total REs minus PUCCH DM-RS REs**) that may be used in **each PUCCH transmission**.

For the **processing** of the **payload REs** of the **5 PUCCH resource** `formats` click 7.8.1.

For how a **UE determines** what **PUCCH resource** to use for each **PUCCH transmission** click 7.6.4.

For more detail on **PUCCH** `formats`, refer to clauses 6.3.2 and 6.4.1.3 of TS 38.211, clauses 9.2.1 and 9.2.2 of TS 38.213, and clause 5.3.3 of TS 38.300.

The following two web pages contain **several examples** of **PUCCH** `formats` that are **very illustrative**:

- https://www.sharetechnote.com/html/lte_toolbox/Matlab_LteToolbox_5G_PUCCH0.html
- https://www.sharetechnote.com/html/lte_toolbox/Matlab_LteToolbox_5G_FrameStructure_UL.html#Format_0_Ex01

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_PUCCH.html
- https://www.sharetechnote.com/html/5G/5G_UCI.html
- https://www.rfwireless-world.com/5G/5G-NR-UCI.html

### 7.6.5.2 PUCCH resource common configuration

***If*** a **UE** does **not** have its **dedicated PUCCH resource configuration** (click 7.6.5.3) provided by a `PUCCH-ResourceSet` (click 7.9.6) **instance** from `PUCCH-Config` (click 7.9.4), ***then***, a **PUCCH resource set** is provided by `pucch-ResourceCommon` from `PUCCH-ConfigCommon` (click 7.9.3) for transmission of **HARQ-ACKs** in an

**initial Uplink bandwidth part** of "$N_{RB}^{BWP}$" **PRBs**. The value of `pucch-ResourceCommon` is an **index** to a **row** of table 9.2.1-1 of TS 38.213:

| Index | PUCCH format | First symbol interval | Symbol intervals | PRB offset | Set of initial CS indexes |
|---|---|---|---|---|---|
| **0** | 0 | 12 | 2 | 0 | {0, 3} |
| **1** | 0 | 12 | 2 | 0 | {0, 4, 8} |
| **2** | 0 | 12 | 2 | 3 | {0, 4, 8} |
| **3** | 1 | 10 | 4 | 0 | {0, 6} |
| **4** | 1 | 10 | 4 | 0 | {0, 3, 6, 9} |
| **5** | 1 | 10 | 4 | 2 | {0, 3, 6, 9} |
| **6** | 1 | 10 | 4 | 4 | {0, 3, 6, 9} |
| **7** | 1 | 4 | 10 | 0 | {0, 6} |
| **8** | 1 | 4 | 10 | 0 | {0, 3, 6, 9} |
| **9** | 1 | 4 | 10 | 2 | {0, 3, 6, 9} |
| **10** | 1 | 4 | 10 | 4 | {0, 3, 6, 9} |
| **11** | 1 | 0 | 14 | 0 | {0, 6} |
| **12** | 1 | 0 | 14 | 0 | {0, 3, 6, 9} |
| **13** | 1 | 0 | 14 | 2 | {0, 3, 6, 9} |
| **14** | 1 | 0 | 14 | 4 | {0, 3, 6, 9} |
| **15** | 1 | 0 | 14 | $floor(N_{RB}^{BWP} / 4)$ | {0, 3, 6, 9} |

For **FR2-2 bands**, the **quantity** "$N_{RB}$" of **RBs** for the **PUCCH resource set** is provided, if **present**, by `nrofPRBs` from `PUCCH-ConfigCommon` (click 7.9.3). If it is **absent**, "$N_{RB}$ =1".

This **PUCCH resource set** includes **16 resources**, **each** with its **PUCCH** `format`, the **slot index** of its **first symbol interval**, its **duration** in **symbol intervals**, its **PRB offset** and its **cyclic shift index set**.

The **UE** uses **frequency hopping** if `useInterlacePUCCH-PUSCH` from from `BWP-UplinkCommon` (click 2.2.8.9) is **absent**. If **present**, the **UE** does **not** use **frequency hopping**.

An **orthogonal cover code** with **index 0** is used for a **PUCCH resource** with `format1` in Table 9.2.1-1, **except** for table **index** values **3**, **7**, or **11** when `useInterlacePUCCH-PUSCH` from `BWP-UplinkCommon` (click 2.2.8.9) is **present**.

The **UE** transmits the **PUCCH** using the **same spatial domain transmission filter** as for a **PUSCH transmission scheduled** by an **Uplink grant** for **Msg2** (**Random Access Response**, refer to clause 8.3 of TS 38.213).

***If*** a **UE** is **not** provided **any** of `pdsch-HARQ-ACK-Codebook`, `pdsch-HARQ-ACK-Codebook-r16`, or `pdsch-HARQ-ACK-OneShotFeedback`, ***then***, the **UE** generates **at most 1 HARQ-ACK information bit**.

For how the **UE determines** what **PUCCH resource** to use for a **specific PUCCH** transmission click 7.6.3.

For more detail on the **PUCCH resource common configuration**, refer to clause 9.2.1 of TS 38.213.

### 7.6.5.3 PUCCH resource dedicated configuration and options

It is possible to **configure** up to **56** **PUCCH resources** in the **PUCCH resource sets** (click 7.6.7) of **each** of its **Uplink** (or **SUL**) **bandwidth part**. **Each PUCCH resource must** belong to **1** of the **32 PUCCH resource types** (click 7.6.6.8). **Even** if **two different PUCCH resources** are of the **same type**, they **differ** in the **values** of some of their **parameters**, that are presented in this section.

**Each UE** may have its **own specific PUCCH configuration** for **each** of its **Uplink** (or **SUL**) **bandwidth parts**. **Up to 2 PUCCH configurations** can be configured for a **UE** per **PUCCH group**, where the **first PUCCH configuration** is associated with a **PUCCH** of **priority** index **0** (**low**) and the **second PUCCH configuration** is associated with a **PUCCH** of **priority** index **1** (**high**).

The **PUCCH resources** may have **some combinations** of the following **5 dedicated configuration options**:

- **Intra-slot frequency hopping option** (click 7.6.5.3.4).
- **Inter-slot repeat option** (click 7.6.5.3.5).
- **Inter-slot frequency hopping option** (click 7.6.5.3.6).
- **DM-RS bundling hopping option** (click 7.6.5.3.7).
- **Interlace option** (click 7.6.5.3.8).

For how the **UE determines** what **PUCCH resource** to use for a **specific PUCCH** transmission click 7.6.3.

For more detail on the **PUCCH resource dedicated configuration**, refer to clause 9.2.1 of TS 38.213.

For more detail on the allowed combinations of **PUCCH resource parameters**, refer to TS 38.331, and to the text in clause 9.2.1 of TS 38.213, starting in the sentence "*If a UE has dedicated PUCCH resource configuration, the UE is provided by higher layers with one or more PUCCH resources.*" in page 109 and ending right before the sentence "*If the format indicates PUCCH-format0…*" in page 110.

#### 7.6.5.3.1 All the PUCCH resources of each bandwidth part

**All** the **PUCCH resources** of **each Uplink** (or **SUL**) **bandwidth part** are configured within **1 instance** of **PUCCH-Config** (click 7.9.4), as follows:

- **Each instance** of **PUCCH-Config** contains **resourceToAddModList**, which is a **SEQUENCE OF** up to **128** **nameless** parameters, **each** of which is an **instance** of **PUCCH-Resource** (click 7.9.7). **Each such instance** of **PUCCH-Resource** is a **definition** of **1 PUCCH resource**, that may be **extended** with other parameters. Even though there are **128** elements in the **SEQUENCE OF**, the **PUCCH resource sets** (click 7.6.7) of **each** of the **UE**'s **Uplink** (or **SUL**) **bandwidth part** may only contain up to **56** **PUCCH resources**.

**Any PUCCH resource** may be **extended** with the **parameters** from **1 instance** of **PUCCH-ResourceExt-v1610** (click 7.9.9), as follows:

- **Each instance** of **PUCCH-Config** (click 7.9.4) **may** contain **resourceToAddModListExt-v1610**, which is a **SEQUENCE OF** up to **128** **nameless** parameters, **each** of which is an **instance** of **PUCCH-ResourceExt-**

v1610. **Each such instance** of PUCCH-ResourceExt-v1610 **extends** the configuration of the **PUCCH resource** that is in its **same sequential position** within resourceToAddModList.

For the **configuration** of **each** and **every PUCCH resource**, see the **next subsection**.

#### 7.6.5.3.2 Each and every PUCCH resource

**Each** and **every PUCCH resource** is **configured** with **1 instance** of PUCCH-resource (click 7.9.7), plus some **other data type instances**. **Each** and **every instance** of PUCCH-Resource **instance** has the following **2 mandatory parameters** from its (**plus** the format **mandatory parameter**, click 7.6.5.3.3):

- pucch-ResourceID

  An **integer** that **uniquely identifies each PUCCH resource** that has been configured within **each Uplink** (or **SUL**) **bandwidth part**.

- startingPRB

  It is the **PRB index** of the **lowest frequency PRB** of the **PUCCH resource**.

The **2** above **generic parameters**, plus the format-**specific** ones from 7.6.5.3.3, configure the **basic variant** (click 7.6.6.1).

#### 7.6.5.3.3 Every PUCCH resource with the same format

**Every PUCCH resource** with the **same** format is **configured** with **1 value** of **parameter** format from its **instance** of PUCCH-resource (click 7.9.7). The format parameter is a **CHOICE** of **1** among the **5 parameters** format0, format1, format2, format3 or format4. The **chosen parameter** among format0 to format4 determines the format of the corresponding **PUCCH resource**. The **data types** of the said **5 parameters** are, respectively, PUCCH-format0, PUCCH-format1, PUCCH-format2, PUCCH-format3 or PUCCH-format4 (click 7.9.8). The **instance** of the **data type** of the **chosen parameter** configures the corresponding format.

Notice that the **4 parameters** format1, format2, format3 and format4 have the **same name** as **4 other parameters** from PUCCH-Config (click 7.9.4), that are described a few paragraphs below. In spite of having the **same name**, they are **different parameters**, from a **different data type**, having **different data types**, and with **different semantics**.

The **data type instance** (among PUCCH-format0 to PUCCH-format4, click 7.9.8) of the **chosen** parameter (among format0 to format4, respectively) **adds** the following **parameters** to the corresponding **PUCCH resource**:

- To a **PUCCH resource** of **any** format:
  - startingSymbolIndex

    It is the **symbol interval slot index** where the **first symbol** of the **PUCCH resource** is located. In case subslotLengthForPUCCH-r16 from PUCCH-Config (click 7.9.4) is **present**, the **symbol interval slot index** is rather:

    startingSymbolIndex ***mod*** subslotLengthForPUCCH-r16

- `nrofSymbols`

  It is the **quantity** of **symbol intervals** of the **PUCCH resource** in `format1`, `format3` and `format4` (i.e., **long** `formats`). In case `subslotLengthForPUCCH-r16` from `PUCCH-Config` (click 7.9.4) is **present**, the **quantity** of **symbol intervals** is rather `subslotLengthForPUCCH-r16`.

- `nrofPRBs`

  It is the **quantity** of **PRBs** of the **PUCCH resource**. This parameter is **mandatory** in **every PUCCH resource** with `format2` or `format3`.

  For **FR2-2 only**, **any PUCCH resource** with `format0`, `format1` or `format4` **may** be **extended** with `nrofPRBs-r17` from `PUCCH-ResourceExt-v1610` (click 7.9.9).

- To a **PUCCH resource** of `format0` or `format2` (**short** `formats`):
  - `initialCyclicShift`
- To a **PUCCH resource** of `format0`:
  - `timeDomainOCC`

    It is **assumed** that it is an **index** for an **orthogonal cover code** of the **PUCCH resource**.

**Each** and **every PUCCH resource** (of a given **bandwidth part**) that has the **same** `format` **shares** the **same values** of a few **parameters** from a `PUCCH-FormatConfig` (click 7.9.10) **instance** and/or from a `PUCCH-FormatConfigExt-r17` (click 7.9.11) **instance**. These **shared values** are configured, for **each** of the **5** `formats`, with the following parameters from the `PUCCH-Config` (click 7.9.4) **instance** of the **bandwidth part**: `format0-r17`, `format1`, `format2`, `format3`, and `format4`, with **data type** `PUCCH-FormatConfig`, and `format2Ext-r17`, `format3Ext-r17` and `format4Ext-r17`, with **data type** `PUCCH-FormatConfigExt-r17`.

Notice that these **4 parameters** `format1`, `format2`, `format3` and `format4` have the **same name** as the **4 other parameters** in the `format` `CHOICE` from `PUCCH-resource` (click 7.9.7) that have been described a few paragraphs above. In spite of having the **same name**, they are **different parameters**, from a **different data type**, having **different data types**, and with **different semantics**.

The **parameters** whose **values** are **shared** by **each** and **every PUCCH resource** (of a given **bandwidth part**) with the **corresponding** `format` are:

- `maxCodeRate`, from the `PUCCH-FormatConfig` (click 7.9.10) **instance** of `format2`, `format3` or `format4`.

  It is **only applicable** to **PUCCH resources** with `format2`, `format3` or `format4`.

- `maxCodeRateLP-r17`, from the `PUCCH-FormatConfigExt-r17` (click 7.9.11) **instance** of `format2Ext-r17`, `format3Ext-r17` or `format4Ext-r17`.

  It is a **second maximum code rate**, in **addition** to `maxCodeRate`.

It is **only applicable** to **PUCCH resources** with `format2`, `format3` or `format4`, and, **only** when `uci-MuxWithDiffPrio-r17` from `PhysicalCellGroupConfig` (click 3.7.5) is **present**.

- `pi2BPSK`, from the `PUCCH-FormatConfig` (click 7.9.10) **instance** of `format3` or `format4`.

  It is **only applicable** to **PUCCH resources** with `format3` or `format4`.

- `simultaneousHARQ-ACK-CSI`, from **the** `PUCCH-FormatConfig` (click 7.9.10) **instance** of `format2`, `format3` or `format4`.

  It is **only** applicable to **PUCCH resources** with `format2`, `format3` or `format4`.

The above **parameters**, plus the **2 generic** ones from 7.6.5.3.2, configure the **basic variant** (click 7.6.6.1).

Recall from 5.9.7 that **all** the **parameters** from **any ASN.1 data type** **may** be present in **any** of its **instances**. Therefore, the `format` **restrictions** indicated in the above bullet points do **not** arise from the **RRC ASN.1 specification**, but **rather** from the **specific technical** characteristics of **each** `format`.

#### 7.6.5.3.4 Intra-slot frequency hopping option for each PUCCH resource

**Each PUCCH resource** is **individually** configured with the **intra-slot frequency hopping option**, when the **2** following **parameters** are **present** in its `PUCCH-Resource` (click 7.9.7) **instance**:

- `intraslotFrequencyHopping`
- `secondHopPRB`

The **intra-slot frequency hopping option** is applicable to `format1` to `format4`, and also to `format0` if it has **2 symbol intervals** (i.e., when its `nrofSymbols`=2).

The **intra-slot frequency hopping option** is **required** for the **intra-slot frequency hopping variant** (click 7.6.6.3) and for the **repeated intra-slot frequency hopping variant** (click 7.6.6.4).

#### 7.6.5.3.5 Inter-slot repeat option for every or each PUCCH resource

The **inter-slot repeat option** (click 7.6.6.5) is **set** in the **3** following cases:

- In **each** and **every PUCCH resource** (of a given **bandwidth part**) that has the **same** `format` when `nrofSlots` is **present** in the `PUCCH-FormatConfig` (click 7.9.10) **instance** of **its** `format`. In this case, the **quantity** "$N_{PUCCH}^{repeat}$" of **consecutive slots** is the value of `nrofSlots`, and the **slot symbol index** of the **first PUCCH symbol interval** in **each slot** is the value of `startingSymbolIndex`.

- In **each PUCCH resource** where `subslotLengthForPUCCH-R16` is **present** in the `PUCCH-FormatConfig` (click 7.9.10) **instance** of **its** `format`. In this case, the **quantity** "$N_{PUCCH}^{repeat}$" of **consecutive slots** is the value of `nrofSlots`, and the **slot symbol index** of the **first PUCCH symbol interval** in **each slot** is the value of `startingSymbolIndex` ***mod*** `subslotLengthForPUCCH-r16`

- In a specific **PUCCH transmission** when it is **dynamically scheduled** by a **DCI message** (click 6.9) and `pucch-RepetitionNrofSlots-r17` from `PUCCH-`

> `ResourceExt-v1610` (click 7.9.9) is **larger** than `1` in the configuration of the **PUCCH resource** of the **PUCCH transmission**. In this case, the **quantity** "$N_{PUCCH}^{repeat}$" of **consecutive slots** is the value of `pucch-RepetitionNrofSlots-r17`, and the **slot symbol index** of the **first PUCCH symbol interval** in **each slot** is the value of `startingSymbolIndex`.

The parameter `startingSymbolIndex` is from the corresponding `PUCCH-format0` to `PUCCH-format4` (click 7.9.8) **data type**, and its configuration is described in 7.6.5.3.2.

The **inter-slot repeat option** is **required** for the **repeated basic variant** (click 7.6.6.2), for the **repeated intra-slot frequency hopping variant** (click 7.6.6.4), for the **inter-slot frequency hopping variant** (click 7.6.6.5) and for the **DM-RS bundling variant** (click 7.6.6.6).

For more detail on how the **UE** determines the value of "$N_{PUCCH}^{repeat}$", refer to the last part of clause 9.2.6 of TS 38.213, starting in the sentence "*For unpaired spectrum, the UE determines the* $N_{PUCCH}^{repeat}$ *slots…*".

For more detail on the **PUCCH repetition procedure**, refer to clause 9.2.6 of TS 38.213.

#### 7.6.5.3.6 Inter-slot frequency hopping option: every PUCCH resource with the same `format`

**Each** and **every PUCCH resource** (of a given **bandwidth part**) that has the **same** `format` is **configured** with the **inter-slot frequency hopping option** when the **inter-slot repeat variant** is **option** (click 7.6.5.3.5) and `interslotFrequencyHopping` is **present** in the `PUCCH-FormatConfig` (click 7.9.10) **instance** of **its** `format`.

In this case, the **frequency location** of the **second hop** is the value of `secondHopPRB` within the `PUCCH-Resource` (click 7.9.7) **instance** of the **PUCCH resource**.

The **inter-slot repeat option** is **required** for the **inter-slot frequency hopping variant** (click 7.6.6.5) and for the **DM-RS bundling variant** (click 7.6.6.6).

#### 7.6.5.3.7 DM-RS bundling option: every PUCCH resource

**Every PUCCH resource** is configured with the **DM-RS bundling** when the **inter-slot frequency hopping option** is **set** (click 7.6.5.3.6) in the **PUCCH resource** and `pucch-DMRS-Bundling-r17` from `DMRS-BundlingPUCCH-Config-r17` (click 7.9.16), within `PUCCH-Config` (click 7.9.4), is **present**.

The **DM-RS bundling option** is **required** for the **DM-RS bundling variant** (click 7.6.6.6).

#### 7.6.5.3.8 Interlace option: each and every PUCCH resource in each cell

**Each and every PUCCH resource** in a given **cell** are **configured** with the **interlace option** when **PUCCH-PUSCH interlace** (click 2.2.4.8) is **set** in that given **cell.**

**PUCCH-PUSCH interlace** is **either set** or **not set** in **all Uplink** (or **SUL**) **bandwidth parts** of the **same serving cell**. Consequently, the **same PUCCH-PUSCH interlace configuration** applies to **all** the **PUCCH transmissions** (i.e., to **all** the **PUCCH resources**, click 7.6.5.3) and to **all** the **PUSCH transmissions** (click 7.11.3) in **each serving cell**.

**Setting** the **interlace option** implies that the following **parameters** from the corresponding **instance** of `PUCCH-format0` to `PUCCH-format4` (click 7.9.8) are **added** to **all** the **PUCCH resources** of the corresponding `format`:

- `occ-Length`

  It is the **orthogonal cover code length** of the **PUCCH resource**. It **only** applies to `format4` (it only exists in `PUCCH-format4`).

  **All** `formats` may be **extended** to include the **additional** parameter `occ-Length-v1610` from `PUCCH-ResourceExt-v1610` (click 7.9.9) that has the **same semantics** as `occ-Length` and may take its **same values**.

- `occ-Index`

  It indicates the **orthogonal cover code index** of the **PUCCH resource**. It **only** applies to `format4` (it only exists in `PUCCH-format4`). For more detail, refer to clause 9.2.1 of TS 38.213.

  **All** `formats` may be **extended** to include the **additional** parameter `occ-Index-v1610` from `PUCCH-ResourceExt-v1610` (click 7.9.9) that has the **same semantics** as `occ-Index` and may take its **same values**.

**Setting** the **interlace option** implies that the following **parameters** from `PUCCH-ResourceExt-v1610` (click 7.9.9) may be **added** to **each** and **every PUCCH resource**:

- `rb-setIndex-r16`
- `interlace0-r16`
- `interlace1-v1610`

### 7.6.6 PUCCH resource `format` variants and types

**Each** of the **7 PUCCH resource** `format` **variants** is a **refinement** of the **characteristics** of **each** the **5 PUCCH resource** `formats` (click 7.6.5.1). **Each PUCCH resource** `format` **variant** is a **valid combination** of the **5 PUCCH resource** `format` **options** (click 7.6.5.3).

The **following restrictions** apply to the **combined** configuration of `format` **options**:

- The **intra-slot frequency hopping option** and the **inter-slot frequency hopping option** are **mutually exclusive** (refer to clause 9.2.6 of TS 38.213).
- The **inter-slot repeat option** is **required** for the **inter-slot frequency hopping option**.
- The **inter-slot frequency hopping option** is **required** for the **DM-RS bundling option**.
- The **interlace option** is **mutually exclusive** with **any** of the **other options**.

**Each** of the **7** `format` **variants** is **applicable** to **most**, but **not all**, `formats`. The result is **32 different PUCCH resource types**. **Each PUCCH resource type** can be defined as a pair of (`format` and **applicable variant**). **Each PUCCH transmission conforms** to **1** among the **32 possible PUCCH resource types**.

The following table lists the **7** `format` **variants** (1st column), indicating for **each** of them: a) a **link** (between parentheses) to its **specific subsection**; b) the **PUCCH resource**

**formats** to which it is applicable (2nd column); c) its **combination** of **0** or **more** **format** **options** (last 5 columns). The table shows the **32 different PUCCH resource types** (**5 types** in **each** of the **first 4 rows** and **4 types** in **each** the **last 3 rows**). The table is:

| | | **format options** | | | | |
|---|---|---|---|---|---|---|
| **format variant** | **Applicable to formats** (7.6.5.1) | **IntraFH** (7.6.5.3.4) | **Repeat** (7.6.5.3.5) | **InterFH** (7.6.5.3.6) | **DM-RS bundling** (7.6.5.3.7) | **Interlace** (7.6.5.3.8) |
| **Basic** (7.6.6.1) | All (0 to 4) | No | | | | |
| **Repeated basic** (7.6.6.2) | 1, 3, 4<br>2 sis=> 0, 2 | Yes | | No | | |
| **Intra-slot frequency hopping** (7.6.6.2) | 1, 3, 4<br>2 sis=> 0, 2 | Yes | No | | | |
| **Repeated intra-slot frequency hopping** (7.6.6.4) | 1, 3, 4<br>2 sis=> 0, 2 | Yes | | No | | |
| **Inter-slot frequency hopping** (7.6.6.5) | 0, 1, 3, 4 | No | Yes | | No | |
| **DM-RS bundling** (7.6.6.7) | 0, 1, 3, 4 | No | Yes | | | No |
| **Interlace** (7.6.6.7) | 0 to 3 | No | | | | Yes |

Some **relevant remarks** on this table are:

- "sis" stands for "**symbol intervals**".
- "**IntraFH**" stands for the "**Intra-slot frequency hopping**" option.
- "**Repeat**" stands for the "**Inter-slot repeat**" option.
- "**InterFH**" stands for the "**Inter-slot frequency hopping**" option.

For a description of the **5** **formats**, click 7.6.5.1.

For how the **UE determines** what **PUCCH resource** to use for a **specific PUCCH transmission** click 7.6.3.

These **7 allowed combinations** of **format** **options** (**each** indicated in **bold font**) that correspond to the **7** **format** **variants** are **presented** in a **hierarchical** way as follows:

1. Interlace option not set

   This case has the following **2** subcases:

   1.1. Inter-slot repeat option not set

   This case has the following **2 variants**:

   **1.1.1. Basic variant** (click 7.6.6.1)

   This **variant** has **no options**.

   This **variant** does **not** have the **intra-slot frequency hopping option**, **nor** the **inter-slot repeat option**, **nor the interlace option**, because **none** of them has been **set**. This **variant** does **not** have the **inter-slot frequency hopping option** because this **option requires** the **inter-slot repeat option**. This **variant** does **not** have the **DM-RS bundling option** because this **option** requires the **inter-slot frequency hopping option**.

**1.1.2. Intra-slot frequency hopping variant** (click 7.6.6.2)

This **variant only** has the **intra-slot frequency hopping option**.

This **variant** does **not** have the **inter-slot repeat option** because this **option** has **not** been **set**. This **variant** does **not** have the **inter-slot frequency hopping option** because this **option requires** the **inter-slot repeat option**. This **variant** does **not** have the **DM-RS bundling option** because this **option** requires the **inter-slot frequency hopping option**. This **variant** does **not** have the **interlace option** because this **option** is **mutually exclusive** with **any** of the **other options**.

1.2. Inter-slot repeat option is set

This case has the following **4 variants**:

**1.2.1. Repeated basic variant** (click 7.6.6.2)

This **variant only** has the **inter-slot repeat option**.

This **variant** does **not** have the **intra-slot frequency hopping option**, **nor** the **inter-slot frequency hopping option** because they have **not** been **set**. This **variant** does **not** have the **DM-RS bundling option** because this option requires the **inter-slot frequency hopping** option. It does **not** have the **interlace option** because **interlace** is **mutually exclusive** with **any** of the **other variants**.

**1.2.2. Repeated intra-slot frequency hopping variant** (click 7.6.6.4)

This **variant** has the **intra-slot frequency hopping option** and the **inter-slot repeat option**.

This **variant** does **not** have the **inter-slot frequency hopping option** because it has **not** been **set**. This **variant** has **no DM-RS bundling** because this **option** requires **inter-slot frequency hopping**. It does **not** have the **interlace option** because this option is **mutually exclusive** with **any** of the **other options**.

1.2.3. Inter-slot repeat variant and inter-slot frequency hopping options are set

This case has the following **2 variants**:

**1.2.3.1 Inter-slot frequency hopping variant** (click 7.6.6.5)

This **variant** has the **inter-slot frequency hopping option** (plus the **inter-slot repeat option**, because it is **required** for **inter-slot frequency hopping**).

This **variant** does **not** have the **DM-RS bundling** option because it has **not** been **set**. This **variant** does **not** have the has **no intra-slot frequency hopping** because it is **mutually exclusive** with **inter-slot frequency hopping**. This **variant** does **not** have the **interlace option** because this **option** is **mutually exclusive** with **any** of the **other options**.

**1.2.3.2 DM-RS bundling variant** (click 7.6.6.6)

This **variant** has the **DM-RS bundling option** (plus the **inter-slot frequency hopping option**, because it is required for **DM-RS**

**bundling**, plus the **inter-slot repeat option**, because it is **required** for **inter-slot frequency hopping**).

This **variant** has **no intra-slot frequency hopping** because it is **mutually exclusive** with **inter-slot frequency hopping**. It has **no interlace** because **interlace** is **mutually exclusive** with **any** of the **other options**.

2. **Interlace variant** (click 7.6.6.8)

   This **variant only** has the **interlace option**.

   This **variant** does **not** have **any other option** because the **interlace option** is **mutually exclusive** with **any** of the **other options**.

**Each** of the following **7 subsections** presents the **shape**, **size** and **location** of a **PUCCH resource** for the corresponding **`format` variant** and for **each PUCCH resource `format`** to which the **variant** is applicable.

For more detail on the **allowed combinations** of **PUCCH resource options**, refer to TS 38.331, and to the text in clause 9.2.1 of TS 38.213, starting in the sentence "*If a UE has dedicated PUCCH resource configuration, the UE is provided by higher layers with one or more PUCCH resources.*" in page 109 and ending right before the sentence "*If the format indicates PUCCH-format0…*" in page 110.

Source web pages:

- https://www.rfwireless-world.com/5G/5G-NR-UCI.html
- https://www.sharetechnote.com/html/5G/5G_PUCCH.html

### 7.6.6.1 Basic variant

The **RE characteristics** (click 7.6.3) of a **PUCCH resource** with the **basic variant** are:

- Its "$N_{PUCCH}^{repeat}$" is **1 slot**.
- It has **1 rectangle** of **REs** within the **slot**.
- The **quantity** of **CRBs** of the **rectangle** of **REs** within the **slot** corresponds to:
  - **`nrofPRBs`** or **`nrofPRBs-r17`**
- The **first symbol interval** of the **rectangle** of **REs** within the **slot** corresponds to:
  - **`startingSymbolIndex`** or:

    **`startingSymbolIndex`** ***mod*** **`subslotLengthForPUCCH-r16`**
- The **quantity** "$N_{syin}^{PUCCH}$" of **symbol intervals** of the **rectangle** of **REs** within the **slot** corresponds to:
  - **`nrofSymbols`** or **`subslotLengthForPUCCH-r16`**

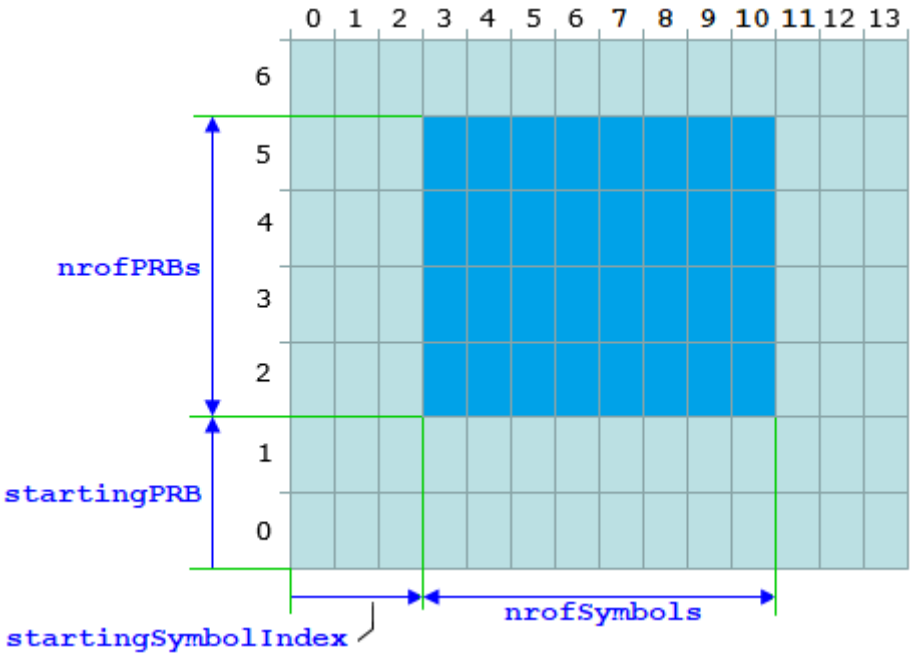
  For more detail on the above parameters, click 7.6.5.3.3.
- The **CRB index** of the **lowest frequency CRB** of the **rectangle** of **REs** within the **slot** corresponds to:
  - **`startingPRB`**

  For more detail on this parameter, click 7.6.5.3.2.

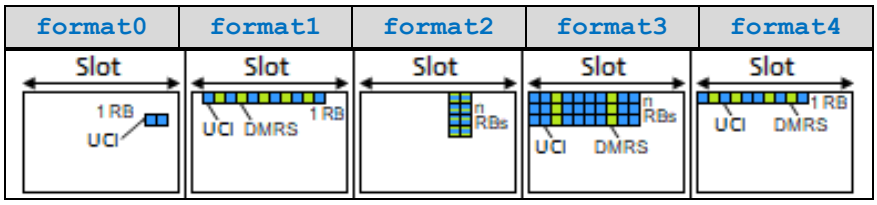
- The **content type** (**payload** or **DM-RS**) of **each RE** of the **PUCCH resource** depends on the **PUCCH resource** `format` (click 7.6.5.1).

For the `format` **options** that characterize the **basic variant**, click 7.6.6.

The following figure shows (as a blue rectangle) an example of this case of **PUCCH resource** defined by `format=format3`, `startingPRB=2`, `nrofPRBs=4`, `startingSymbolIndex=3` and `nrofSymbols=8`




The following table shows examples of the **basic variant** for its applicable **PUCCH** `formats` (**all PUCCH** `formats`):




For **common remarks** on this table and on other similar ones, click 7.6.6.8.

For **two summary tables** of **PUCCH resource** `formats`, their **variants** and their **parameters**, click 7.6.5.1.

### 7.6.6.2 Repeated basic variant

The **repeated basic variant** consists of **repeating** the **PUCCH resource** of the **basic variant** over "$N_{PUCCH}^{repeat}$" **consecutive slots**.

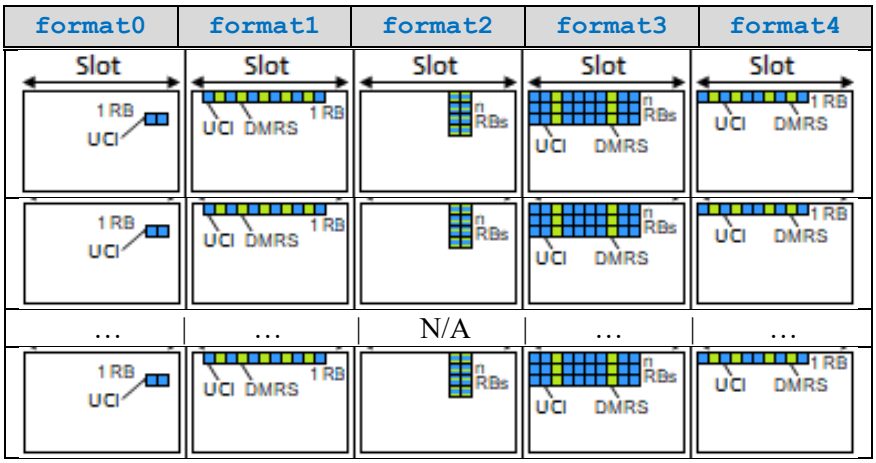
The **RE characteristics** (click 7.6.3) of a **PUCCH resource** with the **repeated basic variant** are:

- "$N_{PUCCH}^{repeat}$" is determined by the **inter-slot repeat option** (click 7.6.5.3.5). It is an **even number** within a **certain range**. "$N_{PUCCH}^{repeat}$" is the **same** for **each** and **every PUCCH resource** (within the corresponding **bandwidth part**) that has the **same** `format`.

- It has **1 rectangle** of **REs** within **each** and **every repeated slot**, and the rectangle's **size**, **shape** and **location** are determined by **exactly** the **same parameters** as in the **basic variant** (click 7.6.6.1).
- The **content type** (**payload** or **DM-RS**) of **each RE** of the **PUCCH resource** depends on the **PUCCH resource** `format` (click 7.6.5.1).
- The **DM-RS** are **recalculated** in **each** and **every repeated slot**.

For the `format` **options** that characterize the **repeated basic variant**, click 7.6.6.

The following table shows examples of the **repeated basic variant** for its **applicable PUCCH** `formats` (**all PUCCH** `formats`):




For **common remarks** on this table and on other similar ones, click 7.6.6.8.

For **two summary tables** of **PUCCH resource** `formats`, their **variants** and their **parameters**, click 7.6.5.1.

### 7.6.6.3 Intra-slot frequency hopping variant

The **intra-slot frequency hopping variant** consists of **splitting** the **PUCCH resource** of the **basic variant** (click 7.6.6.1) in **2 consecutive sequences** of **symbol intervals** and **placing each** of these **2 sequences** in a **different frequency location** within the **slot**.

The **RE characteristics** (click 7.6.3) of a **PUCCH resource** with the **intra-slot frequency hopping variant** are:

- Its "$N_{PUCCH}^{repeat}$" is **1 slot**.
- It has **2 rectangles** of **REs** within the **slot**.
- The **quantity** of **CRBs** of **each** of the **2 rectangles** of **REs** within the **slot** corresponds to:
  - `nrofPRBs` or `nrofPRBs-r17`
- For the **first rectangle** of **REs**, the **first symbol interval** within the **slot** corresponds to:
  - `startingSymbolIndex` or:

    `startingSymbolIndex` ***mod*** `subslotLengthForPUCCH-r16`

- For the **first rectangle** of **REs**, the **quantity** "$S_1$" of **symbol intervals** within the **slot** is "***floor***($N_{syin}^{PUCCH}$/2). The value of "$N_{syin}^{PUCCH}$" is the **same** as in the **basic variant**, which corresponds to:
  - `nrofSymbols` or `subslotLengthForPUCCH-r16`

  For more detail on the above parameters, click 7.6.5.3.3.
- For the **first rectangle** of **REs**, the **CRB index** of its **lowest frequency CRB** corresponds to:
  - `startingPRB`

  For more detail on this parameter, click 7.6.5.3.2.
- For the **second rectangle** of **REs**, the **first symbol interval** within the **slot** corresponds to "$S_1$" (see above), which is the **next symbol interval** after the **first rectangle**. Consequently, the **2 rectangles** are **consecutive** in terms of **symbol intervals**.
- For the **second rectangle** of **REs**, the **quantity** of **symbol intervals** within the **slot** is "***ceiling***($N_{syin}^{PUCCH}$/2). The value of "$N_{syin}^{PUCCH}$" is the **same** as in the **basic variant**, which corresponds to:
  - `nrofSymbols` or `subslotLengthForPUCCH-r16`

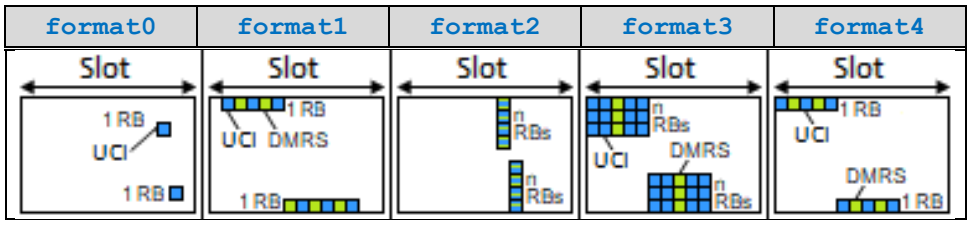
  Consequently, adding the **quantity** of **symbol intervals** of **both rectangles** results in "$N_{syin}^{PUCCH}$".

  For more detail on the above parameters, click 7.6.5.3.3.
- For the **second rectangle** of **REs**, the **CRB index** of its **lowest frequency CRB** corresponds to:
  - `secondHopPRB`

  For more detail on this parameter, click 7.6.5.3.4.
- The **content type** (**payload** or **DM-RS**) of **each RE** of the **PUCCH resource** depends on the **PUCCH resource** `format` (click 7.6.5.1).

For the `format` **options** that characterize the **intra-slot frequency hopping variant**, click 7.6.6.

The following table shows examples of the **intra-slot frequency hopping variant** for its **applicable PUCCH** `formats` (**all PUCCH** `formats`):




Some **relevant remarks** on this table are:

- Notice how the **total quantity** of **symbol intervals** in the **PUCCH resource** is **split** between the **2 rectangles**, according to the formula above. This **does not**

happen in the **inter-slot frequency hopping variant** (click 7.6.6.5) where the **total quantity** of **symbol intervals** is the **same** in the **2 hops**.

- Notice how the **2 rectangles** are **consecutive** in terms of **symbol intervals**.
- For common remarks on this table and on other similar ones, click 7.6.6.8.

For more detail on the **intra-slot frequency hopping variant**, refer to clause 9.2.1 of TS 38.213.

### 7.6.6.4 Repeated intra-slot frequency hopping variant

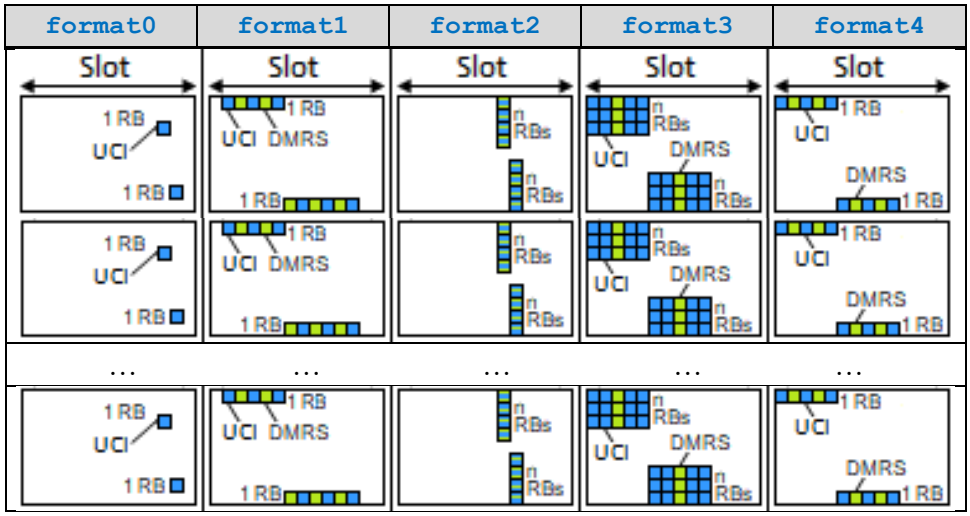
The **repeated intra-slot frequency hopping variant** consists of **repeating** the **PUCCH resource** of the **intra-slot frequency hopping variant** over "$N_{PUCCH}^{repeat}$" **consecutive slots**.

The **RE characteristics** (click 7.6.3) of a **PUCCH resource** with the **repeated intra-slot frequency hopping variant** are:

- "$N_{PUCCH}^{repeat}$" is determined by the **inter-slot repeat option** (click 7.6.5.3.5). It is an **even number** within a **certain range**. "$N_{PUCCH}^{repeat}$" is the **same** for **each** and **every PUCCH resource** (within the corresponding **bandwidth part**) that has the **same** `format`.
- It has **2 rectangles** of **REs** within **each** and **every repeated slot**, and the rectangle's **size**, **shape** and **location** are determined by **exactly** the **same parameters** as in the **intra-slot frequency hopping variant** (click 7.6.6.3).
- The **content type** (**payload** or **DM-RS**) of **each RE** of the **PUCCH resource** depends on the **PUCCH resource** `format` (click 7.6.5.1).
- The **DM-RS** are **recalculated** in **each** and **every repeated slot**.

For the `format` **options** that characterize the **repeated intra-slot frequency hopping variant**, click 7.6.6.

The following table shows examples of the **repeated intra-slot frequency hopping variant** for its **applicable PUCCH** `formats` (**all PUCCH** `formats`):




For **common remarks** on this table and on other similar ones, click 7.6.6.8.

### 7.6.6.5 Inter-slot frequency hopping variant

The **inter-slot frequency hopping variant** consists of **repeating** the **PUCCH resource** of the **basic variant** over "$N_{PUCCH}^{repeat}$" **consecutive slots** but **alternating** the **frequency location** of the **rectangle** of **REs every other slot**.

The **RE characteristics** (click 7.6.3) of a **PUCCH resource** with the **inter-slot frequency hopping variant** are:

- "$N_{PUCCH}^{repeat}$" is determined by the **inter-slot repeat option** (click 7.6.5.3.5). It is an **even number** within a **certain range**. "$N_{PUCCH}^{repeat}$" is the **same** for **each** and **every PUCCH resource** (within the corresponding **bandwidth part**) that has the **same** `format`.
- It has **1 rectangle** of **REs** within **each** and **every repeated slot**, and the rectangle's **first symbol interval**, **quantity** of **symbol intervals** and **quantity** of **CRBs** are determined by **exactly** the **same parameters** as in the **basic variant** (click 7.6.6.1).
- For the **rectangle** of **REs** in the **first slot**, and in **every other slot** within the "$N_{PUCCH}^{repeat}$" **repeated slots**, the **CRB index** of its **lowest frequency CRB** corresponds to:
  - `startingPRB`

  For more detail on this parameter, click 7.6.5.3.2.
- For the **rectangle** of **REs** in the **second slot**, and in **every other slot** within the "$N_{PUCCH}^{repeat}$" **repeated slots**, the **CRB index** of its **lowest frequency CRB** corresponds to:
  - `secondHopPRB`

  For more detail on this parameter, click 7.6.5.3.6.

  This type of **inter-slot frequency hopping** is called "**per slot**".

  In case a **PUCCH transmission** does **not** take place (click 7.6.6.5) in **1** or **more** of the **repeated slots**, this does **not affect** the way in which the **frequency location alternates** in **every other slot** within the "$N_{PUCCH}^{repeat}$" **repeated slots**.
- The **content type** (**payload** or **DM-RS**) of **each RE** of the **PUCCH resource** depends on the **PUCCH resource** `format` (click 7.6.5.1).
- The **DM-RS** are **recalculated** in **each** and **every repeated slot**.

For the `format` **options** that characterize the **inter-slot frequency hopping variant**, click 7.6.6.

The following table shows examples of the **inter-slot frequency hopping variant** for its **applicable** `formats`: `format0`, `format1`, `format3` and `format4`:

| format0 | format1 | format2 | format3 | format4 |
|---|---|---|---|---|
| Slot<br>1 RB<br>UCI | Slot<br>UCI DMRS<br>1 RB | N/A | Slot<br>n RBs<br>UCI DMRS | Slot<br>1 RB<br>UCI DMRS |
| UCI Repetition<br>1 RB | UCI Repetition<br>1 RB | N/A | UCI Repetition<br>n RBs | UCI Repetition<br>1 RB |
| … | … | N/A | … | … |
| UCI Repetition<br>1 RB | UCI Repetition<br>1 RB | N/A | UCI Repetition<br>n RBs | UCI Repetition<br>1 RB |

Some **relevant remarks** on the table above are:

- Notice how the **total quantity** of **symbol intervals** in the **PUCCH resource** is **the same** in the **two hops** (the rectangles shown in both consecutive slots). This **does not** happen in **intra-slot frequency hopping**, where the **total quantity** of **symbol intervals** is **split** between the **two hops**.
- For **common remarks** on this table and on other similar ones, click 7.6.6.8.

For more detail on the **inter-slot frequency hopping variant**, refer to clause 9.2.6 of TS 38.213.

### 7.6.6.6 DM-RS bundling variant

The **DM-RS bundling variant** consists of the **inter-slot frequency hopping variant**, with **two modifications**: a) the **frequency hopping** is performed **every interval** of "$N_{PUCCH}^{interval}$" **slots**, **instead** of **every other slot**; b) The **DM-RS signals** are **exactly** the **same** in **all** the **slots** within **each interval** of "$N_{PUCCH}^{interval}$" **slots** and are **only recalculated** for the **first slot** of **each interval**.

The **RE characteristics** (click 7.6.3) of a **PUCCH resource** with the **DM-RS bundling variant** are:

- "$N_{PUCCH}^{repeat}$" is determined by the **inter-slot repeat option** (click 7.6.5.3.5). It is an **even number** within a **certain range**. "$N_{PUCCH}^{repeat}$" is the **same** for **each** and **every PUCCH resource** (within the corresponding **bandwidth part**) that has the **same** `format`.
- It has **1 rectangle** of **REs** within **each** and **every repeated slot**, and the rectangle's **first symbol interval**, **quantity** of **symbol intervals** and **quantity** of **CRBs** are determined by **exactly** the **same parameters** as in the **basic variant** (click 7.6.6.1), which are also the **same parameters** as in the **inter-slot frequency hopping variant** (click 7.6.6.5).

- For the **rectangle** of **REs** in the **first interval** of "$N_{PUCCH}^{interval}$" **slots**, and in **every other interval** of "$N_{PUCCH}^{interval}$" **slots**, within the "$N_{PUCCH}^{repeat}$" **repeated slots**, the **CRB index** of its **lowest frequency CRB** corresponds to:
  - `startingPRB`

  For more detail on this parameter, click 7.6.5.3.2.
- For the **rectangle** of **REs** in the **second interval** of "$N_{PUCCH}^{interval}$" **slots**, and in **every other interval** of "$N_{PUCCH}^{interval}$" **slots**, within the "$N_{PUCCH}^{repeat}$" **repeated slots**, the **CRB index** of its **lowest frequency CRB** corresponds to:
  - `secondHopPRB`

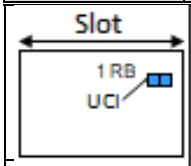
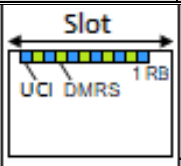
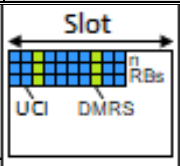
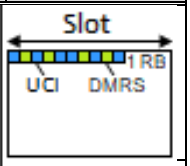
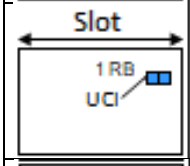
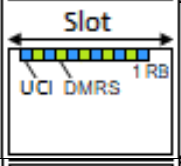
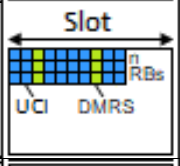
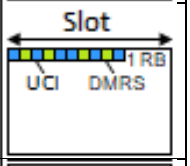
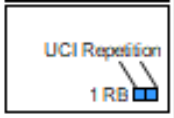
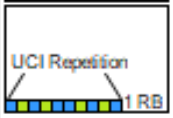
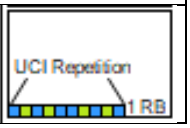
  For more detail on this parameter, click 7.6.5.3.6.

  This type of **inter-slot frequency hopping** is called "**per slot-interval**".

  In case a **PUCCH transmission** does **not** take place (click 7.6.6.5) in **1** or **more** of the **repeated slots**, this does **not affect** the way in which the **frequency location alternates** in **every other interval** of "$N_{PUCCH}^{interval}$" **slots**, within the "$N_{PUCCH}^{repeat}$" **repeated slots**.
- The **content type** (**payload** or **DM-RS**) of **each RE** of the **PUCCH resource** depends on the **PUCCH resource** `format` (click 7.6.5.1).
- The **DM-RS** are **recalculated** for the **first slot** of **each interval** of "$N_{PUCCH}^{interval}$" **slots**, and are kept **exactly** the **same** in **all** the **slots** within **each interval** of "$N_{PUCCH}^{interval}$" **slots**.

For the `format` **options** that characterize the **DM-RS bundling variant**, click 7.6.6.

The following table shows examples of the **DM-RS bundling variant** for its **applicable** `formats`: `format0`, `format1`, `format3` and `format4`:

| format0 | format1 | format2 | format3 | format4 |
|---|---|---|---|---|
|  |  | N/A |  |  |
|  |  | N/A |  |  |
|  |  | N/A |  |  |
|  |  | N/A |  |  |
| … | … | N/A | … | … |
|  |  | N/A |  |  |

Some **relevant remarks** on the table above are:

- Notice how the **total quantity of symbol intervals** in the **PUCCH resource** is **the same** in the **two hops** (the rectangles shown in both consecutive slots). This **does not** happen in **intra-slot frequency hopping**, where the total quantity of symbol intervals is **split** between the **two hops**.
- It is assumed that the **quantity** "$N_{PUCCH}^{interval}$" of **slots** in **each slot-interval** is **2**.
- It is assumed that the **quantity** "$N_{PUCCH}^{repeat}/N_{PUCCH}^{interval}$"is an **even integer**.
- For **common remarks** on this table and on other similar ones, click 7.6.6.8.

For more detail on the **DM-RS bundling variant**, refer to clause 9.2.6 of TS 38.213.

### 7.6.6.7 Interlace variant

The **PUCCH resource interlace** `format` **variant** consists of the **PUCCH RBs** and the **PUSCH RBs** in the **same slot** being **interleaved**. The **interleaving** is based on **specific PRB patterns** called "**PRB interlaces**" (click 2.2.4.7). The **interlace variant** is **set** when **PUCCH-PUSCCH interlace** is **set** (click 2.2.4.8) and it applies to **all** the **PUCCH resources** in the **serving cell**.

In the **interlace variant**, the **PUCCH resource REs** are located in `nrofSymbols` **symbol intervals**, over **all** the **PRBs** from **1** or **2 PRB interlaces** (click 2.2.4.7). **Each** such **PRB interlace** contains **10** or **11 non-contiguous PRBs**.

The said **1** or **2 PRB interlaces** for **PUCCH transmissions** within the **active Uplink** (or **SUL**) **bandwidth part** are the **intersection** between the **PRB set** identified by `rb-SetIndex-r16` and the **interlace** identified by `interlace0-r16`. In case parameter `interlace1-v1610`, then, an **additional set** of **PRBs** available is the **intersection** between the **PRB set** provided by `rb-SetIndex-r16` and the **interlace** identified by `interlace1-v1610`.

For more detail on any of these **parameters**, click 7.6.5.3.8:

Source web pages:

- https://gigayasawireless.github.io/toolkit5G/api/5G_Toolkit/ResourceMapping/PUCCH/Format1/ResourceMappingFormat1.html

### 7.6.6.8 Common remarks on the PUCCH resource example tables

Some **relevant remarks** common to the examples in the tables from 7.6.6.1, 7.6.6.2, 7.6.6.3, 7.6.6.4, 7.6.6.5 and 7.6.6.6 are:

- **Each small square** in the figures is **1 symbol interval** over **1 resource block** (**RB**). It therefore contains **12 resource elements** (**REs**).
- The **REs** with **PUCCH payload**, that contain the **Uplink control information** (**UCI**), are depicted in **blue** (with an indication "**UCI**").
- The **REs** with **PUCCH demodulation reference signals** (**DM-RS**) are depicted in **green** (with an indication "**DM-RS**").
- `format0` does **not** have **DM-RS**.
- For the location of **PUCCH DM-RS REs** within **the PUCCH resource REs**, click 7.6.9.

- **PUCCH resource** `format0` **and** `format2` are depicted with **2 symbol intervals**, but they also support **1 symbol interval**.
- When the **figure** of a given example contains **several rows**, it represents that the **PUCCH resource repeats** the **PUCCH transmission** over "$N_{PUCCH}^{repeat}$" **slots**. **Each consecutive row** represents **1 onsecutive slot** (except for rows separated by an ellipsis). The **slots** advance **top** (**earlier**) to **down** (**later**). The **first** (**top**) **slot** has **index 0** within the "$N_{PUCCH}^{repeat}$" **slots**. The **first** (**top**) **slot** is an **even slot**. It is assumed that the **quantity** "$N_{PUCCH}^{repeat}$" is **even**.

### 7.6.7 PUCCH resource sets

A **PUCCH resource set** is a `SEQUENCE OF` **PUCCH resource identifiers** (of **data type** `PUCCH-ResourceId`, click 7.9.7.1).

**Each PUCCH resource set** is defined as an **instance** of **data type** `PUCCH-ResourceSet` (click 7.9.6) and it is **identified** by the **value** of its parameter `pucch-ResourceSetId`, which can be from `0` to `3`. The **maximum quantity** of **UCI information bits** that the **UE** can transmit using **any** of the **PUCCH resources** within a `PUCCH-ResourceSet` **instance** is the value of parameter `maxPayloadSize` from the `PUCCH-ResourceSet` **instance**.

**Each UE**'s **Uplink bandwidth part** can be configured **up to** `4` `PUCCH-ResourceSet` **instances** with parameter `resourceSetToAddModList` from `PUCCH-Config` (click 7.9.4). The **maximum quantity** of **PUCCH resources** is `32` in the **first** **PUCCH resource set** and it is `8` in **each** of the **3 non-first PUCCH resource sets**, resulting in a **total** of `56` **PUCCH resources**. For the **first PUCCH resource set**, the maximum quantity of **UCI** information **bits** is **2**. **PUCCH** `format0` and `format1` are **only** allowed for **PUCCH resources** in the **first** **PUCCH resource set**. **PUCCH** `format2` to `format4` are **only** allowed for **PUCCH resources** in the **3 non-first PUCCH resource sets**.

For more detail on **PUCCH resource sets**, refer to clause 9.2 of TS 38.213.

### 7.6.8 Several PUCCH transmissions in the same slot

It is possible to perform **more** than **1 PUCCH transmission** in the **same slot** under some system conditions.

#### 7.6.8.1 In different symbol intervals

If a **UE** transmits **2 PUCCH resources** in the **same slot**, they **must** be in **different symbol intervals**.

A **UE may** transmit **1** or **2 PUCCHs** on a **serving cell** in **different symbol intervals** within the **same slot**.

***If*** [[the **UE** transmits **2 PUCCHs** in a **slot**] and [`ackNackFeedbackMode-r16` from `PhysicalCellGroupConfig` (click 3.7.5) with value "`separate`" is **absent**]], ***then***, **at least 1** of the **2 PUCCHs** uses **either** `format0` or `format2`.

***If*** [`ackNackFeedbackMode-r16` with value "`separate`" is **present**], ***then***, the **UE** may transmit up to **2 PUCCHs** in **different symbol intervals** within the **same slot** that may contain **HARQ (N)ACK information**.

If the **UE** transmits **two PUCCHs** in the **same slot**, there are **two cases**, depending on whether parameter `ackNackFeedbackMode-r16` from `PhysicalCellGroupConfig` (click 3.7.5) is **present** or **absent**:

- The parameter is **absent**

  **At least one** of the **two PUCCHs transmitted** in **the slot must** use **either** `format0` or `format2`.

- The parameter is **present**

  The **UE** may **transmit** in **the slot** two **PUCCHs** with **HARQ-ACK** information in **different symbol intervals**.

### 7.6.8.2 UCI reporting of different priorities

***If*** **all** the following conditions **hold** for a **UE**:

- **Any** of the parameters `pucch-ConfigurationList-r16`, `pucch-ConfigurationListMulticast1-r17` or `pucch-ConfigurationListMulticast2-r17` is **present**. **All** of them are from `BWP-UplinkDedicated` (click 2.2.8.10) and have **data type** `PUCCH-ConfigurationList` (click 7.9.5).
- Parameter `uci-MuxWithDiffPrio-r17` from `PhysicalCellGroupConfig` (click 3.7.5) is **present**, and
- would perform **overlapping PUCCH transmissions** that include a **first PUCCH transmission** with "$O_{ACK,0}$" **HARQ-ACK information bits** of **priority 0** and a **second PUCCH transmission** with "$O_{ACK,1}$" **HARQ-ACK information bits** of **priority 1**
- if the **PUCCH resource** for the **second PUCCH transmission** has `format2`, `format3` or `format4` and **additionally** includes "$O_{SR,1}$" **SR** bits of **priority 1**, the "$O_{ACK,1}$" bits are replaced by "$O_{ACK,1}+O_{SR,1}$" where "$O_{SR,1}$" is determined according to clause 9.2.5.1,

***then***, the **UE determines**

- a **PUCCH resource set** from its **second** `PUCCH-Config` **instance** using "$O_{UCI}=O_{ACK,0}+O_{ACK,1}$" as described in clause 9.2.1, and a **PUCCH resource** from the **PUCCH resource set** as described in clause 9.2.3 where a **DCI Format**, if any, triggers a **PUCCH transmission of priority 1**, or
- a **PUCCH resource** from the **second** `sps-PUCCH-AN-List` using "$O_{UCI}=O_{ACK,0}+O_{ACK,1}$" as described in clause 9.2.1, and
- **multiplexes** the "$O_{ACK,0}$" and "$O_{ACK,1}$" **HARQ-ACK information bits** in a same **PUCCH transmission** using the **PUCCH resource**.

All the referred clauses are of TS 38.213.

For more detail on **UCI reporting** of **different priorities**, refer to clause 9.2.5.3 of TS 38.213.

### 7.6.9 PUCCH scheduling of payload PREs and DM-RS PREs

**PUCCH DM-RS** are the **DM-RS** (click 4.11.3.2.1) within **PUCCH**. They are characterized by their **contents** and by their **scheduling** within the **PUCCH PREs** that is described in this section.

The following **6 symbols** (in addition to the ones from 7.5) will be used for the **5 PUCCH formats**:

- "$N_{syin}^{PUCCH\text{-}Pld,n}$" is the **quantity** of **payload symbol intervals** of the **PUCCH transmission**, for the corresponding **format** "**n**".
- "$M_{sc}^{PUCCH\text{-}Pld,n}$" is the **quantity** of **payload subcarriers** of the **PUCCH transmission**, for the corresponding **format** "**n**".
- "$N_{RE}^{PUCCH\text{-}Pld,n}$" is the **quantity** of **payload resource elements** of the **PUCCH transmission**, for the corresponding **format** "**n**":

  $$N_{RE}^{PUCCH\text{-}Pld,n} = N_{syin}^{PUCCH\text{-}Pld,n} \bullet M_{sc}^{PUCCH\text{-}Pld,n}$$

- "$N_{syin}^{PUCCH\text{-}DM\text{-}RS,n}$" is the **quantity** of **payload symbol intervals** of the **PUCCH transmission**, for the corresponding **format** "**n**".
- "$M_{sc}^{PUCCH\text{-}DM\text{-}RS,n}$" is the **quantity** of **payload subcarriers** of the **PUCCH transmission**, for the corresponding **format** "**n**".
- "$N_{RE}^{PUCCH\text{-}DM\text{-}RS,n}$" is the **quantity** of **payload resource elements** of the **PUCCH transmission**, for the corresponding **format** "**n**":

  $$N_{RE}^{PUCCH\text{-}DM\text{-}RS,n} = N_{syin}^{PUCCH\text{-}DM\text{-}RS,n} \bullet M_{sc}^{PUCCH\text{-}DM\text{-}RS,n}$$

Consequently:

- The **quantity** "$N_{syin}^{PUCCH\text{-}DM\text{-}RS,n}$" of **PUCCH symbol intervals fully devoted** to **PUCCH DM-RS**, for the corresponding **format** "**n**" is:

  $$N_{syin}^{PUCCH\text{-}DM\text{-}RS,n} = N_{syin}^{PUCCH,n} - N_{syin}^{PUCCH\text{-}Pld,n}$$

- The **quantity** "$M_{sc}^{PUCCH\text{-}DM\text{-}RS,n}$" of **PUCCH subcarriers fully devoted** to **PUCCH DM-RS**, for the corresponding **format** "**n**" is:

  $$M_{sc}^{PUCCH\text{-}DM\text{-}RS,n} = M_{sc}^{PUCCH,n} - M_{sc}^{PUCCH\text{-}Pld,n}$$

- The **quantity** "$N_{RE}^{PUCCH\text{-}DM\text{-}RS,n}$" of **PUCCH resource elements fully devoted** to **PUCCH DM-RS**, for the corresponding **format** "**n**" is:

  $$N_{RE}^{PUCCH\text{-}DM\text{-}RS,n} = N_{RE}^{PUCCH,n} - N_{RE}^{PUCCH\text{-}Pld,n}$$

The **scheduling** of **PUCCH DM-RS PREs** within the **PUCCH resource PREs** depends on the **PUCCH resource format**. This also determines the **quantity** of **payload PUCCH PREs** and the **ratio** of **payload PUCCH PREs** divided by the **total PUCCH PREs**. The specific values for **each format** are detailed in the following subsections.

For more detail on **PUCCH DM-RS scheduling**, refer to clause 6.4.1.3 of TS 38.211.

Source web pages:

- https://www.rfwireless-world.com/5G/5G-NR-UCI.html

### 7.6.9.1 `format0` scheduling of payload PREs and DM-RS PREs

`format0` does **not** contain **PUCCH DM-RS** "$N_{RE}^{PUCCH\text{-}DM\text{-}RS,1}$=0". The reason is that `format0` does **not** use any **modulation**. It rather conveys the **UCI bits** as a **low-PAPR type 1 complex number sequence** (click 7.8.2.12).

Consequently:

- The **quantity** of `format0` **payload PUCCH subcarriers** "$M_{sc}^{PUCCH\text{-}Pld,1}$" is the **same** as the **quantity** of `format0` **PUCCH subcarriers** "$M_{sc}^{PUCCH,1}$":

$$M_{sc}^{PUCCH\text{-}Pld,1} = M_{sc}^{PUCCH,1}$$

- The **quantity** of `format0` **payload PUCCH symbol intervals** "$N_{syin}^{PUCCH\text{-}Pld,1}$" is the **same** as the **quantity** of `format0` **PUCCH symbol intervals** "$N_{syin}^{PUCCH,1}$":

$$N_{syin}^{PUCCH\text{-}Pld,1} = N_{syin}^{PUCCH,1}$$

- The **quantity** of **payload PUCCH PREs** "$N_{RE}^{PUCCH\text{-}Pld,1}$" is the **product** of the **quantity** of **PUCCH symbol intervals** "$N_{syin}^{PUCCH,1}$" and the **quantity** of **PUCCH subcarriers** "$M_{sc}^{PUCCH,1}$":

$$N_{RE}^{PUCCH\text{-}Pld,1} = N_{syin}^{PUCCH,1} \bullet M_{sc}^{PUCCH,1}$$

- The **ratio** of **payload PUCCH PREs** "$N_{RE}^{PUCCH\text{-}Pld,1}$" divided by the **total PUCCH PREs** "$N_{RE}^{PUCCH,1}$" is **1**:

$$N_{RE}^{PUCCH\text{-}Pld,1} / N_{RE}^{PUCCH,1} = 1$$

### 7.6.9.2 `format1` scheduling of payload PREs and DM-RS PREs

**PUCCH DM-RS** are placed in **all** the **PREs** (i.e., **all** the **subcarriers**) of **1 out** of **2 symbol intervals** within **PUCCH**, starting with the **first symbol interval** in **PUCCH**:

$$N_{syin}^{PUCCH\text{-}Pld,1} = \mathit{floor}(N_{syin}^{PUCCH,1} / 2)$$

$$N_{RE}^{PUCCH\text{-}Pld,1} = \mathit{floor}(N_{RE}^{PUCCH,1} / 2)$$

The following table shows the resulting **quantity** of **payload PUCCH PREs** "$N_{RE}^{PUCCH\text{-}Pld,2}$" and the **ratio** of **payload PUCCH PREs** "$N_{RE}^{PUCCH\text{-}Pld,2}$" divided by the

**total PUCCH PREs** "$N_{RE}^{PUCCH,2}$" for the values of **PUCCH symbol intervals** "$N_{syin}^{PUCCH,2}$":

| | PUCCH format1 Payload PREs | |
|---|---|---|
| **PUCCH duration "$N_{syin}^{PUCCH,2}$" [sy. in.]** | **Payload PREs / subcarriers** | **Ratio payl./total** |
| 4 | 2 | 1/2 |
| 5 | 2 | 2/5 |
| 6 | 3 | 1/2 |
| 7 | 3 | 3/7 |
| 8 | 4 | 1/2 |
| 9 | 4 | 4/9 |
| 10 | 5 | 1/2 |
| 11 | 5 | 5/11 |
| 12 | 6 | 1/2 |
| 13 | 6 | 6/13 |
| 14 | 7 | 1/2 |

Notice that the **quantity** of **payload PUCCH PREs** is expressed **divided** by the **quantity** of **PUCCH subcarriers**.

### 7.6.9.3 format2 scheduling of payload PREs and DM-RS PREs

**PUCCH DM-RS** are placed in **all** the **PREs** of **all** the **PUCCH symbol intervals** of **1 out 3 PUCCH subcarriers**, starting with the **subcarrier** with **index 1** from **common resource block 0**, counting upwards. The lowest subcarrier has **index 0**. Of course, the **PUCCH DM-RS must** be **within** the **PUCCH PREs** (click 7.6.9).

Consequently:

- The **quantity** of **payload PUCCH PREs** "$N_{RE}^{PUCCH\text{-}Pld,2}$" is the **product** of the **quantity** of **symbol intervals** "$N_{syin}^{PUCCH,2}$" and the **quantity** of **payload subcarriers** "$M_{sc}^{PUCCH\text{-}Pld,2}$":

$$N_{RE}^{PUCCH\text{-}Pld,2} = N_{syin}^{PUCCH,2} \bullet M_{sc}^{PUCCH\text{-}Pld,2}$$

$$N_{RE}^{PUCCH\text{-}Pld,2} = N_{syin}^{PUCCH,2} \bullet 2/3 \bullet M_{sc}^{PUCCH,2}$$

- The **ratio** of **payload PUCCH PREs** divided by the **total PUCCH PREs** is **2/3**:

$$N_{RE}^{PUCCH\text{-}Pld,2} \; / \; N_{RE}^{PUCCH,2} = \mathbf{2/3}$$

### 7.6.9.4 format3 and format4 scheduling of payload PREs and DM-RS PREs

**PUCCH DM-RS** are placed in **all** the **PREs** (i.e., **all** the **subcarriers**) of **some specific PUCCH symbol intervals**. The **first symbol interval within PUCCH** has index **0**. The

indexes of the specific **symbol intervals within PUCCH** that contain **DM-RS** are indicated in table 6.4.1.3.3.2-1 of TS 38.211., copied below:

| | **format3 and format4: DM-RS within PUCCH [PUCCH symbol index]** | | | |
|---|---|---|---|---|
| | **No additional DM-RS** | | **Additional DM-RS** | |
| **PUCCH [sy.int.]** | **No frequency hopping** | **Frequency hopping** | **No frequency hopping** | **Frequency hopping** |
| **4** | 1 | 0, 2 | 1 | 0, 2 |
| **5** | 0, 3 | | | |
| **6** | 1, 4 | | | |
| **7** | 1, 4 | | | |
| **8** | 1, 5 | | | |
| **9** | 1, 6 | | | |
| **10** | 2, 7 | | 1, 3, 6, 8 | |
| **11** | 2, 7 | | 1, 3, 6, 9 | |
| **12** | 2, 8 | | 1, 4, 7, 10 | |
| **13** | 2, 9 | | 1, 4, 7, 11 | |
| **14** | 3, 10 | | 1, 5, 8, 12 | |

The **existence** or **not** of **additional DM-RS** in **PUCCH** `format3` and `format4` is determined by the parameter `additionalDMRS`, from `PUCCH-FormatConfig` (click 7.9.10).

Consequently, the **quantity** of **payload PUCCH PREs** and the **ratio** of **payload PUCCH PREs** (between parenthesis) is presented in the following table

| | **PUCCH format3 and format4: Payload PUCCH PREs (ratio payload/total)** | | | |
|---|---|---|---|---|
| | **No additional DM-RS** | | **Additional DM-RS** | |
| **PUCCH [sy.int.]** | **No frequency hopping** | **Frequency hopping** | **No frequency hopping** | **Frequency hopping** |
| **4** | 3•12RBs (3/4) | 2•12RBs (1/2) | 3•12RBs (3/4) | 2•12RBs (1/2) |
| **5** | 3•12RBs (3/5) | | | |
| **6** | 4•12RBs (2/3) | | | |
| **7** | 5•12RBs (5/7) | | | |
| **8** | 6•12RBs (3/4)) | | | |
| **9** | 7•12RBs (7/9) | | | |
| **10** | 8•12RBs (4/5) | | 6•12RBs (3/5) | |
| **11** | 9•12RBs (9/11) | | 7•12RBs (7/11) | |
| **12** | 10•12RBs (5/6) | | 8•12RBs (2/3) | |
| **13** | 11•12RBs (11/13) | | 9•12RBs (9/13) | |
| **14** | 12•12RBs (6/7) | | 10•12RBs (5/7) | |

## 7.7 PUCCH payload contents: UCI messages (also sent through PUSCH)

An **Uplink Control Information** (**UCI**) **message** is a **set** of **control information** sent by a **UE to** the **NR node**.

The **bitlength** "$N_{bit}^{UCI}$" of **1 UCI message** depends on whether it is sent through **PUSCH** or **PUCCH**, and in case it is sent through **PUCCH**, on the **resource** `format`

- **UCI sent through PUSCH**: **1** ≤ $N_{bit}^{UCI}$ ≤ ***11,000** [bits].
- **UCI sent through PUCCH**, with `format0` or `format1`: $N_{bit}^{UCI}$ ≤ **2** [bits].
- **UCI sent through PUCCH**, with `format2` or `format3`: **3** ≤ $N_{bit}^{UCI}$ ≤ ***170** [bits].
- **UCI sent through PUCCH**, with `format4`: **3** ≤ $N_{bit}^{UCI}$ ≤ **1,706** [bits].

The values with **asterisk** "*" are **configuration dependant**.

**Each UCI message** contains a certain **combination** of **4 content types** (click 7.7.1).

**Each UCI message** is **sent** through **PUCCH but can also be sent** through **PUSCH** if certain conditions (click 4.12.8) are met.

For **UCI messages** sent through **PUCCH**, click 7.8.

For **UCI messages** sent through **PUSCH**, click 4.12.8.

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_UCI.html
- https://www.sharetechnote.com/html/5G/5G_PUCCH.html
- https://www.rfwireless-world.com/5G/5G-NR-UCI.html

## 7.7.1 Content types of UCI messages

**Each UCI message** contains a certain **combination** of the following **4 content types**:

- **HARQ-ACK**s (click 7.7.1.1)
- **Scheduling request** (**SR**, click 7.7.1.2)
- **Channel state information** (**CSI**, click 7.7.1.3)
- **Last Resource Retransmission** (**LRR**, click 7.7.1.4)

**Each UCI message** sent **through PUCCH** is **encoded** and **modulated** (click 7.7) for its transmission.

**Each UCI message** sent **through PUSCH** is **multiplexed** (click 4.12.8) with the **PUSCH data** for its transmission.

For more detail on **content types** of **UCI messages**, refer to clause 6.3 of TS 38.212 and clause 9.2 and 9.3 of TS 38.213.

### 7.7.1.1 HARQ-ACK

It conveys the **acknowledgement**s (**ACK**) and/or **negative acknowledgements** (**NACK**) from the **Hybrid Automatic Repeat reQuest** (**HARQ**, click 4.5.6) **protocol**. It may have **1** or **several bits**.

The **bitlength** for **HARQ-ACKs** depends on the **HARQ mode** (click 4.5.6.4):

- **Transport block** (**TB**) **mode**: **1 HARQ-ACK bit** for **each transport block** (i.e., for **each codeword**). The **HARQ-ACK** information has **1** or **2 bits**.
- **Code block group** (**CBG**) **mode**: **1 HARQ-ACK bit** for **each code block group**. The **HARQ-ACK** information has **several bits**.

**Each HARQ-ACK requires 1 bit**. If the bit value is "**1**", it is an **ACK**. If the bit value is "**0**", it is a **NACK**.

The **HARQ-ACKs** are part of **HARQ**, within the **Low-MAC** (click 4.5.4.2), within the **MAC sublayer** (click 11.11).

For what **combinations** of **UCI message types** may be sent, and for the **bit sequence** of the resulting **UCI message**, click 7.7.2.

For more detail on **HARQ-ACK** in **PUCCH UCI messages**, refer to clauses 9.1 (for **CBG mode**) and 9.2.3 (for **transport block mode**) of TS 38.213.

### 7.7.1.2 Scheduling request (SR)

**Each PUCCH scheduling request (SR) transmission** is used by the **UE** to **request PUCCH transmission resources** for the transmission of a **new transport block**. For the **scheduling** of **PUSCH HARQ retransmissions**, click 4.5.6.6.

**UCI messages containing SR** information **cannot** be sent through **PUSCH**.

When the **UE** has **data** to send through **PUSCH**, it transmits a **scheduling request** at the **earliest transmission occasion** (click 4.11.4.2) for a **scheduling request**. **Each pending** (i.e., already **transmitted**, but still **unattended**) **scheduling request** is **retransmitted** up to a **maximum quantity** of **retransmission**. This **maximum quantity** of **retransmissions** is configured with `sr-TransMax` from `SchedulingRequestToAddMod` (click 7.9.15). There may be **up to** `8` `SchedulingRequestToAddMod` **instances**, **all** of them **common** to **all** the **cells** in **1 cell group**.

**Each PUCCH scheduling request (SR) transmission** has **0 bits** (**implicit SR**), or **some bit(s)** (**explicit SR**). For **explicit SR**, it may have **1 bit** (**single-bit SR**) or **several bits** (**multi-bit SR**).

**Each explicit SR** may be **positive** (bit value "**1**") or **negative** (bit value "**0**"). Normally, a **UE** that has **data** to **send** transmits a **SR** to **request** the **NR node** a **PUSCH assignment**. However, there are scenarios where the **NR node expects** an **SR** from the **UE**, but the **UE** has **no data** to transmit. In such cases, **instead** of remaining **silent** (which could cause **confusion**), the **UE transmits** a **negative SR** to **explicitly** indicate that **no PUSCH assignment** is needed.

For what **combinations** of **UCI message types** may be sent, and for the **bit sequence** of the resulting **UCI message**, click 7.7.2.

The **SR** signaling is part of the **High-MAC** (click 4.5.4.1), within the **MAC sublayer** (click 4.5.4)

For more detail on **PUCCH scheduling requests**, refer to clause 5.4.4. of TS 38.321.

### 7.7.1.3 Channel state information (CSI) in PUCCH

It conveys the **channel state information** (**CSI**, click 2.4.5) from **measurements** and **estimations** performed by the **UE**. It has **several bits**.

There are several **overall bit structure types** of **CSI UCI message**, depending on the **type** of **CSI** (e.g., **PMI** or **RI/LI/CQI/CRI** or **CRI/SSBRI/RSRP**) carried by the **UCI message**. The **detailed bit structure** for **each type** varies further depending on some additional factors.

The **CSI** signaling is part of the **channel estimator** (click 4.5.7.6).

For what **combinations** of **UCI message types** may be sent, and for the **bit sequence** of the resulting **UCI message**, click 7.7.2.

#### 7.7.1.4 Last Resource Retransmission (LRR)

It indicates the **NR node** that this **slot** contains the **last repetition** of the **UCI message**, within the **repetitions** of **inter-slot repeat** (click 7.6.6.5). It has **0 bits** (**implicit LRR**) or **1 bit** (**explicit LRR**). The **implicit LRR** (sometimes called "**resource-based indication**") can be signaled by using a **specific PUCCH resource**.

For the **bit sequence** of **last resource retransmission** within a **UCI message**, click 7.7.2.

The **LRR bit** is especially relevant in **PUCCH multiplexing** scenarios where **multiple UEs share** the **same PUCCH resources** and need an **efficient** method to manage **HARQ feedback retransmissions**.

For more detail on **LRR** and **link recovery**, refer to clause 6 of TS 38.213.

For more detail on **LRR encoding**, refer to clauses 9.2, 9.2.4 and 9.2.5.1 of TS 38.213.

For what **combinations** of **UCI message types** may be sent, and for the **bit sequence** of the resulting **UCI message**, click 7.7.2.

**LRR** is **not mentioned** even once in TS 38.211, nor in TS 38.212, nor in TS 38.214.

Along clause 9.2 of TS 38.213 any reference to **SR** is **applicable** to **SR and/or** to **LRR**.

### 7.7.2 Contents of UCI messages

**Multiplexing** of **several UCIs** in **1 PUCCH transmission** is supported when the **UCIs coincide** in **time**, and **either**:

- The **UCIs** have the **same priority** (**high/low**).
- The **HARQ-ACK UCI** has **priority index 0** (**low**) and the other **UCI** has **priority index 1** (**high**).

When the previous **conditions hold**, **each UCI message** contains **1 combination** between the **4 UCI content types** (click 7.7.1): "**HARQ-ACK**", "**SR**", "**CSI**" and "**LRR**".

A **HARQ-ACK** is **sent** as bit value "**1**" while a **HARQ-NACK** is **sent** as bit value "**0**". For the case of "**SR**" and "**LRR**", **only positive** ones are **transmitted** through **PUCCH**, and are **sent** as bit value "**0**". The "**CSI**" information has a **more complex coding**.

**Not all combinations** of information are **allowed**. The **allowed combinations** depend on: a) the **delivery channel** (**PUCCH** or **PUSCH**); b) if the **delivery** is with **repetitions** over **multiple slots** and c) for the case of **PUCCH**, on the **PUCCH resource** `format` (click 7.6.5.1).

This section describes the case of **delivery** through **PUCCH without repetitions**.

For the case of **UCI** through **PUCCH**, the following table summarizes the **allowed combinations** of **UCI content types** in **1 PUCCH UCI message**:

| | **Allowed UCI content type combinations in PUCCH** | | | | |
|---|---|---|---|---|---|
| **PUCCH** `format` (7.6.5.3.2) | `format0` | `format1` | `format2` | `format3` | `format4` |
| `format` **length** | **Short** | **Long** | **Short** | **Long** | |
| **UCI content type combinations** (7.7.1) | ACK, SR, LRR | ACK, SR, LRR and any pairwise combination | Any comb. of ACK, SR, LRR, CSI ACK/SR+CSI only if `simultaneousHARQ-ACK-CSI` (7.6.5.3.3) is **present** | | |
| **UCI message [bits]** (2nd row: in FR2-2) | 1 or 2 | | 3 to large | 3 to large | 3 to moderate |

For **PUCCH** `format2`, `format3` and `format4`, when **several UCI content types** are combined in **1 UCI message**, the **order** of the **bit sequences** of the **content types** within the **bit sequence** of the **UCI message** is:

1. **HARQ-ACK**.
2. **Scheduling request** (**SR**).
3. **Channel State Information** (**CSI**).
4. **Last resource retransmission** (**LRR**).

For an **UCI message** delivered through **PUCCH**, the **UCI message bitlength** is a **relevant factor** for the decision of what **PUCCH resource** `format` to use (refer to the table above) and **what PUCCH resource** to use (click 7.6.4).

For the case of **UCI** through **PUSCH**, **any combination** of the **4 UCI content types** is allowed in **1 PUSCH UCI message**. For more detail on **UCI** through **PUSCH**, refer to clause 6.3.2 of TS 38.212 and clause 9.3 of TS 38.213.

The **control system directly** on **top** of **PUCCH** (there is **no** higher layer **transport channel** on top) produces the **Uplink Control Information** (**UCI**) **messages** for the **NR node**. **Each UCI message** is **sent** in **1 slot**.

A relevant difference between **PUCCH** and **PDCCH** is that **DCI** messages can only be sent through **PDCCH**. However, **UCI messages** (that do **not** contain an **SR**) can be sent through **either PUCCH** or **PUSCH**.

For the relationship between **UCI messages** (**contents**, **bitlength**, …) and **PUCCH resource** `formats`, click 7.8.1.

For more detail on **multiplexing** of **several UCIs** in **1 PUCCH transmission**, refer to clause 5.3.3 of TS 38.300.

For more detail on **PUCCH UCI messages**, refer to clauses 6.3.1 and 6.3.2.1 of TS 38.212 and clause 9.2 and 9.3 of TS 38.213. Clause 6.3.1 of TS 38.212 **only** applies to **UCI messages** through **PUCCH** with `format2`, `format3` or `format4`.

## 7.8 PUCCH payload processing

**PUCCH payload processing** depends on the **PUCCH resource** `format`.

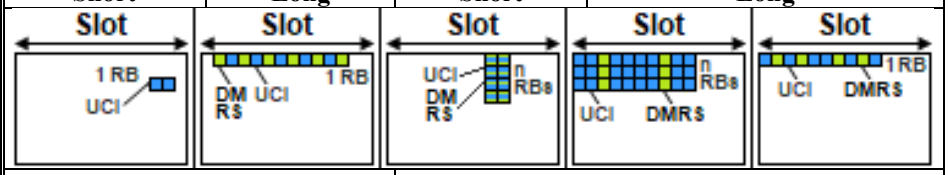
For a summary of the **PUCCH payload processing** in **each** of the **5** `formats`, click 7.8.1.

For more detail on **PUCCH payload processing**, refer to clause 5.3.3 (overview) of TS 38.300 and clauses 5.2.1 (segmentation), 5.3.1 and 5.3.3 (encoding), 5.4.1 (rate-matching), 5.5 (concatenation), 6.3.1 (overall) of TS 38.212, and clause 6.3.2 (modulation) of TS 38.211.

### 7.8.1 PUCCH resource `format` payload processing summary

The **characteristics** and relevant **parameters** of the **UCI message processing** , when delivered through **PUCCH**, for the **5 PUCCH** `formats` are **summarized** in the table below.

For a **summary table** of the **PUCCH scheduling** of its **payload PREs** and its **DM-RS PREs** for the **5 PUCCH resource** `formats` and its **7 variants**, click 7.6.5.1.

| | **PUCCH resource payload processing** | | | | |
|---|---|---|---|---|---|
| **PUCCH** `format` (7.6.5.3.2) | `format0` | `format1` | `format2` | `format3` | `format4` |
| `format` **length** | **Short** | **Long** | **Short** | **Long** | |
| **Examples of PUCCH resources with the basic variant** (7.6.6.1) |  | | | | |
| **Configurable in which PUCCH resource sets** | Only in **first** resource set | | Only in **non-first** resource set | | |
| **UCI content type combinations** (7.7.1) | ACK, SR, LRR and any pairwise combination | | Any combination of ACK, SR, LRR, CSI<br>ACK+CSI only if `simultaneousHARQ-ACK-CSI` (7.6.5.3.3) is **present** | | |
| **UCI message bitlength [bits]** (7.8.2.1) | 1 or 2 | | 3 to 1,706 | 3 to 1,706 | 3 to 170* |
| **Code block segmentation and/or CRC** (7.8.2.2) | No | | Only if $N_{bit}^{UCI} \geq 12$ | | |
| **{UCI message [bits]} => Channel coding** (7.8.2.3) | No channel coding | | {3 to 11} Reed-Muller<br>{>11} CRC & Polar | | |
| `maxCodeRate` **and** `maxCodeRateLP-r17` (7.6.5.3.3) | N/A | | `0.08`, `0.15`, `0.25`, `0.35`, `0.45`, `0.6` or `0.8` | | |
| **Rate matching** (7.8.2.4) | No | | Yes | | |
| **Scrambling** (7.8.2.7) | No | | Yes, with sequence "$c(n)$" (11.20.1) | | |
| **{UCI message [bits]} => Modulation** (7.8.2.8) | N/A | {2 bits}: QPSK<br>{1 bit}: BPSK | QPSK | QPSK or<br>$\pi/2$-BPSK if `pi2BPSK` is **present** (7.6.5.3.3) | |
| **Modul. Order [bits/sym]** | N/A | 2 or 1 | 2 | 2 or 1 | |
| **Frequency-spreading** (7.8.2.9) | Equivalent | Yes | No | | |
| `initialCyclicShift` (7.6.5.3.2) | `0` to `11` | | N/A | | |
| **UE multiplexing technique(s)** (7.8.2.10) | Cyclic Shift | CS and TD-OCC | Time-Domain Orthogonal Cover Code | | |
| **UE multiplexing capacity** | Up to 6 UEs with 1-bit payload | **No fr. hopp.:** up to 84 UEs<br>**Freq. hopp.:** up to 36 UEs | If shared spectrum in **FR1** and 1 interlace, up to 4 UEs | | Up to 4 UEs |
| `timeDomainOCC` (7.6.5.3.2) | N/A | `0` to `6` | N/A | | |
| `occ-Length` (7.6.5.3.8) | N/A | | | | `2`, `4` |
| `occ-Length-v1610` (7.6.5.3.8) | N/A | `2`, `4` | `2` | `2`, `4` | |

| | PUCCH resource payload processing | | | | |
|---|---|---|---|---|---|
| **PUCCH `format`** (7.6.5.3.2) | `format0` | `format1` | `format2` | `format3` | `format4` |
| `occ-Index` (7.6.5.3.8) | N/A | | | | `0`, `1`, `2`, `3` |
| `occ-Index-v1610` (7.6.5.3.8) | N/A | `0`, `1`, `2`, `3` | | | |
| **Option of** `useInterlacePUCCH-PUSCH-r16` (7.6.5.3.8) | If shared spectrum in **FR1**, yes | | | | No |
| `interlace0-r16` (7.6.5.3.8) | Single interlace | | First or single interlace | | N/A |
| `interlace1-v1610` (7.6.5.3.8) | N/A | | Second interlace | | N/A |
| `rb-setIndex-r16` (7.6.5.3.8) | `0` to `4` | | | | |
| **DFT precoding** (7.8.2.12) | No | | | Yes | |
| **Low-PAPR sequence and cyclic shift hopping** (7.8.2.13) | Yes | | No | Yes | |

Some **relevant remarks** on this table are:

- Row "**Configurable in which PUCCH resource sets**" indicates, for **each PUCCH resource** `format`, in **which PUCCH resource sets** it **can** be **configured**.
- Row "**Modulation**": **PUCCH** `format0` does **not** use a **modulation**, (see the next bullet). The **modulation** used depends on the **UCI message bitlength** (indicated between curly brackets).
- Row "**Low PAPR t1 spreading**": indicates if the information is **frequency-spread** using **low-PAPR type 1 complex number sequences** (click 11.20.2) of the **same length** "$M_{ZC}$" as the **quantity** of **PUCCH subcarriers**. In **PUCCH** `format0`, the **2** or **4 bit values** are **directly mapped** to **2** or **4 low-PAPR type 1sequences** "$ms(j)$". In **PUCCH** `format1`, the **1 modulated symbol** "$ms(0)$" is the multiplier of **1 low-PAPR type 1sequence**.
- Row "**Channel coding**": the **UCI message bitlength** values (in bits) that are indicated include the **UCI message CRC** (if present). **PUCCH** `format0` and `format1` do not use a **channel code** because they do **not** use a **modulation**. They **rather** use **sequences** and **very robust line coding**. Refer to clauses 5.3.3.1 and 5.3.3.2 of TS 38.212.
- Row "**Multiple access technique**": **transform precoding** (click 4.12.12) is applied to **PUCCH** `format3` and `format4`.
- Row "`occ-Length-v1610`". This parameter determines the **time-spreading factor** "$N_{SF}^{PUCCH,n}$" for **some** `formats`. It may take **values** "`2` or `4`". It may be configured to **all** `formats`. However:
  - `format0` does **not** use **OCCs** (click 7.8.2.10) and consequently, this parameter does **not** apply to `format0`.
  - `format2` can have **at most 2 symbol intervals** (refer to `nrofSymbols`, click 7.6.5.3.3) and consequently, **only** value "`2`" applies to `format2`.

For more detail on **PUCCH payload processing**, refer to clauses 6.3.2 and 6.4.1.3 of TS 38.211, clause 6.3 of TS 38.212, and clause 5.3.3 of TS 38.300.

The following two web pages contain **several examples** of **PUCCH** `formats` that are **very illustrative**:

- https://www.sharetechnote.com/html/lte_toolbox/Matlab_LteToolbox_5G_PUCCH0.html
- https://www.sharetechnote.com/html/lte_toolbox/Matlab_LteToolbox_5G_FrameStructure_UL.html#Format_0_Ex01

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_PUCCH.html
- https://www.sharetechnote.com/html/5G/5G_UCI.html
- https://www.rfwireless-world.com/5G/5G-NR-UCI.html

## 7.8.2 PUCCH payload Tx processing

The **physical layer processes each UCI message** (click 4.12.1) to obtain **1 PUCCH codeword**. The **PUCCH codeword** is **further processed** to obtain **1 symbol sequence**. This **symbol sequence** is **mapped** to the corresponding **payload Resource Elements** of the **PUCCH transmission**. The **PUCCH transmission** is done in **antenna port 2,000** (click 4.11.4.8.3).

Along the **initial stages**, **each** of the variables "$i_{xyz}$" (e.g., "$i_{uci}$" or "$i_{cb}$") is the **bit index** that identifies **each bit** of the corresponding **bit sequence** "$xyz(i_{xyz})$" (e.g., "$uci(i_{uci})$" or "$cb(i_{cb})$"). **Each** such variable "$i_{xyz}$" ranges from **0** to the **bitlength** "$N_{bit}^{XYZ}$" (e.g., "$N_{bit}^{UCI}$" or "$N_{bit}^{CB}$") **minus 1** of the corresponding **bit sequence** "$xyz(i_{xyz})$".

Along the **final stages**, **each** of the variables "$j_{qrs}$" (e.g., "$j_{ms}$" or "$j_{fss}$") is the **symbol index** that identifies **each symbol** of the corresponding **symbol sequence** "$qrs(j_{qrs})$" (e.g., "$ms(j_{ms})$" or "$fss(j_{fss})$"). **Each** such variable "$j_{qrs}$" ranges from **0** to the **symbol length** "$N_{symb}^{QRS}$" (e.g., "$N_{symb}^{MS}$" or "$N_{symb}^{FSS}$") **minus 1** of the corresponding **symbol sequence** "$qrs(j_{qrs})$".

**Each UCI message** to be **sent** is processed by the **physical layer** in **up** to **12 stages**, as follows:

1. **Transform each UCI message** into **1 sequence** of "$N_{blk}^{CBS}$" **code block(s)**, where **each** has its **CRC**. **Each UCI message** "$uci(i_{uci})$" has **bitlength** "$N_{bit}^{UCI}$". **Each code block bit sequence** "$cb_n(i_{cb})$" has the same **bitlength** "$N_{bit}^{CB}$". The variable "$n$" is the **block index** within **code block sequence** "$cb_n(i_{cb})$", and "$n$" ranges from **0** to "$N_{blk}^{CBS}$-1".
   For more detail, click 7.8.2.2.
2. **Channel encode each code block** into **1 encoded block**, resulting in **1 sequence** of "$N_{blk}^{CBS}$" **encoded block(s)**. **Every code block bit sequence** "$cb_n(i_{cb})$" has **bitlength** "$N_{bit}^{CB}$". **Every encoded block bit sequence** "$eb_n(i_{eb})$" has the **same bitlength** "$N_{bit}^{EB}$", **substantially larger** than "$N_{bit}^{CB}$". The variable "$n$" is the **block index**, within the **encoded block sequence** "$eb_n(i_{eb})$" and "$n$" ranges from **0** to "$N_{blk}^{CBS}$-1".
   For more detail, click 7.8.2.3.

3. **Rate match: adjust** the **bitlength** of each **encoded block** to **match** the **available PUCCH transmission resources**, which produces **1 rate-matched block**. **Each encoded block** "$eb_n(i_{eb})$" has the same **bitlength** "$N_{bit}^{EB}$". **Each rate-matched block** "$rm_n(i_{rm})$" has **bitlength** "$N_{bit}^{RMn}$" (usually **smaller** than "$N_{bit}^{EB}$").
   For more detail, click 7.8.2.4.

4. **Concatenate sequentially every rate-matched block** into **1 codeword**. **Each rate-matched block** "$rm_n(i)$" has **bitlength** "$N_{bit}^{RMn}$". The resulting **codeword** "$cw(i)$" has **bitlength** "$N_{bit}^{CW}$", where "$N_{bit}^{CW} = \Sigma_n N_{bit}^{RMn}$".
   For more detail, click 7.8.2.5.

5. **Reorder** the **UCI codeword bits**: assign a **specific ordering** to the **codeword bits** corresponding to **CSI** of **2 parts** or **UCIs** with **different priority indexes**, to produce a **reordered codeword**. The **codeword** "$cw(i)$" and the **reordered codeword** "$rcw(i)$" have the **same bitlength** "$N_{bit}^{CW}$".
   For more detail, click 7.8.2.6.

6. **Scramble** the **reordered codeword** to produce the **scrambled codeword**. The **reordered codeword** "$rcw(i)$" and the **scrambled codeword** "$scw(i)$" have the **same bitlength** "$N_{bit}^{CW}$".
   For more detail, click 7.8.2.7.

7. **Modulate** (click 11.2.1) **every** "$Q_m$" **consecutive bits** into **1 symbol** of the corresponding **digital modulation constellation**, to obtain the **modulated sequence**. The **symbol length** "$N_{symb}^{MS}$" of the **modulated sequence** "$ms(j_{ms})$" is the **bitlength** "$N_{bit}^{UCI}$" or "$N_{bit}^{CW}$" divided by the **modulation order** "$Q_m$" (except in `format0`).
   For more detail, click 7.8.2.8.

8. **Frequency-spread** the **modulated sequence**, resulting in a **frequency-spread sequence** (click 11.20.2). The **modulated sequence** "$ms(j_{ms})$" has **symbol length** "$N_{symb}^{MS}$". The **frequency-spread sequence** "$fss(j_{fss})$" has **symbol length** "$N_{symb}^{FSS}$", that depends on the **PUCCH** `format`.
   For more detail, click 7.8.2.9.

9. **Cyclic shift** (**CS**) and/or **time-spread** (**TD-OCC**) **either** the **modulated sequence** or the **frequency-spread sequence** (depending on the `format`). When **time-spreading** is **applied**, the **result** is the **time-spread sequence**. The **resulting time-spread sequence** "$tss(j_{tss})$" has **symbol length** "$N_{symb}^{TSS}$", that is the **product** of the **previous symbol length** and the **time-spreading factor** "$N_{SF}^{PUCCH,n}$" of the corresponding **PUCCH** `format` "`n`":

   $N_{symb}^{TSS} = N_{SF}^{PUCCH,n} \bullet N_{symb}^{MS}$ **or**

   $N_{symb}^{TSS} = N_{SF}^{PUCCH,n} \bullet N_{symb}^{FSS}$, as corresponds.

   For more detail, click 7.8.2.10.

10. **Map each symbol either** from the **modulated sequence** "$ms(j_{ms})$", from the **frequency-spread sequence** "$fss(j_{fss})$" or from the **time-spread sequence** "$tss(j_{tss})$" (as corresponds) to the **PUCCH payload PREs** in **port** "$(sc, si, ap)_\mu$". The **sequency length** "$N_{symb}^{MS}$", "$N_{symb}^{FSS}$" or "$N_{symb}^{TSS}$" (respectively) is the **same** as the **quantity** of **PUCCH payload PREs** "$N_{RE}^{PUCCH\text{-}Pld,n}$".
    For more detail, click 7.8.2.11.

11. **DFT precode** (sometimes called "**transform precode**", click 4.12.12) the **PUCCH PREs**, resulting in **low-PAPR type 1 complex number sequence** "$r_{u,v}^{(\alpha,0,m)}(j)$".
    For more detail, click 7.8.2.12.
12. **Sequency (group) hop** and **cyclic shift hop** the **low-PAPR type 1 complex number sequence** "$r_{u,v}^{(\alpha,0,m)}(j)$".
    For more detail, click 7.8.2.13.

For a **mapping** between the **terms** (e.g., **protected block**) and the **symbols** (e.g., "$N_{blk}^{CBS}$") of **PUCCH processing** used in **this book** and the corresponding **terms** and **symbols** (e.g., ***A***, ***C***, …) used in the **TS documents** click 4.12.17.

For more detail on **PUCCH payload processing**, refer to clauses 6.3.2 and 6.4.1.3 of TS 38.211, clause 6.3 of TS 38.212, and clause 5.3.3 of TS 38.300.

For the **NR node functional blocks**, the **NR node physical functions**, the **payload processing** of the **physical binary channels**, the **protocol architecture** of the **New Radio** and the **data flow** through the **protocol architecture**, click 4.3.

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_UCI.html
- https://www.rfwireless-world.com/5G/5G-NR-UCI.html
- https://www.sharetechnote.com/html/5G/5G_PUCCH.html

### 7.8.2.1 UCI message bitlength

The **UCI message content** (click 7.7.2) is delivered **through PUCCH** as the **bit sequence** "$uci(i_{uci})$" with **bit index** "$i_{uci}$" and **bitlength** "$N_{bit}^{UCI}$".

For the **PUCCH total PREs** (i.e., the **quantity** of **PREs scheduled** for **each PUCCH transmission**) and for the **PUCCH payload PREs** (i.e., the **PUCCH total PREs minus** the **PUCCH DM-RS PREs**), click 7.6.5.1.

**Each PUCCH `format`** may convey **1 UCI message** (click 7.7) with a **different maximum bitlength**, as it is now detailed.

**PUCCH `format0`** and **`format1`**

Their **UCI message** "$uci(i_{uci})$" **bitlength** "$N_{bit}^{UCI}$" is **1** or **2 bits**: "$N_{bit}^{UCI} \in \{0,1\}$".

**PUCCH `format2`**, **`format3`** and **`format4`**

Their **UCI message** "$uci(i_{uci})$" **bitlength** "$N_{bit}^{UCI}$" ranges from **3** to "$N_{bit}^{UCI}{}_{max}^{2}$", "$N_{bit}^{UCI}{}_{max}^{3}$" and "$N_{bit}^{UCI}{}_{max}^{4}$". The values of "$N_{bit}^{UCI}{}_{max}^{2}$", "$N_{bit}^{UCI}{}_{max}^{3}$" and "$N_{bit}^{UCI}{}_{max}^{4}$" depending on the **`format`** and on several options, like the **symbol intervals** and **resource blocks** of the **PUCCH transmission**, **interlaced mapping**, **OCCs** and **code rate**.

### 7.8.2.2 PUCCH segmentation and/or CRC attachment

This consist of **transforming each UCI message** into **1 sequence** of "$N_{blk}^{CBS}$" **code block(s)**, where a **specific CRC** "$CRC_n^{CB(q)}$" **can** be included into **each** and **every code block**. **Each UCI message** "$uci(i_{uci})$" has **bitlength** "$N_{bit}^{UCI}$". **Each code block bit sequence** "$cb_n(i_{cb})$" has the **same bitlength** "$N_{bit}^{CB}$". The variable "$n$" is the **block index** within the **code block sequence** "$cb_n(i_{cb})$", and "$n$" ranges from **0** to "$N_{blk}^{CBS}$-1"

**Each PUCCH `format` may perform different code block segmentation of the UCI message**, as it is now detailed.

**PUCCH `format0` and `format1`**

There is **no code block segmentation** and **no CRC attachment**. Consequently:

$N_{blk}^{CBS}$=1, (i.e., "$n$=0")

$cb_0(i_{cb}) = uci(i_{uci})$

$N_{bit}^{CB} = N_{bit}^{UCI} \in \{0,1\}$

**PUCCH `format2`, `format3` and `format4`**

We have the following **3 cases**:

**When** ($N_{bit}^{UCI} \leq 11$):

There is **no segmentation** and **no CRC attachment**. Consequently:

$N_{blk}^{CBS}$=1, (i.e., "$n$=0")

$cb_0(i) = uci(i)$

$N_{bit}^{CB} = N_{bit}^{UCI}$

**When** ($12 \leq N_{bit}^{UCI} \leq 1012$) and (($12 \leq N_{bit}^{UCI} \leq 360$) or ($N_{bit}^{RM} \leq 1088$)):

There is **no segmentation** but there is **CRC attachment** (see below). Consequently:

$N_{blk}^{CBS}$=1, (i.e., "$n$=0")

$cb_0(i) = uci(i)$ for $i \in \{0..(N_{bit}^{UCI}-1)\}$

$N_{bit}^{CB} = N_{bit}^{UCI} + N_{bit}^{CRC\text{-}CB}$

**When** ($1013 \leq N_{bit}^{UCI}$) or (($360 \leq N_{bit}^{UCI}$) and ($1088 \leq N_{bit}^{RM}$)):

There is **segmentation** into **2 code blocks** and there is **CRC attachment** (see below). Consequently:

$N_{blk}^{CBS}$=2, (i.e., "$n \in \{0,1\}$")

$N_{bit}^{CB} = ceiling(N_{bit}^{UCI}/2)$

$cb_0(i) = 0$ for $i \in \{0..(N_{bit}^{CB}-floor(N_{bit}^{UCI}/2)-1)\}$

$cb_0(i) = uci(i)$ for $i \in \{(N_{bit}^{CB}-floor(N_{bit}^{UCI}/2))..(N_{bit}^{CB}-1)\}$

$cb_1(i-N_{bit}^{CB}) = uci(i)$ for $i \in \{N_{bit}^{CB}..(N_{bit}^{UCI}-1)\}$

Some **relevant remarks** on the above data are:

- The **2 code blocks** have the **same bitlength** "$N_{bit}^{CB}$".
- In case that "$N_{bit}^{UCI}$" is **odd**, a **padding bit** set to **0** is added as the **first bit** of the **first code block**.

**CRC attachment**

When there is **CRC** attachment, a **CRC** is **attached** at the **end** of **each** of the **1** or **2 code block(s)**. This **CRC** is computed over the bits **0** to "$N_{bit}^{CB}$-1" of **each code block**. The **CRC** is an **integral part** of **each code block**.

The **CRC bitlength** and **generator polynomial** depends on the **UCI bitlength**, as follows:

- **When** ($12 \leq N_{bit}^{UCI} \leq 19$)

  The **CRC bitlength** is **6 bits**, and the **generator polynomial** is "$g_{crc6}(i_{cb})$".

- **When** ($20 \leq N_{bit}^{UCI}$)

  The **CRC bitlength** is **11 bits**, and the **generator polynomial** is "$g_{crc11}(i_{cb})$".

For more detail on **PUCCH code block segmentation and CRC attachment**, refer to clauses 6.3.1.2, 5.2.1 and 5.1 of TS 38.212.

### 7.8.2.3 PUCCH channel coding

This consists of **channel encoding** **each** **code block** into **1 encoded block**, resulting in **1 sequence** of "$N_{blk}^{CBS}$" **encoded block(s)**. **Every** **code block** **bit sequence** "$cb_n(i_{cb})$" has the **same bitlength** "$N_{bit}^{CB}$". **Every** **encoded block** **bit sequence** "$eb_n(i_{eb})$" has the **same bitlength** "$N_{bit}^{EB}$", **substantially larger** than "$N_{bit}^{CB}$". The variable "$n$" is the **block index**, within the **encoded block sequence** "$eb_n(i_{eb})$" and "$n$" ranges from **0** to "$N_{blk}^{CBS}$-1".

The **channel coding** used for **each UCI message** depends on the **transmission** being through **PUCCH** or through **PUSCH**, on the **PUCCH** `format` and on the **UCI message bitlength** "$N_{bit}^{UCI}$". The following table summarizes the different cases.

| | | UCI channel code | | |
|---|---|---|---|---|
| | | UCI through PUCCH | | UCI through PUSCH |
| Bitlength "$N_{bit}^{UCI}$" of UCI message | Bitlength "$N_{bit}^{CW}$" of UCI codeword | `format0` `format1` | `format2` `format3` `format4` | |
| $N_{bit}^{UCI} = 1$ | $N_{bit}^{CW} = 1$ | None | N/A | Repetition |
| $N_{bit}^{UCI} = 2$ | $N_{bit}^{CW} = 2$ | None | N/A | Simplex |
| $3 \leq N_{bit}^{UCI} \leq 11$ | $3 \leq N_{bit}^{CW} \leq 18$ | N/A | Reed-Muller | Reed-Muller |
| $12 \leq N_{bit}^{UCI} \leq 1{,}706$ | $18 \leq N_{bit}^{CW} \leq 864$ | N/A | CRC & Polar | CRC & Polar |

For the specifics of **channel coding** in the different **PUCCH** `formats`, click 7.8.2.3.1.

For **repetition** **coding**, click 7.8.2.3.2.

For **simplex** **coding**, click 7.8.2.3.3.

For **Reed-Muller** **coding**, click 11.2.3.5.

For **polar** **coding**, click 11.2.3.4.

For more detail on **UCI message channel coding**, refer to clause 5.3.3 of TS 38.300 and clauses 6.3.1.3, 5.3.1 and 5.3.3 of TS 38.212.

#### 7.8.2.3.1 Channel coding in different `formats`

**Each PUCCH** `format` may use **different channel coding techniques**, as it is now detailed.

**PUCCH `format0`**

It does **not** use **channel coding**. Consequently:

$N_{blk}^{CBS}$=1 (i.e., $n$=0)

$eb_0(i_{eb})=cb_0(i_{cb})=uci(i_{uci})$

$N_{bit}^{EB}=N_{bit}^{CB0}=N_{bit}^{UCI} \in \{0,1\}$

For its transmission over **2 symbol intervals**, the **low-PAPR type 1 sequence** is **repeated** (click 7.8.2.11) in **each** of the **2 symbol intervals**. This is **not** a **repetition channel coding** because it does **not** allow a **majority decision**. This is rather a **diversity technique** in the **time domain**, and the **2 Rx sequences** in **both symbol intervals** are **analogically added**.

**PUCCH `format1`**

It does **not** use **channel coding**. Consequently:

$N_{blk}^{CBS}$=1 (i.e., $n$=0)

$eb_0(i_{eb}) = cb_0(i_{cb}) = uci(i_{uci})$

$N_{bit}^{EB} = N_{bit}^{CB0} = N_{bit}^{UCI} \in \{0,1\}$

**PUCCH `format2`, `format3` and `format4`**

**`format2`**, **`format3`** and **`format4`** use **Reed-Muller coding** (click 11.2.3.5) for **each code block** with **3** to **11 bits** and **Polar coding** (click 11.2.3.4) for **each code block** with **12** or **more bits**.

The **maximum usable code rate** is configured (click 7.6.5.3.3) to **each UE** with **`maxCodeRate`** from the **`PUCCH-FormatConfig`** (click 7.9.10) **instance** of **`format2`**, **`format3`** or **`format4`**. A **second maximum usable code rate** may be configured with **`maxCodeRateLP-r17`** from the **`PUCCH-FormatConfigExt-r17`** (click 7.9.11) **instance** of **`format2Ext-r17`**, **`format3Ext-r17`** or **`format4Ext-r17`**. **Each** of them may take the values **`0.08`**, **`0.15`**, **`0.25`**, **`0.35`**, **`0.45`**, **`0.6`** or **`0.8`**.

For more detail on **`format2`**, **`format3`** and **`format4`** **channel coding**, refer to clause 6.3.1.3 of TS 38.212.

### 7.8.2.3.2 Repetition coding

The **encoding** of a **1-bit code block** "$cb(0)$" to produce an **encoded block** "$eb(i_{eb})$" with **bitlength** "$N_{bit}^{EB}=Q_m$" bits, is shown in the following table:

| $Q_m$ | $N_{bit}^{EB}$ | encoded block | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | $eb(0)$ | $eb(1)$ | $eb(2)$ | $eb(3)$ | $eb(4)$ | $eb(5)$ | $eb(6)$ | $eb(7)$ |
| 1 | 1 | $cb(0)$ | | | | | | | |
| 2 | 2 | $cb(0)$ | $y$ | | | | | | |
| 4 | 4 | $cb(0)$ | $y$ | $x$ | $x$ | | | | |
| 6 | 6 | $cb(0)$ | $y$ | $x$ | $x$ | $x$ | $x$ | | |
| 8 | 8 | $cb(0)$ | $y$ | $x$ | $x$ | $x$ | $x$ | $x$ | $x$ |

The **bit values** "$x$" and "$y$" within the **encoded block** are **placeholders** until the **scrambling stage** (click 4.12.9), where they are used to **maximize** the **Euclidean distance between** the **modulation symbols** that **carry** the information bits.

Recall that "$Q_m$" is the **modulation order** (click 11.2.1).

**Repetition coding** is **1 type** of **block code** to **encode small block lengths** (refer to clause 5.3.3 of TS 38.212).

For more detail on **repetition coding**, refer to clause 5.3.3.1 of TS 38.212.

#### 7.8.2.3.3 Simplex coding

The **encoding** of a **2-bit code block** "$cb(0)$, $cb(1)$" to produce an **encoded block** "$eb(i_{eb})$" with **bitlength** "$N_{bit}^{EB}=3 \bullet Q_m$" bits, is shown in the following table:

| $Q_m$ | $N_{bit}^{EB}$ | encoded block<br>$eb(0)$ $eb(1)$ … $eb(3Q_m-2)$ $eb(3Q_m-1)$ | | |
|---|---|---|---|---|
| 1 | 3 | $cb(0)$ $cb(1)$ | $cb(2)$ | |
| 2 | 6 | $cb(0)$ $cb(1)$ | $cb(2)$ $cb(0)$ | $cb(1)$ $cb(2)$ |
| 4 | 12 | $cb(0)$ $cb(1)$ $x$ $x$ | $cb(2)$ $cb(0)$ $x$ $x$ | $cb(1)$ $cb(2)$ $x$ $x$ |
| 6 | 18 | $cb(0)$ $cb(1)$ $x$ $x$ $x$ $x$ | $cb(2)$ $cb(0)$ $x$ $x$ $x$ $x$ | $cb(1)$ $cb(2)$ $x$ $x$ $x$ $x$ |
| 8 | 24 | $cb(0)$ $cb(1)$ $x$ $x$ $x$ $x$ $x$ $x$ | $cb(2)$ $cb(0)$ $x$ $x$ $x$ $x$ $x$ $x$ | $cb(1)$ $cb(2)$ $x$ $x$ $x$ $x$ $x$ $x$ |

The **bit values** "$x$" within the **encoded block** are **placeholders** until the **PUSCH scrambling stage** (click 4.12.9), where they are used to **maximize** the **Euclidean distance between** the **modulation symbols** that **carry** the information bits. For the PUCCH scrambling stage, click

The value of "$cb(2)$" is obtained as:

$$cb(2) = (cb(0) + cb(1)) \bmod 2 = cb(0) \oplus cb(1)$$

Recall that "$Q_m$" is the **modulation order** (click 11.2.1).

**Simplex coding** is **1 type** of **block code** to **encode small block lengths** (refer to clause 5.3.3 of TS 38.212).

For more detail on **simplex coding**, refer to clause 5.3.3.2 of TS 38.212.

### 7.8.2.4 PUCCH rate-matching

It consists of **adjusting** (usually **reducing**) the **bitlength** of **each encoded block** to **match** the **available PUCCH transmission resources**, producing **1 rate-matched block**. **Every encoded block** "$eb_n(i_{eb})$" has **bitlength** "$N_{bit}^{EB}$". **Each rate-matched block** "$rm_n(i_{rm})$" has **bitlength** "$N_{bit}^{RBn}$" (usually **smaller** than "$N_{bit}^{EB}$").

**Each PUCCH** `format` may use **different rate matching techniques**, as it is now detailed.

**PUCCH** `format0` and `format1`

They do **not** use **rate matching**. Consequently:

$$rm_0(i_{rm}) = eb_0(i_{eb})$$

$$N_{bit}^{RB0} = N_{bit}^{EB} = N_{bit}^{CB0} = N_{bit}^{UCI} \in \{0,1\}$$

**PUCCH** `format2`, `format3` and `format4`

They **use rate-matching**.

For more detail on `format2`, `format3` and `format4` **rate matching**, refer to clause 6.3.1.4 of TS 38.212.

### 7.8.2.5 PUCCH concatenation of rate-matched blocks into 1 codeword

It consists of **concatenating sequentially every rate-matched block** into **1 codeword**. **Each rate-matched block** "$rm_n(i_{rm})$" has **bitlength** "$N_{bit}^{RBn}$". The resulting **codeword** "$cw(i_{cw})$" has **bitlength** "$N_{bit}^{CW}$", where "$N_{bit}^{CW} = \Sigma_n N_{bit}^{RBn}$".

**Each PUCCH** `format` may use a **different rate-matched block concatenation**, as it is now detailed.

**PUCCH** `format0` and `format1`

They do **not** use **concatenation** of **rate-matched blocks**, because they do **not** use **code block segmentation** (click 4.12.4). Consequently:

$$cw(i_{cw}) = rm_0(i_{rm})$$

$$N_{bit}^{CW} = N_{bit}^{RB0} = N_{bit}^{EB} = N_{bit}^{CB0} = N_{bit}^{UCI} \in \{0,1\}$$

**PUCCH** `format2`, `format3` and `format4`

They use **concatenation** of **rate-matched blocks**.

For more detail on `format2`, `format3` and `format4` **rate-matched block concatenation**, refer to clause 6.3.1.5 of TS 38.212.

### 7.8.2.6 PUCCH reordering of UCI coded bits

It consists of **assigning** a **specific ordering** to the **codeword bits** that have been produced from **CSI** of **2 parts** or **UCIs** with **different priority indexes**, . The result of the reordering is the **reordered codeword**. The **codeword** "$cw(i)$" and the **reordered codeword** "$rcw(i)$" have the **same bitlength** "$N_{bit}^{CW}$".

Notice that in TS 38.212 this is called "**multiplexing**" instead of **reordering**.

**Each PUCCH** `format` may use **different reordering** of **UCI coded bits**, as it is now detailed.

**PUCCH** `format0` and `format1`

They do **not** use **reordering** of **UCI coded bits**. Consequently:

$$rcw(i_{cw}) = cw(i_{cw})"$$

$$N_{bit}^{CW} = N_{bit}^{RB0} = N_{bit}^{EB} = N_{bit}^{CB0} = N_{bit}^{UCI} \in \{0,1\}$$

**PUCCH** `format2`, `format3` and `format4`

They **may use reordering** of **UCI coded bits**. When they do **not** use **reordering** of **UCI coded bits** "$rcw(i) = cw(i)$".

For more detail on `format2`, `format3` and `format4` **reordering** of **UCI coded bits**, refer to clause 6.3.1.6 of TS 38.212.

### 7.8.2.7 PUCCH scrambling

It consists of **scrambling** the **reordered codeword** to produce the **scrambled codeword**. The **reordered codeword** "$rcw(i_{cw})$" and the **scrambled codeword** "$scw(i_{cw})$" have the **same bitlength** "$N_{bit}^{CW}$".

**Each PUCCH** `format` may use a different **scrambling**, as it is now detailed.

**PUCCH `format0` and `format1`**

`format0` and `format1` do **not** use **any scrambling**. Consequently:

$scw(i_{cw}) = rcw(i_{cw})$

$N_{bit}^{CW} = N_{bit}^{RB0} = N_{bit}^{EB} = N_{bit}^{CB0} = N_{bit}^{UCI} \in \{0,1\}$".

**PUCCH `format2`, `format3` and `format4`**

**Scrambling** is based on the **pseudo-random binary sequence** "$c(n)$" (click 11.20.1), whose variable "$c_{init}$" is computed as:

$c_{init}$ = **C-RNTI** • $2^{15} + n_{ID}$

The variable "$n_{ID}$" is `dataScramblingIdentityPUSCH` from `PUSCH-Config` (click 7.15.5), if **present**. If it is **absent**, "$n_{ID}$" is the **physical cell identity** (click 3.5).

In the **scrambling procedure**, the "$y$" **place-holder bits** from the **channel coding stage** (click 7.8.2.3) are replaced by a **repetition** of the **previously scrambled bit**.

For more detail on **PUCCH `format2`, `format3` and `format4` scrambling**, refer to clauses 6.3.2.5.1 or 6.3.2.6.1 of TS 38.211 (**both** clauses are **identical**).

### 7.8.2.8 PUCCH modulation

It consists of **modulating** (click 11.2.1) **every** "$Q_m$" **consecutive bits** of the **scrambled codeword** onto **1 symbol** of the corresponding **digital modulation constellation**, to obtain the **modulated sequence**. The **scrambled codeword** "$scw(i_{cw})$" has **bitlength** "$N_{bit}^{CW}$". The **modulated sequence** "$ms(j_{ms})$" has **symbol length** "$N_{symb}^{MS}=N_{bit}^{CW}/Q_m$", where "$Q_m$" is the **modulation order**.

The **symbol length** "$N_{symb}^{MS}$" of the **modulated sequence** "$ms(j_{ms})$" **must** be a **multiple** of the **quantity** of **PUCCH subcarriers** "$M_{sc}^{PUCCH,n}$", because "$ms(j_{ms})$" **must occupy** an **integer quantity** of **PUCCH symbol intervals**, where "`n`" is the corresponding **PUCCH `format`**.

**Each PUCCH `format`** may use different **modulation(s)**, as it is now detailed.

**PUCCH `format0`**

It does **not** use **any modulation**. It **rather maps each 1-bit** or **2-bits UCI message** "$uci(i_{uci})$" onto **1 low-PAPR type 1** "$r_{u,v}^{(\alpha,0,m)}(j_{ms})$" (click 11.20.2) **complex number** (i.e., **symbol**) **sequence**:

$ms(j_{ms}) = f(uci(i_{uci})) = r_{u,v}^{(\alpha,0,m)}(j_{ms})$

For consistency with the other `formats`, the `format0` **symbol sequence** "$ms(j_{ms})$" is **also called** the **modulated sequence**.

The **UCI message bit sequence** "$uci(i_{uci})$" has **bitlength** "$N_{bit}^{UCI} \in \{1,2\}$". The `format0` **modulated sequence** "$ms(j_{ms})=r_{u,v}^{(\alpha,0,m)}(j_{ms})$" **length** is "$N_{symb}^{MS}$" **symbols**.

The different variables are:

- "$N_{symb}^{MS}$" is the `format0` **modulated sequence length**. It is the **quantity** of `format0` **PUCCH subcarriers** "$M_{sc}^{PUCCH,0}$", which for `format0` is the **same** as the **payload subcarriers**:

  $N_{symb}^{MS} = M_{sc}^{PUCCH,0} = M_{sc}^{PUCCH\text{-}Pld,0}$

In **both FR1** and **FR2-1** "$M_{sc}^{PUCCH,0}=12$". In **FR2-2** "$M_{sc}^{PUCCH,0}=12M_{RB}^{PUCCH,0}$", where "$1 \leq M_{RB}^{PUCCH,0} \leq 16$", and "$M_{RB}^{PUCCH,0}$" is `nrofPRBs-r17` from `PUCCH-ResourceExt-v1610` (click 7.9.9).

- "$m$" is determined by the **sequence length** "$N_{symb}^{MS}$" using the expression "$M_{ZC}(0, m)=N_{symb}^{MS}$" (click 11.20.2).
- "$u$", "$v$" are (respectively) the **group sequence index** and the **sequence index**. They are determined by the **sequence (group) hopping** (click 7.8.2.13.1).
- "$\alpha$" is the **sequential symbol shift**. It is **determined** by the **cyclic shift hopping** (click 7.8.2.13.2), **parametrized** by variable "$m_{CS}$", whose value depends on the **UCI contents**, as it is now indicated.

For an **UCI message** with **SR** or **LRR**, only **positive SR** or **LRR** are **transmitted**, and "$m_{CS}=0$".

For **HARQ-ACK(s)**, the value of "$m_{CS}$" is provided by the following two tables (9.2.3-**3** and **-4** of TS 38.213):

| 1 HARQ-ACK | Bit value of HARQ-ACK | |
|---|---|---|
| | 0 | 1 |
| $m_{CS}$ | 0 | 6 |

| 2 HARQ-ACKs | Bit value of the 2 HARQ-ACKs | | | |
|---|---|---|---|---|
| | 00 | 01 | 10 | 11 |
| $m_{CS}$ | 0 | 3 | 6 | 9 |

where bit value "**0**" is a **HARQ-NACK**, while bit value "**1**" is a **HARQ-ACK**.

For more detail on **PUCCH** `format0` **transmission** of **HARQ-ACK**, refer to clause 9.2.3 of TS 38.213.

For more detail on **PUCCH** `format0` **transmission** of **SR** or **LRR**, refer to clause 9.2.4 of TS 38.213.

For more detail on **PUCCH** `format0` **sequence generation**, refer to clause 6.3.2.3.1 of TS 38.211.

**PUCCH** `format1`

It uses **BPSK modulation** for **1-bit UCI** and **QPSK** for **2-bit UCI**. The **scrambled codeword** "$scw(i_{cw})$" is the same as the **UCI message bit sequence** "$uci(i_{uci})$", and it has **bitlength** of **1** or **2** bits. The `format1` **modulated sequence** "$ms(0)$" has a **symbol length**: "$N_{symb}^{MS}$" of **only 1 symbol**:

$$N_{symb}^{MS} = N_{bit}^{CW}/Q_m = 1$$

For the **inter-slot repeat option** (click 7.6.5.3.5), the **same symbol** "$ms(0)$" is used for **all** the **repeated slots**.

For more detail on **PUCCH** `format1` **modulation**, refer to clause 6.3.2.4.1 of TS 38.211.

**PUCCH** `format2`

It uses **QPSK modulation**. The **scrambled codeword** "$scw(i_{cw})$" has **bitlength** "$N_{bit}^{CW}$". The `format2` **modulated sequence** has **symbol length** "$N_{symb}^{MS}=N_{bit}^{CW}/Q_m=N_{bit}^{CW}/2$".

Its **symbol length** "$N_{symb}^{MS}$" is also the **quantity** of `format2` **payload PUCCH subcarriers** "$M_{sc}^{PUCCH\text{-}Pld,2}$":

$$N_{symb}^{MS} = N_{bit}^{CW}/2 = M_{sc}^{PUCCH\text{-}Pld,2}$$":

For more detail on **PUCCH** `format2` **modulation**, refer to clause 6.3.2.5.2 of TS 38.211.

**PUCCH** `format3` and `format4`

They use **QPSK**, **except** if `pi2BSK` from `PUCCH-FormatConfig` (click 7.9.10) is **present**, in which case they use **π/2-BPSK**.

The **scrambled codeword** "$scw(i_{cw})$" has **bitlength** "$N_{bit}^{CW}$". The `format2` and `format3` **modulated sequence** has **symbol length** "$N_{symb}^{MS}=N_{bit}^{CW}/Q_m$", which is "$N_{symb}^{MS}=N_{bit}^{CW}/2$" for **QPSK** and "$N_{symb}^{MS}=N_{bit}^{CW}$" for **π/2-BPSK**.

Their **modulated sequence length** "$N_{symb}^{MS}$" is a **multiple** of the **quantity** of **PUCCH subcarriers** "$M_{sc}^{PUCCH,3,4}$", which is the **same** as the **PUCCH payload subcarriers** "$M_{sc}^{PUCCH\text{-}Pld,3,4}$":

$$N_{symb}^{MS} = M_{sc}^{PUCCH,3} = M_{sc}^{PUCCH\text{-}Pld,3}$$

$$N_{symb}^{MS} = M_{sc}^{PUCCH,4} = M_{sc}^{PUCCH\text{-}Pld,4}$$

Their **modulated sequence length** "$N_{symb}^{MS}$" is their **time-spread sequence length** "$N_{symb}^{TSS}$" **divided** by their **time-spreading factor** "$N_{SF}^{PUCCH,3,4}$":

$$N_{symb}^{MS} = N_{symb}^{TSS} / N_{SF}^{PUCCH,3,4}$$

**QPSK** has **double modulation order** (**2** bits/symbol) than **π/2-BPSK** (**1** bit/symbol), but **π/2-BPSK** is more **robust** to **noise**, and it has a **very low PAPR**.

For more detail on **PUCCH** `format3` and `format4` **modulation**, refer to clause 6.3.2.6.2 of TS 38.211.

### 7.8.2.9 PUCCH frequency-spreading with low-PAPR type 1 sequence

It consists of **spreading** the **modulated sequence** in the **frequency domain**, resulting in a **longer frequency-spread low-PAPR type 1** (click 11.20.2) **sequence**. The **modulated sequence** "$ms(j_{ms})$" has **symbol length** "$N_{symb}^{MS}$". The **symbol length** "$N_{symb}^{FSS}$" of the **frequency-spread sequence** "$fss(j_{fss})$" depends on the **PUCCH** `format`.

The **symbol length** "$N_{symb}^{FSS}$" of the **frequency-spread sequence** "$fss(j_{fss})$" **must** be a **multiple** of the **quantity** of **PUCCH subcarriers** "$M_{sc}^{PUCCH,n}$", because "$fss(j_{fss})$" **must occupy** an **integer quantity** of **PUCCH symbol intervals**, where "`n`" is the corresponding **PUCCH** `format`.

**Some PUCCH** `formats` may use **frequency-spreading** with a **low-PAPR type 1** "$r_{u,v}^{(\alpha,\delta,m)}(j_{fss})$" **complex number sequence** (with "$\delta$=0") as it is now detailed.

**PUCCH** `format0`

It does **not** use **frequency-spreading**. Consequently, it does **not** have a **frequency-spread sequence** "$fss(j_{fss})$".

However, as indicated in 7.8.2.8, `format0` **maps each 1-bit** or **2-bits UCI message** onto **1 low-PAPR type 1** (click 11.20.2) **complex number sequence** "$r_{u,v}^{(\alpha,0,m)}(j_{ms})$". This is not **frequency-spreading**, but the result is **conceptually equivalent**.

For more detail on **PUCCH `format0` sequence generation**, refer to clause 6.3.2.3.1 of TS 38.211.

**PUCCH `format1`**

It **spreads** its **1 modulated symbol** "$ms(0)$" (click 7.8.2.8) in the **frequency domain** by **multiplying** "$ms(0)$" by **each successive symbol** of the **low-PAPR type 1** (click 11.20.2) "$r_{u,v}^{(\alpha,0,m)}(j_{fss})$" **complex number sequence**:

$$fss(j_{fss}) = r_{u,v}^{(\alpha,0,m)}(j_{fss}) \bullet ms(0)$$

where:

- "$ms(0)$" is the **PUCCH `format1` single modulated symbol**.
- "$r_{u,v}^{(\alpha,0,m)}(j_{fss})$" is the **frequency-spreading low-PAPR type 1 sequence**.

  Its **symbol length** is "$M_{ZC}(0, m)$", which is the same as **quantity** of **subcarriers** "$M_{sc}^{PUCCH,1}$" of this **`format1` PUCCH transmission**. Since the value of "$M_{ZC}(0, m)$" is known, the value of "$m$" is determined by it.

  Its **group sequence index** "$u$" and its **sequence index** "$v$" are determined by the **sequence (group) hopping** (click 7.8.2.13.1).

  Its **shift** "$\alpha$" is **determined** by the **cyclic shift hopping** (click 7.8.2.13.2).
- "$fss(j_{fss})$" is the resulting **frequency-spread sequence**.
- "$j_{fss}$" is the **symbol index** of the **frequency-spreading sequence** "$r_{u,v}^{(\alpha,0,m)}(j_{fss})$" and of the **frequency-spread sequence** "$fss(j_{fss})$". It **ranges** from **0** to **quantity** of **PUCCH subcarriers** (minus 1) "$M_{sc}^{PUCCH,1}-1$".

For more detail on **PUCCH `format1` frequency-spreading**, refer to clause 6.3.2.4.1 of TS 38.211.

**PUCCH `format2`**

It does **not** use **frequency-spreading**. Consequently, it does **not** have a **frequency-spread sequence** "$fss(j_{fss})$".

Additionally, its **final waveform** is **not** a **low-PAPR type 1 sequence**.

**PUCCH `format3` and `format4`**

They do **not** use **frequency-spreading**. Consequently, they do **not** have a **frequency-spread sequence** "$fss(j_{fss})$".

Even though they do **not** use **frequency-spreading**, their **final waveform** is a **low-PAPR type 1 sequence**, as a result of their **specific DFT precoding** (click 7.8.2.12).

### 7.8.2.10 PUCCH cyclic shifts and time-spreading (TD-OCC) for UE multiplexing

**All** the **PUCCH `formats`** may allow to **multiplex** the **UCI messages** from **several UEs** over the **same PUCCH transmission resources**. The **2 multiplexing techniques** that **may** be **used** are:

- **Cyclic shift (CS)**
- **Time-domain (TD) orthogonal cover codes (OCCs)**

**Cyclic shift (CS)**

It consists of **modifying** the **shift** "$\alpha$" of the **low-PAPR type 1 sequence** "$r_{u,v}^{(\alpha,\delta,m)}(n)$" (click 11.20.2) in a **different way** for **each UE**. The **cyclic shifted sequences** of **different UEs** are **orthogonal**, which implements **code-division multiple access** (click 11.9).

**Time-domain (TD) orthogonal cover codes (OCCs)**

It consists of **time-spreading either** the **modulated sequence** "$ms(j_{ms})$" or the **frequency-spread sequence** "$fss(j_{fss})$" (as corresponds) with a **specific time-spreading sequence** "$w_s(j_w)$". The **time-spreading sequences** "$w_s(j_w)$" of **different UEs** are **orthogonal**, which implements **code-division multiple access** (click 11.9). The **result** of the **time-spreading** is the **time-spread sequence** "$tss(j_{tss})$".

- "$s$" is the **identifier** of **each time-spreading sequence** that **can** be used in a **given** `format`. **Each UE** should use a **different** value of "$s$".
- "$j_w$" is the **symbol index** of the **time-spreading sequence** "$w_s(j_w)$". It **ranges** from **0** to the corresponding **time-spreading factor** "$N_{SF}^{PUCCH,n}$" minus 1:

  $j_w \in \{0..(N_{SF}^{PUCCH,n} - 1)\}$

- "$N_{SF}^{PUCCH,n}$" is the **time-spreading factor** of the corresponding **PUCCH** `format` "n". It is the **quantity** of **mapped symbol intervals after** the **time-spreading** for **each mapped symbol interval before** the **time-spreading**. Consequently, it is **also** the **ratio** between the **symbol length** "$N_{symb}^{MS}$" of the **modulated sequence** "$ms(j_{ms})$" or the **symbol length** "$N_{symb}^{FSS}$" of the **frequency-spread sequence** "$fss(j_{fss})$" (as corresponds) and the **symbol length** "$N_{symb}^{TSS}$" of the **resulting time-spread sequence** "$tss(j_{tss})$":

  $N_{SF}^{PUCCH,n} = N_{symb}^{TSS} / N_{symb}^{MS}$, **or**,

  $N_{SF}^{PUCCH,n} = N_{symb}^{TSS} / N_{symb}^{FSS}$, as corresponds,

- "$j_{tss}$" is the **symbol index** of the **time-spread sequence** "$tss(j_{tss})$". It **ranges** from **0** to the **symbol length** "$N_{symb}^{TSS}$", minus 1, of the **time-spread sequence** "$tss(j_{tss})$":

  $j_{tss} \in \{0..(N_{symb}^{TSS}-1)\}$

- "$j_{ms}$" is the **symbol index** of the **modulated sequence** "$ms(j_{ms})$". It **ranges** from **0** to "$N_{symb}^{MS}$-1":

  $j_{ms} \in \{0..(N_{symb}^{MS}-1)\}$

- "$j_{fss}$" is the **symbol index** of the **frequency-spread sequence** "$fss(j_{fss})$". It **ranges** from **0** to "$N_{symb}^{FSS}$-1":

  $j_{fss} \in \{0..(N_{symb}^{FSS}-1)\}$

- "$N_{symb}^{TSS}$" is the **symbol length** of the **time-spread sequence** "$tss(j_{tss})$". It is the **time-spreading factor** "$N_{SF}^{PUCCH,n}$" multiplied by the **symbol length** of the **sequence before** the **time-spreading**:

  $N_{symb}^{TSS} = N_{SF}^{PUCCH,n} \bullet N_{symb}^{MS}$ **or**

  $N_{symb}^{TSS} = N_{SF}^{PUCCH,n} \bullet N_{symb}^{FSS}$, as corresponds.

"$N_{symb}^{TSS}$" is also the **quantity** of **PUCCH payload PREs** for the corresponding `format` "`n`":

$$N_{symb}^{TSS} = N_{RE}^{PUCCH\text{-}Pld,n}$$

- **Each** of "$N_{symb}^{MS}$", "$N_{symb}^{FSS}$" and "$N_{symb}^{TSS}$" **must** be a **multiple** of the **quantity** of **PUCCH subcarriers**, because **each** of their **3 corresponding sequences must fit** in an **integer quantity** of **symbol intervals**. Consequently, **each** of them **must** be a **multiple** of **12** (the **quantity** of **subcarriers** in a **resource block**).

**TD-OCCs** may be used **with**, or **without**, **interlaced transmission**. **TD-OCCs** are an **integral** part of **PUCCH operation**.

The **TS documents** use the term "**block-wise spreading**" or "**block-wise time-spreading**" for what is just called "**time-spreading**" in this book.

The **NR node demultiplexes** the **UCI message** corresponding to **each UE**.

**Multiplexing technique(s) for each PUCCH `format`**

**Each PUCCH `format`** may use **different UE multiplexing technique(s)**, as it is now detailed.

**PUCCH `format0`**

It **only** uses **cyclic shift multiplexing**. It does **not** use **OCC multiplexing** (which is used by **all** the **other PUCCH `formats`**). Consequently, it does **not** have a **time-spread sequence** "$tss(j_{tss})$", and it **only** has the **`format0` modulated sequence** "$ms(j_{ms})$".

For a **1-bit UCI message**, **1 different shift** is used for **each** of the **2 bit-values**. Since the **length** of the **low-PAPR type 1 sequence** of **`format0`** is **12**, this allows for **up to 6 UEs multiplexed** over the **same resources**.

**PUCCH `format1`**

It uses **both cyclic shift multiplexing** and **OCC multiplexing**. This combination provides **`format1`** with the **highest UE multiplexing capacity**:

- **Without frequency hopping** (click 7.6.5.3)

  **Up to 84 UEs multiplexed** over the **same resources**.

- **With frequency hopping**

  **Up to 36 UEs multiplexed** over the **same resources**.

For the **OCC multiplexing**, **`format1`** is **time-spread** by **multiplying every symbol** of its **frequency-spread sequence** "$fss(j_{fss})$" (click 7.8.2.9) by **each successive symbol** of the corresponding **time-spreading sequence** "$w_s(j_w)$":

$$tss(j_w \bullet M_{sc}^{PUCCH,1} + j_{fss}) = w_s(j_w) \bullet fss(j_{fss})$$

where:

- "$w_s(j_w)$" for **`format1`** is provided in table 6.3.2.4.1-2 of TS 38.211.
- "$s$" for **`format1`** is determined in clause 9.2.1 of TS 38.213.
- "$j_{fss}$" **ranges** from **0** to "$N_{symb}^{FSS}$-1". Consequently, for **`format1`** "$j_{fss}$" ranges from **0** to "$M_{sc}^{PUCCH,1}$-1" (see below).

- "$j_{tss}$" for `format1` is a **function** of "$j_w$" (the **symbol index** of its **time-spreading sequence** "$w_s(j_w)$", see above) and of "$j_{fss}$" (the **symbol index** of its **frequency-spread sequence** "$fss(j_{fss})$", see above), that is defined by the expression:

  $$j_{tss}(j_w, j_{fss}) = j_w \bullet M_{sc}^{PUCCH,1} + j_{fss}$$

  Recall from 7.8.2.10 that "$j_{tss}$" ranges from **0** to "$N_{symb}^{TSS}$-1". Consequently, "$N_{symb}^{TSS}$-1" is the **same** as the **expression** above over the **maximum value** of "$j_w$" and the **maximum value** of "$j_{fss}$".

- "$N_{symb}^{FSS}$" for `format1` is the **same** as the **quantity** of **PUCCH subcarriers** "$M_{sc}^{PUCCH,1}$":

  $$N_{symb}^{FSS} = M_{sc}^{PUCCH,1}$$

- "$N_{SF}^{PUCCH,1}$" for `format1` is the **same** as the **quantity** of `format1` **PUCCH payload symbol intervals** "$N_{syin}^{PUCCH\text{-}Pld,1}$":

  $$N_{SF}^{PUCCH,1} = N_{syin}^{PUCCH\text{-}Pld,1}$$

  which from 7.6.9 is **half** (rounded down) the **total quantity** of `format1` **PUCCH symbol intervals** "$N_{syin}^{PUCCH,1}$":

  $$N_{SF}^{PUCCH,1} = N_{syin}^{PUCCH\text{-}Pld,1} = floor(N_{syin}^{PUCCH,1} / 2)$$

- "$N_{symb}^{TSS}$" equals "$N_{SF}^{PUCCH,n} \bullet N_{symb}^{FSS}$", which for `format1` (see above) is:

  $$N_{symb}^{TSS} = N_{SF}^{PUCCH,n} \bullet M_{sc}^{PUCCH,1}$$

For more detail on **PUCCH** `format1` **time-spreading**, refer to clause 6.3.2.4.1 of TS 38.211.

**PUCCH** `format2`

It can **only** use **OCC multiplexing**. It is **time-spread only** if `interlace1-v1610` is **not present** and `occ-Length-v1610` is **present**. **Both** parameters are from `PUCCH-ResourceExt-v1610` (click 7.9.9).

When **PUCCH** `format2` is **not time-spread**, it does **not** have a **time-spread sequence** "$tss(j_{tss})$", and it **only** has its **modulated sequence** "$ms(j_{ms})$".

When **PUCCH** `format2` is **time-spread**, this is done by **multiplying every symbol** of its **modulated sequence** "$ms(j_{ms})$" (click 7.8.2.8) by **each successive symbol** of the corresponding **time-spreading sequence** "$w_s(j_w)$":

$$tss(j_w \bullet N_{SF}^{PUCCH,2} + j_{ms}) = w_s(j_w) \bullet ms(j_{ms})$$

where:

- "$w_s(j_w)$" for `format2` is provided in tables 6.3.2.5.2A-1 and 6.3.2.5.2A-2 of TS 38.211,

- "$s$" for `format2` is determined as:

  $$s = (n_0 + n_{IRB}) \bmod N_{SF}^{PUCCH,2}$$

  where "$n_0$" is provided by `occ-Index-v1610` and "$n_{IRB}$" is the **interlaced resource block number** as defined in clause 4.4.4.6 of TS 38.211, within the **interlace** provided by `interlace0-r16`. **Both** parameters are from `PUCCH-ResourceExt-v1610` (click 7.9.9).

- "$j_{ms}$" **ranges** from 0 to "$N_{symb}^{MS}$-1". Consequently, for `format2` "$j_{ms}$" ranges from 0 to "$M_{sc}^{PUCCH,2}$-1" (see below).
- "$j_{tss}$" for `format2` is a **function** of "$j_w$" (the **index** of its **time-spreading sequence** "$w_s(j_w)$", see above) and of "$j_{ms}$" (the **symbol index** of its **modulated sequence** "$ms(j_{ms})$", see above), that is defined by the expression:

  $$j_{tss}(j_w, j_{ms}) = j_w \bullet N_{SF}^{PUCCH,2} + j_{ms}$$

  Recall from 7.8.2.10 that "$j_{tss}$" ranges from 0 to "$N_{symb}^{TSS}$-1". Consequently, "$N_{symb}^{TSS}$-1" is the **same** as the **expression** above over the **maximum value** of "$j_w$" and the **maximum value** of "$j_{ms}$".
- "$N_{symb}^{MS}$" for `format2` is the **same** as the **quantity** of **PUCCH subcarriers** "$M_{sc}^{PUCCH,2}$":

  $$N_{symb}^{MS} = M_{sc}^{PUCCH,2}$$

- "$N_{SF}^{PUCCH,2}$" for `format2` takes the value "2" and it is configured with `occ-Length-v1610` (see above).

For more detail on **PUCCH** `format2` **time-spreading**, refer to clause 6.3.2.5.2A of TS 38.211.

**PUCCH** `format3`

It can **only** use **OCC multiplexing**. It is **time-spread only** if **interlaced mapping** is **set** and a **single interlace** is **present**.

**When PUCCH** `format3` is **not time-spread**, it does **not** have a **time-spread sequence** "$tss(j_{tss})$", and it **only** has its **modulated sequence** "$ms(j_{ms})$".

**When PUCCH** `format3` is **time-spread**, this is done by **multiplying each symbol** of its **modulated sequence** "$ms(j_{ms})$" (click 7.8.2.8) by **1 symbol** of the corresponding **time-spreading sequence** "$w_s(j_w)$" in a **specific order**:

$$tss(si \bullet M_{sc}^{PUCCH,3} + sc) = w_s(floor(sc \bullet (N_{SF}^{PUCCH,3} / M_{sc}^{PUCCH,3})) \bullet ms(si \bullet (M_{sc}^{PUCCH,3} / N_{SF}^{PUCCH,3}) + sc \ mod \ (M_{sc}^{PUCCH,3} / N_{SF}^{PUCCH,3}))$$

where:

- "$w_s()$" for `format3` and for "$N_{SF}^{PUCCH,3} > 1$" are provided in tables 6.3.2.6.3-1 and 6.3.2.6.3-2 of TS 38.211.
- "$s$" for `format3` is determined in clause 9.2.1 of TS 38.213.
- "$j_{ms}$" for `format3` is interpreted as a **function** of "$si$" (the **PUCCH payload symbol interval index**) and of "$sc$" (the **PUCCH payload subcarrier index**), that is defined by the expression:

  $$j_{ms}(si, sc) = si \bullet (M_{sc}^{PUCCH,3} / N_{SF}^{PUCCH,3}) + sc \ mod \ (M_{sc}^{PUCCH,3} / N_{SF}^{PUCCH,3})$$

  Recall from the beginning of this section that the value of "$j_{ms}$" ranges from 0 to "$N_{symb}^{MS}$-1".
- "$j_w$" for `format3` is interpreted a **function** of "$sc$" (the **PUCCH subcarrier index**), that is defined by the expression:

  $$\text{“}j_w(sc)\text{”} = floor(sc \bullet (N_{SF}^{PUCCH,3} / M_{sc}^{PUCCH,3}))$$

Recall from 7.8.2.10 that "$j_w$" ranges from 0 to "$N_{SF}^{PUCCH,3}$-1". Consequently, "$N_{SF}^{PUCCH,3}$-1" is the **same** as the **expression** above over the **maximum value** of "$sc$".

- "$j_{tss}$" is determined for `format3` as a **function** of "$si$" (see below) and "$sc$" (see below), defined by the expression:

  $$j_{tss}(si, sc) = si \bullet M_{sc}^{PUCCH,3} + sc$$

  Recall from 7.8.2.10 that "$j_{tss}$" ranges from 0 to "$N_{symb}^{TSS}$-1". Consequently, "$N_{symb}^{TSS}$-1" is the **same** as the **expression** above over the **maximum value** of "$si$" and the maximum value of "$sc$".

- "$sc$" may be interpreted as the **payload subcarrier index** within `format3` **PUCCH**. It **ranges** from 0 to the **quantity** of `format3` **PUCCH payload subcarriers** (minus 1) "$M_{sc}^{PUCCH\text{-}Pld,3}$-1".

- "$si$" may be interpreted as the `format3` **PUCCH payload symbol interval index**. It **ranges** from 0 to the **quantity** of **PUCCH payload symbol intervals** (minus 1) "$N_{syin}^{PUCCH\text{-}Pld,3}$-1".

- "$N_{SF}^{PUCCH,3}$" is `1`, `2` or `4`. Its value `2` or `4` is provided by `occ-Length-v1610` from `PUCCH-ResourceExt-v1610` (click 7.9.9), if **present**. If **absent**, the value is `1`, in which case there is **no time-spreading**, and the result of the above expression is a **second scrambling**.

Be aware that TS 38.211 uses the term "**block-wise time-spreading**," instead of **OCC multiplexing**.

For more detail on **PUCCH** `format3` **OCC multiplexing**, refer to clause 6.3.2.6.3 of TS 38.211.

**PUCCH** `format4`

It **only** uses **OCC multiplexing**. It is **block-wise spread** in the **time domain** by **multiplying each symbol** of its **modulated sequence** "$ms(j_{ms})$" (click 7.8.2.8) by **1 symbol** of the corresponding **time-spreading sequence** "$w_s(j_w)$" in a **specific order**:

$$tss(si \bullet M_{sc}^{PUCCH,4} + sc) = w_s(floor(\bullet(N_{SF}^{PUCCH,4} / M_{sc}^{PUCCH,4})) \bullet ms(si \bullet (M_{sc}^{PUCCH,4} / N_{SF}^{PUCCH,4}) + sc \ mod \ (M_{sc}^{PUCCH,4} / N_{SF}^{PUCCH,4}))$$

where:

- "$w_s(j_w)$" for `format4` is provided in tables 6.3.2.6.3-1 and 6.3.2.6.3-2 of TS 38.211.

- "$s$" for `format4` is determined in clause 9.2.1 of TS 38.213.

- "$j_{ms}$" for `format4` is interpreted as a **function** of "$si$" (the **PUCCH payload symbol interval index**) and of "$sc$" (the **PUCCH payload subcarrier index**), that is defined by the expression:

  $$j_{ms}(si, sc) = si \bullet (M_{sc}^{PUCCH,4} / N_{SF}^{PUCCH,4}) + sc \ mod \ (M_{sc}^{PUCCH,4} / N_{SF}^{PUCCH,4})$$

  Recall from the beginning of this section that the value of "$j_{ms}$" ranges from 0 to "$N_{symb}^{MS}$-1".

- "$j_w$" for `format4` is interpreted a **function** of "$sc$" (the **PUCCH subcarrier index**), that is defined by the expression:

  "$j_w(sc)$" = $floor(sc \cdot (N_{SF}^{PUCCH,4} / M_{sc}^{PUCCH,4}))$

  Recall from 7.8.2.10 that "$j_w$" ranges from **0** to "$N_{SF}^{PUCCH,4}$-1". Consequently, "$N_{SF}^{PUCCH,4}$-1" is the **same** as the **expression** above over the **maximum value** of "$sc$".

- "$j_{tss}$" is determined for `format4` as a **function** of "$si$" (see below) and "$sc$" (see below), defined by the expression:

  $j_{tss}(si, sc) = si \cdot M_{sc}^{PUCCH,4} + sc$

  Recall from 7.8.2.10 that "$j_{tss}$" ranges from **0** to "$N_{symb}^{TSS}$-1". Consequently, "$N_{symb}^{TSS}$-1" is the **same** as the **expression** above over the **maximum value** of "$si$" and the maximum value of "$sc$".

- "$sc$" may be interpreted as the **payload subcarrier index** within `format4` **PUCCH**. It **ranges** from **0** to the **quantity** of `format4` **PUCCH payload subcarriers** (minus 1) "$M_{sc}^{PUCCH\text{-}Pld,4}$-1".

- "$si$" may be interpreted as the `format4` **PUCCH payload symbol interval index**. It **ranges** from **0** to the **quantity** of **PUCCH payload symbol intervals** (minus 1) "$N_{syin}^{PUCCH\text{-}Pld,4}$-1".

- "$N_{SF}^{PUCCH,4}$" is **2** or **4**, as provided by `occ-Length-v1610`.

Be aware that TS 38.211 uses the term "**block-wise time-spreading**" instead of **OCC multiplexing** or **time-spreading**.

For more detail on **PUCCH** `format4` **OCC multiplexing**, refer to clause 6.3.2.6.3 of TS 38.211.

### 7.8.2.11 PUCCH RE mapping

It consists of **mapping each symbol either** from the **modulated sequence** "$ms(j_{ms})$" (click 7.8.2.8), from the **frequency-spread sequence** "$fss(j_{fss})$" (click 7.8.2.9) or from the **time-spread sequence** "$tss(j_{tss})$"(click 7.8.2.10), as corresponds, to the **PUCCH payload PREs** in **port** "$(sc, si, ap)_\mu$".

The **sequency length** "$N_{symb}^{MS}$", "$N_{symb}^{FSS}$" or "$N_{symb}^{TSS}$" (respectively) is the **same** as the **quantity** of **PUCCH payload PREs** "$N_{RE}^{PUCCH\text{-}Pld,n}$".

Regarding **power control**, **every PUCCH RE** is multiplied by the **amplitude scaling** "$\beta_{PUCCH,n}$" (refer to TS 38.213), where subindex "**n**" identifies the corresponding **PUCCH** `format`.

For **non-interlaced transmission**, the **sequence symbol mapping** is done by advancing **low** to **high frequency** in the **PUCCH payload subcarriers** in **each subsequent PUCCH payload symbol interval**. Explaining this in slightly more detail, the **first sequence symbol** is mapped to the **PUCCH payload** (click 7.6.5.1) **RE** in the **first PUCCH payload symbol interval** and in the **PUCCH payload lowest frequency subcarrier**. **Subsequent sequence symbols** are **mapped** to **PUCCH payload PREs** obtained by **increasing** the **PUCCH payload subcarrier upwards 1** by **1**. When the **highest frequency PUCCH payload subcarrier** is reached, it passes to the **lowest**

**frequency one** in the **next PUCCH payload symbol interval**. Expressing this in a formal way:

$$(sc, si, ap)_{\mu} = mft(si\ M_{sc}^{PUCCH\text{-}Pld,n} + sc)$$

where:

- "$(sc, si, ap)_{\mu}$" is a **PUCCH payload Resource Element** in **port**.
- "$sc$" is the **PUCCH payload subcarrier index** within **each PUCCH payload symbol interval**. It **ranges** from **0** to the **quantity** of **PUCCH payload subcarriers** (minus 1) "$M_{sc}^{PUCCH\text{-}Pld,n}$-1". It is **incremented** until it **reaches** its **maximum range value** "$M_{sc}^{PUCCH\text{-}Pld,n}$-1". It then goes **back to 0** and "$si$" is **incremented** in **1**.
- "$si$" is the **PUCCH payload symbol interval index**. It **ranges** from **0** to the **quantity** of **PUCCH payload symbol intervals** (minus 1) "$N_{syin}^{PUCCH\text{-}Pld,n}$-1". It **starts** with value **0** and is **incremented** as indicated in the **previous bullet**.
- "$p$" is the **antenna port**, which is **2,000**.
- "**μ**" is the **PUCCH subcarrier spacing**.
- "$mft()$" is **either** the **modulated sequence** "$ms(j_{ms})$" (click 7.8.2.8), or the **frequency-spread sequence** "$fss(j_{fss})$" (click 7.8.2.9) or the **time-spread sequence** "$tss(j_{tss})$", as corresponds.
- "$j_{ms}$", "$j_{fss}$" and "$j_{tss}$" are the corresponding **sequence symbol index**. **Each** of them **ranges** from **0** to the **quantity** of **sequence symbols** (minus 1), which are (respectively) "$N_{symb}^{MS}$-1", "$N_{symb}^{FSS}$-1" and "$N_{symb}^{TSS}$-1".
- "$M_{sc}^{PUCCH\text{-}Pld,n}$" is the **quantity** of **PUCCH payload subcarriers** in **PUCCH `format` "`n`"**.
- "$N_{syin}^{PUCCH\text{-}Pld,n}$" is the **quantity** of **PUCCH payload symbol intervals** in **PUCCH `format` "`n`"**.

Some **specific remarks** on **each `format`** follow.

**PUCCH `format0`**

Its **specific remarks** are:

- "$mft()$" is in this case the **modulated sequence** "$ms(j_{ms})$" (click 7.8.2.8). The reason is that **`format0`** uses **neither frequency-spreading**, **nor time-spreading**.
- The **PUCCH payload PREs** in **port** "$(sc, si, ap)_{\mu}$" include **all** the **PUCCH PREs** because **`format0`** does **not** contain **DM-RS** (click 7.6.6.6).
- In case of **2 PUCCH symbol intervals**, the **modulated sequence** "$ms(j_{ms})$" is **repeated** in **each** of them, as a **diversity technique** (**both symbol intervals** are **analogically added** at the **NR node**).
- In case of **intra-slot frequency hopping** (click 7.6.6.3), **each hop** contains a **repetition** of the **modulated sequence** "$ms(j_{ms})$".

For **interlaced transmission**, the **mapping** that has been described at the beginning of this section for **non-interlaced transmission** is **repeated** for **each resource block** in the **interlace** and in the **active bandwidth part** over the **assigned physical resource blocks**

according to clause 9.2.1 of TS 38.213, with the **resource-block dependent sequence** generated according to clause 6.3.2.2 of TS 38.211.

For more detail on **PUCCH `format0` RE mapping**, refer to clause 6.3.2.3.2 of TS 38.211.

**PUCCH `format1`**

Its **specific remarks** are:

- "***mfi***()" is in this case the **time-spread sequence** "***tss***($j_{tss}$)" (click 7.8.2.8). The reason is that **`format1`** uses **time-spreading**.
- In case of **intra-slot frequency hopping** (click 7.6.6.3), **half** (**rounded down**) of the **PUCCH symbol intervals** are transmitted in the **first hop**, and the **remaining PUCCH symbol intervals** are transmitted in the **second hop**.

For **interlaced transmission**, the **mapping** that has been described at the beginning of this section for **non-interlaced transmission** is **repeated** for **each resource block** in the **interlace** and in the **active bandwidth part** over the **assigned physical resource blocks** according to clause 9.2.1 of TS 38.213, with the **resource-block dependent sequence** generated according to clause 6.3.2.2 of TS 38.211.

For more detail on **PUCCH `format1` RE mapping**, refer to clause 6.3.2.4.2 of TS 38.211.

**PUCCH `format2`**

Its **specific remarks** are:

- "***mfi***()" is in this case **either** the **modulated sequence** "***ms***($j_{ms}$)" (click 7.8.2.8) or the **time-spread sequence** "***tss***($j_{tss}$)" (click 7.8.2.10). The reason is that **`format2`** does **not** use **frequency-spreading** and it may **optionally** use **time-spreading**.
- In case of **intra-slot frequency hopping** (click 7.6.6.3), the **first PUCCH symbol interval** is transmitted in the **first hop**, and the **second PUCCH symbol interval** is transmitted in the **second hop**.

For more detail on **PUCCH `format2` RE mapping**, refer to clause 6.3.2.5.3 of TS 38.211.

**PUCCH `format3`**

Its **specific remarks** are:

- "***mfi***()" is in this case **either** the **modulated sequence** "***ms***($j_{ms}$)" (click 7.8.2.8) or the **time-spread sequence** "***tss***($j_{tss}$)" (click 7.8.2.10). The reason is that **`format3`** does **not** use **frequency-spreading** and it may **optionally** use **time-spreading**.
- In case of **intra-slot frequency hopping** (click 7.6.6.3), **half** (**rounded down**) of the **PUCCH symbol intervals** are transmitted in the **first hop**, and the **remaining PUCCH symbol intervals** are transmitted in the **second hop**.

For more detail on **PUCCH `format3` RE mapping**, refer to clause 6.3.2.6.5 of TS 38.211.

**PUCCH `format4`**

Its **specific remarks** are:

- "$mft()$" is in this case the **time-spread sequence** "$tss(j_{tss})$" (click 7.8.2.10). The reason is that **`format4` always** uses **time-spreading**.
- In case of **intra-slot frequency hopping** (click 7.6.6.3), **half** (**rounded down**) of the **PUCCH symbol intervals** are transmitted in the **first hop**, and the **remaining PUCCH symbol intervals** are transmitted in the **second hop**.

For more detail on **PUCCH `format3` RE mapping**, refer to clause 6.3.2.6.5 of TS 38.211.

### 7.8.2.12 PUCCH DFT (transform) precoding

The usage of **DFT precoding** (sometimes called **transform precoding**, click 4.12.12) implements **DFT-s-ODFM** (click 11.16) which is **more robust** to **channel impairments**.

**Each PUCCH `format`** may be subject to a **DFT precoding**, as it is now detailed.

**PUCCH `format0` and `format1`**

They are **not** subject to a **DFT precoding** because it does **not** provide a **sufficiently high robustness gain** (due to the **relatively low quantity** of **REs** of **these `formats`**) to justify the required resources.

**PUCCH `format2`**

It is **not** subject to a **DFT precoding**. Its **waveform** is **not** a **low-PAPR type 1 sequence**.

**PUCCH `format3` and `format4`**

They **are** subject to **DFT precoding** because it provides a **sufficiently high robustness gain** (due to the **relatively high quantity** of **REs** of **these `formats`**) to justify the required resources. Notice that these **`formats`** are transmitted over **`4`** to **`14`** **symbol intervals** and **`1`** to **`16`** **RBs** (and even **20 RBs** for the case of **2 interlaces**).

The **specific DFT precoding** that is applied results in a **low-PAPR type 1** "$r_{u,v}^{(\alpha,\delta,m)}(j)$" (click 11.20.2) **complex number sequence**.

### 7.8.2.13 PUCCH sequence (group) and cyclic shift hopping

**PUCCH transmissions** with **`format0`**, **`format1`**, **`format3`**, and **`format4`** are conformed as a **low-PAPR type 1** "$r_{u,v}^{(\alpha,\delta,m)}(j)$" (click 11.20.2) **complex number sequence** (with "$\delta$=0").

**Sequence** (**group**) **hopping** (click 7.8.2.13.1) consists of a **pseudo-random variation** of the **sequence group** "$u$" and the **sequence** "$v$" of the **low-PAPR type 1 sequence** "$r_{u,v}^{(\alpha,\delta,m)}(j)$" of **each PUCCH transmission**.

**Cyclic shift hopping** (click 7.8.2.13.2) consists of a **pseudo-random variation** of the **phase shift** "$\alpha$" of the **low-PAPR type 1 sequence** "$r_{u,v}^{(\alpha,\delta,m)}(j)$" of **each PUCCH transmission**.

**Sequence** (**group**) **hopping** and **cyclic shift hopping** implement a **diversity** (click 11.8) scheme to **improve reliability**.

#### 7.8.2.13.1 Sequence (group) hopping: computing "*u*" and "*v*"

The **sequence group** "***u***" and the **sequence index** "***v***" vary with the **slot index** "$n_{s,f}^{\mu}$" within the **radio frame** for **numerology** "**μ**" as follows:

$u = (f_{gh} + f_{ss}) \; mod \; 30$

and "***v***", "$f_{gh}$" and "$f_{ss}$" depend on `pucch-GroupHopping` from `PUCCH-ConfigCommon` (click 7.9.3) as follows:

- ***If*** `pucch-GroupHopping` = `neither`, ***then***:
  - $f_{gh} = 0$
  - $f_{ss} = n_{ID} \; mod \; 30$
  - $v = 0$
- ***If*** `pucch-GroupHopping` = `enable`, ***then***:
  - $f_{gh} = (\Sigma^{7}_{i=0} \; 2^{i} \; c(8(2n_{s,f}^{\mu} + n_{hop}) + i)) \; mod \; 30$
  - $f_{ss} = n_{ID} \; mod \; 30$
  - $v = 0$

  where "***c***(***n***)" is **initialized** at the **beginning** of **each radio frame** with "$c_{init} = floor(n_{ID} \; /30)$".
- ***If*** `pucch-GroupHopping` = `disable`, ***then***:
  - $f_{gh} = 0$
  - $f_{ss} = n_{ID} \; mod \; 30$
  - $v = c(2n_{s,f}^{\mu} + n_{hop})$

  where "***c***(***n***)" is **initialized** at the **beginning** of **each radio frame** with "$c_{init} = floor(n_{ID} \; /30)$".

where in the expressions above:

- "$n_{ID}$" is provided by `hoppingID` from `PUCCH-ConfigCommon` (click 7.9.3), ***if*** **present**. ***Else***, "$n_{ID}$" equals the **physical cell identity** "$N_{ID}^{cell}$".
- "***c***(***n***)" is the **pseudo-random sequence** defined in 11.20.1, that in this case is used as a **pseudo-random** binary value "**0**" or "**1**"
- "$n_{hop}$" is the **intra-slot frequency hopping index**. ***If*** **intra-slot frequency hopping** (click 7.6.5.3) is **set**, "$n_{hop}$=0" for the **first hop** and "$n_{hop}$=1" for the **second hop**. ***If*** it is **not set**, "$n_{hop}$=0".

For more detail on **PUCCH sequence hopping**, refer to clause 6.3.2.2.1 of TS 38.211.

#### 7.8.2.13.2 Cyclic shift hopping: computing "α"

The **cyclic shift** "***α***" varies with the **symbol interval index** "***si***" in the **PUCCH transmission** (starting with **0**) and with the **slot index** "$n_{s,f}^{\mu}$" within **the radio frame** for **numerology** "**μ**" as follows:

$\alpha = 2\pi / N_{sc}^{RB} \; ((m_0 + m_{cs} + m_{int} + n_{cs}(n_{s,f}^{\mu} \; , si + l')) \; mod \; N_{sc}^{RB}$

$\alpha = \pi/6 \; ((m_0 + m_{cs} + m_{int} + n_{cs}(n_{s,f}^{\mu} \; , si + l')) \; mod \; 12$

where:

- "$N_{sc}^{RB}$" is the **quantity** of **subcarriers** per **RB**, which is **12**. The factors "$2\pi/N_{sc}^{RB}$" and "***mod*** $N_{sc}^{RB}$" **jointly** cause a **complete cyclic shift** in the **frequency domain every RB**.
- "$l'$" is the **symbol interval index** in the **slot** where the **first symbol** of the **PUCCH transmission** is **located**.
- "$m_0$" depends on the **PUCCH** `format`. For `format0`, it depends (click 7.8.2.9) on the **UCI message**. For `format1`, clause 6.3.2.2.2 of TS 38.211 says that it is defined in TS 38.213, but I have not been able to find if there. For `format3` and `format4`, it is related to the **PUCCH DM-RS signal** (refer to clause 6.4.1.3.3.1 of TS 38.331).
- "$m_{cs}$" is **0**, except for `format0`, where it **depends** (click 7.8.2.9) on the **UCI message**.
- "$m_{int}$" depends on the **PUCCH** `format` and on **interlace**. It is "$5n_{IRB\mu}$" for `format0` and `format1` when **interlace** is **set** (click 7.6.5.3.8). It is 0 otherwise.
- "$5n_{IRB\mu}$" is the **resource block index** within the **interlace**, for numerology "**μ**".
- "$n_{cs}()$" is a **function** with **2 variables** defined as:

  $$n_{cs}(n_{s,f}^{\mu}, si+l') = \Sigma_{i=0}^{7} 2^i \, c(8N_{symb}^{slot} n_{s,f}^{\mu} + 8(si+l') + i)$$

  where:

  - "$n_{ID}$" is provided by `hoppingID` from `PUCCH-ConfigCommon` (click 7.9.3), if **present**. ***Else***, "$n_{ID}$" equals the **physical cell identity** "$N_{ID}^{cell}$".
  - "$c(n)$" is the **pseudo-random sequence** defined in 11.20.1, that in this case it is used as a **pseudo-random** binary value "**0**" or "**1**"
  - "$n_{hop}$" is the **intra-slot frequency hopping index**. ***If*** **intra-slot frequency hopping** (click 7.6.5.3.4) is **set**, "$n_{hop}$=0" for the **first hop** and "$n_{hop}$=1" for the **second hop**. ***If*** it is **not set**, "$n_{hop}$=0".

For more detail on **PUCCH cyclic shift hopping**, refer to clause 6.3.2.2.2 of TS 38.211.

#### 7.8.2.14 PUCCH interlaces

For **shared spectrum** in **FR1**, **PUCCH** `format0`, `format1`, `format2` and `format3` are extended to use resource in **1 PRB interlace** (up to **2 interlaces** for `format2` and `format3`) in **1 RB set**.

### 7.8.3 PUCCH payload Rx processing

The **PUCCH payload Rx processing** at the **receiver** (the **NR node**) is the **reciprocal** to the **PUCCH payload Tx processing** at **each transmitter** (**each UE**).

A specific procedure that the **NR node** has to carry out is to **demultiplex** the transmissions from **different UEs** (click 7.6.5.1 and 7.8.2.10) over the **same PUCCH resources**.

### 7.8.4 Mapping to PUCCH processing terms used in TS documents

The **terms** and **symbols** (e.g., $a_i$, $b_i$, $c_i$, $A$, $B$, $C$, …) used in the **TS documents** do **not help** to **understand** the explanations about the **PUCCH payload processing**. This is why

in **this book** the **terms** and **symbols** used in **TS documents** have been **replaced** by **more illustrative terms** and **symbols**. The same has been done for the **terms** and **symbols** used **PDSCH/PUCSH payload processing** (click 4.12.9).

The following table presents a **mapping** between the **PUCCH payload processing terms** and **symbols** in **this book** and in the **TS documents**, to facilitate the **cross-checking** of information between **this book** and the **TS documents**:

| | Terminology mapping | |
|---|---|---|
| **Element** | **This book** | **TS 38.211 and 38.212** |
| **Term** | **UCI message** | **UCI message** |
| **Bit sequences** | $uci(i_{uci})$ | $a_i$ |
| **Bit sequence index** | $i_{tb}$ | $_i$ (subindex) |
| **Bit seq. length [bits]** | $N_{bit}^{UCI}$ | $A$ |
| **Term** | **code block sequence** | **sequence** of **code blocks** |
| **Quantity of blocks** | $N_{blk}^{CBS}$ | $C$ |
| **Block index** | $n$ | $r$ |
| **Term** | **code block** of **seq.** | **code block** |
| **Bit sequences** | $cb_{cn}(i_{cb})$ | $c_{ri}$ |
| **Bit sequence index** | $i_{cb}$ | $_i$ (subindex) |
| **Bit seq. length [bits]** | $N_{bit}^{CB}$ | $K_r$ |
| **Term** | **CRC**$^{CB}$ of **code block** | **CRC** of **code block** |
| **Bit sequence** | $crc_{CBcn}(i_{crccbcn})$ | $p_{ri}$ |
| **Bit seq. length [bits]** | $N_{bit}^{CRC\text{-}CB}$ | $L$ |
| **Term** | **encoded block sequence** | **sequence** of **encoded code blocks** |
| **Term** | **encoded block** | **encoded code block** |
| **Stored bit sequences** | $eb_n(i_{eb})$ | $d_{ri}$ |
| **Bit sequence index** | $i_{eb}$ | $_i$ (subindex) |
| **Bit seq. length [bits]** | $N_{bit}^{EB}$ | $N_r$ |
| **Term** | **rated-matched block sequence** | **sequence** of **rate-matched code blocks** |
| **Term** | **rate-matched block** | **rate-matched code block** |
| **Bit sequence** | $rb_{tn}(i_{rb})$ | $f_{ri}$ |
| **Bit sequence index** | $i_{rb}$ | $_i$ (subindex) |
| **Bit seq. length [bits]** | $N_{bit}^{RBtn}$ | $E_r$ |
| **Term** | **codeword** | **codeword** |
| **Bit sequence** | $cw(i_{cw})$ | $g_i$ (in TS 38.212)<br>$b_i$ (in TS 38.211) |
| **Bit sequence index** | $i_{cw}$ | $_i$ (subindex) |
| **Bit seq. length [bits]** | $N_{bit}^{CW}$ | $G$ (in TS 38.212)<br>$M_{bit}$ (in TS 38.211) |
| **Term** | **reordered codeword** | **codeword** |
| **Bit sequence** | $rcw(i_{cw})$ | $g_i$ (in TS 38.212)<br>$b_i$ (in TS 38.211) |
| **Bit sequence index** | $i_{rcw}$ | $_i$ (subindex) |
| **Bit seq. length [bits]** | $N_{bit}^{CW}$ | $G$ (in TS 38.212) |

| | Terminology mapping | |
|---|---|---|
| **Element** | **This book** | **TS 38.211 and 38.212** |
| **Term** | **scrambled codeword** | **scrambled codeword** |
| **Bit sequence** | $scw(i_{cw})$ | $b_{\sim i}$ |
| **Bit sequence index** | $i_{cw}$ | $_i$ (subindex) |
| **Bit seq. length [bits]** | $N_{bit}^{CW}$ | $M_{bit}$ |
| **Term** | **modulated sequence** | **Modulated sequence** |
| **Symbol sequence** | $ms(j_{ms})$ | $d_i$ |
| **Symbol sequence index** | $j_{ms}$ | $_i$ (subindex) |
| **Sym. seq. length [symbs]** | $N_{symb}^{MS}$ | $M_{symb}$ |
| **Term** | **Layers** | **Layers** |
| **Quantity of layers** | $N_{layer}$ | $N_L$ |
| **Term** | **frequency-spread sequence** | **Layer mapped sequence** |
| **Symbol sequence** | $fss(j_{fss})$ | $y_i$ |
| **Symbol sequence index** | $j_{fss}$ | $_i$ (subindex) |
| **Sym. seq. length [symbs]** | $N_{symb}^{FSS}$ | `format` **dependent** |
| **Term** | **time-spread sequence** | **Layer mapped sequence** |
| **Symbol sequence** | $tss(j_{tss})$ | $y_i$ |
| **Symbol sequence index** | $j_{fss}$ | $_i$ (subindex) |
| **Sym. seq. length [symbs]** | $N_{symb}^{TSS}$ | `format` **dependent** |

Some **relevant remarks** on this table are:

- The term "***i***" are the **bit index** within **each** and **every bit sequence**, **both** in **this book** and in **TS documents**. In **this book** it is expressed as a **variable** "bit(***i***)", while in the **TS document** it is expressed as a **subindex** "bit$_i$".
- The term "***j***" is the **symbol index** within **each** and **every symbol sequence**, in **this book**, while in **TS documents** it is "***i***". In **this book** is expressed as a **variable** "symbol(***j***)", while in the **TS document** is expressed as a **subindex** "symbol$_i$".

## 7.9 PUCCH parameters

The **UE**'s **Physical Uplink Control Channel** (**PUCCH**, click 7.5) is configured with:

- **Common parameters** that are **broadcast** (or **unicast**) within `SIB1`, as described in section 7.9.1.
- **Common parameters** that are **unicast** within the **Non-SI parameter group**, as described in section 7.9.2.

**PUCCH** configuration does **not** have **any parameter** from `MIB` (click 8.12.3).

Possibly existing **PUCCH parameters** within **other SIBs** and **posSIBs** (click 5.3.5) are **not addressed** in this book.

For an overall explanation on **UE configuration parameters** click 5. For a brief overview of **ASN.1**, click 5.9.

### 7.9.1 PUCCH parameters within SIB1

**PUCCH** is configured with **parameters** within **SIB1** (click 5.3.4) in the following case:

- In **UE**'s **primary cell**, for **PUCCH** in the **Uplink initial bandwidth part during initial access**.

For **PUCCH** parameters within **SIB1**, the **most relevant related data types** are:

- **BWP-UplinkCommon** (click 2.2.8.9): common parameters

  **Only one** of the parameters of this data type, which is **useInterlacePUCCH-PUSCH-r16**, is **relevant** for **PUCCH**.

- **PUCCH-ConfigCommon** (click 7.9.3): common parameters

The **ancestor tree** (click 5.9.8) of these **most relevant related data types** up to **SIB1** is:

```
                                    SIB1 (click 5.7.4)
servingCellConfigCommon               ServingCellConfigCommonSIB (0,1) (click 3.9.3)
  uplinkConfigCommon                    UplinkConfigCommonSIB (click 2.2.6.5)
    initialUplinkBWP                      BWP-UplinkCommon (click 2.2.8.9)
      pucch-ConfigCommon                    PUCCH-ConfigCommon (click 7.9.3)
  supplementaryUplink                 UplinkConfigCommonSIB (click 2.2.6.5)
    initialUplinkBWP                      BWP-UplinkCommon (click 2.2.8.9)
      pucch-ConfigCommon                    PUCCH-ConfigCommon (click 7.9.3)
```

For the **ancestor trees** of **SIB1** **up to** the **top**, click 5.3.4.

### 7.9.2 PUCCH parameters within the Non-SI parameter group

**PUCCH** is configured with **parameters unicast** within the **Non-SI parameter group** (click 5.3.6) in the following cases:

- In **UE**'s **primary cell**, **PUCCH** for the **additional bandwidth parts** (i.e., **other** than the **initial bandwidth part**).
- In the **primary-secondary cell** and in **secondary cells**, **PUCCH** for **all bandwidth parts**.
- In **UE**'s **primary cell**, **PUCCH** for the **initial bandwidth part**, if configuration changes are done **after initial access**.

For **PUCCH** parameters within the **Non-SI parameter group**, the **most relevant related data types** are:

- **PUCCH-ConfigCommon** (click 7.9.3): common parameters
- **PUCCH-Config** (click 7.9.4): dedicated parameters
- **SchedulingRequestConfig** (click 7.9.14): dedicated parameters
- **SchedulingRequestToAddMod** (click 7.9.15): dedicated parameters

The following **data types** (whose **only parent** is **PUCCH-Config**, click 7.9.4) are also relevant:

- **PUCCH-ResourceSet** (click 7.9.6): dedicated parameters
- **PUCCH-Resource** (click 7.9.7): dedicated parameters

- `PUCCH-ResourceExt-v1610` (click 7.9.9): dedicated parameters
- `PUCCH-FormatConfig` (click 7.9.10): dedicated parameters
- `PUCCH-FormatConfigExt-r17` (click 7.9.11): dedicated parameters
- `SchedulingRequestResourceConfig` (click 7.9.12): dedicated parameters
- `SchedulingRequestResourceConfigExt-v1610` and `-v1700` (click 7.9.13): dedicated parameters
- `DMRS-BundlingPUCCH-Config-r17` (click 7.9.16): dedicated parameters

The following **5 data types** (whose **only parent** is `PUCCH-Resource`) are also relevant:

- `PUCCH-format0` to `PUCCH-format4` (click 7.9.8): dedicated parameters

The following **data type** (whose **only parent** is `SchedulingRequestConfig`, click 7.9.14) is also relevant:

- `SchedulingRequestToAddMod` (click 7.9.15): dedicated parameters

The **ancestor tree** (click 5.9.8) of these **most relevant related data types** up to `CellGroupConfig` is:

```
                                CellGroupConfig (click 3.7.3)
mac-CellGroupConfig               MAC-CellGroupConfig (click 3.7.4)
  schedulingRequestConfig           SchedulingRequestConfig (click 7.9.14)
    schedulingRequestToAddModList
                                      SchedulingRequestToAddMod (click 7.9.15)
spCellConfig                      SpCellConfig (click 3.9.5)
  reconfigurationWithSync           ReconfigurationWithSync (0,1) (click 3.9.8)
    spCellConfigCommon                ServingCellConfigCommon (click 3.9.6)
      uplinkConfigCommon                UplinkConfigCommon (click 2.2.6.9)
        initialUplinkBWP                  BWP-UplinkCommon (click 2.2.8.9)
          pucch-ConfigCommon                PUCCH-ConfigCommon (click 7.9.3)
      supplementaryUplinkConfig         UplinkConfigCommon (0,1) (click 2.2.6.9)
        initialUplinkBWP                  BWP-UplinkCommon (click 2.2.8.9)
          pucch-ConfigCommon                PUCCH-ConfigCommon (click 7.9.3)
  spCellConfigDedicated             ServingCellConfig (click 3.9.7)
    uplinkConfig                      UplinkConfig (0,1) (click 2.2.8.4)
      initialUplinkBWP                  BWP-UplinkDedicated (0,1) (click 2.2.8.10)
        pucch-Config                      PUCCH-Config (0,1) (click 7.9.4)
        sl-PUCCH-Config-r16               PUCCH-Config (0,1) (click 7.9.4)
        pucch-ConfigMulticast1-r17        PUCCH-Config (0,1) (click 7.9.4)
        pucch-ConfigMulticast2-r17        PUCCH-Config (0,1) (click 7.9.4)
        pucch-ConfigurationList-r16
                                          PUCCH-ConfigurationList-r16 (0,1)
                                                                  (click 7.9.5)
                                            PUCCH-Config (1..2) (click 7.9.4)
        pucch-ConfigurationListMulticast1-r17
                                          PUCCH-ConfigurationList-r16 (0,1)
                                                                  (click 7.9.57.9.4)
                                            PUCCH-Config (1..2) (click 7.9.4)
        pucch-ConfigurationListMulticast2-r17
                                          PUCCH-ConfigurationList-r16 (0,1)
                                                                  (click 7.9.5)
                                            PUCCH-Config (1..2) (click 7.9.4)
```

```
    uplinkBWP-ToAddModList       BWP-Uplink (0..4) (click 2.2.8.8)
      bwp-Common                  BWP-UplinkCommon (0,1) (click 2.2.8.9)
        pucch-ConfigCommon          PUCCH-ConfigCommon (click 7.9.3)
      bwp-Dedicated               BWP-UplinkDedicated (0,1) (click 2.2.8.10)
        pucch-Config                PUCCH-Config (0,1) (click 7.9.4)
        sl-PUCCH-Config-r16         PUCCH-Config (0,1) (click 7.9.4)
        pucch-ConfigMulticast1-r17 PUCCH-Config (0,1) (click 7.9.4)
        pucch-ConfigMulticast2-r17 PUCCH-Config (0,1) (click 7.9.4)
        pucch-ConfigurationList-r16
                                    PUCCH-ConfigurationList-r16 (0,1)
                                                                (click 7.9.5)
                                      PUCCH-Config (1..2) (click 7.9.4)
        pucch-ConfigurationListMulticast1-r17
                                    PUCCH-ConfigurationList-r16 (0,1)
                                                                (click 7.9.5)
                                      PUCCH-Config (1..2) (click 7.9.4)
        pucch-ConfigurationListMulticast2-r17
                                    PUCCH-ConfigurationList-r16 (0,1)
                                                                (click 7.9.5)
                                      PUCCH-Config (1..2) (click 7.9.4)
  supplementaryUplink         UplinkConfig (0,1) (click 2.2.8.4)
    initialUplinkBWP            BWP-UplinkDedicated (0,1) (click 2.2.8.10)
      pucch-Config                PUCCH-Config (0,1) (click 7.9.4)
      sl-PUCCH-Config-r16         PUCCH-Config (0,1) (click 7.9.4)
      pucch-ConfigMulticast1-r17 PUCCH-Config (0,1) (click 7.9.4)
      pucch-ConfigMulticast2-r17 PUCCH-Config (0,1) (click 7.9.4)
      pucch-ConfigurationList-r16
                                  PUCCH-ConfigurationList-r16 (0,1)
                                                              (click 7.9.5)
                                    PUCCH-Config (1..2) (click 7.9.4)
      pucch-ConfigurationListMulticast1-r17
                                  PUCCH-ConfigurationList-r16 (0,1)
                                                              (click 7.9.5)
                                    PUCCH-Config (1..2) (click 7.9.4)
      pucch-ConfigurationListMulticast2-r17
                                  PUCCH-ConfigurationList-r16 (0,1)
                                                              (click 7.9.5
                                    PUCCH-Config (1..2) (click 7.9.4)
    uplinkBWP-ToAddModList       BWP-Uplink (0..4) (click 2.2.8.8)
      bwp-Common                  BWP-UplinkCommon (click 2.2.8.9)
        pucch-ConfigCommon          PUCCH-ConfigCommon (click 7.9.3)
      bwp-Dedicated               BWP-UplinkDedicated (0,1) (click 2.2.8.10)
        pucch-Config                PUCCH-Config (0,1) (click 7.9.4)
        sl-PUCCH-Config-r16         PUCCH-Config (0,1) (click 7.9.4)
        pucch-ConfigMulticast1-r17 PUCCH-Config (0,1) (click 7.9.4)
        pucch-ConfigMulticast2-r17 PUCCH-Config (0,1) (click 7.9.4)
        pucch-ConfigurationList-r16
                                    PUCCH-ConfigurationList-r16 (0,1)
                                                                (click 7.9.5)
                                      PUCCH-Config (1..2) (click 7.9.4)
        pucch-ConfigurationListMulticast1-r17
                                    PUCCH-ConfigurationList-r16 (0,1)
                                                                (click 7.9.5)
                                      PUCCH-Config (1..2) (click 7.9.4)
        pucch-ConfigurationListMulticast2-r17
                                    PUCCH-ConfigurationList-r16 (0,1)
                                                                (click 7.9.5)
                                      PUCCH-Config (1..2) (click 7.9.4)
```

```
sCellToAddModList              SCellConfig (0..maxNrofSCells) (click 3.9.4)
  sCellConfigCommon              ServingCellConfigCommon (0,1) (click 3.9.6)
    uplinkConfigCommon             UplinkConfigCommon (click 2.2.6.9)
      initialUplinkBWP               BWP-UplinkCommon (click 2.2.8.9)
        pucch-ConfigCommon             PUCCH-ConfigCommon (click 7.9.3)
    supplementaryUplinkConfig    UplinkConfigCommon (0,1) (click 2.2.6.9)
      initialUplinkBWP               BWP-UplinkCommon (click 2.2.8.9)
        pucch-ConfigCommon             PUCCH-ConfigCommon (click 7.9.3)
  sCellConfigDedicated           ServingCellConfig (click 3.9.7)
    uplinkConfig                   UplinkConfig (0,1) (click 2.2.8.4)
      initialUplinkBWP               BWP-UplinkDedicated (click 2.2.8.10)
        pucch-Config                   PUCCH-Config (0,1) (click 7.9.4)
        sl-PUCCH-Config-r16            PUCCH-Config (0,1) (click 7.9.4)
        pucch-ConfigMulticast1-r17   PUCCH-Config (0,1) (click 7.9.4)
        pucch-ConfigMulticast2-r17   PUCCH-Config (0,1) (click 7.9.4)
        pucch-ConfigurationList-r16
                                       PUCCH-ConfigurationList-r16 (0,1)
                                                                   (click 7.9.5)
                                         PUCCH-Config (1..2) (click 7.9.4)
        pucch-ConfigurationListMulticast1-r17
                                       PUCCH-ConfigurationList-r16 (0,1)
                                                                   (click 7.9.5)
                                         PUCCH-Config (1..2) (click 7.9.4)
        pucch-ConfigurationListMulticast2-r17
                                       PUCCH-ConfigurationList-r16 (0,1)
                                                                   (click 7.9.5)
                                         PUCCH-Config (1..2) (click 7.9.4)
      uplinkBWP-ToAddModList       BWP-Uplink (0..maxNrofBWPs) (click 2.2.8.8)
        bwp-Common                     BWP-UplinkCommon (click 2.2.8.9)
          pucch-ConfigCommon             PUCCH-ConfigCommon (click 7.9.3)
        bwp-Dedicated                  BWP-UplinkDedicated (click 2.2.8.10)
          pucch-Config                   PUCCH-Config (0,1) (click 7.9.4)
          sl-PUCCH-Config-r16            PUCCH-Config (0,1) (click 7.9.4)
          pucch-ConfigMulticast1-r17   PUCCH-Config (0,1) (click 7.9.4)
          pucch-ConfigMulticast2-r17   PUCCH-Config (0,1) (click 7.9.4)
          pucch-ConfigurationList-r16
                                         PUCCH-ConfigurationList-r16 (0,1)
                                                                     (click 7.9.5)
                                           PUCCH-Config (1..2) (click 7.9.4)
          pucch-ConfigurationListMulticast1-r17
                                         PUCCH-ConfigurationList-r16 (0,1)
                                                                     (click 7.9.5)
                                           PUCCH-Config (1..2) (click 7.9.4)
          pucch-ConfigurationListMulticast2-r17
                                         PUCCH-ConfigurationList-r16 (0,1)
                                                                     (click 7.9.5)
                                           PUCCH-Config (1..2) (click 7.9.4)
```

```
supplementaryUplink          UplinkConfig (0,1) (click 2.2.8.4)
  initialUplinkBWP             BWP-UplinkDedicated (click 2.2.8.10)
  uplinkBWP-ToAddModList       BWP-Uplink (0..maxNrofBWPs) (click 2.2.8.8)
    bwp-Common                   BWP-UplinkCommon (click 2.2.8.9)
      pucch-ConfigCommon           PUCCH-ConfigCommon (click 7.9.3)
    bwp-Dedicated                BWP-UplinkDedicated (click 2.2.8.10)
      pucch-Config                 PUCCH-Config (0,1) (click 7.9.4)
      sl-PUCCH-Config-r16          PUCCH-Config (0,1) (click 7.9.4)
      pucch-ConfigMulticast1-r17 PUCCH-Config (0,1) (click 7.9.4)
      pucch-ConfigMulticast2-r17 PUCCH-Config (0,1) (click 7.9.4)
      pucch-ConfigurationList-r16
                                   PUCCH-ConfigurationList-r16 (0,1)
                                                          (click 7.9.5)
                                     PUCCH-Config (1..2) (click 7.9.4)
      pucch-ConfigurationListMulticast1-r17
                                   PUCCH-ConfigurationList-r16 (0,1)
                                                          (click 7.9.5)
                                     PUCCH-Config (1..2) (click 7.9.4)
      pucch-ConfigurationListMulticast2-r17
                                   PUCCH-ConfigurationList-r16 (0,1)
                                                          (click 7.9.57.9.4)
                                     PUCCH-Config (1..2) (click 7.9.4)
```

Some **relevant remarks** on this **ancestor tree** are:

- Be aware that `CellGroupConfig` can be found within the **RRC NR messages** `RRCReconfiguration`, `RRCResume` and `RRCSetup` (click 5.8.3.1).
- For the **ancestor trees** of `CellGroupConfig`, **up to** some **RRC NR messages** click 3.7.1 and **up to** some **RRC inter-node messages** click 3.7.2.

### 7.9.3 `PUCCH-ConfigCommon` data type

It can be **both broadcast** (or **unicast**) within `SIB1` (click 5.3.4) and **unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `SIB1`, click 7.9.1.

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 7.9.2.

**All** its parameters are **common** (click 5.1.3).

**Each** of its **instances** contains the **common parameters** that configure **PUCCH**.

**Each** of its **instances** configures the **UE** with **most PUCCH elements**, of its corresponding **bandwidth part** (**initial** or **additional**), of **common configuration**, of its corresponding **cell** (**special** or **secondary**), of its corresponding **cell group** (**master** or **secondary**). The "**corresponding**" elements are determined by the **specific place**, within the overall **ASN.1 specification**, where this **data type** has been **instantiated**.

#### 7.9.3.1 `PUCCH-ConfigCommon` parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `intra-SlotFH-r17 ENUMERATED {fromLowerEdge, fromUpperEdge}`: optional, common

  When **absent**, **intra-slot frequency hopping** is **set** in the **RedCap-specific initial bandwidth part** (in case this **BWP** has been configured) for the **RedCap UEs**.

When **present**, **intra-slot frequency hopping** is **not set** in the said **bandwidth part**. In this case, **each PUCCH resource** is **mapped** to a **single PRB** on **one edge** of the **Uplink** (or **SUL**) **bandwidth part** and the **parameter value** indicates whether the **PRB index** in the **PRB mapping** is counted in **increasing order** from the **lower edge** or in **decreasing order** from the **upper edge**.

- `nrofPRBs INTEGER (1..16)`: optional, common

  It indicates the **quantity** of **PRBs** used per **PUCCH resource** for **PUCCH** `format0` and `format1` **in FR2-2** (refer to clause 9.2.1 of TS 38.213).

- `pucch-ResourceCommon INTEGER (0..15)`: optional, common

  It selects an entry from the **16-row table 9.2.1-1** (of TS 38.213), where **each row** configures a **set** of **common PUCCH resources** and **parameters**.

  The **UE** uses the resulting **PUCCH resources** until it is provided with a **dedicated** `PUCCH-Config` **instance** on the **Uplink initial bandwidth part**. Once the network provides a **dedicated** `PUCCH-Config` **instance** for that **bandwidth part**, the **UE** applies that one instead of the one obtained from this parameter (refer to clause 9.2 of TS 38.213).

- `pucch-GroupHopping ENUMERATED {neither, enable, disable}`: optional, common

  It configures **group hopping** and **sequence hopping** for **all** the **PUCCH resources** with `format0`, `format1`, `format3` and `format4`. Value `neither` **disables both group hopping** and **sequence hopping**. Value `enable` **enables group hopping** and **disables sequence hopping**. Value `disable` **disables group hopping** and **enables sequence hopping** (refer to clause 6.3.2.2 of TS 38.211).

For more detail on these parameters, refer to the table "*PUCCH-ConfigCommon field descriptions*" of TS 38.331.

### 7.9.3.2 `PUCCH-ConfigCommon` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
PUCCH-ConfigCommon ::= SEQUENCE {
  pucch-ResourceCommon INTEGER (0..15) OPTIONAL,     -- Cond InitialBWP-Only
  pucch-GroupHopping ENUMERATED {neither, enable, disable},
  hoppingId INTEGER (0..1023) OPTIONAL,                                -- Need R
  p0-nominal INTEGER (-202..24) OPTIONAL,                              -- Need R
  ...,
  [[
  nrofPRBs INTEGER (1..16) OPTIONAL,                                   -- Need R
  intra-SlotFH-r17 ENUMERATED {fromLowerEdge, fromUpperEdge} OPTIONAL,
                                                                       -- Need R
  pucch-ResourceCommonRedCap-r17 INTEGER (0..15) OPTIONAL,
                                                  -- Cond InitialBWP-RedCap
  additionalPRBOffset-r17 ENUMERATED {n2, n3, n4, n6, n8, n9, n10, n12}
                          OPTIONAL                                     -- Need S
  ]]
}
```

## 7.9.4 `PUCCH-Config` data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 7.9.2.

Its **only parent data type** (directly, or indirectly through `PUCCH-ConfigurationList-r16`, click 7.9.5) is `BWP-UplinkDedicated` (click 2.2.8.10), which contains the **dedicated configuration** for **each Uplink** or **SUL bandwidth part**. **Data type** `BWP-UplinkDedicated` is considered an **indirect parent** of `PUCCH-Config` because a few of the parameters of `BWP-UplinkDedicated` contain `2` **instances** of `PUCCH-Config` using the "**shorthand**"

**All** of its parameters are **dedicated** (click 5.1.3).

**Each** of its **instances** defines: a) several parameters for **all PUCCH resources** (click 7.6.5); b) a certain **quantity** of **individual PUCCH resources**; and c) a certain quantity of **PUCCH resource sets** (click 7.6.7).

**Each** of its **instances** configures the **UE** with **most PUCCH elements**, of its corresponding **bandwidth part** (**initial** or **additional**), of **dedicated configuration**, of its corresponding **cell** (**special** or **secondary**), of its corresponding **cell group** (**master** or **secondary**). The "**corresponding**" elements are determined by the **specific place**, within the overall **ASN.1 specification**, where this **data type** has been **instantiated**.

#### 7.9.4.1 `PUCCH-Config` parameters

Some of the **relevant** parameters of `PUCCH-Config` (indicating for each of them its data type and semantics) are:

- `dmrs-BundlingPUCCH-Config-r17` `DMRS-BundlingPUCCH-Config-r17` (click 7.9.16): optional, dedicated

  It contains the configuration parameters for **PUCCH DM-RS bundling** (click 7.6.5.3.7), for **each** and **every PUCCH resource** (of the corresponding **bandwidth part**).

- `format0-r17` and `format1` to `format4` `PUCCH-FormatConfig` (click 7.9.10): optional, dedicated

  **Each** of them configures the **parameter values** for **each** and **every PUCCH resource** (of the corresponding **bandwidth part**) that have the `format` identified by the **parameter name**. Be aware that **PUCCH** `format2` **to** `format4` may be **extended** with **one additional** parameter (see the next bullet).

  Be aware that its **son data type** `PUCCH-Resource` (click 7.9.7) contains a `CHOICE` between parameters `format0` to `format4`, that should **not be mistaken** with these parameters: they have the **same identifier** (except for `format0`), but they are **different parameters**, with **different data type** and **semantics**.

- `format2Ext-r17`, `format3Ext-r17` and `format4Ext-r17` `PUCCH-FormatConfigExt-r17` (click 7.9.11): optional, dedicated

  **Each** of them configures **one additional parameter** for **each** and **every PUCCH resource** (of the corresponding **bandwidth part**) that have the `format` identified by the **parameter name**.

  These parameters `format2Ext-r17` to `format4Ext-r17` are used to **add one more parameter** (`maxCodeRateLP-r17`, the only one within `PUCCH-FormatConfig-r17` click 7.9.11) to the parameters configured with (respectively) `format2` to `format4`, from the previous bullet point.

- `multi-CSI-PUCCH-ResourceList SEQUENCE (SIZE (1..2)) OF PUCCH-ResourceId` (click 7.9.7): optional, dedicated

  It does **not** have a **field description** in TS 38.331. It is then **assumed** that it contains **1** or **2 PUCCH resource identifiers** for **multiple CSI reports** (click 7.6.3).

- `resourceSetToAddModList SEQUENCE (SIZE (1..maxNrofPUCCH-ResourceSets)) OF PUCCH-ResourceSet` (click 7.9.6): optional, dedicated

  It **adds** new **PUCCH resource sets** to the corresponding **UE**'s **Uplink bandwidth part**. **Each** of its **nameless** parameters of **data type** `PUCCH-ResourceSet` (click 7.9.6) defines **one PUCCH resource set**.

  The constant `maxNrofPUCCH-ResourceSets` is `4`, which is **maximum quantity** of **PUCCH resource sets** in the corresponding **UE**'s **Uplink bandwidth part**.

- `resourceToAddModList SEQUENCE (SIZE (1..maxNrofPUCCH-Resources)) OF PUCCH-Resource` (click 7.9.7): optional, dedicated

  It **adds** new **PUCCH resources**, together with the semantics definition of their parameters, to the corresponding **UE**'s **Uplink bandwidth part**. **Each** of its **nameless** parameters of **data type** `PUCCH-Resource` (click 7.9.7) defines the parameters of **one individual PUCCH resource**.

  The constant `maxNrofPUCCH-Resources` is `128`, which is **maximum quantity** of **PUCCH resources** in the corresponding **UE**'s **Uplink bandwidth part**.

- `resourceToAddModListExt-v1610 SEQUENCE (SIZE (1..maxNrofPUCCH-Resources)) OF PUCCH-ResourceExt-v1610` (click 7.9.9): optional, dedicated

  **Each** of its **nameless parameters extends** the **PUCCH resource** from `resourceToAddModList` that is in the same sequential position (within both `SEQUENCE OF`), with some **additional** parameters. Consequently, the **quantity** of **nameless fields** within this parameter **must** be the same as the one within `resourceToAddModList`.

  The **additional** parameters provided by these parameters are **extensions** to the specification of a **PUCCH resource** performed in TS 38.331 **version 16.10**.

  The constant `maxNrofPUCCH-Resources` is `128`, which is **maximum quantity** of **PUCCH resources** in the corresponding **UE**'s **Uplink bandwidth part**.

- `resourceToAddModListExt-v1610 SEQUENCE (SIZE (1..maxNrofPUCCH-Resources)) OF SchedulingRequestResourceConfigExt-v1610` (click 7.9.13): optional, dedicated

- `resourceToAddModListExt-v1700 SEQUENCE (SIZE (1..maxNrofPUCCH-Resources)) OF SchedulingRequestResourceConfigExt-v1700` (click 7.9.13): optional, dedicated

  **Each** of these **3 parameters adds** new **scheduling request resources**, to the corresponding **UE**'s **Uplink or SUL bandwidth part**. **Each** of their **nameless** parameters defines the parameters of **1 individual scheduling request resource**.

  The constant `maxNrofSR-Resources` is `8` which is **maximum quantity** of **scheduling request resources** in the corresponding **UE**'s **Uplink bandwidth part**.

- `schedulingRequestResourceToAddModList SEQUENCE (SIZE (1..maxNrofSR-Resources)) OF SchedulingRequestResourceConfig` (click 7.9.12): optional, dedicated
- `subslotLengthForPUCCH-r16 CHOICE`: optional, dedicated

  It indicates the quantity of **PUCCH symbol intervals** for **subslot-based PUCCH** feedback, for **each** and **every PUCCH resource** (of the corresponding **bandwidth part**).

  Its data type is a `CHOICE` between parameters `normalCP-r16` and `extendedCP-r16`, both of which are an `ENUMERATED`. The **former** may take the value of `2` **or** `7` **symbol intervals** while the **latter** may take the value of `2` **or** `6` **symbol intervals**.

  For more detail about how it is used by the **UE**, click 7.6.5.3.

For more detail on these parameters, refer to the table "*PUCCH-Config field descriptions*" of TS 38.331.

### 7.9.4.2 `PUCCH-Config` ASN.1 specification

**Part** of the specification of `PUCCH-Config` is:

```
PUCCH-Config ::= SEQUENCE {
  resourceSetToAddModList SEQUENCE (SIZE (1..maxNrofPUCCH-ResourceSets))
                          OF PUCCH-ResourceSet OPTIONAL,           -- Need N
  resourceSetToReleaseList SEQUENCE (SIZE (1..maxNrofPUCCH-ResourceSets))
                           OF PUCCH-ResourceSetId OPTIONAL,        -- Need N
  resourceToAddModList SEQUENCE (SIZE (1..maxNrofPUCCH-Resources))
                       OF PUCCH-Resource OPTIONAL,                 -- Need N
  resourceToReleaseList SEQUENCE (SIZE (1..maxNrofPUCCH-Resources))
                        OF PUCCH-ResourceId OPTIONAL,              -- Need N
  format1 SetupRelease {PUCCH-FormatConfig} OPTIONAL,              -- Need M
  format2 SetupRelease {PUCCH-FormatConfig} OPTIONAL,              -- Need M
  format3 SetupRelease {PUCCH-FormatConfig} OPTIONAL,              -- Need M
  format4 SetupRelease {PUCCH-FormatConfig} OPTIONAL,              -- Need M
  schedulingRequestResourceToAddModList SEQUENCE
       (SIZE (1..maxNrofSR-Resources))
       OF SchedulingRequestResourceConfig OPTIONAL,                -- Need N
  schedulingRequestResourceToReleaseList SEQUENCE
       (SIZE (1..maxNrofSR-Resources))
       OF SchedulingRequestResourceId OPTIONAL,                    -- Need N
  multi-CSI-PUCCH-ResourceList SEQUENCE (SIZE (1..2))
       OF PUCCH-ResourceId OPTIONAL,                               -- Need M
  dl-DataToUL-ACK SEQUENCE (SIZE (1..8))
       OF INTEGER (0..15) OPTIONAL,                                -- Need M
  spatialRelationInfoToAddModList SEQUENCE
       (SIZE (1..maxNrofSpatialRelationInfos))
       OF PUCCH-SpatialRelationInfo OPTIONAL,                      -- Need N
  spatialRelationInfoToReleaseList SEQUENCE
       (SIZE (1..maxNrofSpatialRelationInfos))
       OF PUCCH-SpatialRelationInfoId OPTIONAL,                    -- Need N
  pucch-PowerControl PUCCH-PowerControl OPTIONAL,                  -- Need M
  ...,
  [[
  resourceToAddModListExt-v1610 SEQUENCE
       (SIZE (1..maxNrofPUCCH-Resources))
       OF PUCCH-ResourceExt-v1610 OPTIONAL,                        -- Need N
  dl-DataToUL-ACK-r16 SetupRelease {DL-DataToUL-ACK-r16}
                      OPTIONAL,                                    -- Need M
  ul-AccessConfigListDCI-1-1-r16
       SetupRelease {UL-AccessConfigListDCI-1-1-r16} OPTIONAL,     -- Need M
  subslotLengthForPUCCH-r16 CHOICE {
    normalCP-r16    ENUMERATED {n2, n7},
    extendedCP-r16 ENUMERATED {n2, n6}
```

```
    }
    other parameters (not shown)
    schedulingRequestResourceToAddModListExt-v1610
           SEQUENCE (SIZE (1..maxNrofSR-Resources))
           OF SchedulingRequestResourceConfigExt-v1610 OPTIONAL      -- Need N
    ]],
    [[
    format0-r17 SetupRelease {PUCCH-FormatConfig} OPTIONAL,          -- Need M
    format2Ext-r17 SetupRelease {PUCCH-FormatConfigExt-r17}
                   OPTIONAL,                                         -- Need M
    format3Ext-r17 SetupRelease {PUCCH-FormatConfigExt-r17}
                   OPTIONAL,                                         -- Need M
    format4Ext-r17 SetupRelease {PUCCH-FormatConfigExt-r17}
                   OPTIONAL,                                         -- Need M
    other parameters (not shown)
    schedulingRequestResourceToAddModListExt-v1700
           SEQUENCE (SIZE (1..maxNrofSR-Resources))
           OF SchedulingRequestResourceConfigExt-v1700 OPTIONAL      -- Need N
    dmrs-BundlingPUCCH-Config-r17 SetupRelease {DMRS-BundlingPUCCH-Config-r17}
                                  OPTIONAL,                          -- Need M
    other parameters (not shown)
    ]]
}
```

where `maxNrofPUCCH-ResourceSets` is `4`, which is the **maximum quantity** of **PUCCH resource sets**; `maxNrofPUCCH-Resources` is `128`; `maxNrofSR-Resources` is `64`, which is **maximum quantity** of **SR resources**; and `maxNrofSpatialRelationInfos` is `8`.

### 7.9.5 `PUCCH-ConfigurationList-r16` data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 7.9.2.

Its **only parent data type** is `BWP-UplinkDedicated` (click 2.2.8.10), which contains the **dedicated configuration** for **each Uplink** or **SUL bandwidth part**.

`PUCCH-ConfigurationList-r16` is a **shorthand** for `2` `PUCCH-Config` **instances**, as you may see from its **ASN.1 specification**:

```
PUCCH-ConfigurationList-r16 ::= SEQUENCE (SIZE (1..2)) OF PUCCH-Config
```

These `2` `PUCCH-Config` **instances** are used for multiplexing **low-priority HARQ-ACK** and **high-priority UCI** in a **PUCCH**, allowing for the reporting of **UCI** of **2 different priorities** (click 7.6.8.2).

For more detail on **UCI reporting** with **different priorities**, refer to clause 9.2.5.3 of TS 38.213.

### 7.9.6 `PUCCH-ResourceSet` data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 7.9.2.

Its **only parent data type** is `PUCCH-Config` (click 7.9.4).

**All** its parameters are **dedicated** (click 5.1.3).

**Each** of its **instances contains** a **set** of **PUCCH resources** (identified by their **values** of `PUCCH-ResourceSetId`) and indicates the **maximum payload size** in bits (`maxPayloadSize`) supported by the **set**.

#### 7.9.6.1 PUCCH-ResourceSet parameters

Its **three parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- **maxPayloadSize** **INTEGER (4..256)**: optional, dedicated

  It is the **maximum quantity** of **UCI information bits** that the **UE** may transmit using **this** **PUCCH-ResourceSet** **instance**. Its **value must** be a **multiple of 4**, within the specified range.

  In a **PUCCH occurrence**, the **UE chooses** the **first** of its **PUCCH-ResourceSet** **instances** which **supports** the **quantity** of **bits** that the **UE** wants to **transmit**. The **order** of the **PUCCH-ResourceSet** **instances** is determined by the **value** of their parameter **pucch-ResourceSetId**, and **not** by their position in the **SEQUENCE OF**. The **instance** with value 0 is the **first instance**, and the one with the **largest** value is the **last instance**.

  Parameter **maxPayloadSize** is **absent both** in the **first and last** **PUCCH-ResourceSet** **instances**, because of the way the **UE** selects the **instance** to be used (click 7.6.7).

  For more detail on **maxPayloadSize**, to clause 9.2.1 of TS 38.213.

- **pucch-ResourceSetId** **PUCCH-ResourceSetId**: mandatory, dedicated

  It does **not** have a **field description** in TS 38.331. It is then **assumed** that it is the **identifier** of this **PUCCH-ResourceSet** **instance**.

  Its **data type** **PUCCH-ResourceSetId** is an **INTEGER (0..maxNrofPUCCH-ResourceSets-1)**. The constant **maxNrofPUCCH-ResourceSets-1** is **3**, which is **maximum quantity**, minus **1**, of **PUCCH resource sets** in the corresponding **UE**'s **Uplink bandwidth part**.

- **resourceList** **SEQUENCE (SIZE (1..maxNrofPUCCH-ResourcesPerSet)) OF** **PUCCH-ResourceId**: mandatory, dedicated

  It is the **set** of the **identifiers** of the **PUCCH resources within** this **PUCCH resource set**. The **UE chooses** a **PUCCH resource** from this **set** as specified in clause 9.2.3 of TS 38.213. Note that this is a **set** of **PUCCH-ResourceId** (i.e., **set** of **identifiers**) and **not** of **PUCCH-Resource** **instances**. The actual **PUCCH-Resource** **instances** are configured in parameter **resourceToAddModList**, from **PUCCH-Config** (click 7.9.4). The **order** of the **identifiers** is within the **SEQUENCE OF** is **irrelevant**, and this is why we say it is a **set**, and **not** a **list**.

  **PUCCH-ResourceId** values corresponding to **PUCCH-Resource** **instances** with either **format0** or **format1** are **only** allowed in the **first** **PUCCH resource set**, i.e., when **pucch-ResourceSetId = 0**. This **first** **set** may contain **from 1 to 32** **PUCCH-ResourceId** values.

  **PUCCH-ResourceId** values corresponding to **PUCCH-Resource** **instances** with **format2**, **format3** or **format4** are **only** allowed in **subsequent** **PUCCH resource sets**, i.e., when **pucch-ResourceSetId > 0**. If it is **present**, **each** of these **subsequent** **sets** may contain **from 1 to 8** **PUCCH-ResourceId** values.

The constant **maxNrofPUCCH-ResourcesPerSet** is **32**, which is the **maximum quantity** of **PUCCH resources** within the **first** **PUCCH resource set**.

Its **data type** **PUCCH-ResourceSetId** is an **INTEGER (0..maxNrofPUCCH-ResourceSets-1)**. The constant **maxNrofPUCCH-ResourceSets-1** is **3**, which is **maximum quantity**, minus **1**, of **PUCCH resource sets** in the corresponding **UE**'s **Uplink bandwidth part**.

For more detail on these parameters, refer to the table "*PUCCH-ResourceSet field descriptions*" of TS 38.331.

#### 7.9.6.2 PUCCH-ResourceSet ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
PUCCH-ResourceSet ::= SEQUENCE {
  pucch-ResourceSetId PUCCH-ResourceSetId,
  resourceList SEQUENCE (SIZE (1..maxNrofPUCCH-ResourcesPerSet))
               OF PUCCH-ResourceId,
  maxPayloadSize INTEGER (4..256) OPTIONAL                    -- Need R
}
```

where **maxNrofPUCCH-ResourcesPerSet** is **32**.

### 7.9.7 PUCCH-Resource data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** **CellGroupConfig** (click 3.7.3), click 7.9.2.

Its **only parent data type** is **PUCCH-Config** (click 7.9.4).

**All** its parameters are **dedicated** (click 5.1.3).

**Each** of its **instances** defines the parameters of **1 PUCCH resource**, belonging to the **corresponding UE**'s **Uplink** or **SUL bandwidth part**. The parameters of **any given PUCCH resource** may be **extended** with the parameters from **PUCCH-ResourceExt-v1610** (click 7.9.9). This **extension** is done with the **optional** parameter **resourceToAddModListExt-v1610** from **PUCCH-Config** (click 7.9.4).

The parameters within this **data type apply** to the **PUCCH resource irrespective** of its **format**.

#### 7.9.7.1 PUCCH-Resource parameters

For a description of some **restrictions** on the **combination** of **PUCCH-Resource** **parameters**, click 7.9.10.

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- **format** **CHOICE**: mandatory, dedicated

  It indicates the **format** (click 7.6.5.1) of this **PUCCH-Resource** **instance**, and the corresponding configuration **parameters**. It applies to **all PUCCH** **formats**.

  Its **data type** is a **CHOICE** between parameters **format0** to **format4**, with **data types** **PUCCH-format0** to **PUCCH-format4** (click 7.9.8), respectively. Notice that its **parent data type** **PUCCH-Config** (click 7.9.4) contains parameters **format0-r17** and **format1** to **format4**, that should **not be mistaken** with

these parameters: they have the **same identifier** (except for `format0`), but they are **different parameters**, with **different data type** and **semantics**.

- `intraslotFrequencyHopping` **ENUMERATED** `{enabled}`: optional, dedicated

  When **present**, it **sets** the **intra-slot frequency hopping option** (click 7.6.5.3.4) in this **PUCCH resource** (**irrespective** of its `format`). It is **not applicable** in a **PUCCH resource** with **only 1 symbol interval** (which may **only** happen if this **PUCCH resource** has `format0` or `format2` (**short** `formats`).

  A **UE cannot** be configured with **intra-slot** and **inter-slot frequency hopping** at the same time.

- `pucch-ResourceId` `PUCCH-ResourceId`: mandatory, dedicated

  It is an **identifier** of this **PUCCH resource** (**irrespective** of its `format`). It applies to **all PUCCH** `formats`.

  Its **data type** `PUCCH-ResourceId` is an **INTEGER** `(0..maxNrofPUCCH-Resources-1)`. The constant `maxNrofPUCCH-Resources-1` is `127`, which is the **maximum quantity**, minus **1**, of `PUCCH-Resource` **instances** within **each** of the **UE**'s **Uplink** or **SUL bandwidth part** and **serving cell**.

- `secondHopPRB` `PRB-Id`: optional, dedicated

  When **present**, it is the **PRB index** of the **lowest frequency PRB** in the **second** of **inter-slot frequency hopping** of this **PUCCH resource** (**irrespective** of its `format`). It is **only applicable** when **either** the **intra-slot frequency hopping option** (click 7.6.5.3.4) **or** the **inter-slot frequency hopping option** (click 7.6.5.3.6) is **set**. It is **not applicable** in a **PUCCH resource** with **only 1 symbol interval** (which may **only** happen if this **PUCCH resource** has `format0` or `format2` (**short** `formats`)).

  Its **data type** `PRB-Id` is an **INTEGER** `(0..maxNrofPhysicalResourceBlocks-1)`. The constant `maxNrofPhysicalResourceBlocks-1` is `274`, which is **maximum quantity**, minus **1**, of **PRBs** in **each** of the **UE**'s **bandwidth parts**.

- `startingPRB` `PRB-Id`: mandatory, dedicated

  It is the **PRB index** of the **lowest frequency PRB** of this **PUCCH resource** (**irrespective** of its `format`) in its **only hop** (**no frequency hopping** is **set**) or in its **first hop** (**intra** or **inter slot frequency hopping** is **set**).

  Its **data type** `PRB-Id` is an **INTEGER** `(0..maxNrofPhysicalResourceBlocks-1)`. The constant `maxNrofPhysicalResourceBlocks-1` is `274`, which is **maximum quantity**, minus **1**, of **PRBs** in **each** of the **UE**'s **bandwidth parts**.

For more detail on these parameters, refer to the table "*PUCCH-Resource, PUCCH-ResourceExt field descriptions*" of TS 38.331 and to clause 9.2.1 of TS 38.213.

#### 7.9.7.2 PUCCH-Resource ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
PUCCH-Resource ::= SEQUENCE {
  pucch-ResourceId PUCCH-ResourceId,
  startingPRB PRB-Id,
  intraSlotFrequencyHopping ENUMERATED {enabled} OPTIONAL,        -- Need R
  secondHopPRB PRB-Id OPTIONAL,                                   -- Need R
  format CHOICE {
    format0 PUCCH-format0,
    format1 PUCCH-format1,
    format2 PUCCH-format2,
    format3 PUCCH-format3,
    format4 PUCCH-format4
  }
}
```

### 7.9.8 PUCCH-format0 to PUCCH-format4 data types

They are **unicast** within the **Non-SI parameter group** (click 5.3.6).

For their **ancestor tree up to** CellGroupConfig (click 3.7.3), click 7.9.2.

Their **only parent data type** is PUCCH-Resource (click 7.9.7).

**All** their parameters are **dedicated** (click 5.1.3).

**Each** of their **instances** determines the format of its **enclosing instance** of PUCCH-Resource (which defines a **PUCCH resource**). It also provides **configuration parameters** for this specific **PUCCH resource**, that complement the **parameters** that apply to **all PUCCH Resources** of each given format, defined in parameters format0-r17 and format1 to format4 from PUCCH-Config (click 7.9.4).

#### 7.9.8.1 PUCCH-format0 to PUCCH-format4 parameters

**Only** PUCCH-format3 has a table of **field descriptions** (with **only one field**) in TS 38.331. **None** of the other data types PUCCH-format0, 1, 2 or 4 has a table of **field descriptions**.

**These data types** have **several same-name parameters**, with the **same data type** and **same semantics**. This is why **their parameters** are **explained together**, right below.

**All** their **parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- initialCyclicShift **INTEGER(0..11)**: mandatory, dedicated

  It does **not** have a **field description** in TS 38.331. It is then **assumed** that it is the value of "$m_0$," which is the **initial cyclic shift** for the **sequence generation** in **UCI** transmission of (click 7.7) for the **PUCCH resource**. It **only** applies to **PUCCH** format0 and format2 (**short** formats), because it only exists in PUCCH-format0 and in PUCCH-format1.

- nrofPRBs **INTEGER(1..16)**: mandatory, dedicated

  It is **one** of the **few** (**quasi**)**same-name parameters** from **different data types** that has a **field description**. However, its **field description** (specific for PUCCH-format-3) **only** indicates that the **UE** shall **ignore** this parameter when format-v1610 from PUCCH-ResourceExt-v1610 (click 7.9.9) is **present**. **All** its **other** (**quasi**)**same-name parameters** from **different data types** have a **different**

**context** to the one of this parameter, so their **semantics** are **not applicable** to this parameter.

It is then **assumed** that `nrofPRBs` is the **quantity** of **PRBs** of the **PUCCH resource**.

Parameter `nrofPRBs` **only** applies to **PUCCH** `format2` and `format3` (it only exists in `PUCCH-format2` and `PUCCH-format3`). It may take the values `1` to `16` for **PUCCH** `format0` or `format2` and `1` to `16`, except **7** and **11**, for `format3`. Parameter `nrofPRBs` is ignored for `format3` when `format-v1610` from `PUCCH-ResourceExt-v1610` (click 7.9.9) is **present**. Parameter `format-v1610` **chooses either** `interlace1-v1610` or `occ-v1610`.

For **FR2-2 only**, **PUCCH** `format0`, `format1` and `format4` are **extended** to include parameter `nrofPRBs-r17` from `PUCCH-ResourceExt-v1610` (click 7.9.9) that has the same semantics as `nrofPRBs`. The values of `nrofPRBs-r17` may be from `1` to `16` for **PUCCH** `format0` and `format1`, and `1` to `16`, except **7** and **11**, for `format4`.

- `nrofSymbols` **`INTEGER`**: mandatory, dedicated

  It does **not** have a **field description** in TS 38.331. There is a **field description** for the **version-different** (click 5.7.2.2) **parameter** `nrofSymbols-r16` from `RSSI-ResourceConfigCLI-r16`, but it does **not apply** to `nrofSymbols`.

  It is then **assumed** that it is the **quantity** of **symbol intervals** of the **PUCCH resource**. It applies to **all PUCCH** `formats` (it exists in `PUCCH-format0` to `PUCCH-format4`). Its possible values are `1` or `2` for **PUCCH** `format0` and `format2` (**short** `formats`), and `4` to `14` for **PUCCH** `format1`, `format3` and `format4` (**long** `formats`).

- `occ-Index` **`ENUMERATED`** **`{`**`n0, n1, n2, n3`**`}`**: mandatory, dedicated

  Its **field description** is in the table "*PUCCH-Resource, PUCCH-ResourceExt field descriptions*" of TS 38.331.

  It indicates the **orthogonal cover code index** of the **PUCCH resource**. It **only** applies to `format4` (it only exists in `PUCCH-format4`). It is **only** applicable when **PUCCH-PUSCH interlace** is **set** (click 2.2.4.8). For more detail, refer to clause 9.2.1 of TS 38.213.

  **All** `formats` may be **extended** to include the **additional** parameter `occ-Index-v1600` from `PUCCH-ResourceExt-v1610` (click 7.9.9) that has the **same semantics** as `occ-Index` and may take its **same values**.

- `occ-Length` **`ENUMERATED`** **`{`**`n2, n4`**`}`**: mandatory, dedicated

  It is the **orthogonal cover code length** of the **PUCCH resource**. It **only** applies to `format4` (it only exists in `PUCCH-format4`). It is **only** applicable when `useInterlacePUCCH-PUSCH-r16`, **either** from `BWP-UplinkCommon` (click 2.2.8.9) or from `BWP-UplinkDedicated` (click 2.2.8.10), is **present**. For more detail, refer to clause 9.2.1 of TS 38.213.

  Its **field description** is in the table "*PUCCH-Resource, PUCCH-ResourceExt field descriptions*" of TS 38.331.

**All** `formats` may be **extended** to include the **additional** parameter `occ-Length-v1600` from `PUCCH-ResourceExt-v1610` (click 7.9.9) that has the **same semantics** as `occ-Length` and may take its **same values**.

- `startingSymbolIndex` `INTEGER`: mandatory, dedicated

  It does **not** have a **field description** in TS 38.331. It is then **assumed** that in **dynamic scheduling** (click 7.6.1) it is the **first symbol interval** of the **PUCCH resource** while in **semi-persistent scheduling** (click 7.6.2) it is the value of variable "$l_0$". It applies to **all PUCCH** `formats` (it exists in `PUCCH-format0` to `PUCCH-format4`). Its possible values are `0` to `13` for **PUCCH** `format0` and `format2` (**short** `formats`), and `0` to `10` for **PUCCH** `format1`, `format3` and `format4` (**long** `formats`).

- `timeDomainOCC` `INTEGER (0..6)`: mandatory, dedicated

  It does **not** have a **field description** in TS 38.331. It is then **assumed** that it is an **index** for an **orthogonal cover code** of the **PUCCH resource**. It **only** applies to **PUCCH** `format1` (it only exists in `PUCCH-format1`).

### 7.9.8.2 `PUCCH-format0` to `PUCCH-format4` ASN.1 specifications

The **complete** specifications of `PUCCH-format0` to `PUCCH-format4` are:

```
PUCCH-format0 ::= SEQUENCE {
  initialCyclicShift INTEGER(0..11),
  nrofSymbols INTEGER (1..2),
  startingSymbolIndex INTEGER(0..13)
}

PUCCH-format1 ::= SEQUENCE {
  initialCyclicShift INTEGER(0..11),
  nrofSymbols INTEGER (4..14),
  startingSymbolIndex INTEGER(0..10),
  timeDomainOCC INTEGER(0..6)
}

PUCCH-format2 ::= SEQUENCE {
  nrofPRBs INTEGER (1..16),
  nrofSymbols INTEGER (1..2),
  startingSymbolIndex INTEGER(0..13)
}

PUCCH-format3 ::= SEQUENCE {
  nrofPRBs INTEGER (1..16),
  nrofSymbols INTEGER (4..14),
  startingSymbolIndex INTEGER(0..10)
}

PUCCH-format4 ::= SEQUENCE {
  nrofSymbols INTEGER (4..14),
  occ-Length ENUMERATED {n2, n4},
  occ-Index ENUMERATED {n0, n1, n2, n3},
  startingSymbolIndex INTEGER(0..10)
}
```

## 7.9.9 `PUCCH-ResourceExt-v1610` data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 7.9.2.

Its **only parent data type** is `PUCCH-Config` (click 7.9.4).

**All** its parameters are **dedicated** (click 5.1.3).

**Each** of its **instances extends** the parameters of **1 PUCCH resource**, belonging to the **corresponding UE**'s **bandwidth part**. This **extension** is done with the **optional** parameter `resourceToAddModListExt-v1610` from `PUCCH-Config` (click 7.9.4). Recall that the parameters of **each PUCCH resource** are defined with `PUCCH-Resource` (click 7.9.7).

#### 7.9.9.1 `PUCCH-ResourceExt-v1610` parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `format-v1610` **`CHOICE`**: optional, dedicated

  It is a **`CHOICE` between** `interlace1-v1610` and `occ-v1610` (see below).

- `format-v1700` **`SEQUENCE`**: optional, dedicated

  It is a **`SEQUENCE`** that contains **only one** parameter: `nrofPRBs-r17` (see below).

- `interlace0-r16` **`CHOICE`**: mandatory (within optional), dedicated

  It identifies the **only** interlace of the **PUCCH resource** for **PUCCH** `format0` and `format1`, and the **first** interlace for `format2` and `format3`. It applies to **all PUCCH** `formats` except `format4`.

  Its **data type** is a **`CHOICE`** between `scs15` and `scs30`. The **chosen parameter** configures the **subcarrier spacing**. Whitin each choice of subcarrier spacing, the possible values determine the interlace. The values of `scs15` (for **15 kHz SCS**) can be from `0` to `9`, and the ones of `scs30` (for **30 kHz SCS**) can be from `0` to `4`.

- `interlace1-v1610` **`INTEGER (0..9)`**: choice, dedicated

  It identifies the **second** interlace of the **PUCCH resource**. It **only** applies to **PUCCH** `format2` and `format3`.

  Its possible values are from `0` to `9`. For **15k Hz SCS**, its values **must** satisfy the following expression:

  $$\texttt{interlace1-v1610} = (\texttt{interlace0-r16} + X) \ \textit{mod}\ 10$$

  where "$X \in \{1, -1, 5\}$".

- `interlaceAllocation-r16` **`SEQUENCE`** : optional, dedicated

  It is a **`SEQUENCE`** that contains `rb-setIndex-r16` and `interlace0-r16`

- `nrofPRBs-r17` **`INTEGER (1..16)`**: mandatory (within optional), dedicated

  For **FR2-2 only**, it **adds** this parameter to this **PUCCH resource**. It **only** applies to `format0`, `format1` and `format4`. It is quantity of **PRBs** of the **PUCCH resource** (i.e., the same semantics as `nrofPRBs` from `PUCCH-format2` and `PUCCH-format3`, click 7.9.8). The values of `nrofPRBs-r17` may be from `1` to `16` for **PUCCH** `format0` and `format1`, and `1`, `2`, `3`, `4`, `5`, `6`, `8`, `9`, `10`, `12`, `15` or `16` for `format4`.

- `occ-Index-v1610` **`ENUMERATED {n0, n1, n2, n3}`**: optional, dedicated

  It is an **extension** of `occ-Index` from `PUCCH-Resource` (click 7.9.7) for this **PUCCH resource**. It has the same semantics as `occ-Index` (the **orthogonal cover**

**code index**) and may take its same values (`0`, `1`, `2` or `3`). However, `occ-Index` is **not** applicable to **PUCCH** `format2`, while `occ-Index-v1610` is applicable to **all PUCCH** `formats`.

- `occ-Length-v1610` **`ENUMERATED {n2, n4}`**: optional, dedicated

  It is an **extension** of `occ-Length` from `PUCCH-Resource` (click 7.9.7) for this **PUCCH resource**. It has the same semantics as `occ-Length` (the **orthogonal cover code length**) and may take its same values (`2` or `4`). However, `occ-Length` is **not** applicable to **PUCCH** `format2`, while `occ-Length-v1610` is applicable to **all PUCCH** `formats`.

- `occ-v1610` **`SEQUENCE`**: choice, dedicated

  It is **only** applicable when `useInterlacePUCCH-PUSCH-r16`, **either** from `BWP-UplinkCommon` (click 2.2.8.9) **or** from `BWP-UplinkDedicated` (click 2.2.8.10), is **present**.

  It is a **`SEQUENCE`** that contains `occ-Length-v1610` and `occ-Index-v1610` (see above).

- `pucch-RepetitionNrofSlots-r17` **`INTEGER (1..16)`**: optional, dedicated

  It is the **quantity** of **slots** for the **PUCCH repeat** in the **PUCCH resource**, with associated scheduling **DCI** corresponding to **Rel-17 dynamic PUCCH repetition**. In some contexts, the quantity of **repeated slots** is denoted by "$N_{PUCCH}^{repeat}$".

  It has the same semantics (the **quantity of slots** for the **PUCCH repeat**) as `nrofSlots` from `PUCCH-FormatConfig` (click 7.9.10). However, `nrofslots` applies to **each** and **every PUCCH resource** (of the corresponding **bandwidth part**) that has the corresponding `format`, while `pucch-RepetitionNrofSlots-r17` applies to **only 1 specific PUCCH resource** with associated scheduling **DCI**. Also, `nrofslots` does **not** apply to **PUCCH** `format2`, while `pucch-RepetitionNrofSlots-r17` applies to **all PUCCH** `formats`.

  For a **PUCCH resource**, if **both** the parameter `pucch-RepetitionNrofSlots-r17` and the parameter `nrofSlots` are present, then `nrofSlots` is ignored and `pucch-RepetitionNrofSlots-r17` is applied. If `pucch-RepetitionNrofSlots-r17` is absent in a **PUCCH resource** with associated scheduling **DCI**, then `nrofSlots` is applied.

  For more detail on this parameter, click 7.6.6.1.

- `rb-setIndex-r16` **`INTEGER (0..4)`**: mandatory (within optional), dedicated

  It indicates the **RB set** that intersects with the **interlace** (click 2.2.4.6) `interlace0-r16` (and if a **second** interlace is defined by `interlace1-v1610`, also with it) to determine which are the **PRBs** of the **PUCCH resource**. Its possible values are `0` to `4`. It applies to **all PUCCH** `formats`.

  An **RB set** is a **range** of **contiguous CRBs**. An **RB set group** is a **group** of **RB sets** that do **not overlap**. For more detail, refer to clause 11.1.1 of TS 38.213, clause 9.3.1.230 of TS 38.473 and the data type `AvailableRB-SetsPerCell`, from "*AvailableRB-SetsPerCell field descriptions*" of TS 38.331.

For more detail on these parameters, refer to the table "*PUCCH-Resource, PUCCH-ResourceExt field descriptions*" of TS 38.331 and to clause 9.2.1 of TS 38.213.

#### 7.9.9.2 PUCCH-ResourceExt-v1610 ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
PUCCH-ResourceExt-v1610 ::= SEQUENCE {
  interlaceAllocation-r16 SEQUENCE {
    rb-SetIndex-r16 INTEGER (0..4),
    interlace0-r16 CHOICE {
      scs15 INTEGER (0..9),
      scs30 INTEGER (0..4)
    }
  } OPTIONAL,                                                    --Need R
  format-v1610 CHOICE {
    interlace1-v1610 INTEGER (0..9),
    occ-v1610 SEQUENCE {
      occ-Length-v1610 ENUMERATED {n2, n4} OPTIONAL,             -- Need M
      occ-Index-v1610 ENUMERATED {n0, n1, n2, n3} OPTIONAL       -- Need M
    }
  } OPTIONAL,                                                    -- Need R
  ...,
  [[
  format-v1700 SEQUENCE {nrofPRBs-r17 INTEGER (1..16)}   OPTIONAL, -- Need R
  pucch-RepetitionNrofSlots-r17 ENUMERATED {n1,n2,n4,n8} OPTIONAL  -- Need R
  ]]
}
```

### 7.9.10 PUCCH-FormatConfig data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 7.9.2.

Its **only parent data type** is `PUCCH-Config` (click 7.9.4).

**All** its parameters are **dedicated** (click 5.1.3).

**Each** of its **instances** contains the **configuration parameters** for **each** and **every PUCCH resource** that have the `format` that is **identified** by the **parameter** that **instantiates** this **data type**. This **data type** is **instantiated** in parameters `format0-r17` to `format4`, **all** of them from the `PUCCH-Config` (click 7.9.4) **instance** of the **corresponding UE**'s **bandwidth part**.

For **PUCCH** `format2` **to** `format4`, there may be **one additional** configuration **parameter** `maxCodeRateLP-r17` from `PUCCH-FormatConfigExt-r17` (click 7.9.11). This additional parameter is configured within parameters `format2Ext-r17`, `format3Ext-r17` and `format4Ext-r17`, **all** of them from `PUCCH-Config` (click 7.9.4) and with **data type** of `PUCCH-FormatConfigExt-r17` (click 7.9.11).

#### 7.9.10.1 PUCCH-FormatConfig parameters

**All** of its **parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `additionalDMRS` **ENUMERATED** `{true}`: optional, dedicated

  When **present** it **sets additional DM-RS** (click 7.6.8) in **each** and **every PUCCH resource** (of the corresponding **bandwidth part**) that has the corresponding `format` (see above).

  It is **only** applicable to **PUCCH** `format3` and `format4`.

For more detail, refer to clause 6.4.1.3.3.2 of TS 38.211.

- `interslotFrequencyHopping` **ENUMERATED** `{enabled}`: optional, dedicated

  When **present** it **sets inter-slot frequency hopping** in **each** and **every PUCCH resource** (of the corresponding **bandwidth part**) that has the corresponding `format` (see above).

  It is applicable to **all** `formats` **except** `format2`.

  It may **only** be present if the **inter-slot repeat option** (click 7.6.5.3.5) is **set**. A **UE cannot** be configured with **intra-slot** and **inter-slot frequency hopping** at the same time.

- `maxCodeRate` `PUCCH-MaxCodeRate`: optional, dedicated

  It determines the **maximum code rate** in **each** and **every PUCCH resource** (of the corresponding **bandwidth part**) that has the corresponding `format` (see above).

  It is applicable to **all** `formats`.

  Its **data type** `PUCCH-MaxCodeRate` is an **ENUMERATED** `{zeroDot08, zeroDot15, zeroDot25, zeroDot35, zeroDot45, zeroDot60, zeroDot80}`, where these enumerated items represent the values `0.08`, `0.15`, `0.25`, `0.35`, `0.45`, `0.60` or `0.80`.

  Notice that there is a **complementary** parameter `maxCodeRateLP-r17` from `PUCCH-FormatConfigExt-r17` (click 7.9.11), that has the **same data type**.

- `nrofSlots` **ENUMERATED** `{n2, n4, n8}`: optional, dedicated

  It is the **quantity** "$N_{PUCCH}^{repeat}$" of repeated **slots** in the **inter-slot repeat option** (click 7.6.5.3.5) for **each** and **every PUCCH resource** (of a given **bandwidth part**) that has the corresponding `format` (see above). Its enumerated values represent `2`, `4` or `8` **slots**.

  It is applicable to **all PUCCH** `formats` **except** `format2`.

  Notice that parameter `pucch-RepetitionNrofSlots-r17` from `PUCCH-ResourceExt-v1610` (click 7.9.9), has the **same semantics** (with possible values `1` to `16`) but it is specific to **1 PUCCH resource**, and it may be applied to **all PUCCH** `formats`, including `format2`.

- `pi2BPSK` **ENUMERATED** `{enabled}`: optional, dedicated

  When **present** it **sets** the **possible usage** of **π/2-PBSK** modulation in **each** and **every PUCCH resource** (of the corresponding **bandwidth part**) that has the corresponding `format` (see above).

  It is applicable to **all PUCCH** `formats`.

- `simultaneousHARQ-ACK-CSI` **ENUMERATED** `{true}`: optional, dedicated

  When **present** it **sets** the **simultaneous transmission** of **CSI** and **HARQ-ACK feedback with** or **without SR** in **each** and **every PUCCH resource** (of the corresponding **bandwidth part**) that has the corresponding `format` (see above).

It is **only** applicable to **PUCCH** `format2`, `format3` or `format4`.

For more detail, refer to clause 9.2.5 of TS 38.213.

For more detail on these parameters, refer to the table "*PUCCH-FormatConfig, PUCCH-FormatConfigExt field descriptions*" of TS 38.331 and to clause 9.2.1 of TS 38.213.

#### 7.9.10.2 `PUCCH-FormatConfig` ASN.1 specification

Its complete **ASN.1** (click 5.9) specification is:

```
PUCCH-FormatConfig ::= SEQUENCE {
  interslotFrequencyHopping ENUMERATED {enabled} OPTIONAL,     -- Need R
  additionalDMRS ENUMERATED {true} OPTIONAL,                   -- Need R
  maxCodeRate PUCCH-MaxCodeRate OPTIONAL,                      -- Need R
  nrofSlots ENUMERATED {n2, n4, n8} OPTIONAL,                  -- Need S
  pi2BPSK ENUMERATED {enabled} OPTIONAL,                       -- Need R
  simultaneousHARQ-ACK-CSI ENUMERATED {true} OPTIONAL          -- Need R
}
```

### 7.9.11 `PUCCH-FormatConfigExt-r17` data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 7.9.2.

Its **only parent data type** is `PUCCH-Config` (click 7.9.4).

**All** its parameters are **dedicated** (click 5.1.3).

**Each** of its **instances** contains **one additional** configuration **parameter** for **each** and **every PUCCH resource** (of a **given bandwidth part**) that have the `format` identified by the **parameter name** (i.e., `format2Ext-r17`, `format3Ext-r17` and `format4Ext-r17` that **instantiates** this **data type**).

Its **only** parameter is:

- `maxCodeRateLP-r17` `PUCCH-MaxCodeRate`: optional, dedicated

  It configures an **additional maximum code rate** in **each** and **every PUCCH resource** (of the corresponding **bandwidth part**) of the corresponding `format` (identified by the corresponding **parameter name**). It is **only** applicable when `uci-MuxWithDiffPrio-r17` from `PhysicalCellGroupConfig` (click 3.7.5) is **present** and for **PUCCH** `format2`, `format3` and `format4`.

  This parameter `maxCodeRateLP-r17` is **present** in the **second** `PUCCH-Config` (click 7.9.4) **entry** of `PUCCH-ConfigurationList-r16` (click 7.9.5) and is **absent** from the **first** `PUCCH-Config` (click 7.9.4) **entry**. Therefore, in the **second entry**, this parameter is **present** in **addition** to `maxCodeRate` from `PUCCH-FormatConfig` (click 7.9.10).

  The **data type** `PUCCH-ConfigurationList-r16` is a `SEQUENCE OF 2` **nameless** parameters of **data type** `PUCCH-Config`.

  The **data type** `PUCCH-FormatConfig` is **instantiated** in parameters `format1` to `format4`, within `PUCCH-Config` (click 7.9.4). The **data type** of parameters `format1` to `format4` is, obviously, `PUCCH-FormatConfig`.

  This parameter is used for **UCI reporting** of **different priorities** (click 7.6.8.2).

`maxCodeRateLP-r17` has the **same data type** as `maxCodeRate` (see below): `PUCCH-MaxCodeRate`, which is an `ENUMERATED {zeroDot08, zeroDot15, zeroDot25, zeroDot35, zeroDot45, zeroDot60, zeroDot80}`, where these enumerated items represent the values `0.08`, `0.15`, `0.25`, `0.35`, `0.45`, `0.60` or `0.80`.

For more detail on this parameter, refer to the table "*PUCCH-FormatConfig, PUCCH-FormatConfigExt field descriptions*" of TS 38.331.

Its complete **ASN.1** (click 5.9) specification is:

```
PUCCH-FormatConfigExt-r17 ::= SEQUENCE {
  maxCodeRateLP-r17 PUCCH-MaxCodeRate OPTIONAL,              -- Need R
  ...
}
```

### 7.9.12 `SchedulingRequestResourceConfig` data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 7.9.2.

Its **only parent data type** is `PUCCH-Config` (click 7.9.4).

**All** its parameters are **dedicated** (click 5.1.3).

**Each** of its **instances** configures (within the **corresponding UE**'s **bandwidth part**) the **scheduling** (**periodicity** and **offset**) of **PUSCH scheduling request transmission opportunities** of **1 PUCCH resource** with **PUCCH** `format0` or `format1`. The parameters of **each** of its **instances** may be **extended** with the parameters from the **data types** `SchedulingRequestResourceConfigExt-v1610` and/or `-v1700` (click 7.9.13). This **extension** is done (respectively) with the **optional** parameters `schedulingRequestResourceToAddModListExt-v1610` and `-v1700` from `PUCCH-Config` (click 7.9.4).

#### 7.9.12.1 `SchedulingRequestResourceConfig` parameters

**All** of its **parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `periodicityAndOffset` `CHOICE`: optional, dedicated

  It indicates "***SR***$_{PERIODICITY}$" **and** the "***SR***$_{OFFSET}$" for **1 instance** of **semi-persistent scheduling** of **scheduling requests** (click 7.6.2). The value of "***SR***$_{PERIODICITY}$" is indicated by the **parameter** that is **chosen** in the `CHOICE`. The "***SR***$_{PERIODICITY}$" of the **2 first parameters** in the `CHOICE` is expressed in **symbol intervals** and for **all** the **other ones** is expressed in **slots**. The value of "***SR***$_{OFFSET}$" is indicated by the **value** of the **parameter** itself. The value of "***SR***$_{OFFSET}$" is `0` for the **first 3 parameters** of the `CHOICE`, and it may be from `0` to **periodicity** (minus 1) for **each** of the **other parameters**. The following **periodicities** (and their

corresponding **offsets**) may be configured depending on the **PUCCH subcarrier spacing**:

| | | $SR_{PERIODICITY}$ | |
|---|---|---|---|
| **μ** | **SCS [kHz]** | **[sym. inter.]** | **[slots]** |
| **0** | **15** | 2, 7 | 1, 2, 4, 5, 8, 10, 16, 20, 40, 80 |
| **1** | **30** | 2, 7 | 1, 2, 4, 8, 10, 16, 20, 40, 80, 160 |
| **2** | **60** | 2, sym6or7 | 1, 2, 4, 8, 10, 16, 20, 40, 80, 160, 320 |
| **3** | **120** | 2, 7 | 1, 2, 4, 8, 10, 16, 20, 40, 80, 160, 320, 640 |
| **5** | **480** | N/A | 1, 2, 4, 8, 10, 16, 20, 40, 80, 160, 320, 640 |
| **6** | **960** | N/A | 1, 2, 4, 8, 10, 16, 20, 40, 80, 160, 320, 640 |

The parameter "sym6or7" corresponds to **7 symbol intervals** except for the case of **SCS=60 kHz** and **extended cyclic prefix**, where it corresponds to **6 symbol intervals**.

If periodicityAndOffset-r17 from SchedulingRequestResourceConfig-1700 (click 7.9.13) is **present**, any previously configured periodicityAndOffset is **released**, and **vice versa**.

- resource PUCCH-ResourceId (click 7.9.7): optional, dedicated

  It **identifies** (**index**) the PUCCH-Resource (click 7.9.7) **instance** in which the **UE** shall send the **scheduling request** (**SR**). The actual PUCCH-Resource **instance** is configured in the PUCCH-Config (click 7.9.4) **instance** of the same **Uplink bandwidth part** and **serving cell** as this SchedulingRequestResourceConfig **instance**. The PUCCH-Resource (click 7.9.7) **instance must** be of format0 or format1 (click 7.6.5.1). **Other** formats are **not supported** (refer to clause 9.2.4 of TS 38.213).

- schedulingRequestId SchedulingRequestId: mandatory, dedicated

  **Each instance** of SchedulingRequestResourceConfig takes the values of sr-ProhibitTimer and sr-TransMax from the **instance** of SchedulingRequestToAddMod (click 7.9.15) that has this **identifier**.

  Its **data type** SchedulingRequestId is an **INTEGER (0..7)**.

- schedulingRequestResourceId SchedulingRequestResourceId: mandatory, dedicated

  It is the **identifier** of **each instance** of SchedulingRequestResourceConfig.

  Its **data type** SchedulingRequestResourceId is an **INTEGER (1..maxNrofSR-Resources)**. The constant maxNrofSR-Resources is 8, which is the **maximum quantity** of **scheduling request resources** within **each bandwidth part**.

For more detail on these parameters, refer to the table "*SchedulingRequestResourceConfig field descriptions*" of TS 38.331.

#### 7.9.12.2 SchedulingRequestResourceConfig ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
SchedulingRequestResourceConfig ::= SEQUENCE {
  schedulingRequestResourceId SchedulingRequestResourceId,
  schedulingRequestId SchedulingRequestId,
  periodicityAndOffset CHOICE {
    sym2     NULL,
    sym6or7 NULL,
    sl1      NULL,                                    -- Recurs in every slot
    sl2      INTEGER (0..1),
    sl4      INTEGER (0..3),
    sl5      INTEGER (0..4),
    sl8      INTEGER (0..7),
    sl10     INTEGER (0..9),
    sl16     INTEGER (0..15),
    sl20     INTEGER (0..19),
    sl40     INTEGER (0..39),
    sl80     INTEGER (0..79),
    sl160    INTEGER (0..159),
    sl320    INTEGER (0..319),
    sl640    INTEGER (0..639)
  } OPTIONAL,                                        -- Need M
  resource PUCCH-ResourceId OPTIONAL                 -- Need M
}
```

### 7.9.13 SchedulingRequestResourceConfigExt-v1610 and -v1700 data types

They are only **unicast** within the **Non-SI parameter group** (click 5.3.6).

For their **ancestor tree up to** CellGroupConfig (click 3.7.3), click 7.9.2.

Their **only parent data type** is PUCCH-Config (click 7.9.4).

**All** their parameters are **dedicated** (click 5.1.3).

**Each** of their **instances** contains parameters to **extend** SchedulingRequestResourceConfig (click 7.9.12). This **extension** is done with the **optional** parameters schedulingRequestResourceToAddModListExt-v1610 and -v1700 from PUCCH-Config (click 7.9.4).

#### 7.9.13.1 SchedulingRequestResourceConfigExt-v1610 and -v1700 parameters

The **only** parameter (indicating its **data type** and **semantics**) of SchedulingRequestResourceConfigExt-v1610 is:

- phy-PriorityIndex **ENUMERATED** {p0, p1}: optional, dedicated

  It indicates whether this **scheduling request resource** is **high** or **low priority** in **PHY prioritization/multiplexing handling** (refer to clause 9.2.4 of TS 38.213). Value "p0" indicates **low priority** and value "p1" indicates **high priority**.

The **only** parameter (indicating its data type and semantics) of SchedulingRequestResourceConfigExt-v1700 is:

- periodicityAndOffset-r17 **CHOICE**: optional, dedicated

  The **semantics** of **this parameter** is the same as the one of periodicityAndOffset from SchedulingRequestResourceConfig

(click 7.9.12). This **parameter** allows for **additional** "***SR*$_{PERIODICITY}$**" **and** "***SR*$_{OFFSET}$**" **values** over the ones of `periodicityAndOffset`. The following are the **additional periodicities** that may be configured, depending on the **PUCCH subcarrier spacing**:

| μ | SCS [kHz] | *SR*$_{PERIODICITY}$ [slots] |
|---|---|---|
| 5 | 480 | 1280, 2560 |
| 6 | 960 | 1280, 2560, 5120 |

For more detail on these parameters, refer to the table "*SchedulingRequestResourceConfig field descriptions*" of TS 38.331.

### 7.9.13.2 `SchedulingRequestResourceConfigExt-v1610` and `-v1700` ASN.1 specification

Their **complete** specifications are:

```
SchedulingRequestResourceConfigExt-v1610 ::= SEQUENCE {
  phy-PriorityIndex-r16 ENUMERATED {p0, p1} OPTIONAL,              -- Need M
  ...
}

SchedulingRequestResourceConfigExt-v1700 ::= SEQUENCE {
  periodicityAndOffset-r17 CHOICE {
    sl1280 INTEGER (0..1279),
    sl2560 INTEGER (0..2559),
    sl5120 INTEGER (0..5119)
  } OPTIONAL                                                       -- Need M
}
```

## 7.9.14 `SchedulingRequestConfig` data type

It can be **both unicast** within the **Non-SI parameter group** (click 5.3.6) and **sent** within an **RRC inter-node message** (click 3.7.2).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 7.9.2.

For its **ancestor tree up to RRC inter-node message**, click 3.7.2.

Its **only parent data type** is `MAC-CellGroupConfig` (click 3.7.4).

**All** of its parameters are **dedicated** (click 5.1.3).

**Each** of its **instances adds/removes/modifies** a **list** of **instances** of `SchedulingRequestToAddMod` to/from/in the **dedicated configuration** (for **all** the **cells** of **1** of the **UE**'s **cell groups**) of the **PUCCH scheduling requests**.

### 7.9.14.1 `SchedulingRequestConfig` parameters

Its **two parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `schedulingRequestToAddModList SEQUENCE (SIZE (1..maxNrofSR-ConfigPerCellGroup)) OF SchedulingRequestToAddMod` (click 7.9.15): optional, dedicated

  It **contains** a **list** of **instances** of `SchedulingRequestToAddMod` to be **added** or **modified** to/in the **dedicated configuration** (for **all** the **cells** of **1** of the **UE**'s **cell groups**) of the **PUCCH scheduling requests**.

If `schedulingRequestToAddModListExt` from `SchedulingRequestConfig-v1700` is **present**, it contains the **same quantity** of **entries**, and in the **same order**, as `schedulingRequestToAddModList`.

- `schedulingRequestToReleaseList SEQUENCE (SIZE (1..maxNrofSR-ConfigPerCellGroup)) OF SchedulingRequestToAddMod` (click 7.9.15): optional, dedicated

  It **contains** a **list** of **instances** of `SchedulingRequestToAddMod` to be **removed** from the **dedicated configuration** (for **all** the **cells** of **1** of the **UE**'s **cell groups**) of the **PUCCH scheduling requests**.

The value `maxNrofSR-ConfigPerCellGroup` of is `8`, which is the **maximum quantity** of **scheduling request instances** in the corresponding **cell group**.

For more detail on these parameters, refer to the table "*SchedulingRequestConfig field descriptions*" of TS 38.331.

### 7.9.14.2 `SchedulingRequestConfig` ASN.1 specification

Its complete **ASN.1** (click 5.9) specification is:

```
SchedulingRequestConfig ::= SEQUENCE {
  schedulingRequestToAddModList
            SEQUENCE (SIZE (1..maxNrofSR-ConfigPerCellGroup))
            OF SchedulingRequestToAddMod OPTIONAL,                -- Need N
  schedulingRequestToReleaseList
            SEQUENCE (SIZE (1..maxNrofSR-ConfigPerCellGroup))
            OF SchedulingRequestId OPTIONAL                       -- Need N
}
```

## 7.9.15 `SchedulingRequestToAddMod` data type

It can be **both instantiated** within the **Non-SI parameter group** (click 5.3.6) and **sent** within an **RRC inter-node message** (click 3.7.2).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 7.9.2.

For its **ancestor tree up to RRC inter-node message**, click 3.7.2.

Its **only parent data type** is `SchedulingRequestConfig` (click 7.9.14).

**All** of its parameters are **dedicated** (click 5.1.3).

**Each** of its **instances** provides `sr-ProhibitTimer`, which is the **minimum time** before **retransmitting** a **scheduling request** and `sr-TransMax`. Each instance is assigned an **identifier** `schedulingRequestId`, in order to **reference** it from `SchedulingRequestResourceConfig` (click 7.9.12).

### 7.9.15.1 `SchedulingRequestToAddMod` parameters

Its **three parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `schedulingRequestId SchedulingRequestId`: mandatory, dedicated

  It is the **identifier** of the **PUCCH scheduling request** that is **added** or **modified** (with this **data type instance**) to/from the **dedicated configuration** (for **all** the **cells** of **1** of the **UE**'s **cell groups**)

  Its **data type** `SchedulingRequestId` is an `INTEGER (0..7)`.

- `sr-ProhibitTimer` **ENUMERATED**: optional, dedicated

  It is the **minimum time** that the **UE must wait** for a **PUCCH assignment**, **after** having **sent** a **scheduling request**, before **retransmitting** the **scheduling request** (refer to TS 38.321). Its possible values are: `1`, `2`, `4`, `8`, `16`, `32`, `64` or `128` **ms**.

  If `sr-ProhibitTimer-v1700` from `SchedulingRequestToAddModExt-v1700` is **present**, the **UE** shall use its **value**, ignoring `sr-ProhibitTimer`. The possible values of `sr-ProhibitTimer-v1700` are: `192`, `256`, `320`, `384`, `448`, `512`, `576`, `640` or `1082` **ms**.

  If **both** `sr-ProhibitTimer` and `sr-ProhibitTimer-v1700` are **absent**, the **timer value** is `0`.

- `sr-TransMax` **ENUMERATED**: mandatory, dedicated

  It is the **maximum quantity** of **retransmissions** (see the previous bullet) of a **scheduling request** (refer to TS 38.321). Do **not confuse** this **maximum quantity** of **retransmissions** with the **quantity** of **PUCCH transmission repetitions** of the **inter-slot repeat option** (click 7.6.5.3.5). Its possible values are: `4`, `8`, `16`, `32` or `64`.

For more detail on these parameters, refer to the table "*SchedulingRequestToAddMod field descriptions*" of TS 38.331 and to clause 9.2.1 of TS 38.213.

### 7.9.15.2 `SchedulingRequestToAddMod` ASN.1 specification

Its complete **ASN.1** (click 5.9) specification is:

```
SchedulingRequestToAddMod ::=  SEQUENCE {
  schedulingRequestId SchedulingRequestId,
  sr-ProhibitTimer ENUMERATED {ms1, ms2, ms4, ms8, ms16,
                               ms32, ms64, ms128} OPTIONAL,        -- Need S
  sr-TransMax ENUMERATED {n4, n8, n16, n32, n64, spare3, spare2, spare1}
}
```

## 7.9.16 `DMRS-BundlingPUCCH-Config-r17` data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 7.9.2.

Its **only parent data type** is `PUCCH-Config` (click 7.9.4).

**All** its parameters are **dedicated** (click 5.1.3).

**Each** of its **instances** configures **DM-RS bundling** for **PUCCH**, for the corresponding **UE**'s **Uplink bandwidth part**.

### 7.9.16.1 `DMRS-BundlingPUCCH-Config-r17` parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `pucch-DMRS-Bundling-r17` **ENUMERATED** `{enabled}`: optional, dedicated

  If **present**, it indicates that **DM-RS bundling** and **time domain window** for **PUCCH** are **jointly enabled**. For more detail, click 7.6.6.6.

- `pucch-FrequencyHoppingInterval-r17 ENUMERATED {s2, s4, s5, s10}`: optional, dedicated

  It indicates the **quantity of slots** "$N_{PUCCH}^{interval}$" that constitute a **slot interval** for the effects of **inter-slot frequency hopping** over **slot intervals**. For more detail, click 7.6.5.3.7 and 7.6.6.6.

For more detail on these parameters, refer to the table "*DM-RS-BundlingPUCCH-Config field descriptions*" of TS 38.331.

### 7.9.16.2 `DMRS-BundlingPUCCH-Config-r17` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
DMRS-BundlingPUCCH-Config-r17 ::= SEQUENCE {
  pucch-DMRS-Bundling-r17 ENUMERATED {enabled} OPTIONAL,          -- Need R
  pucch-TimeDomainWindowLength-r17 INTEGER (2..8) OPTIONAL,       -- Need S
  pucch-WindowRestart-r17 ENUMERATED {enabled} OPTIONAL,          -- Need R
  pucch-FrequencyHoppingInterval-r17
                      ENUMERATED {s2, s4, s5, s10} OPTIONAL,      -- Need S
  ...
}
```

## 7.10 PUSCH overview

The **Physical Uplink Shared Channel** (**PUSCH**) is used by **each UE** to send **user** data and **higher-layer control** data to the **NR node**.

The most relevant **functions** of **PUSCH** are:

- Send **unicast Uplink user** data to the **NR node**.
- Send **unicast Uplink higher-layer control** data to the **NR node**.

**Two transmission schemes** are supported for **PUSCH**: **codebook-based transmission** and **non codebook-based transmission**. The **transmission scheme** is determined by `txConfig` in `pusch-Config`, **either** set to `codebook` or set to `nonCodebook`. If `txConfig` is **absent**, the **UE only** expects to be **scheduled** with **DCI Format 0_0 messages**. For [[**FR2 bands**] and [the **UE** being in the **RRC_CONNECTED** (click 5.8.2) state] and [in a **bandwidth part** where **none** of the **PUCCH resources** (click 7.6.5) have been configured with an **instance** of `PUCCH-SpatialRelationInfo`]] the **UE only** expects to be **scheduled** with a **DCI Format 0_0 message** when `enableDefaultBeamPLForPUSCH0-0` is set to `enabled`. If **PUSCH** is **scheduled** with a **DCI Format 0_0 message**, the **PUSCH transmission** is based on a **single antenna port**. For more detail on these **2 PUSCH transmission schemes**, refer to clause 6.1.1 of TS 38.214.

**Each PUSCH transmission** takes place in **1** of the **transmission occasions determined** by the **PUSCH scheduling** (click 7.11). **Each slot** can contain **0**, **1** or **several PUSCH transmissions** (click 7.12).

**Each PUSCH transmission** contains **only 1 PUSCH unicast codeword** (click 4.12.1.2) in **1** to **4 layers**.

**Each UL-SCH transport block** (click 4.12.1.1) and **each PUSCH codeword** is subject to a **specific processing** (click 4.12) for its **transmission** and for its **reception** through **PUSCH**. This processing may include the **multiplexing** (click 4.12.8) of **1 PUCCH codeword** with **1 PUSCH codeword**, to be **jointly** sent in **1 PUSCH transmission**.

**PUSCH** is **configured** (click 7.15) with several **parameters** and **data types**.

The **PUSCH description** in this book covers almost nothing about **PUSCH** for **shared spectrum**.

For more detail on **PUSCH**, refer to:

- Clauses 4, 5 and 6.3.1 (**Phys**. and **modulation**) of TS 38.211.
- Clauses 4.1, 5 and 6.2 (**mux**. and **coding**) of TS 38.212.
- Clause 6 (**procedures**) of TS 38.214.
- Clauses 5.3.1 and 5.3.2 (**overall Tx**) and 10.3 (**scheduling**) of TS 38.300.

## 7.11 PUSCH scheduling

**PUSCH Tx scheduling**

**Each UE decides**, for **each** of its **PUSCH transmissions**, its **frequency-domain resources** (click 7.11.3), its **time-domain resources** (click 7.11.4), its **space-domain resources** and its **transmission characteristics**. **Each UE must** select the values of the said **elements** within a **set** of **Tx resources** and **characteristics** that the **NR node** has **previously configured** to the **UE**.

The said **set** of **Tx resources** and **characteristics** is **previously configured** to **each UE either** with **PUSCH Tx semi-persistent scheduling** (click 7.11.16.14.1) or with **PUSCH Tx dynamic scheduling** (click 7.11.2).

Within its **scheduled PUSCH PREs**, **each UE** will **only transmit data** in the **payload PUSCH PREs** because a **few** of the **scheduled PUSCH PREs** will be used for **physical signals** (click 4.11.3.2) **instead** of for **data**.

**PUSCH Rx scheduling**

The **NR node must listen** on **all** the **PUSCH transmission occasions** that the **NR node** has **scheduled** to **each** and **every UE**, **either** as **Tx configured grants** or as **Tx assignments**.

Within its **scheduled Tx PUSCH PREs**, **each UE** will **only transmit data** in the **payload PUSCH PREs** because a **few** of the **scheduled PUSCH PREs** will be used for **physical signals** (click 4.11.3.2) **instead** of for **data**.

For an **overall** description of **scheduling**, **transmission occasions** and **transmissions**, click 4.11.4.

For more detail on **PUSCH scheduling**, refer to clause 10.3 of TS 38.300.

### 7.11.1 PUSCH Tx semi-persistent scheduling

**Semi-persistent scheduling** (click 4.11.4.3) provides **each UE** with a **PUSCH Tx configured grant**, which is a **set** of **periodical transmission occasions**. **Configured grants** are provided to the **UE** using the **RRC** (click 5) **protocol**.

**PUSCH configured grants** are **only** for **unicast**, and there are **2 types**:

- **PUSCH CG-Type 1**

  It is **automatically activated right after** it has been **configured** with **RRC.**

It is **deactivated** with a **DCI message** (click 6.9) of **Format 0_0**, **0_1** or **0_2** with the **last 16 bits** of its **CRC masked** with **CS-RNTI** (click 6.9.10.2).

- **PUSCH CG-Type 2**

  It is **activated** or **deactivated** with a **DCI message** of **Format 0_0**, **0_1** or **0_2** with the **last 16 bits** of its **CRC masked** with **CS-RNTI** (click 6.9.10.2).

**PUSCH semi-persistent scheduling** (i.e., **configured grants**) can be configured with `configuredGrantConfig` and with `configuredGrantConfigToAddModList-r16`, **both** of them from `BWP-UplinkDedicated` (click 2.2.8.10). **PUSCH semi-persistent scheduling** (i.e., a **configured grants**) for **small data transmission** (**SDT**, click 10.9) can be configured with `configuredGrantConfigToAddModList-r17` from `BWP-UplinkDedicatedSDT-r17`. The **data type** of the **first** parameter is `ConfiguredGrant` and the one of the **second** and **third** parameters is a `SEQUENCE OF ConfiguredGrant`.

**PUSCH configured grants** can be **released** with `configuredGrantConfigToReleaseList-r16`. **PUSCH configured grants** of **CG-Type 2** can be **deactivated** with `configuredGrantConfigType2DeactivationStateList-r16`. **Both** parameters are also from `BWP-UplinkDedicated`. **PUSCH configured grants** for **small data transmissions** can be **released** with `configuredGrantConfigToReleaseList-r17` from `BWP-UplinkDedicatedSDT-r17`. The **data type** these **3** parameters is a is a `SEQUENCE OF ConfiguredGrantConfigIndex-r16`.

**Release 17** introduced **PUSCH configured grants** with **repetition**. This consists of **allowing** for **repetitions** of **PUSCH transmissions** scheduled with **configured grants,** even **without** waiting for a **HARQ (N)ACK**. **Transmission repetitions enhance reliability** and **reduce latency**. This is especially useful for **ultra-reliable low-latency communications** (**URLLC**) **services** and **coverage-limited scenarios** (e.g., for **mMTC services** or for **RedCap UEs**). This is particularly valuable in **time-division duplexing** (**TDD** ) where **PUSCH transmission opportunities** are limited or for power-constrained devices.

**PUSCH configured grants** are typically used in **time-critical services** (such as **URLLC**), for the **scheduling** of **HARQ transmissions** during **initial access** (click 8).

When **carrier aggregation** (**CA**, click 10.2) is configured, **at most 1 PUSCH configured grant** can be signaled for **each UE** in **each** of its **cells**.

When **bandwidth adaptation** (**BA**, click 10.1) is configured, **at most 1 PUSCH configured grant** can be signaled for **each** of the **UE**'s **Uplink** (or **SUL**) **bandwidth parts** in **each** of its **cells**.

On **each** of the **UE**'s **cells**, there can be **at most 1 active configured PUSCH grant** at any given time. A **configured PUSCH grant** for **each** of the **UE**'s **cells** can **either** be of **CG-Type 1** or **CG-Type 2**. For **CG-Type 2**, the **activation** and **deactivation** of **configured PUSCH grants** are **independent** among the **UE**'s **cells**.

When **SUL** is configured, a **configured Uplink grant** can **only** be signaled for **either Uplink** or **SUL**.

For an **overall** description of **scheduling**, **transmission occasions** and **transmissions**, click 4.11.4.

For more detail on **PUSCH semi-persistent scheduling**, refer to clause 10.3 of TS 38.300 or to "*Configured Grant for Ultra-Reliable and Low-Latency Communications: Standardization and Beyond*", by M. Gerami and B. Singh, IEEE Communications Standards, Volume 6, Issue, 4, December 2022.

## 7.11.2 PUSCH Tx dynamic scheduling

**PUSCH scheduling requests**

Whenever a **UE** has **PUSCH data** to **transmit** and it **lacks** the required **PUSCH resources**, it **requests them** to the **NR node**. This **request** can be done in the two following ways:

- An **UCI message** (click 7.7) containing a **scheduling request** (**SR**, click 7.7.1.2).
- A **MAC PDU** (click 4.5.5.1) containing a "**Buffer Status Report**" **MAC Control Element** (click 4.5.5.3).

**PUSCH Tx dynamic scheduling**

As the result of a **scheduling request** by a **UE**, **PUSCH Tx dynamic scheduling** provides the requesting **UE** with a **one-shot set** of **1** to **8 transmission assignments**. **Each** such **transmission assignment** allows the **UE** for **1 PUSCH transmission occasion**.

**Each** such **one-shot set** of **transmission assignments** is provided to the **UE either** with **1** (**DCI**) **message** (click 6.9) **or** with a (**normal** or **fallback**) **random access response** (click 9.3.3 and 9.3.7) from a **random access procedure** (click 9).

**Each** such **DCI message**:

- Has **Formats 0_x** .
- **Only** signals **PUSCH unicast transmission assignments**.
- Signals from **1** to **8 PUSCH transmission assignments** to the recipient **UE**.
- Has the **last 16 bits** of its **CRC masked** with one of the **individual RNTIs** that the destination **UE** can manage.
- **Each resource type** is **assigned** with a **specific DCI field**: **a)** the **frequency resources** with "***Frequency domain resource assignment***" (click 7.11.3); **b)** the **time resources** with "***Time domain resource assignment***" (click 7.11.4); and **c)** the **space resources** with "***Antenna ports***".

For an **overall** description of **scheduling**, **transmission occasions** and **transmissions**, click 4.11.4.

For more detail on **DCI message fields**, click 6.9.1.

For more detail on **DCI messages**, click 6.9 and refer to clause 7.3 of TS 38.212.

For more detail on **PUSCH scheduling**, refer to clause 10.3 of TS 38.300.

## 7.11.3 PUSCH frequency-domain resources

The **dynamic scheduling** of **PUSCH frequency-domain resources** is encoded in the **DCI message field** "***Frequency domain resource assignment***", except for a **PUSCH transmission** scheduled **either** by a (**normal** or **fallback**) **random access response**

(click 9.3.3 and 9.3.7) or by a **MsgA** (click 9.3.6). For these **2 exceptions**, refer (respectively) to clause 8.3 of TS 38.213 and to clause 8.1A of TS 38.213.

The **PUSCH frequency resource assignment** is the **same** for **all** the **1** to **8 PUSCH assignments** that may be done with **1 DCI message**.

The following table shows, for **PUSCH assignments**, the **DCI message Formats** where the said **field** is **present**, what is **assigned** by **each** of these **DCI Formats** and what is the **field**'s **bitlength** in **each** of these **DCI Formats**:

| **DCI message field "*Frequency domain resource assignment*"** | | |
|---|---|---|
| **Assigns** | **DCI Format** | **Bitlength** |
| **PUSCH Unicast** | Format 0_0 (6.9.2) | Variable |
| | Format 0_1 (6.9.3) | Variable |
| | Format 0_2 (6.9.4) | Variable |

The "***Frequency domain resource assignment***" contains **1 optional bit** and a **subfield**.

This **1 optional bit** is **only** present when `dynamicSwitch` (click 7.11.3.1) has been configured. When this **bit** is **present**, its value **0** indicates **UFA-Type 0** (see below) and is value **1** indicate **UFA-Type 1**. When this **bit** is **present**, it is the **most significant bit** of the "***Frequency domain resource assignment***" field.

The **bitlength** and **interpretation** of the **subfield** depends on the **DCI message Format**, on the **search space** and on the **type** of **frequency-domain resource allocation**.

There are **3 types** of **Uplink frequency-domain resource allocation**, which are **UFA-Type 0**, **UFA-Type 1** and **UFA-Type 2**:

- **UFA-Type 0** (**bitmap based**, click 7.11.3.2)

  The **subfield** within the "***Frequency domain resource assignment***" is a **bitmap** that indicates a **set** of individual **Resource Block Groups** (**RBGs**, click 2.2.9).

  **UFA-Type 0** is **only supported** when **transform precoding** (click 4.12.12) is **disabled**.

- **UFA-Type 1** (**start-and-length**, click 7.11.3.3)

  The **subfield** within the "***Frequency domain resource assignment***" **is** a **resource indicator value** (**RIV**) that identifies a **block** of **contiguous Physical Resource Blocks** (**PRBs**).

- **UFA-Type 2** (**interlace based**, click 7.11.3.4)

  The **DCI message field** "***Frequency domain resource assignment***" indicates the **PUSCH interlace**.

For the **selection** of the **UFA-Type**, click 7.11.3.1.

**Resource Block indexing**

The **Resource Block indexing** used in **PUSCH frequency-domain resource allocation** corresponds to **either** of the following cases:

- **RB indexing from the currently active Uplink bandwidth part**

  This case is when **either** [the **scheduling DCI message** does **not** include the "***Bandwidth part indicator***" field] **or** [the **UE** does **not support active bandwidth part change** via **DCI messages**].

- **RB indexing from the new active Uplink bandwidth part**

  This case is when [the **scheduling DCI message includes** the "***Bandwidth part indicator***" field] **and** [the **UE** does **supports active bandwidth part change** via **DCI messages**]. The **new active Uplink bandwidth part** is the one indicated in the "***Bandwidth part indicator***" field.

**Regardless** of the case, **RB indexing** starts from the **lowest frequency RB**.

The following table shows, for **PUSCH assignments**, the **DCI message Formats** where the said **field** is **present**, what is **assigned** by **each** of these **DCI Formats** and what is the **field**'s **bitlength** in **each** of these **DCI Formats**:

| **DCI message field "*Bandwidth part indicator*"** | | |
|---|---|---|
| **Assigns** | **DCI Format** | **Bitlength** |
| **PUSCH Unicast** | Format 0_1 (6.9.3) | 0 to 2 |
| | Format 0_2 (6.9.46.9.7) | 0 to 2 |

The **UE** shall upon detection of **PDCCH** intended for the **UE** determine **first** the corresponding **Uplink bandwidth part** and **then** the **resource allocation** within the **bandwidth part**.

For more detail on **DCI message fields**, click 6.9.1.

For more detail on **DCI messages**, click 6.9 and refer to clause 7.3 of TS 38.212.

For more detail on **Uplink frequency resource allocation**, refer to clause 6.1.2.2 of TS 38.214.

### 7.11.3.1 Selection of the type of PUSCH frequency-domain resource allocation

The **UE** selects the **type** of **PUSCH frequency-domain resource allocation** (i.e., the **UFA-Type**) according to the following rules.

**UFA-Type 0 is used in the following cases:**

- When [the **scheduling DCI message** is **Format 0_1**]

  and [`useInterlacePUCCH-PUSCH-r16` from `BWP-UplinkCommon` (click 2.2.8.9) is **not present**]

  and [ [`resourceAllocation` from the `PUSCH-Config` (click 7.15.5) **instance** associated to `pusch-Config` is set to `resourceAllocationType0`]

  or [ [`resourceAllocation` from the `PUSCH-Config` (click 7.15.5) **instance** associated to `pusch-Config` is set to `dynamicSwitch`]

  and [the **most significant bit** of the "***Frequency domain resource***" **field** from the **scheduling DCI message** is **0**]]].

- When [the **scheduling DCI message** is **Format 0_2**]

  and [ [`resourceAllocationDCI-0-2-r16` from the `PUSCH-Config` (click 7.15.5) **instance** associated to `pusch-Config` is set to `resourceAllocationType0`]

  or [ [`resourceAllocationDCI-0-2-r16` from the `PUSCH-Config` (click 7.15.5) **instance** associated to `pusch-Config` is set to `dynamicSwitch`]

  and [the **most significant bit** of the "***Frequency domain resource***" **field** from the **scheduling DCI message** is **0**]]].

Recall that **UFA-Type 0** is **only** supported when **transform precoding** (click 4.12.12) is **disabled**.

**UFA-Type 1 is used in the following cases:**

- When [the **scheduling DCI message** is **Format 0_0**]

  and [`useInterlacePUCCH-PUSCH-r16` from `BWP-UplinkCommon` (click 2.2.8.9) **and** from `BWP-UplinkDedicated` (click 2.2.8.102.2.8.9) are **not present**]

- When [the **scheduling DCI message** is **Format 0_1**]

  and [`useInterlacePUCCH-PUSCH-r16` from `BWP-UplinkCommon` (click 2.2.8.9) is **not present**]

  and [ [`resourceAllocation` from the `PUSCH-Config` (click 7.15.5) **instance** associated to `pusch-Config` is set to `resourceAllocationType1`]

  or [ [`resourceAllocation` from the `PUSCH-Config` (click 7.15.5) **instance** associated to `pusch-Config` is set to `dynamicSwitch`]

  and [the **most significant bit** of the "***Frequency domain resource***" **field** from the **scheduling DCI message** is **1**]]].

- When [the **scheduling DCI message** is **Format 0_2**]

  and [ [`resourceAllocationDCI-0-2-r16` from the `PUSCH-Config` (click 7.15.5) **instance** associated to `pusch-Config` is set to `resourceAllocationType1`]

  or [ [`resourceAllocationDCI-0-2-r16` from the `PUSCH-Config` (click 7.15.5) **instance** associated to `pusch-Config` is set to `dynamicSwitch`]

  and [the **most significant bit** of the "***Frequency domain resource***" **field** from the **scheduling DCI message** is **1**]]].

**UFA-Type 2 is used in the following cases:**

- When [the **DCI scheduling message** is **Format 0_0**]

  and [`useInterlacePUCCH-PUSCH-r16` **either** from `BWP-UplinkCommon` (click 2.2.8.9) or from `BWP-UplinkDedicated` (click 2.2.8.102.2.8.9) is **present**].

- When [the **DCI scheduling message** is **Format 0_1**]

  and [`useInterlacePUCCH-PUSCH-r16` from `BWP-UplinkDedicated` (click 2.2.8.10) is **present**].

**Recall** that **CRB interlaces** (click 2.2.4.7) are **only** defined for **numerologies** "**μ**=0" and "**μ**=1".

Summarizing the above description

- **Each PUSCH transmission** that is **scheduled** with a **DCI message** with **Format 0_1** or **0_2** uses the **UFA-Type** that is **set** in the **parameter** "`resourceAllocation...`" that corresponds to the **DCI Format**. ***If*** this corresponding parameter is set to `dynamicSwitch`, ***then***, the **PDSCH reception** uses the **DFA-Type** that is indicated in the "***Frequency domain resource***" **field**.

  As an **exception**, ***if*** [the scheduling **DCI message** is **Format 0_1**] and [`useInterlacePUCCH-PUSCH-r16` from `BWP-UplinkDedicated` (click 2.2.8.10) is **present**], ***then***, **UFA-Type 2** is used.

- **Each PUSCH transmission** that is **scheduled** with a **DCI message** with **Format 0_0** uses **DFA-Type 1**, **except** if `useInterlacePUCCH-PUSCH-r16` **either** from `BWP-UplinkCommon` (click 2.2.8.9) or from `BWP-UplinkDedicated` (click 2.2.8.102.2.8.9) is **present**, in which case **DFA-Type 2** is **used**.

It is **assumed** that when `pusch-Config` is set to `dynamicSwitch` the **UE supports dynamic switch**.

The **UE** expects that `useInterlacePUCCH-PUSCH-r16` is **present either** from **none** or from **both** `BWP-UplinkCommon` (click 2.2.8.9) and `BWP-UplinkDedicated` (click 2.2.8.10).

For more detail on **DCI message fields**, click 6.9.1.

For more detail on **DCI messages**, click 6.9 and refer to clause 7.3 of TS 38.212.

For more detail on **Uplink frequency resource allocation**, refer to clause 6.1.2.2 of TS 38.214.

### 7.11.3.2 PUSCH UFA-Type 0 frequency-domain resource allocation

**UFA-Type 0** is **bitmap-based**. In **UFA-Type 0**, the **subfield** (click 7.11.3) within the **DCI message field** "***Frequency domain resource assignment***" is a **bitmap**. This **bitmap** indicates which **Resource Block Groups** (**RBGs**, click 2.2.9) are **assigned** to the **UE** for the **1** to **8 PUSCH transmission**(**s**). The **bitmap** contains **1 bit** for **each** and **every RBG** in the **bandwidth part**, from **RBG0** (**lowest frequency**) to **RBGn** (**highest frequency**). **RBG0** is mapped to the **most significant bit** of the **bitmap**. If a given bit value is **1**, it implies that the corresponding **RBG** is **assigned** to the **UE**.

**Different UEs** can **share** the **same RBGs** as long as their **PUSCH time-domain resources** are different.

The **RBs** in the **RBGs** are **physical resource blocks** (**PRBs**), because in **PUSCH** there is **no interleaved VRB-to-PRB mapping**, which implies that the **VRBs** are **the same** as **PRBs**.

**UFA-Type 0** is **supported only** when **transform precoding** (click 4.12.12) is **disabled**.

For more detail on **DCI message fields**, click 6.9.1.

For more detail on **DCI messages**, click 6.9 and refer to clause 7.3 of TS 38.212.

For more detail on **PUSCH frequency resource allocation UFA-Type 0**, refer to clause 6.1.2.2.1 of TS 38.214.

### 7.11.3.3 PUSCH UFA-Type 1 frequency-domain resource allocation

**UFA-Type 1** is **start-and-length based**.

In **UFA-Type 1**, the **subfield** (click 7.11.3) within the **DCI field** "***Frequency domain resource assignment***" is a **resource indicator value** (**RIV**). This **RIV** indicates the **lowest frequency PRB** "$RB_{start}$", and the **quantity** "$L_{RBs}$" of **contiguous PRBs** that are **assigned** to the **UE**.

The **RIV** value is defined by:

***If*** $(L_{RBs} - 1) \leq floor(N_{RB}^{BWP} / 2)$

***then*** $\text{RIV} = N_{RB}^{BWP} (L_{RBs} - 1) + RB_{start}$

***else*** $\text{RIV} = N_{RB}^{BWP} (N_{RB}^{BWP} - L_{RBs} + 1) + (N_{RB}^{BWP} - 1 - RB_{start})$

where "$1 \leq L_{RBs} \leq (N_{RB}^{BWP} - RB_{start})$" and "$N_{RB}^{BWP}$" is the **quantity** of **resource blocks** of the **corresponding** (click 7.11.3) **bandwidth part**.

The **bitlength** of the **RIV** is the **smallest one** that can contain the **required quantity** of **values**.

**Case 1: derived DCI Format 1_0**

***If*** the **DCI size** for a **scheduling DCI message Format 1_0** received in a **UE-specific search space** (**USS**) is **derived** from the **size** of a **scheduling message DCI Format 1_0** received in a **common search space** (**CSS**) but it is **applied** to an **active bandwidth part with RB size** "$N_{RB}^{BWP.active}$", ***then***, the **UFA-Type 1** field consists of a **resource indicator value** (**RIV**) corresponding to a **starting resource block** "$RB_{start}$":

$RB_{start} = 0, K, 2K, \ldots, (N_{RB}^{BWP,initial} - 1) \bullet K$

and a **length** "$L_{RBs}$" expressed in **contiguous physical resource blocks** (**PRBs**):

$L_{RBs} = K, 2K, \ldots, N_{RB}^{BWP,initial} \bullet K$

This **RIV** value is defined by:

***If*** $(L'_{RBs} - 1) \leq floor(N_{RB}^{BWP,initial} / 2)$

***then*** $\mathbf{RIV} = N_{RB}^{BWP,initial} (L'_{RBs} - 1) + RB'_{start}$

***else*** $\mathbf{RIV} = N_{RB}^{BWP,initial} (N_{RB}^{BWP,initial} - L'_{RBs} + 1) + (N_{RB}^{BWP,initial} - 1 - RB'_{start})$

where "$L'_{RBs} = L_{RBs} / K$", "$RB'_{start} = RB_{start} / K$", "$L'_{RBs} \leq (N_{RB}^{BWP,initial} - RB'_{start})$" and the value of "$K$" is given by:

***If*** $N_{RB}^{BWP,active} \leq N_{RB}^{BWP,initial}$

***then*** $K = 1$

***else*** $K = max(1, 2, 4, 8)$ such that "$K \leq floor(N_{RB}^{BWP,active} / N_{RB}^{BWP,initial})$"

**Case 2: DCI Format 0_2 in RGBs**

***If*** the **scheduling DCI message** is **Format 0_2**, ***then***, the **UFA-Type 1** field consists of a **resource indicator value** (**RIV**) corresponding to a **starting resource block group** (**RBG**, click 2.2.9) "$RBG_{start}$":

$RBG_{start} = 0, 1, \ldots, (N_{RBG}^{BWP} - 1)$

and a **length** "$L_{RBGs}$" expressed in **contiguous resource block groups** (**RBGs**):

$L_{RBGs} = 0, 1, \ldots, N_{RBG}^{BWP}$

where the **RBGs** are defined as described in 2.2.9, with the **only difference** that in this case "$P$" takes the value of `resourceAllocationType1GranularityDCI-0-2-r16` (from `PUSCH-Config`, click 7.15.5) if **present**, and if **absent** "$P$=1".

This **RIV** value is defined by:

***If*** $(L_{RBGs} - 1) \leq floor(N_{RBG}^{BWP} / 2)$

***then*** $\mathbf{RIV} = N_{RBG}^{BWP} (L_{RBGs} - 1) + RBG_{start}$

***else*** $\mathbf{RIV} = N_{RBG}^{BWP} (N_{RBG}^{BWP} - L_{RBGs} + 1) + (N_{RBG}^{BWP} - 1 - RBG_{start})$

where "$1 \leq L_{RBGs} \leq (N_{RBG}^{BWP} - RBG_{start})$".

For more detail on **DCI message fields**, click 6.9.1.

For more detail on **DCI messages**, click 6.9 and refer to clause 7.3 of TS 38.212.

For more detail on **PUSCH frequency resource allocation UFA-Type 1**, refer to clause 6.1.2.2.2 of TS 38.214.

### 7.11.3.4 PUSCH UFA-Type 2 frequency-domain resource allocation

**UFA-Type 2** is **interlace based**.

In **UFA-Type 2**, the **subfield** (click 7.11.3) within the **DCI field** "***Frequency domain resource assignment***" determines which **interlace(s)** (click 2.2.4.7) are assigned to the **PUSCH transmission**.

**PUCCH-PUSCH interlace** (click 2.2.4.8) is **either set** or **not set** in **all Uplink** (or **SUL**) **bandwidth parts** of the **same serving cell**. Consequently, the **same PUCCH-PUSCH interlace configuration** applies to **all** the **PUCCH transmissions** (i.e., to **all** the **PUCCH resources**, click 7.6.5.3) and to **all** the **PUSCH transmissions** (click 7.11.3) in **each serving cell**.

Regarding **PUSCH**, this implies that **frequency-domain resource allocation** with **UFA-Type 2** is used for **cell-specific PUSCH**, e.g., **PUSCH** scheduled by **RAR Uplink grant** (refer to clause 8.3 of TS 38.213 and to clause 6.1.2.2 of TS 38.214) .

**Recall** that **CRB interlaces** (click 2.2.4.7) are **only** defined for **numerologies** "**μ**=0" and "**μ**=1".

For more detail on **DCI message fields**, click 6.9.1.

For more detail on **DCI messages**, click 6.9 and refer to clause 7.3 of TS 38.212.

For more detail on **PUSCH frequency resource allocation UFA-Type 2**, refer to clause 6.1.2.2.3 of TS 38.214.

### 7.11.4 PUSCH time-domain resources

The **dynamic scheduling** of **PUSCH frequency-domain resources** is encoded in **either** of the following **2 fields**:

- The **field** "***Time domain resource assignment***" of a **DCI message Format 0_0**, **0_1** or **0_2**.
- The **4-bit field** "**PUSCH time resource allocation**" (click 4.5.5.7.3) of a (**normal** or **fallback**) **random access response** (click 9.3.3 and 9.3.7) from a **random access procedure** (click 9).

**Adding 1** to the **value** "***m***" contained in **either** of the **said fields** results in an **index**. This **index** "***m***+1" **identifies 1 entry** from a **PUSCH time-domain resource allocation list**. The **list** to use is **1 among several ones**.

The **UE first** determines **which PUSCH time-domain resource allocation list** to **use**, as described in 7.11.4.1.

The **UE then** extracts the corresponding **entry** from the **list** and **applies** it, as described in 7.11.4.26.14.4.2. The resulting **list entry** contains from **1** to **8 PUSCH time-domain resources**. **Each PUSCH time-domain resource** corresponds to **1** of the **1** to **8 PUSCH assignments** that can be performed with the **scheduling DCI** or **RAR message**. **Each PUSCH time-domain resource** indicates the **time** between the **reception** of the **scheduling message** and the **actual transmission** of the corresponding **PUSCH transmission**. The **list entry** contains **other** configuration **information**, such as the **PUSCH DM-RS mapping type** and it can **also** contain **other** configuration information, such as for **PUSCH repetition**.

The following table shows, for **PUSCH scheduling**, the **DCI message Formats** where the said **field** is **present**, what is **assigned** by **each** of these **DCI Formats** and what is the **field**'s **bitlength** in **each** of these **DCI Formats**:

| **DCI message field "*Time domain resource assignment*"** | | |
|---|---|---|
| **Assigns** | **DCI Format** | **Bitlength** |
| **PUSCH Unicast** | Format 0_0 (6.9.2) | 4 |
| | Format 0_1 (6.9.3) | 0 to 6 |
| | Format 0_2 (6.9.4) | 0 to 6 |

The **field**'s **bitlength** in **each case** is sufficient to **index all** the **existing entries** in the corresponding **PUSCH time-domain resource allocation list**.

**Each DCI Format 0_1 message** can **schedule** from **1** to **8 PUSCH transmissions** (i.e., from **1** to **8 PUSCH assignments**).

For more detail on **DCI messages**, click 6.9 and refer to clause 7.3 of TS 38.212.

For more detail on **PUSCH resource allocation** in the **time domain**, refer to clause 6.1.2.1 of TS 38.214.

### 7.11.4.1 PUSCH time-domain resource allocation lists

For **each incoming DCI Format 0_x message** or **RAR message**, the **UE** selects **1 PUSCH time-domain resource allocation list** among the following ones:

- **Default A** for **normal cyclic prefix** (table 6.1.2.1.1-2 of TS 38.214). The **list** in this table has **16 entries**.
- **Default A** for **extended cyclic prefix** (table 6.1.2.1.1-3 of TS 38.214). The **list** in this table has **16 entries**.
- The **list** contained in `pusch-TimeDomainAllocationList` from `PUSCH-ConfigCommon` (click 7.15.4). This **list** can have **up to** `16` **entries**.
- The **list** contained in `pusch-TimeDomainAllocationList` from `PUSCH-Config` (click 7.15.5). This **list** can have **up to** `16` **entries**.
- The **list** contained in `pusch-TimeDomainAllocationListDCI-0-1-r16`, also from `PUSCH-Config`. This **list** can have **up to** `64` **multi-entries**, where **each multi-entry** corresponds to a **multiple PUSCH assignment** that contains `1` to `8` **entries**.
- The **list** contained in `pusch-TimeDomainAllocationListDCI-0-2-r16` also from `PUSCH-Config`. This **list** can have **up to** `64` **multi-entries**, where **each multi-entry** corresponds to a **multiple PUSCH assignment** that contains `1` to `8` **entries**.
- The **list** contained in `pusch-TimeDomainAllocationListForMultiPUSCH-r16` also from `PUSCH-Config`. This **list** can have **up to** `64` **multi-entries**, where **each multi-entry** corresponds to a **multiple PUSCH assignment** that contains `1` to `8` **entries**.

The **first** and **second parameters** above have **data type** `PUSCH-TimeDomainResourceAllocationList` (see within 7.15.4.1 and 7.15.5.1), which is a `SEQUENCE OF 1 to 16` **instances** of `PUSCH-TimeDomainResourceAllocation` (click 7.15.6). The **last 3** parameters have **data type** `PUSCH-TimeDomainResourceAllocationList-r16` (see within 7.15.5.1), which is a `SEQUENCE OF 1 to 64` **instances** of `PUSCH-TimeDomainResourceAllocation-r16` (see within 7.15.5.1). This last **data type** is a `SEQUENCE OF 1 to 8` **instances** of `PUSCH-Allocation-r16` (click 7.15.7).

The specific **PUSCH time-domain resource allocation list** to use is determined by **either** of the following **3 tables** of TS 38.214. Which **table** to use among **these 3** depends

on the **search space type**, on the **DCI Format** of the **scheduling message** and on the **masking RNTI**, as indicated below:

For a **common search space** or for a **DCI Format 0_0 message** in a **UE-specific search space**, the **PUSCH time-domain resource allocation list** to use is determined by table 6.1.2.1.1-1:

| RNTI | PDCCH search space | pusch-TimeDomainAllocationList from PUSCH-ConfigCommon (7.15.4) | pusch-TimeDomainAllocationList or pusch-TimeDomainAllocationList-r16 from PUSCH-Config (7.15.5) | PUSCH time-domain resource allocation list to be used |
|---|---|---|---|---|
| **PUSCH scheduled by MAC RAR as described in clause 8.2 of TS 38.213 or MAC fallbackRAR as described in clause 8.2A of 38.213 or for MsgA PUSCH transmission** | | **Absent** | **-** | Default A |
| | | **Present** | **-** | pusch-TimeDomainAllocationList from PUSCH-ConfigCommon |
| **C-RNTI, MCS-C-RNTI, TC-RNTI, CS-RNTI** | **Any CSS associated with CORESET#0** | **Absent** | **-** | Default A |
| | | **Present** | **-** | pusch-TimeDomainAllocationList from PUSCH-ConfigCommon |
| **C-RNTI, MCS-C-RNTI, TC-RNTI, CS-RNTI** | **Any CSS not associated with CORESET#0 or DCI Format 0_0 in USS** | **Absent** | **Absent** | Default A |
| | | **Present** | **Absent** | pusch-TimeDomainAllocationList from PUSCH-ConfigCommon |
| | | **Present or Absent** | **Present** | pusch-TimeDomainAllocationList or pusch-TimeDomainAllocationList-r16 from PUSCH-Config |

For a **DCI Format 0_1 message** in a **UE-specific search space**, with the **last 16 bits** of its **CRC masked** with **either** of **C-RNTI**, **MCS-C-RNTI**, **CS-RNTI** or **SP-CSI-RNTI**,

the **PUSCH time-domain resource allocation list** to use is determined by table 6.1.2.1.1-1A:

| pusch-TimeDomainAllocationList from PUSCH-ConfigCommon (7.15.4) | pusch-TimeDomainAllocationList or pusch-TimeDomainAllocationList-r16 from PUSCH-Config (7.15.5) | pusch-TimeDomainAllocationList DCI-0-1 from PUSCH-Config (7.15.5) | pusch-TimeDomainAllocationListForMultiPUSCH from PUSCH-Config (7.15.5) | PUSCH time-domain resource allocation list to be used |
|---|---|---|---|---|
| Absent | Absent | Absent | Absent | Default A |
| Present | Absent | Absent | Absent | pusch-TimeDomainAllocationList from PUSCH-ConfigCommon |
| Present or Absent | Present | Absent | Absent | pusch-TimeDomainAllocationList or pusch-TimeDomainAllocationList-r16 from PUSCH-Config |
| Present or Absent | Absent | Present | - | pusch-TimeDomainAllocationListDCI-0-1 from PUSCH-Config |
| Present or Absent | Absent | - | Present | pusch-TimeDomainAllocationListForMultiPUSCH from PUSCH-Config |

For a **DCI Format 0_2 message** in a **UE-specific search space**, with the **last 16 bits** of its **CRC masked** with **either** of **C-RNTI**, **MCS-C-RNTI**, **CS-RNTI** or **SP-CSI-RNTI**, the **PUSCH time-domain resource allocation list** to use is determined by table 6.1.2.1.1-1B:

| pusch-TimeDomainAllocationList from PUSCH-ConfigCommon (7.15.4) | pusch-TimeDomainAllocationList or pusch-TimeDomainAllocationList-r16 from PUSCH-Config (7.15.5) | pusch-TimeDomainAllocationListDCI-0-2 from PUSCH-Config (7.15.5) | PUSCH time-domain resource allocation list to be used |
|---|---|---|---|
| Absent | Absent | Absent | Default A |
| Present | Absent | Absent | pusch-TimeDomainAllocationList from PUSCH-ConfigCommon |
| Present or Absent | Present | Absent | pusch-TimeDomainAllocationList or pusch-TimeDomainAllocationList-r16 from PUSCH-Config |
| Present or Absent | Absent | Present | pusch-TimeDomainAllocationListDCI-0-2 from PUSCH-Config |

### 7.11.4.2 PUSCH time-domain resource entries

The **UE** has already obtained **1 PUSCH time-domain resource allocation list** as it has been detailed in 7.11.4.1.

The **UE** then selects the "***m***+1" **list entry**, where "***m***" is the value from **either [**the field "***Time domain resource assignment***" (click 6.9.1) from the **scheduling DCI message**], or [the **4-bit field** "**PUSCH time resource allocation**" (click 4.5.5.7.3) from the scheduling (**normal** or **fallback**) **random access response** (click 9.3.3 and 9.3.7) from a **random access procedure** (click 9)].

The **selected** "***m***+1" **list entry** provides the **UE** with the **slot offset** "$K_2$" and with the **time-domain resource assignment(s) either** for **1 PUSCH transmission** or for **1** to **8 PUSCH transmissions** (for the case of a **multi-PUSCH assignment**). A **multi- PUSCH assignment** is **only** performed with a **PUSCH time-domain resource allocation list** that is **contained** in an **instance** of the **data type** `PUSCH-TimeDomainResourceAllocationList-r16` (see within 7.15.5.1).

Notice that for the case of **multi-PUSCH assignment**, **all** the **assigned PUSCH transmissions** share a **single value** of the **slot offset** "$K_2$".

**Each PUSCH time-domain resource assignment** (corresponding to **1 PUSCH transmission**) provides the following information.

- **Either** [a **start-and-length indicator** (**SLIV**)]

  or [[the **first symbol interval** "***S***"] and [the **quantity** of **allocated symbol intervals** "***L***"]]

  that identifies the **symbol intervals** of the **PUSCH transmission**.

- The **PUSCH DM-RS mapping type**: `typeA` or `typeB`.

- If `numberOfSlotsTBoMS-r17` from `PUSCH-Allocation-r16` (click 7.15.7) is **present**, the **quantity** of **slots used** for **transport block size** (**TBS**) **determination** (click 6.16.3).

- If `numberOfRepetitions-r16` and/or `numberOfRepetitionsExt-r16` (**both** from `PUSCH-Allocation-r16`, click 7.15.7) is **present**, the **quantity** of **repetitions** (click 7.8.2.3.2) to be **applied** to the **PUSCH transmission**.

**Obtaining the PUSCH transmission slot index "$K_s$"**

The value of the **slot index** "$K_s$" for the **PUSCH transmission** depends on the **scheduling case**, as follows:

- **Transport block scheduling**

  This case is when a **UE**

  [[[is **scheduled** to **transmit** a **transport block**] and [**no CSI report**]]

  [[by a **DCI message**] or [by a (**normal** or **fallback**) **random access response** (click 9.3.3 and 9.3.7)]]

  **or**

  [[is **scheduled** to **transmit** a **transport block** and a **CSI report(s)**]

  [by a **DCI message**]],

In this case, the **slot offset** "$K_2$" is indicated in the **selected entry** from the **selected PUSCH time-domain resource allocation list** (see further below).

The **UE** then determines the **slot index** "$K_s$" for the **PUSCH transmission** by adding the **slot offset** "$K_2$" to the **slot index** where the **scheduling message** was **received**.

- **Aperiodic CSI only scheduling**

  This case is when a **UE**

  [[is **scheduled** to **transmit** an **A-CSI report** (with **no transport block**)]

  [by a **DCI message** that contains a "***CSI request***" **field**]]

  In this case, the **slot offset** "$K_2$" is determined as:

  $K_2 = max_j\ Y_j(m+1)$

  where "$m$+1" is the **index** of the **selected list entry** (see above), "$j \in \{0..N_{Rep}-1\}$" and "$Y_j$" are the various values within the corresponding parameter, as follows:

  - ***If*** [**PUSCH** is scheduled by **DCI Format 0_1**] and [either `reportSlotOffsetListDCI-0-1-r16` or `reportSlotOffsetListDCI-0-1-r17` is **present**],

    ***then***,

    the corresponding parameter is **either** `reportSlotOffsetListDCI-0-1-r16` or `reportSlotOffsetListDCI-0-1-r17`.

  - ***If*** [**PUSCH** is scheduled by **DCI Format 0_2**] and [either `reportSlotOffsetListDCI-0-2-r16` or `reportSlotOffsetListDCI-0-2-r17` is **present**],

    ***then***,

    the corresponding parameter is **either** `reportSlotOffsetListDCI-0-2-r16` or `reportSlotOffsetListDCI-0-2-r17`.

  - ***Otherwise***, the corresponding parameter is `reportSlotOffsetList` or `reportSlotOffsetList-r17`.

  The **UE** then determines the **slot index** "$K_s$" for the **PUSCH transmission** using the **expressions** indicated in clause 6.1.2.1 of TS 38.214.

**Obtaining the slot offset "$K_2$" from a list entry**

For the case of a **PUSCH time-domain resource allocation list** (click 7.11.4.1) contained in a **RRC** parameter, the value of "$K_2$" is explicitly provided **either** by `k2` from `PUSCH-TimeDomainResourceAllocation` (click 7.15.6) or by `extendedK2-r17` from `PUSCH-Allocation-r16` (click 7.15.7).

However, for the case of a **PUSCH time-domain resource allocation list** (click 7.11.4.1) obtained from a **default table**, the value of "$K_2$" is **indicated** as an **expression** of variable

"***j***" (e.g., "***j***", "***j***+1" or "***j***+2"). The value of "***j***" is determined by the **PUSCH numerology**, as follows:

| $\mu_{PUSCH}$ | ***j*** |
|---|---|
| **0** | 1 |
| **1** | 1 |
| **2** | 2 |
| **3** | 3 |
| **5** | 11 |
| **6** | 21 |

As one example of a **PUSCH time-domain resource allocation list** obtained from a **default table**, the following is table **Default A** for **normal cyclic prefix** (which is table 6.1.2.1.1-2 of TS 38.214):

| Row index | PUSCH DM-RS mapping type | $K_2$ | ***S*** | ***L*** |
|---|---|---|---|---|
| **1** | `typeA` | ***j*** | 0 | 14 |
| **2** | `typeA` | ***j*** | 0 | 12 |
| **3** | `typeA` | ***j*** | 0 | 10 |
| **4** | `typeB` | ***j*** | 2 | 10 |
| **5** | `typeB` | ***j*** | 4 | 10 |
| **6** | `typeB` | ***j*** | 4 | 8 |
| **7** | `typeB` | ***j*** | 4 | 6 |
| **8** | `typeA` | ***j***+1 | 0 | 14 |
| **9** | `typeA` | ***j***+1 | 0 | 12 |
| **10** | `typeA` | ***j***+1 | 0 | 10 |
| **11** | `typeA` | ***j***+2 | 0 | 14 |
| **12** | `typeA` | ***j***+2 | 0 | 12 |
| **13** | `typeA` | ***j***+2 | 0 | 10 |
| **14** | `typeB` | ***j*** | 8 | 6 |
| **15** | `typeA` | ***j***+3 | 0 | 14 |
| **16** | `typeA` | ***j***+3 | 0 | 10 |

**Default tables** do **not** perform **multi-PUSCH assignments**. Consequently, **each entry** in this **table** contains **1 PDSCH time-domain resource**.

**PUSCH transmission repetitions**

The **repetitions** of a **PUSCH transmission** depend on the **repetition type**, either `puschRep-TypeA` or `puschRep-TypeB` and on a number of variables and cases. For more detail on this, refer to clause 6.1.2.1 of TS 38.214.

For more detail on **PUSCH time-domain resources**, refer to clause 6.1.2.1 of TS 38.214.

### 7.11.5 PUSCH scheduling of payload PREs, DM-RS PREs, PT-RS PREs, SRS PREs and empty PREs

The **PUSCH scheduled Physical Resource Elements** (**PREs**) defined in the previous subsections can contain **PUSCH payload PREs**, **PUSCH Demodulation Reference Signals** (**DM-RS**) **PREs**, **PUSCH Phase-Tracking Reference Signals** (**PT-RS**) **PREs**, **Sounding Reference Signals** (**SRS**) **PREs** and **PDSCH empty** (no signal) **PREs**.

The **PUSCH payload PREs** are the **PUSCH scheduled PREs not** used for **any** of the other purposes just indicated. In more detail, the **PUSCH payload PREs** are the **PUSCH scheduled PREs** that are **not** used for:

- **PUSCH DM-RS**, nor for **DM-RS** intended for other co-scheduled **UEs** (refer to clause 6.4.1.1.3 of TS 38.211).
- **PUSCH PT-RS** (refer to clause 6.4.1.2.2 of TS 38.211).
- **SRS** (refer to clause 6.4.1.4.3 of TS 38.211).

The **NR node** receives the **PUSCH DM-RS PREs** that every **PUSCH transmission** contains, to support the demodulation of the **PUSCH payload PREs**.

The **scheduling** of **PUSCH DM-RS PREs**, **within** the **PUSCH** scheduled **PREs**, is **independently** specified for the **time domain** (click 7.11.5.1), for the **frequency domain** (click 7.11.5.2) and for the **space domain** (click 7.11.5.3).

For the contents of **PUSCH DM-RS** and **PT-RS**, click 7.14.

For more detail on the **scheduling** of **PUSCH DM-RS**, refer to clauses 6.4.1.1.1 and 6.4.1.1.3 of TS 38.211 and to clause 6.2.2 of TS 38.214.

For more detail on the **scheduling** of **PUSCH PT-RS**, refer to clauses 6.4.1.2.1 AND 6.4.1.2.2 of TS 38.211 and 6.2.3 of TS 38.214.

### 7.11.5.1 PUSCH DM-RS time-domain scheduling

The **symbol intervals** that contain **DM-RS** are identified with variable "*si*", that can contain **several values**. The **time reference** for "*si*" depends on the **PUSCH DM-RS mapping type**, as follows:

- For **PUSCH DM-RS mapping** `typeA`:

  "*si*" is indicated **relative** to the **first symbol interval either** of the **slot**, when **frequency hopping** is **disabled**, or of **each hop**, when it is **enabled**.

- For **PUSCH DM-RS mapping** `typeB`:

  "*si*" is indicated **relative** to the **first symbol interval either** of the **PUSCH transmission**, when **frequency hopping** is **disabled**, or of **each hop**, when it is **enabled**.

The **PUSCH DM-RS mapping type** is determined by:

- `mappingType` from `PUSCH-TimeDomainResourceAllocation` (click 7.15.6), within **either** `PUSCH-ConfigCommon` (click 7.15.4) or `PUSCH-Config` (click 7.15.5). This is a **mandatory** parameter.
- `mappingType-r16` from `PUSCH-Allocation-r16` (click 7.15.7), within `PUSCH-Config` (click 7.15.5). This is an **optional** parameter.
- `mappingTypeMsgA-PUSCH-r16` from `MsgA-PUSCH-Resource-r16`, within `BWP-ConfigCommon` (click 2.2.8.6), for the case of **MsgA** (click 9.3.6), during a **random access procedure** (click 9). This is an **optional** parameter.

The **data type** of these **3** parameters is an `ENUMERATED{typeA, typeB}`.

The **value(s)** of the **symbol interval index** "*si*" of the **DM-RS PREs** is/are obtained as:

$$si = \bar{l} + si'$$

Notice that TS 38.211 uses the notation "***l***" instead of "***si***" and "***l'***" instead of "***si'***".

The **value(s)** of "***si'***" represent **single** or **double symbol DM-RS**:

- **Single-symbol DM-RS** <=> "***si'***=**0**"

  **Each value** of "$\overline{l}$" implies **1 symbol interval** for **DM-RS**.

  **Single-symbol DM-RS** is **used** when [`maxLength` from `DMRS-UplinkConfig` (click 7.15.9) is **absent**] or when [[it is **present**] and [**single-symbol** is indicated by the "***Antenna ports***" field of the **scheduling DCI message** (click 6.9)]] or when [[it is a **MsgA** (click 9.3.6) **transmission**] and [`msgA-MaxLength-r16` from `MsgA-DMRS-Config-r16` is **absent**]].

- **Double-symbol DM-RS** <=> "***si'*** ∈ {**0**, **1**}"

  **Each value** of "$\overline{l}$" implies **2 consecutive symbol intervals** for **DM-RS**.

  **Double-symbol DM-RS** are **used** when `maxLength` from `DMRS-DownlinkConfig` (click 6.18.11) is **present**, and **double-symbol** is indicated by the "***Antenna ports***" field of the **scheduling DCI message** (click 6.9).

The following table shows, for **PUSCH scheduling**, the **DCI message Formats** where the said **field** is **present**, what is **assigned** by **each** of these **DCI Formats** and what is the **field**'s **bitlength** in **each** of these **DCI Formats**:

| **DCI message field "*Antenna ports*"** | | |
|---|---|---|
| **Assigns** | **DCI Format** | **Bitlength** |
| **PUSCH Unicast** | Format 0_1 (6.9.3) | 2 to 5 |
| | Format 0_2 (6.9.4) | 0, 2 to 5 |

The **semantics** of the "***Antenna ports***" **field** for **PUSCH** has **many cases**, and it is detailed in tables 7.3.1.1.2-**6**, **-6A**, **7**, **-7A** and **-8** to **-23** of TS 38.212.

The **1** or **several value(s)** contained in "$\overline{l}$" are obtained from the following tables from TS 38.212:

- **For single-symbol DM-RS and intra-slot frequency hopping disabled**

  Table 6.4.1.1.3-3 (**1** to **4** values).

- **For double-symbol DM-RS and intra-slot frequency hopping disabled**

  Table 6.4.1.1.3-4 (**1** to **2** values).

- **For single-symbol DM-RS and intra-slot frequency hopping enabled**

  Table 6.4.1.1.3-4 (**1** to **2** values).

For more detail on the **time-domain scheduling** of **PUSCH DM-RS**, refer to clauses 6.4.1.1.1 and 6.4.1.1.3 of TS 38.211 and to clause 6.2.2 of TS 38.214.

### 7.11.5.2 PUSCH DM-RS frequency-domain scheduling

The **PUSCH subcarriers** that contain **DM-RS** are identified with variable "*sc*", that contains **multiple values**. The **expression** for "*sc*" depends on **transform precoding** and on the **DM-RS frequency configuration type**, as follows:

- For [[**transform precoding disabled**] and [**DM-RS frequency configuration UDM-Type 1**]] or for [**transform precoding enabled**]:

  $sc = 4n + 2sc' + \Delta$

- For [[**transform precoding disabled**] and [**DM-RS frequency configuration UDM-Type 2**]]:

  $sc = 6n + sc' + \Delta$

where "$sc' \in \{0, 1\}$" and "$n \in \{0, 1, 2, \ldots\}$". For the **subcarrier offset** "**Δ**" see below.

Notice that TS 38.211 uses the notation "*k*" instead of "*sc*" and "*k'*" instead of "*sc'*".

The **DM-RS frequency configuration type** is determined by `dmrs-Type` from `DMRS-UplinkConfig` (click 7.15.9). If it is **absent**, the **DM-RS frequency configuration** is **DDM-Type 1**. If it is **present**, the **configuration** is **DDM-Type 2**.

The **reference point** for **subcarrier** "*sc*=0" is:

- ***If* transform precoding** (click 4.12.12) is **disabled**, ***then***,

  *sc*=0 is **subcarrier 0** of **Common Resource Block 0** (click 2.2.4).

- ***Else***,

  *sc*=0 is **subcarrier 0** of the **lowest frequency Resource Block** of the **scheduled PUSCH frequency-domain allocation**.

The value of **subcarrier offset** "**Δ**" depends on the **pre-precoding port number** "*p*" and is obtained from table 6.4.1.1.3-1 for **PUSCH DM-RS** configuration **UDM-Type 1** and from table 6.4.1.1.3-2 for **UDM-Type 1** (**both** tables from TS 38.211). These tables are copied below:

Table 6.4.1.1.3-1: variables for **PUSCH DM-RS** configuration **UDM-Type 1**

| *p* | CDM group λ | Δ | $w_f(sc')$ | | $w_t(si')$ | |
|---|---|---|---|---|---|---|
| | | | *sc'* = 0 | *sc'* = 1 | *si'* = 0 | *si'* = 1 |
| **0** | 0 | 0 | +1 | +1 | +1 | +1 |
| **1** | 0 | 0 | +1 | -1 | +1 | +1 |
| **2** | 1 | 1 | +1 | +1 | +1 | +1 |
| **3** | 1 | 1 | +1 | -1 | +1 | +1 |
| **4** | 0 | 0 | +1 | +1 | +1 | -1 |
| **5** | 0 | 0 | +1 | -1 | +1 | -1 |
| **6** | 1 | 1 | +1 | +1 | +1 | -1 |
| **7** | 1 | 1 | +1 | -1 | +1 | -1 |

Table 6.4.1.1.3-2: variables for **PUSCH DM-RS** configuration **UDM-Type 2**

| $\tilde{p}$ | CDM group λ | Δ | $w_f(sc')$ | | $w_t(si')$ | |
|---|---|---|---|---|---|---|
| | | | $sc' = 0$ | $sc' = 1$ | $sc' = 0$ | $sc' = 1$ |
| **0** | 0 | 0 | +1 | +1 | +1 | +1 |
| **1** | 0 | 0 | +1 | -1 | +1 | +1 |
| **2** | 1 | 2 | +1 | +1 | +1 | +1 |
| **3** | 1 | 2 | +1 | -1 | +1 | +1 |
| **4** | 2 | 4 | +1 | +1 | +1 | +1 |
| **5** | 2 | 4 | +1 | -1 | +1 | +1 |
| **6** | 0 | 0 | +1 | +1 | +1 | -1 |
| **7** | 0 | 0 | +1 | -1 | +1 | -1 |
| **8** | 1 | 2 | +1 | +1 | +1 | -1 |
| **9** | 1 | 2 | +1 | -1 | +1 | -1 |
| **10** | 2 | 4 | +1 | +1 | +1 | -1 |
| **11** | 2 | 4 | +1 | -1 | +1 | -1 |

Notice that TS 38211 uses the notation "$\tilde{p}$" with "~" on top, instead of "$\tilde{p}$".

The relationship between the **antenna ports** "$p$" and the **pre-precoding antenna ports** "$\tilde{p}$" is determined by the **PUSCH precoding matrix** (click 4.12.13).

The corresponding **pre-precoding antenna port(s)** "$\tilde{p}$" is/are entered into the corresponding table above to obtain **subcarrier offset** "**Δ**". The **subcarrier indexes** "$sc$" (where **PDSCH DM-RS** are located) are **shifted upwards** in the value of "**Δ**". **Each value** of "**Δ**" therefore indicates a different **PUSCH DM-RS group**.

The result of the **DM-RS frequency scheduling procedure** (as explained up to now) is the following:

- For **UDM-Type 1**

  Table 6.4.1.1.3-1 and "$sc = 4n + 2sc' + \Delta$"

  **1 PUSCH DM-RS symbol** is placed in **even subcarrier indexes** (i.e., **0**, **2**, **4**, …), plus **offset** "**Δ**". The value of "**Δ**" is **0** or **1** depending on the **pre-precoding antenna port** "$\tilde{p}$", as indicated in the table above.

  **UDM-Type 1** with **single-symbol DM-RS** supports **4 DM-RS pre-precoding antenna ports** "$\tilde{p}$" (**0** to **3**) and with **double-symbol DM-RS** supports **8 DM-RS pre-precoding antenna port** "$\tilde{p}$" (**0** to **7**).

- For **UDM-Type 2** and **transform precoding enabled**:

  Table 6.4.1.1.3-2 and "$sc = 4n + 2sc' + \Delta$"

  **1 PUSCH DM-RS symbol** is placed in **even subcarrier indexes** (i.e., **0**, **2**, **4**, …), plus **offset** "**Δ**". The value of "**Δ**" is **0**, **2**, or **4** depending on the **pre-precoding antenna port** "$\tilde{p}$", as indicated in the table above.

  **UDM-Type 2** with **single-symbol DM-RS** supports **6 DM-RS pre-precoding antenna port** "$\tilde{p}$" (**0** to **5**) and with **double-symbol DM-RS** supports **12 DM-RS pre-precoding antenna port** "$\tilde{p}$" (**0** to **11**).

- For **UDM-Type 2** and **transform-precoding disabled**:

  Table 6.4.1.1.3-2 and "$sc = 6n + sc' + \Delta$"

**1 PUSCH DM-RS symbol** is placed in **subcarrier indexes** {**0**, **1**, **6**, **7**, **11**, **12**, …}, plus **offset** "**Δ**". The value of "**Δ**" is **0**, **2**, or **4** depending on **pre-precoding antenna port** "***p***", as indicated in the table above.

**UDM-Type 2** with **single-symbol DM-RS** supports **6 DM-RS pre-precoding antenna port** "***p***" (**0** to **5**) and with **double-symbol DM-RS** supports **12 DM-RS pre-precoding antenna port** "***p***" (**0** to **11**).

Notice that clauses 6.4.1.1 of TS 38.211 and 6.2.2 of TS 38.214 contain **clarifications** and **particular** cases that **modify** the **overall procedure** presented in this section.

For more detail on the **time-domain scheduling** of **PUSCH DM-RS**, refer to clauses 6.4.1.1.1 and 6.4.1.1.3 of TS 38.211 and to clause 6.2.2 of TS 38.214.

### 7.11.5.3 PUSCH DM-RS space-domain scheduling

The usage of **single-symbol PUSCH DM-RS** or **double-symbol PDSCH DM-RS** has an impact on the **supported pre-precoding antenna ports**, as indicated in table 6.4.1.1.3-5 of TS 38.211:

| | ***si'*** | **Supported pre-precoding antenna ports "*p*"** | |
|---|---|---|---|
| | | **Configuration UDM-Type 1** | **Configuration UDM-Type 2** |
| **Single-symbol DM-RS** | 0 | 0 to 3 | 0 to 5 |
| **Double-symbol DM-RS** | 0, 1 | 0 to 7 | 0 to 11 |

This table shows which **DM-RS pre-precoding antenna ports** can be used for **each combination** of **DM-RS frequency configuration type** and **single/double DM-RS symbol**. For **single-symbol DM-RS**, **up to 4 DM-RS pre-precoding antenna ports** can be used for configuration **UDM-Type 1** and a **up to 6 DM-RS pre-precoding antenna ports** can be used for configuration **UDM-Type 2**. For **double-symbol DM-RS**, **up to 8 DM-RS pre-precoding antenna ports** can be used for configuration **UDM-Type 1** and a **up to 12 DM-RS pre-precoding antenna ports** can be used for configuration **UDM-Type 2**.

When `maxLength` from `DMRS-UplinkConfig` (click 7.15.9) is **absent**, **single-symbol DM-RS** are used. If it is **present**, the usage of **single-symbol** or **double-symbol DM-RS** is determined by the scheduling **DCI message**.

This information is used to **select** which **antenna port table** is associated with **DCI Format 0_1** for setting up **SISO/MIMO.**

For more detail on the **space-domain scheduling** of **PUSCH DM-RS**, refer to clauses 6.4.1.1.1 and 6.4.1.1.3 of TS 38.211 and to clause 6.2.2 of TS 38.214.

## 7.12 PUSCH payload contents

**Each PUSCH transmission** contains **1 unicast PUSCH codeword** in **1** to **4 layers**.

For a **new transmission**, **each PUSCH codeword** contains **1 encoded transport block** (click 4.12.1.1). **Each PUSCH codeword** can also be **multiplexed** (click 4.12.8) with **1 PUCCH codeword** for their **joint delivery** in **1 PUSCH transmission**.

For a **HARQ retransmission** (click 4.5.6.6), **each codeword** contains **all** or **part** of **the** above information.

For an overview of the **data flow** across the **user-plane protocol stack**, click 4.10.4.

## 7.13 PUSCH payload processing

**PUSCH payload processing** is here presented in a **very high-level way**, but **PDSCH payload processing** (click 6.16) is presented with **substantial detail**.

**PUSCH payload Tx processing**

For the **Tx processing** of the **PUSCH payload** by the **UE**, the **UE first** has to determine the following variables:

- **Modulation** and **target code rate**

  For more detail, refer to clause 6.1.4.1 of TS 38.214.

- **Redundancy version**

  For more detail, refer to clauses 6.1.2.1 and 6.1.4 of TS 38.214.

- **Size** of the **transport block**

  For more detail, refer to clause 6.1.4.2 of TS 38.214.

- **LDPC base graph**

  For more detail, refer to clause 6.2.2 of TS 38.212.

**After** having **determined** the above **variables**, the **Tx processing** of the **PUSCH payload** by the **UE** is **jointly described** with the **Tx processing** of the **PDSCH payload** by the **NR node** because **both processes** are **quite similar**. Whenever there is a **difference** between the **Tx processing** of the **PUSCH payload** and the one of the **PDSCH payload** it is **explicitly indicated**. For this joint description **PDSCH/PUSCH payload Tx processing**, click 4.12.2.

**PUSCH payload Rx processing**

The **Rx processing** of the **PUSCH payload** by the **NR node** is **jointly described** with the **Rx processing** of the **PDSCH payload** by the **UE** because **both processes** are **quite similar**. Whenever there is a **difference** between the **Rx processing** of the **PUSCH payload** and the one of the **PDSCH payload** it is **explicitly indicated**. For this joint description **PDSCH/PUSCH payload Rx processing**, click 4.12.16.

For more detail on **PUSCH payload processing** through **PUSCH**, refer to clauses 5.2.2 (segmentation), 5.3.2 and 5.3.3 (encoding), 5.4.2 (rate-matching), 5.5 (concatenation) and 6.3.2 (overall) of TS 38.212, and clause 6.3.1 (modulation) of TS 38.211.

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_PTRS_DL.html

## 7.14 PUSCH DM-RS and PT-RS contents

**PUSCH DM-RS contents**

The **PUSCH demodulation reference signals** (**DM-RS**) are the **DM-RS** (click 4.11.3.2.1) within **each PUSCH transmission**. These **signals** are characterized by their **scheduling** (click 7.11.5) within the **PUSCH transmission** and by their **contents**.

For more detail on **PUSCH DM-RS contents**, refer to clause 6.4.1.1.1 of TS 38.211.

**PUSCH PT-RS contents**

The **PUSCH phase-tracking reference signals** (**PT-RS**) are the **PT-RS** (click 4.11.3.2) within **each PUSCH transmission**. These **signals** are characterized by their **scheduling** within the **PUSCH transmission** and by their **contents**.

The **PUSCH PT-RS** are configured with parameter `phaseTrackingRS` from `DMRS-UplinkConfig` (click 7.15.9). If it is **absent**, the **UE must not** transmit **PUSCH PT-RS**.

The **contents** of **PUSCH PT-RS** depend on whether **transform-precoding** (click 4.12.12) is **enabled** or **not**.

For more detail on **PUSCH PT-RS contents** with **transform-precoding disabled**, refer to clause 6.4.1.2.1.1 of TS 38.211.

For more detail on **PUSCH PT-RS contents** with **transform-precoding enabled**, refer to clause 6.4.1.2.1.2 of TS 38.211.

## 7.15 PUSCH parameters

The **Physical Uplink Shared Channel** (**PUSCH**, click 7.10) is configured with:

- **Common** parameters that are **broadcast** within `MIB`, as described in section 7.15.1.º
- **Common parameters** that are **broadcast** (or **unicast**) within `SIB1`, as described in section 7.15.2.
- **Common** and **dedicated** parameters that are **unicast** within the **non-SI parameter group**, as described in section 7.15.3.

Possibly existing **PUSCH parameters** within **other SIBs and posSIBs** are **not addressed** in this book.

For an overall explanation on **UE configuration parameters** click 5. For a brief overview of **ASN.1**, click 5.9.

### 7.15.1 PUSCH parameters within `MIB`

**PUSCH** is configured with **parameters broadcast** in `MIB` (click 5.3.3), within **SSB**, in the following case:

- In **UE**'s **primary cell**, **during cell selection**.

The corresponding parameter is `dmrs-TypeA-Position` from `MIB` (click 8.12.3).

For this parameter **broadcast** in `MIB`, its **ancestor tree up to** the **top** is:

```
                                BCCH-BCH-Message (click 5.8.5.1)
message                           BCCH-BCH-MessageType (click 5.8.5.1)
  mib                               MIB (click 8.12.3)
    dmrs-TypeA-Position               ENUMERATED {pos2, pos3}
```

### 7.15.2 PUSCH parameters within `SIB1`

**PUSCH** is configured with **parameters broadcast** in a **RRC** `SIB1` **message** (click 5.3.4) in the following case:

- In **UE**'s **primary cell**, for **PDSCH** in the **Downlink initial bandwidth part during initial access**.

The **RRC** `SIB1` **message** is sent within a **RRC** `BCCH-DL-SCH-Message` **class** (click 5.8.5.2).

The **most relevant related data type** is:

- `PUSCH-ConfigCommon` (click 7.15.4): common parameters

The following **data types**, that are **descendants** of **both** `PUSCH-ConfigCommon` and of `PUSCH-Config` (click 7.15.5), are also relevant:

- `PUSCH-TimeDomainResourceAllocationList` (see within 7.15.4.1 and 7.15.5.1): instantiated as common
- `PUSCH-TimeDomainResourceAllocation` (click (7.15.6): instantiated as common

The **ancestor tree** (click 5.9.8) of this **most relevant related data type** up to `SIB1` is:

```
                                  SIB1 (click 5.7.4)
servingCellConfigCommon             ServingCellConfigCommonSIB (0,1) (click 3.9.3)
  uplinkConfigCommon                  UplinkConfigCommonSIB (click 2.2.6.5)
    initialUplinkBWP                    BWP-UplinkCommon (click 2.2.8.92.2.8.6)
      pusch-ConfigCommon                  PUSCH-ConfigCommon (0,1) (click 7.15.4)
```

For the **ancestor trees** of `SIB1` **up to** the **top**, click 5.3.4.

## 7.15.3 PUSCH parameters within the Non-SI parameter group

**PUSCH** is configured with **common** and **dedicated parameters** that are **unicast** within the **Non-SI parameter group** (click 5.3.6) in the following cases:

- In **UE**'s **primary cell**, **PUSCH** for the **additional bandwidth parts** (i.e., **other** than the **initial bandwidth part**).
- In the **primary-secondary cell** and in **secondary cells**, **PUSCH** for **all bandwidth parts**.
- In **UE**'s **primary cell**, **PUSCH** for the **initial bandwidth part**, if configuration changes are done **after initial access**.

The **most relevant related data types** are:

- `PUSCH-ConfigCommon` (click 7.15.4): dedicated parameters
- `PUSCH-Config` (click 7.15.5): dedicated parameters
- `PUSCH-ServingCellConfig` (click 7.15.8): dedicated parameters

The following **data types**, **descendants** of `PUSCH-ConfigCommon` (click 7.15.4) and of `PDSCH-Config` (click 7.15.5), are also relevant:

- `PUSCH-TimeDomainResourceAllocationList` (see within 7.15.4.1 and 7.15.5.1): instantiated as dedicated
- `PUSCH-TimeDomainResourceAllocation` (click 7.15.6): instantiated as dedicated

The following **data types**, **descendants** of `PDSCH-Config` (click 7.15.5), are also relevant:

- `PUSCH-TimeDomainResourceAllocationList-r16` (see within 7.15.5.1): dedicated parameters

- **PUSCH-TimeDomainResourceAllocation-r16** (see within 7.15.5.1): dedicated parameters
- **PUSCH-Allocation-r16** (click 7.15.7): dedicated parameters

For **PUSCH** parameters within the **Non-SI parameter group**, we show the following **2 ancestor trees**.

**First**, the **ancestor tree** (click 5.9.8) of the **most relevant related data types** (see above), up to **CellGroupConfig** is:

```
                                CellGroupConfig (click 3.7.3)
spCellConfig                      SpCellConfig (click 3.9.5)
  reconfigurationWithSync       ReconfigurationWithSync (0,1) (click 3.9.8)
    spCellConfigCommon            ServingCellConfigCommon (click 3.9.6)
      uplinkConfigCommon            UplinkConfigCommon (click 2.2.6.9)
        initialUplinkBWP              BWP-UplinkCommon (click 2.2.8.9)
          pusch-ConfigCommon            PUSCH-ConfigCommon (click 7.15.4)
      supplementaryUplinkConfig   UplinkConfigCommon (click 2.2.6.9)
        initialUplinkBWP              BWP-UplinkCommon (click 2.2.6.9)
          pusch-ConfigCommon            PUSCH-ConfigCommon (click 7.15.4)
  spCellConfigDedicated         ServingCellConfig (click 3.9.7)
    uplinkConfig                  UplinkConfig (click 2.2.8.42.2.8.3)
      initialUplinkBWP              BWP-UplinkDedicated (click 2.2.8.10)
        pusch-Config                  PUSCH-Config (0,1) (click 7.15.5)
      uplinkBWP-ToReleaseList       BWP-Id (0..4) (click 2.2.8.13)
      uplinkBWP-ToAddModList        BWP-Uplink (0..4) (click 2.2.8.8)
        bwp-Common                    BWP-UplinkCommon (click 2.2.6.9)
          PUSCH-ConfigCommon            PUSCH-ConfigCommon (click 7.15.4618.5)
        bwp-Dedicated                 BWP-UplinkDedicated (click 2.2.8.102.2.8.7)
          pusch-Config                  PUSCH-Config (0,1) (click 7.15.5)
      pusch-ServingCellConfig       PUSCH-ServingCellConfig (0,1) (click 7.15.8)
    supplementaryUplink           UplinkConfig (click 2.2.8.3)
      initialUplinkBWP              BWP-UplinkDedicated (click 2.2.8.10)
        pusch-Config                  PUSCH-Config (0,1) (click 7.15.5)
      uplinkBWP-ToReleaseList       BWP-Id (0..4) (click 2.2.8.13)
      uplinkBWP-ToAddModList        BWP-Uplink (0..4) (click 2.2.8.8)
        bwp-Common                    BWP-UplinkCommon (click 2.2.6.9)
          PUSCH-ConfigCommon            PUSCH-ConfigCommon (click 7.15.4618.5)
        bwp-Dedicated                 BWP-UplinkDedicated (click 2.2.8.102.2.8.7)
          pusch-Config                  PUSCH-Config (0,1) (click 7.15.5)
      pusch-ServingCellConfig       PUSCH-ServingCellConfig (0,1) (click 7.15.8)
sCellToAddModList               SCellConfig (0..maxNrofSCells) (click 3.9.4)
  sCellConfigCommon               ServingCellConfigCommon (0,1) (click 3.9.6)
    uplinkConfigCommon              UplinkConfigCommon (click 2.2.6.9)
      initialUplinkBWP                BWP-UplinkCommon (click 2.2.8.9)
        pusch-ConfigCommon              PUSCH-ConfigCommon (click 7.15.4)
    supplementaryUplinkConfig     UplinkConfigCommon (click 2.2.6.9)
      initialUplinkBWP                BWP-UplinkCommon (click 2.2.8.9)
        pusch-ConfigCommon              PUSCH-ConfigCommon (click 7.15.4)
```

```
sCellConfigDedicated          ServingCellConfig (click 3.9.7)
  uplinkConfig                  UplinkConfig (click 2.2.8.42.2.8.3)
    initialUplinkBWP              BWP-UplinkDedicated (click 2.2.8.10)
      pusch-Config                  PUSCH-Config (0,1) (click 7.15.5)
    uplinkBWP-ToReleaseList       BWP-Id (0..4) (click 2.2.8.13)
    uplinkBWP-ToAddModList        BWP-Uplink (0..4) (click 2.2.8.8)
      bwp-Common                    BWP-UplinkCommon (click 2.2.6.9)
        PUSCH-ConfigCommon            PUSCH-ConfigCommon (click 7.15.4618.5)
      bwp-Dedicated                 BWP-UplinkDedicated (click 2.2.8.102.2.8.7)
        pusch-Config                  PUSCH-Config (0,1) (click 7.15.5)
    pusch-ServingCellConfig       PUSCH-ServingCellConfig (0,1) (click 7.15.8)
  supplementaryUplink           UplinkConfig (click 2.2.8.3)
    initialUplinkBWP              BWP-UplinkDedicated (click 2.2.8.10)
      pusch-Config                  PUSCH-Config (0,1) (click 7.15.5)
    uplinkBWP-ToReleaseList       BWP-Id (0..4) (click 2.2.8.13)
    uplinkBWP-ToAddModList        BWP-Uplink (0..4) (click 2.2.8.8)
      bwp-Common                    BWP-UplinkCommon (click 2.2.6.9)
        PUSCH-ConfigCommon            PUSCH-ConfigCommon (click 7.15.4618.5)
      bwp-Dedicated                 BWP-UplinkDedicated (click 2.2.8.102.2.8.7)
        pusch-Config                  PUSCH-Config (0,1) (click 7.15.5)
    pusch-ServingCellConfig       PUSCH-ServingCellConfig (0,1) (click 7.15.8)
```

Some **relevant remarks** on this **first ancestor tree** are:

- For the **ancestor trees** of `CellGroupConfig`, **up to** some **RRC NR messages** click 3.7.1 and **up to** some **RRC inter-node messages** click 3.7.2.
- The constant `maxNrofSCells` is `31`, which is the **maximum quantity** of **UE**'s **secondary cells** (i.e., **other** than the **special cells**) in **each cell group**
- The constant `maxNrofBWPs` is `4`, which is the **maximum quantity** of **UE**'s **additional bandwidth parts** (i.e., **other** than the **initial bandwidth part**).

**Second**, the **additional ancestor tree** (click 5.9.8) of `PUSCH-Config`, **up to** the **top**, is:

```
                                  DL-DCCH-Message (click 5.8.5.4)
message                             DL-DCCH-MessageType (click 5.8.5.4)
  c1                                  CHOICE
    rrcRelease                          RRCRelease (click 5.8.5.4)
      rrcRelease                          RRCRelease-IEs
        suspendConfig                       SuspendConfig (0,1)
          sdt-Config-r17                      SDT-Config-r17 (0,1)
            sdt-MAC-PHY-CG-Config-r17           SDT-CG-Config-r17 (0,1)
                                                OCTET STRING (CONTAINING
                                                SDT-MAC-PHY-CG-Config-r17)
              cg-SDT-ConfigInitialBWP-DL-r17 BWP-UplinkDedicatedSDT-r17
                pusch-Config-r17                PUSCH-Config (0,1) (6.18.6)
```

Some **relevant remarks** on this **second ancestor tree** are:

- This `PUSCH-Config` instance is contained in `RRCRelease` **message**, in addition to the ones contained in `RRCReconfiguration`, `RRCResume` and `RRCSetup` (click 5.8.5.11) **messages** from the **first ancestor tree**.
- This `PUSCH-Config` instance is devoted to **small data transmission** (**SDT**).

## 7.15.4 `PUSCH-ConfigCommon` data type

It can be **both broadcast** (or **unicast**) within `SIB1` (click 5.3.4) and **unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `SIB1` (click 7.15.2).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 7.15.3.

Its **only parent data type** is `BWP-UplinkCommon` (click 2.2.8.9).

**All** its parameters are **common** (click 5.1.3).

**Each** of its **instances** defines several low-level parameters for **PUSCH**, for its corresponding **bandwidth part** (**initial** or **additional**), of **common configuration**, of its corresponding **cell** (**special** or **secondary**), of its corresponding **cell group** (**master** or **secondary**).

### 7.15.4.1 `PUSCH-ConfigCommon` parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `groupHoppingEnabledTransformPrecoding ENUMERATED {enabled}`: optional, common

  It configures **group hopping** for **PUSCH DM-RS** with **transform-precoder**. It is a **cell-specific parameter**. For more detail, refer to clause 6.4.1.1.2 of TS 38.211.

- `msg3-DeltaPreamble INTEGER (-1..6)`: optional, common

  It indicates the **power offset** between **Msg3** and the **RACH preamble transmission**, as follows:

  **Power offset** = 2 • `parameter_value` expressed in **decibels** (**dB**)

  For more detail, refer to clause 7.1 of TS 38.213.

- `pusch-TimeDomainAllocationList PUSCH-TimeDomainResourceAllocationList`: optional, common

  It is a **list** of **PUSCH time-domain resource assignments** (click 7.11.4 and refer to clause 6.1.2.1 of TS 38.214).

  Its **data type** is a `SEQUENCE OF PUSCH-TimeDomainResourceAllocation` (click 7.15.6), defined as follows:

```
PUSCH-TimeDomainResourceAllocationList ::= SEQUENCE
      (SIZE(1..maxNrofUL-Allocations)) OF
      PUSCH-TimeDomainResourceAllocation
```

  where `maxNrofUL-Allocations` is `16`, which is the **maximum quantity** of **entries** in the **list**.

  This **parameter** can be configured for **each Uplink bandwidth part**.

  Notice that there is a **same-name** (**dedicated**) **parameter**, with the **same data type** and the **same semantics**, from `PUSCH-Config` (click 7.15.5).

For more detail on these parameters, refer to the table "*PUSCH-ConfigCommon field descriptions*" of TS 38.331.

### 7.15.4.2 PUSCH-ConfigCommon ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
PUSCH-ConfigCommon ::= SEQUENCE {
  groupHoppingEnabledTransformPrecoding ENUMERATED {enabled}
                                        OPTIONAL,                   -- Need R
  pusch-TimeDomainAllocationList PUSCH-TimeDomainResourceAllocationList
                                 OPTIONAL,                          -- Need R
  msg3-DeltaPreamble INTEGER (-1..6) OPTIONAL,                      -- Need R
  p0-NominalWithGrant INTEGER (-202..24) OPTIONAL,                  -- Need R
  ...
}
```

## 7.15.5 PUSCH-Config data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 7.15.3.

Its **2 parent data types** are `BWP-Uplink` (click 2.2.8.8) and `BWP-UplinkDedicatedSDT-r17`.

**All** its parameters are **dedicated** (click 5.1.3).

**Each** of its **instances** configures the **UE** with **most PUSCH variables**, of its corresponding **bandwidth part** (**initial** or **additional**), of **dedicated configuration**, of its corresponding **cell** (**special** or **secondary**), of its corresponding **cell group** (**master** or **secondary**). The "**corresponding**" variables are determined by the **specific place**, within the overall **ASN.1 specification**, where this **data type** has been **instantiated**.

### 7.15.5.1 PUSCH-Config parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `dataScramblingIdentityPUSCH INTEGER (0..1023)`: optional, dedicated

  It is the identifier "$c_{init}$" to **initialize PUSCH data scrambling**. If it is **absent**, the **UE** applies the **physical cell ID** (refer to clause 6.3.1.1 of TS 38.211).

- `dmrs-UplinkForPUSCH-MappingTypeA DMRS-UplinkConfig`: optional, dedicated

- `dmrs-UplinkForPUSCH-MappingTypeB DMRS-UplinkConfig`: optional, dedicated

  **Each** of these **2 parameters** provides the **PUSCH DM-RS time-domain resource allocation** (click 7.11.5.1) using (respectively) **mapping** `typeA` or `typeB`. The **mapping type** may be chosen **dynamically** with parameter `mappingType` from `PUSCH-TimeDomainResourceAllocation` (click 7.15.6).

- `pusch-TimeDomainAllocationList PUSCH-TimeDomainResourceAllocationList`: optional, dedicated

  It is a **list** of **PUSCH time-domain resource assignments** (click 7.11.4 and refer to clause 6.1.2.1 of TS 38.214).

Its **data type** is a **SEQUENCE OF** **PUSCH-TimeDomainResourceAllocation** (click 7.15.6), defined as follows:

```
PUSCH-TimeDomainResourceAllocationList ::= SEQUENCE
      (SIZE(1..maxNrofUL-Allocations)) OF
      PUSCH-TimeDomainResourceAllocation
```

where **maxNrofUL-Allocations** is **16**, which is the **maximum quantity** of **entries** in the **list**.

This parameter applies to **PUSCH transmissions scheduled** with a **DCI message** with **Formats 0_0** (refer to table 6.1.2.1.1-1 of TS 38.214). In case that **pusch-TimeDomainAllocationListDCI-0-1-r16** from this **same data type** is **absent**, this **parameter** also applies to **DCI messages** with **Format 0_1** (refer to table 6.1.2.1.1-1A of TS 38.214).

The **NR node** does **not configure** this parameter **simultaneously** with **either** of **pusch-TimeDomainAllocationListDCI-0-2-r16**, **pusch-TimeDomainAllocationListDCI-0-2-r16** or **pusch-TimeDomainAllocationListForMultiPUSCH** **all** of them (see the next bullet) from **this same data type**.

This **parameter** can be configured for **each Uplink bandwidth part**.

Notice that there is a **same-name** (**common**) **parameter**, with the **same data type** and the **same semantics**, from **PUSCH-ConfigCommon** (click 7.15.4).

- **pusch-TimeDomainAllocationListDCI-0-1-r16** **PUSCH-TimeDomainResourceAllocationList-r16**: optional, dedicated
- **pusch-TimeDomainAllocationListDCI-0-2-r16** **PUSCH-TimeDomainResourceAllocationList-r16**: optional, dedicated
- **pusch-TimeDomainAllocationListForMultiPUSCH-r16** **PUSCH-TimeDomainResourceAllocationList-r16**: optional, dedicated

**Each** of these **3 parameters** is a **list** of **1** to **64** **multi-entries** of **PUSCH time-domain resource assignments**. **Each** such **multi-entry corresponds** to a **multiple PDSCH time-domain resource assignment** (click 7.11.4 and refer to clause 6.1.2.1 of TS 38.214).

Regarding parameter with **suffix** "**DCI-0-1-r16**", it applies to **PDSCH transmissions** that are **scheduled** with a **DCI message** with **Format 0_1** (refer to table 6.1.2.1.1-1A of TS 38.214).

Regarding parameter with **suffix** "**DCI-0-2-r16**", it applies to **PDSCH transmissions** that are **scheduled** with a **DCI message** with **Format 0_2** (refer to table 6.1.2.1.1-1B of TS 38.214).

Regarding parameter with **suffix** "**MultiPUSCH-r16**", the **NR node** configures **at most 16 entries** in the **list** of this parameter. This parameter is **not configured simultaneously** with **pusch-AggregationFactor** from **this same data type**. This parameter is **not configured simultaneously** with **pusch-TimeDomainAllocationListDCI-0-1-r16** from **this same data type**.

Their **data type** is a **SEQUENCE OF instances** of **PUSCH-TimeDomainResourceAllocation-r16**, and its **ASN.1** definition is:

```
PUSCH-TimeDomainResourceAllocationList-r16 ::= SEQUENCE
          (SIZE(1..maxNrofUL-Allocations-r16)) OF
          PUSCH-TimeDomainResourceAllocation-r16
```

where **maxNrofUL-Allocations** is **64**, which is the **maximum quantity** of **entries** in the **list**.

**Each instance** of the **data type** **PUSCH-TimeDomainResourceAllocation-r16** represents a **multiple PUSCH time-domain resource assignment**. **Each instance** of **PUSCH-TimeDomainResourceAllocation-r16** contains a **list** of **1** to **8** **PUSCH time-domain resource assignments** and their **shared value** of **PUSCH variable** "$K_2$". The **data type** **PUSCH-TimeDomainResourceAllocation-r16** is defined as follows is:

```
PUSCH-TimeDomainResourceAllocation-r16 ::= SEQUENCE {
  k2-r16 INTEGER (0..32) OPTIONAL,                    -- Need S
  puschAllocationList-r16 SEQUENCE
                          (SIZE(1..maxNrofMultiplePUSCHs-r16))
                          OF PUSCH-Allocation-r16,
  ...
}
```

where **maxNrofMultiplePUSCHs-r16** is **8**, which is the **maximum quantity** of **PUSCH time-domain resource assignments** in in **each** of the **sets** of **multi-PUSCH allocations**.

Notice that there is **data type** **MultiPDSCH-TDRA-r17** (see within 6.18.6) that although has a **different name**, it has **reciprocal semantics** (in respect to **PDSCH**) to the ones of **PUSCH-TimeDomainResourceAllocation-r16**.

The **field description** (from "*PUSCH-TimeDomainResourceAllocationList field descriptions*") for **puschAllocationList-r16** is:

- It contains a **set** of **1** to **8** **PUSCH time-domain resource assignments** (**each** of them with **data type** **PUSCH-Allocation-r16**, click 7.15.7) and their **shared value** of **PUSCH variable** "$K_2$" (click 7.11.4.2 and refer to clause 6.1.2.1 of TS 38.214).

  When this **parameter** is within a **data type instance** that is associated **either** to **pusch-TimeDomainAllocationListDCI-0-1-r16** or to **pusch-TimeDomainAllocationListDCI-0-2-r16** (both of them from **PUSCH-Config**, click 7.15.5), the **set** contained in this parameter **only** has **1** **PUSCH time-domain resource assignment** (i.e., the **set only** has **1** **instance** of **PUSCH-Allocation-r16**).

- **resourceAllocation ENUMERATED {resourceAllocationType0, resourceAllocationType1, dynamicSwitch}**: mandatory, dedicated
- **resourceAllocationDCI-0-2-r16 ENUMERATED {resourceAllocationType0, resourceAllocationType1, dynamicSwitch}**: mandatory, dedicated

**Each** of these 2 parameters indicates the **PDSCH frequency-domain resource allocation** of **either UFA-Type 0** or **UFA-Type 1** (click 7.11.3) for **non-fallback DCI**. Parameter **resourceAllocation** applies to **DCI Format 0_1**, while

parameter **resourceAllocationDCI-1-2-r16** applies to **DCI Format 0_2**. For more detail, refer to clause 6.1.2 of TS 38.214.

- **tp-pi2BPSK ENUMERATED {enabled}**: optional, dedicated

  It **enables π/2-BPSK modulation** if **transform-precoding** (click 4.12.12) is **enabled** in **PUSCH**.

  If it is **absent**, **π/2-BPSK modulation** is **disabled**.

For more detail on these parameters, refer to the table "*PUSCH-Config field descriptions*" of TS 38.331.

### 7.15.5.2 PUSCH-Config ASN.1 specification

Its **partial ASN.1** (click 5.9) specification is:

```
PUSCH-Config ::= SEQUENCE {
  dataScramblingIdentityPUSCH INTEGER (0..1023) OPTIONAL,           -- Need S
  txConfig ENUMERATED {codebook, nonCodebook} OPTIONAL, -- Need S
  dmrs-UplinkForPUSCH-MappingTypeA SetupRelease {DMRS-UplinkConfig}
                                   OPTIONAL,                        -- Need M
  dmrs-UplinkForPUSCH-MappingTypeB SetupRelease {DMRS-UplinkConfig}
                                   OPTIONAL,                        -- Need M
  pusch-PowerControl PUSCH-PowerControl OPTIONAL,                   -- Need M
  frequencyHopping ENUMERATED {intraSlot, interSlot} OPTIONAL,      -- Need S
  frequencyHoppingOffsetLists SEQUENCE (SIZE (1..4)) OF INTEGER
                  (1..maxNrofPhysicalResourceBlocks-1) OPTIONAL,    -- Need M
  resourceAllocation ENUMERATED {resourceAllocationType0,
                                 resourceAllocationType1, dynamicSwitch},
  pusch-TimeDomainAllocationList SetupRelease
             {PUSCH-TimeDomainResourceAllocationList} OPTIONAL,     -- Need M
  other parameters (not shown)
  tp-pi2BPSK ENUMERATED {enabled} OPTIONAL, -- Need S
  other parameters (not shown)
  resourceAllocationDCI-0-2-r16 ENUMERATED {resourceAllocationType0,
         resourceAllocationType1, dynamicSwitch} OPTIONAL,          -- Need M
  other parameters (not shown)
  pusch-TimeDomainAllocationListDCI-0-2-r16 SetupRelease
           {PUSCH-TimeDomainResourceAllocationList-r16} OPTIONAL,   -- Need M
  pusch-TimeDomainAllocationListDCI-0-1-r16 SetupRelease
           {PUSCH-TimeDomainResourceAllocationList-r16} OPTIONAL,   -- Need M
  other parameters (not shown)
  pusch-TimeDomainAllocationListForMultiPUSCH-r16 SetupRelease
           {PUSCH-TimeDomainResourceAllocationList-r16} OPTIONAL,   -- Need M
  other parameters (not shown)
}
```

## 7.15.6 PUSCH-TimeDomainResourceAllocation data type

It can be **both broadcast** (or **unicast**) within **SIB1** (click 5.3.4) and **unicast** within the **Non-SI parameter group** (click 5.3.6).

Its **only parent data type** is **PUSCH-TimeDomainResourceAllocationList** (see within 7.15.4.1 and 7.15.5.1). The **only parent data types** of **PUSCH-TimeDomainResourceAllocationList** are **PUSCH-ConfigCommon** (click 7.15.4) and **PUSCH-Config** (click 7.15.5).

**Each** of its **instances** contains **either all common parameters or all dedicated parameters** (click 5.1.3), depending on where it is **instantiated**.

**Each** of its **instances** configures **1** case of **PUSCH time domain allocation** that can be **selected** for **1 PUSCH assignment** within to the corresponding **UE**'s **Uplink** (or **SUL**) **bandwidth part**.

Its **ancestor tree**, within **SIB1** **up to** **PUSCH-ConfigCommon** (click 7.15.4) is:

```
                    PUSCH-ConfigCommon (click 7.15.4)
pusch-TimeDomainAllocationList
                      PUSCH-TimeDomainResourceAllocationList (0,1) (click 7.15.4.1)
                        PUSCH-TimeDomainResourceAllocation
                        (1..maxNrofUL-Allocations) (click 7.15.6)
```

Some **relevant remarks** on this **ancestor tree** are:

- The constant **maxNrofUL-Allocations** is **16**, which is the maximum **quantity** of **Uplink time-domain resource allocations** in the corresponding **list**.
- For the **ancestor tree** of **PUSCH-ConfigCommon** (click 7.15.4) **up to** the **top** click 7.15.2.

Its **ancestor tree**, within the **Non-SI parameter group**, **up to** **PUSCH-Config** (click 7.15.5) is:

```
                    PUSCH-Config (click 7.15.5)
pusch-TimeDomainAllocationList
                      PUSCH-TimeDomainResourceAllocationList (0,1) (click 7.15.5.1)
                        PUSCH-TimeDomainResourceAllocation
                        (1..maxNrofUL-Allocations) (click 7.15.6)
```

Some **relevant remarks** on this **ancestor tree** are:

- The constant **maxNrofUL-Allocations** is **16**, which is the maximum **quantity** of **Uplink time-domain resource allocations** in the corresponding **list**.
- For the **ancestor tree** of **PUSCH-Config** (click 7.15.5) **up to** **CellGroupConfig** (click 3.7.3), click 7.15.3.

Notice that there is the **data type** **PUSCH-Allocation-r16** (click 7.15.7) that has a **similar name** and **extended semantics**

Notice that there is the **data type** **PDSCH-TimeDomainresourceAllocation** (click 6.18.8) that has a **reciprocal name** and **reciprocal semantics**

### 7.15.6.1 PUSCH-TimeDomainResourceAllocation parameters

This **data type** does **not** have a table of **field descriptions** in TS 38.331. However, the table "*PUSCH-TimeDomainResourceAllocationList field descriptions*" contains a **field description** of **all** the **parameters** of **this data type**, and the **corresponding parameters** do **not** belong to **PUSCH-TimeDomainResourceAllocationList**. It is then assumed that the said **field descriptions** correspond to the **parameters** of **this data type**. **Exactly** the same **happens** with **data type** **PUSCH-Allocation-r16** (click 7.15.7).

Its **three parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- **k2** **INTEGER(0..32)**: optional, common **or** dedicated

  It is the **physical layer parameter** "$K_2$" (refer to clause 6.1.2.1 of TS 38.214). ***If*** it is **absent**, ***then***, the **UE** will apply a **value** that depends on what is the **PUSCH subcarrier spacing**:

  - For **15** or **30 kHz**, apply **value** "**1**".
  - For **60 kHz**, apply **value** "**2**".
  - For **120 kHz**, apply **value** "**3**".

Notice that there is a parameter **extendedK2-r17**, with a **similar name**, an **extended range** "0..128" and **exactly** the **same semantics**, from **PUSCH-Allocation-r16** (click 7.15.7). ***If*** the said parameter **extendedK2-r17** is **present**, ***then***, this parameter **k2** is ignored.

- **mappingType ENUMERATED {typeA, typeB}**: mandatory, common **or** dedicated

  It is the **time-domain resource allocation mapping type** (click 7.11.3) for **PUSCH DM-RS**.

  Notice that there is a **version-different** parameter, with the **same data type** and **exactly** the **same field description**, from **PUSCH-Allocation-r16** (click 7.15.7).

  Notice that there is a **same-name** parameter, with the **same data type** and **reciprocal semantics**, from **PDSCH-TimeDomainResourceAllocation** (click 6.18.8).

  Notice that there is a **same-name** parameter, with the **same data type** and **reciprocal semantics**, from **PDSCH-TimeDomainResourceAllocation-r16** (click 6.18.9).

- **startSymbolAndLength INTEGER (0..127)**: mandatory, common **or** dedicated

  It indicates the **start symbol interval** and the **quantity** of **symbol intervals** of the **PUSCH transmission** encoded as a **start-and-length indicator** (**SLIV**). The network configures this parameter so that the allocation does **not cross** the **slot boundary** (refer to clause 6.1.2.1 of TS 38.214).

  Notice that there is a **version-different** parameter, with the **same data type** and **exactly** the **same field description**, from **PUSCH-Allocation-r16** (click 7.15.7).

  Notice that there is a **same-name** parameter, with the **same data type** and **reciprocal semantics**, from **PDSCH-TimeDomainResourceAllocation** (click 6.18.8).

  Notice that there is a **same-name** parameter, with the **same data type** and **reciprocal semantics**, from **PDSCH-TimeDomainResourceAllocation-r16** (click 6.18.9).

For more detail on these parameters, refer to the table "*PUSCH-TimeDomainResourceAllocationList field descriptions*" of TS 38.331.

### 7.15.6.2 PUSCH-TimeDomainResourceAllocation ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
PUSCH-TimeDomainResourceAllocation ::= SEQUENCE {
  k2 INTEGER(0..32) OPTIONAL,                                    -- Need S
  mappingType ENUMERATED {typeA, typeB},
  startSymbolAndLength INTEGER (0..127)
}
```

## 7.15.7 PUSCH-Allocation-r16 data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

Its **only parent data type** is `PUSCH-TimeDomainResourceAllocation-r16` (see within 7.15.5.1). The **only parent data type** of `PUSCH-TimeDomainResourceAllocation-r16` is `PUSCH-TimeDomainResourceAllocationList-r16` (see within 7.15.5.1). The **only parent data type** of `PUSCH-TimeDomainResourceAllocationList-r16` is `PUSCH-Config` (click 7.15.5).

**All** its parameters are **dedicated** (click 5.1.3).

**Each** of its **instances** configures **1** case of **PUSCH time domain allocation** that can be **selected** for **1 PUSCH assignment** within a **multiple PUSCH assignment**, within the corresponding **UE**'s **Uplink** (or **SUL**) **bandwidth part**.

Its **ancestor tree**, within the **Non-SI parameter group**, **up to** `PUSCH-Config` (click 7.15.5) is:

```
                          PUSCH-Config (click 7.15.5)
  pusch-TimeDomainAllocationListDCI-0-2-r16
                            PUSCH-TimeDomainResourceAllocationList-r16 (0..1)
                                                                  (click 7.15.5.1)
                              PUSCH-TimeDomainResourceAllocation-r16
                              (1..maxNrofUL-Allocations-r16) (click 7.15.5.1)
      puschAllocationList-r16 PUSCH-Allocation-r16
                                (1..maxNrofMultiplePUSCHs-r16) (click 7.15.7)
  pusch-TimeDomainAllocationListDCI-0-1-r16
                            PUSCH-TimeDomainResourceAllocationList-r16 (0..1)
                                                                  (click 7.15.5.1)
                              PUSCH-TimeDomainResourceAllocation-r16
                              (1..maxNrofUL-Allocations-r16) (click 7.15.5.1)
      puschAllocationList-r16 PUSCH-Allocation-r16
                                (1..maxNrofMultiplePUSCHs-r16) (click 7.15.7)
  pusch-TimeDomainAllocationListForMultiPUSCH-r16
                            PUSCH-TimeDomainResourceAllocationList-r16 (0..1)
                                                                  (click 7.15.5.1)
                              PUSCH-TimeDomainResourceAllocation-r16
                              (1..maxNrofUL-Allocations-r16) (click 7.15.5.1)
      puschAllocationList-r16 PUSCH-Allocation-r16
                                (1..maxNrofMultiplePUSCHs-r16) (click 7.15.7)
```

Some **relevant remarks** on this **ancestor tree** are:

- The constant `maxNrofUL-Allocations-r16` is `64`, which is the maximum **quantity** of **multi-entries** of **Uplink time-domain resource allocations** in the corresponding **list**.
- The constant `maxNrofMultiplePUSCHs-r16` is `8`, which is the maximum **quantity** of **Uplink time-domain resource allocations** in the corresponding **PUSCH multi-entry**.
- For the **ancestor tree** of `PUSCH-Config` (click 7.15.5) **up to** `CellGroupConfig` (click 3.7.3), click 7.15.3.

Notice that there is the **data type** `PUSCH-TimeDomainResourceAllocation` (click 7.15.7) that has a **similar name** and **simpler semantics**.

Notice that there is the **data type** `PDSCH-TimeDomainresourceAllocation-r16` (click 6.18.9) that has a **somehow reciprocal name** and **reciprocal semantics**.

### 7.15.7.1 `PUSCH-Allocation-r16` parameters

This **data type** does **not** have a table of **field descriptions** in TS 38.331. However, the table "*PUSCH-TimeDomainResourceAllocationList field descriptions*" contains a **field description** of **all** the **parameters** of **this data type**, and the **corresponding parameters** do **not** belong to `PUSCH-TimeDomainResourceAllocationList`. It is then assumed that the said **field descriptions** correspond to the **parameters** of **this data type**. **Exactly** the same **happens** with **data type** `PUSCH-TimeDomainResourceAllocation` (click 7.15.6).

Its **eight parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `extendedK2-r17 INTEGER (0..128)`: optional, dedicated

  It is the **physical layer parameter** "$K_2$" (refer to clause 6.1.2.1 of TS 38.214). For a **PUSCH subcarriers spacing** of **120 kHz**, the possible values are restricted to "`0..32`".

  ***If*** [[this parameter is **absent**] **and**

  [**either** [this instance corresponds to the **first PUSCH assignment** in a **multi-PUSCH assignment** with a **single DCI message**]

  or [this instance corresponds to a **single PUSCH assignment**]]

  ***then***,

  the **UE** will apply a **value** that depends on what is the **PUSCH subcarrier spacing**:

  - For **15** or **30 kHz**, apply **value** "`1`".
  - For **60 kHz**, apply **value** "`2`".
  - For **120 kHz**, apply **value** "`3`".
  - For **480 kHz**, apply **value** "`11`".
  - For **960 kHz**, apply **value** "`21`".

  If the **PUSCH time-domain allocation list** contains **more than 1 entry**, this parameter is **mandatory** in **all** the **entries** in the **list except** in the **first** one, where it is **optional**.

  Notice that there is a parameter `k2`, with a **similar name**, a **reduced range** "`0..32`" and **similar semantics**, from `PUSCH-TimeDomainResourceAllocation` (click 7.15.6). ***If*** this parameter `extendedK2-r17` is **present**, ***then***, the said parameter `k2` is ignored.

  This parameter is subject to `Need S`.

- `length-r16 INTEGER (1..14)`: optional, dedicated

  It is the **quantity** of **consecutive symbol intervals** (within the **slot**) of the **first symbol interval** of a **PUSCH transmission scheduled** with a **DCI message** with **Format 0_1** or **0_2** (click 7.11.4.2 and refer to clause 6.1.2.1 of TS 38.214).

  In an **instance** of this **data type** that corresponds to `pusch-TimeDomainAllocationListForMultiPUSCH-r16`, this parameter is **present**.

In an **instance** of this **data type** that corresponds to `pusch-TimeDomainAllocationListDCI-0-1-r16`, ***if*** `pusch-RepTypeIndicatorDCI-0-1-r16` is **set** to `pusch-RepTypeB`, ***then***, this parameter is **present**. ***Else***, it is **absent**.

In an **instance** of this **data type** that corresponds to `pusch-TimeDomainAllocationListDCI-0-2-r16`, ***if*** `pusch-RepTypeIndicatorDCI-0-2-r16` is **set** to `pusch-RepTypeB`, ***then***, this parameter is **present**. ***Else***, it is **absent**.

This parameter is subject to `Need R`.

- `mappingType-r16` `ENUMERATED {typeA, typeB}`: optional, dedicated

  Its **data type** and **field description** are **exactly** the **same** as the ones of its **version-different parameter** `mappingType` from `PUSCH-TimeDomainResourceAllocation` (click 7.15.6).

  In an **instance** of this **data type** that corresponds to `pusch-TimeDomainAllocationListForMultiPUSCH-r16`, this parameter is **present**.

  In an **instance** of this **data type** that corresponds to `pusch-TimeDomainAllocationListDCI-0-1-r16`, ***if*** `pusch-RepTypeIndicatorDCI-0-1-r16` is **not set** to `pusch-RepTypeB`, ***then***, this parameter is **present**. ***Else***, it is **absent**.

  In an **instance** of this **data type** that corresponds to `pusch-TimeDomainAllocationListDCI-0-2-r16`, ***if*** `pusch-RepTypeIndicatorDCI-0-2-r16` is **not set** to `pusch-RepTypeB`, ***then***, this parameter is **present**. ***Else***, it is **absent**.

  This parameter is subject to `Need R`.

  Notice that there is a **version-different** parameter, with the **same data type** and **reciprocal semantics**, from `PDSCH-TimeDomainResourceAllocation` (click 6.18.8).

  Notice that there is a **version-different** parameter, with the **same data type** and **reciprocal semantics**, from `PDSCH-TimeDomainResourceAllocation-r16` (click 6.18.9).

- `numberOfrepetitions-r16` `ENUMERATED {n1, n2, n3, n4, n7, n8, n12, n16}`: optional, dedicated

  It is the **quantity** of **repetitions** for a **PUSCH transmission** that is **scheduled** with a **DCI message** with **Format 0_1** or **0_2** (refer to clause 6.1.2.1 of TS 38.214). When `numberOfSlotsTBoMS-r17` (see below) is set to `2`, `4` or `8` (i.e., **transport block** processing over multi-slot (**TBoMS**) **PUSCH** is **enabled**), it indicates the **quantity** of **repetitions** of a **single TBoMS**.

  In an **instance** of this **data type** that corresponds to `pusch-TimeDomainAllocationListForMultiPUSCH-r16`, this parameter is **absent**.

In an **instance** of this **data type** that corresponds to `pusch-TimeDomainAllocationListDCI-0-1-r16` or to `pusch-TimeDomainAllocationListDCI-0-2-r16`, this parameter is **present**.

- `numberOfrepetitionsExt-r17 ENUMERATED {n1, n2, n3, n4, n7, n8, n12, n16, n20, n24, n28, n32, spare4, spare3, spare2, spare1}`: optional, dedicated

  It is the **quantity** of **repetitions** for a **PUSCH transmission** that is **scheduled** with a **DCI message** with **Format 0_1** or **0_2** for the case where `pusch-RepTypeIndicatorDCI-0-1-r16` or `pusch-RepTypeIndicatorDCI-0-2-r16` (respectively) is **not set** to `pusch-RepTypeB` (refer to clause 6.1.2.1 of TS 38.214).

  If it is **present**, the parameter `numberOfrepetitions-r16` (see previous bullet) is **ignored** for the case of `pusch-RepTypeA`.

  In an **instance** of this **data type** that corresponds to `pusch-TimeDomainAllocationListForMultiPUSCH-r16`, this parameter is **absent**.

  In an **instance** of this **data type** that corresponds to `pusch-TimeDomainAllocationListDCI-0-1-r16`, ***if*** `pusch-RepTypeIndicatorDCI-0-1-r16` is **not set** to `pusch-RepTypeB`, ***then***, this parameter is **optional**. ***Else***, it is **absent**.

  In an **instance** of this **data type** that corresponds to `pusch-TimeDomainAllocationListDCI-0-2-r16`, ***if*** `pusch-RepTypeIndicatorDCI-0-2-r16` is **not set** to `pusch-RepTypeB`, ***then***, this parameter is **optional**. ***Else***, it is **absent**.

  This parameter is subject to `Need R`.

- `numberOfSlotsTBoMS-r17 ENUMERATED {n1, n2, n4, n8, spare4, spare3, spare2, spare1}`: optional, dedicated

  It is the **quantity** of **slots** allocated for **transport block** processing over a **multi-slot PUSCH transmission** that has been **scheduled** with a **DCI message** with **Format 0_1** or **0_2**.

  ***If*** a **quantity** of **repetitions** "***K***" is configured by `numberOfRepetitions-r16` or by `numberOfRepetitionsExt-r17` (see above), ***then***, the **NR node** configures "***K***" and `numberOfSlotsTBoMS` of entry "***N***" in the **PUSCH time-domain allocation list** such that "$N \bullet K \leq 32$" (refer to clause 6.1.2.1 of TS 38.214).

- `startSymbol-r16 INTEGER (0..13)`: optional, dedicated

  It is the **symbol interval index** (within the **slot**) of the **first symbol interval** of a **PUSCH transmission scheduled** with a **DCI message** with **Format 0_1** or **0_2** (click 7.11.4.2 and refer to clause 6.1.2.1 of TS 38.214).

  In an **instance** of this **data type** that corresponds to `pusch-TimeDomainAllocationListForMultiPUSCH-r16`, this parameter is **present**.

In an **instance** of this **data type** that corresponds to `pusch-TimeDomainAllocationListDCI-0-1-r16`, ***if*** `pusch-RepTypeIndicatorDCI-0-1-r16` is **set** to `pusch-RepTypeB`, ***then***, this parameter is **present**. ***Else***, it is **absent**.

In an **instance** of this **data type** that corresponds to `pusch-TimeDomainAllocationListDCI-0-2-r16`, ***if*** `pusch-RepTypeIndicatorDCI-0-2-r16` is **set** to `pusch-RepTypeB`, ***then***, this parameter is **present**. ***Else***, it is **absent**.

This parameter is subject to `Need R`.

- `startSymbolAndLength-r16 INTEGER (0..127)`: optional, dedicated

  Its **data type** and **field description** are **exactly** the **same** as the ones of its **version-different parameter** `startSymbolAndLength` from `PUSCH-TimeDomainResourceAllocation` (click 7.15.6).

  In an **instance** of this **data type** that corresponds to `pusch-TimeDomainAllocationListForMultiPUSCH-r16`, this parameter is **present**.

  In an **instance** of this **data type** that corresponds to `pusch-TimeDomainAllocationListDCI-0-1-r16`, ***if*** `pusch-RepTypeIndicatorDCI-0-1-r16` is **not set** to `pusch-RepTypeB`, ***then***, this parameter is **present**. ***Else***, it is **absent**.

  In an **instance** of this **data type** that corresponds to `pusch-TimeDomainAllocationListDCI-0-2-r16`, ***if*** `pusch-RepTypeIndicatorDCI-0-2-r16` is **not set** to `pusch-RepTypeB`, ***then***, this parameter is **present**. ***Else***, it is **absent**.

  This parameter is subject to `Need R`.

  Notice that there is a **version-different** parameter, with the **same data type** and **reciprocal semantics**, from `PDSCH-TimeDomainResourceAllocation` (click 6.18.8).

  Notice that there is a **same-name** parameter, with the **same data type** and **reciprocal semantics**, from `PDSCH-TimeDomainResourceAllocation-r16` (click 6.18.9).

For more detail on these parameters, refer to the table "*PUSCH-TimeDomainResourceAllocationList field descriptions*" of TS 38.331.

### 7.15.7.2 `PUSCH-Allocation-r16` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
PUSCH-Allocation-r16 ::= SEQUENCE {
  mappingType-r16 ENUMERATED {typeA, typeB} OPTIONAL,
                                                -- Cond NotFormat01-02-Or-TypeA
  startSymbolAndLength-r16 INTEGER (0..127) OPTIONAL,
                                                -- Cond NotFormat01-02-Or-TypeA
  startSymbol-r16 INTEGER (0..13) OPTIONAL,                   -- Cond RepTypeB
  length-r16 INTEGER (1..14) OPTIONAL,                        -- Cond RepTypeB
  numberOfRepetitions-r16 ENUMERATED {n1, n2, n3, n4, n7, n8,
                                      n12, n16} OPTIONAL, -- Cond Format01-02
  ...,
```

```
    [[
    numberOfRepetitionsExt-r17 ENUMERATED {n1, n2, n3, n4, n7, n8,
                    n12, n16, n20, n24, n28, n32, spare4, spare3,
                    spare2, spare1} OPTIONAL,      -- Cond Format01-02-For-TypeA
    numberOfSlotsTBoMS-r17 ENUMERATED {n1, n2, n4, n8, spare4, spare3,
                                       spare2, spare1} OPTIONAL,      -- Need R
    extendedK2-r17 INTEGER (0..128) OPTIONAL                 -- Cond MultiPUSCH
    ]]
}
```

### 7.15.8 PUSCH-ServingCellConfig data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 7.15.3.

Its **only parent data type** is `UplinkConfig` (click 2.2.8.4).

**All** its parameters are **dedicated** (click 5.1.3).

#### 7.15.8.1 PUSCH-ServingCellConfig parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

**Each** of its **instances** contains **PUSCH dedicated parameters** for a **given UE**, that apply the **same** for **all** the **bandwidth parts** of the corresponding **serving cell**.

**HARQ-related parameters:**

- `codeBlockGroupTransmission SetupRelease {PUSCH-CodeBlockGroupTransmission}`: optional, dedicated

  If it is **present**, **PUSCH HARQ** works in **code block group** (**CBG**) **mode** (click 4.5.6.4).

  If it is **absent**, **PUSCH HARQ** works in **transport block mode**.

  It contains `maxCodeBlockGroupsPerTransportBlock`, which indicates the (**maximum**) **quantity** of **CBGs** for **each PUSCH transport block**. Its **data type** is `ENUMERATED {n2, n4, n6, n8}`, which stands for {2, 4, 6, 8}.

- `nrofHARQ-ProcessesForPUSCH-r17 ENUMERATED {n32}`: optional, dedicated
  If it is **present**, it indicates **32 PUSCH HARQ processes** for this **UE**. It was introduced in **R17**. If it is **absent**, the default value is **16**.

Notice that there are **3 reciprocal parameters** from `PDSCH-ServingCellConfig` (click 6.18.10).

For more detail on **5G HARQ**, click 4.5.6.

For more detail on these parameters, refer to the table "*PUSCH-ServingCellConfig field descriptions*" of TS 38.331.

#### 7.15.8.2 PUSCH-ServingCellConfig ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
PUSCH-ServingCellConfig ::= SEQUENCE {
  codeBlockGroupTransmission SetupRelease
                { PUSCH-CodeBlockGroupTransmission } OPTIONAL,      -- Need M
  rateMatching ENUMERATED {limitedBufferRM} OPTIONAL,              -- Need S
  xOverhead ENUMERATED {xoh6, xoh12, xoh18} OPTIONAL,              -- Need S
  ...,
  [[
  maxMIMO-Layers INTEGER (1..4) OPTIONAL,                          -- Need M
  processingType2Enabled BOOLEAN OPTIONAL                          -- Need M
  ]],
  [[
  maxMIMO-LayersDCI-0-2-r16 SetupRelease { MaxMIMO-LayersDCI-0-2-r16 }
                            OPTIONAL                               -- Need M
  ]],
  [[
  nrofHARQ-ProcessesForPUSCH-r17 ENUMERATED {n32} OPTIONAL,        -- Need R
  uplinkHARQ-mode-r17 SetupRelease { UplinkHARQ-mode-r17 } OPTIONAL  -- Need M
  ]]
}
```

### 7.15.9 DMRS-UplinkConfig data type

Its **only parent data type** is PUSCH-Config (click 7.15.5). For the **ancestor tree** of PUSCH-Config **up to** CellGroupConfig (click 3.7.3), click 7.15.3.

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

**All** its parameters are **dedicated** (click 5.1.3).

**Each** of its **instances** configures a **UE** with the configuration for **either PUSCH DM-RS mapping type** typeA or typeB (see PUSCH-Config **parameters**, click 7.15.5.1).

#### 7.15.9.1 DMRS-UplinkConfig parameters

**All** its **parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- dmrs-AdditionalPosition ENUMERATED {pos0, pos1, pos3}: optional, dedicated

  Its value is used to determine the **Uplink symbol intervals** that **contain PUSCH DM-RS**. If it is **absent**, the **UE** applies the value pos2. For more detail, 6.4.1.1.3 of TS 38.211.

- dmrs-Type ENUMERATED {type2}: optional, dedicated

  It is the **DM-RS frequency-domain configuration type** to be used for **PUSCH** (refer to clause 6.4.1.1.3 of TS 38.211). If it is **present**, the **UE** uses **UDM-Type 2**. If it is **absent**, the **UE** uses **UDM-Type 1**.

- maxLength ENUMERATED {len2}: optional, dedicated

  It indicates the **maximum quantity** of **front-load PUSCH DM-RS symbols**. In other words, it indicates the **quantity** of **consecutive DM-RS symbols** for each **DM-RS position**. If it is **absent**, **single-symbol DM-RS** is used. If it is **present**, **single-symbol DM-RS** or **double symbol DM-RS** is used as indicated by the "***Antenna ports***" field of the **scheduling DCI message** (click 6.9). For more detail, refer to 6.4.1.1.3 of TS 38.211.

- `phaseTrackingRS PTRS-UplinkConfig`: optional, dedicated

  It configures the **PUSCH phase tracking reference signals** (**PT-RS**). If it is **absent**, the **UE must not** transmit **PUSCH PT-RS** (refer to clause 6.2.3 of TS 38.214).

- `transportPrecodingDisabled SEQUENCE`: optional, dedicated

  It is a `SEQUENCE` of the following parameters:

  - `dmrs-Uplink-r16 ENUMERATED {enabled}`: optional, dedicated

    It indicates whether **low PAPR PUSCH DM-RS** is used (refer to 6.4.1.1.1 of TS 38.211).

  - `scramblingID0 INTEGER (0..65535)`: optional, dedicated

  - `scramblingID1 INTEGER (0..65535)`: optional, dedicated

    These two parameters are the **PUSCH DM-RS scrambling initialization 0** and **1** (respectively) when **transform-precoding** (click 4.12.12) is **disabled** (i.e., **PUSCH** uses **CP-OFDMA**). If they are **absent**, the **UE** uses the **physical cell identity** (**PCI**, click 3.5.3.2) of the **cell**. For more detail on these **scrambling ids**, refer to clause 6.4.1.1.1 of TS 38.211.

- `transportPrecodingEnabled SEQUENCE`: optional, dedicated

  It is a `SEQUENCE` of the following parameters:

  - `nPUSCH-Identity INTEGER(0..1007)`: optional, dedicated

    It contains the value of variable "$N_{ID}^{PUSCH}$" that is required to generate the contents of **PUSCH DM-RS** when **transform-precoding** is **enabled**. If it is **absent**, the **UE** uses the **physical cell identity** (**PCI**, click 3.5.3.2) of the **cell**. For more detail on this, refer to clause 6.4.1.1.1 of TS 38.211.

  - `sequenceGroupHopping ENUMERATED {disabled}`: optional, dedicated

    It **disables group-hopping** for **PUSCH transmission except** for **Msg3**, (click 9.3.4). It **overrides** the configuration from the corresponding `PUSCH-ConfigCommon` (click 7.15.4) **instance**.

    If it is **absent**, the **UE** uses for **PUSCH** the **same hopping mode** as for **Msg3**.

    It can **only** be **present** for **PUSCH DM-RS** transmission when [**transform-precoding** is **enabled**] and [**group-hopping** is **enabled** with `groupHoppingEnabledTransformPrecoding` from `PUSCH-ConfigCommon`].

  - `sequenceHopping ENUMERATED {enabled}`: optional, dedicated

    It **enables sequence hopping** for a **PUSCH DM-RS transmission** with **transform-precoding other** than **Msg3** (**sequence hopping** is **always disabled** for **Msg3**).

    If it is **absent**, **sequence hopping** for **PUSCH DM-RS transmission** with **transform-precoding** is **disabled**.

The **NR node** does **not simultaneously** configure **group hopping** and **sequence hopping** (refer to clause 6.4.1.1.2 of TS 38.211).

- `dmrs-UplinkTransformPrecoding-r16 DMRS-UplinkTransformPrecoding-r16`: optional, dedicated

  It indicates that **low PAPR DMRS** should be used for **PUSCH** with **π/2-BPSK modulation**, as specified in clause 6.4.1.1.2 of TS 38.211.

  If it is **absent**, **low PAPR DMRS** should **not** be used for **PUSCH** with **π/2-BPSK modulation**.

  It can **only** be **present** if `tp-pi2BPSK` is **present** in the corresponding **instance** of `PUSCH-Config` (click 7.15.5).

For more detail on these parameters, refer to the table "*DMRS-UplinkConfig field descriptions*" of TS 38.331.

### 7.15.9.2 `DMRS-UplinkConfig` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
DMRS-DownlinkConfig ::= SEQUENCE {
  dmrs-Type ENUMERATED {type2} OPTIONAL,                          -- Need S
  dmrs-AdditionalPosition ENUMERATED {pos0, pos1, pos3} OPTIONAL, -- Need S
  phaseTrackingRS SetupRelease {PTRS-UplinkConfig} OPTIONAL,      -- Need M
  maxLength ENUMERATED {len2} OPTIONAL,                           -- Need S
  transformPrecodingDisabled SEQUENCE {
    scramblingID0 INTEGER (0..65535) OPTIONAL,                    -- Need S
    scramblingID1 INTEGER (0..65535) OPTIONAL,                    -- Need S
    ...,
    [[
    dmrs-Uplink-r16 ENUMERATED {enabled} OPTIONAL                 -- Need R
    ]]
  } OPTIONAL,                                                     -- Need R
  transformPrecodingEnabled SEQUENCE {
    nPUSCH-Identity INTEGER(0..1007) OPTIONAL,                    -- Need S
    sequenceGroupHopping ENUMERATED {disabled} OPTIONAL,          -- Need S
    sequenceHopping ENUMERATED {enabled} OPTIONAL,                -- Need S
    ...,
    [[
    dmrs-UplinkTransformPrecoding-r16 SetupRelease
                    {DMRS-UplinkTransformPrecoding-r16} OPTIONAL  -- Need M
    ]]
  } OPTIONAL,                                                     -- Need R
  ...
}
```

## 7.16 Sounding reference signals (SRS)

**Sounding Reference Signals** (**SRS**) are **transmitted** by the **UE** to help the **gNB** measure the **uplink radio channel**.

Some of the **specific purposes** of **SRS** are.

- **Uplink channel estimation**

  Allows the **NR node** to perform **link adaptation**.

- **Uplink scheduling**

  Allows the **NR node** to select the appropriate **modulation**, **coding**, and **MIMO layers**.

- **Uplink beam management**

  Allows the **NR node** to perform **Uplink beam selection** and **beamforming**, which are particularly important in **TDD bands**.

- **Uplink Reciprocity in TDD**

  Assists the **NR node** in **Downlink beamforming** based on the **Uplink channel measurements**.

- **Mobility Management**

  Assists the **NR node** in **interference management** and **handover decisions**.

**SRS** can be **periodic**, **semi-persistent**, or **aperiodic**.

**SRS** are based on the **pseudo random binary sequence** "***c***()" click 11.20.1.

For more detail on **SRS**, refer to clause 6.4.1.4 of TS 38.211.

# 8. NR initial access procedure

The **initial access** is the **procedure** carried out by **each UE** to gain **access** to the **mobile network** when the **UE** has **no context knowledge** (**band**, **cells**, …) about the **network**. **Initial access** is typically carried out when the **UE** is **powered up** or when the **UE lost** its **serving cell** (e.g., the **UE moved** to another **cell** while in **RRC_IDLE** or in **RRC_INACTIVE**, click 5.8.2). **Initial access finishes** when the **UE** has obtained in **common** and **dedicated configuration** (`RRCSetup`, click 5.8.3.1) and is in the **RRC_CONNECTED state**.

**Initial access** comprises the following **3 sequential procedures**:

1. **Cell selection procedure** (click 8.1)
2. **Initial random access procedure** (click 8.10)
3. **UE configuration and RRC connection establishment** (click 8.11)

An **overview** of **each** of these **procedures** is now presented:

1. **Cell selection procedure** (click 8.1)

   The **UE** acquires the **physical cell identity** (**PCI**, click 5.3), the **PBCH System Information** (click 5.3.2), **beam selection** (click 6.1.2.3), the `MIB` (click 5.3.3) parameter group, and the `SIB1` (click 5.3.4) parameter group, **all** of them for its **primary cell** (click 3.2.2). The **UE** may also **possibly** acquire **some other SIBs** and **posSIBs** (click 5.3.5) parameter groups for its **primary cell**.

   The **UE also** gets the **Downlink common configuration** and its **Downlink carrier synchronization** (**frame**, **subframe**, **slot** and **symbol**) for the **initial bandwidth part** in its **primary cell**.

   Upon completion of the **cell selection procedure**, the **UE** is said to have **camped** in a **cell**. The **UE** can **receive PDCCH** and **PDSCH** information over the **initial bandwidth part** of its **primary cell**. The **UE** can also **transmit** through **PUSCH**, but in a very **restricted** way (**only** for **random access**), because it does **not** have its **C-RNTI**. The **UE** still **cannot** transmit **PUCCH**. Notice that the **network** is still **unaware** of the **presence** of the **UE**.

   The **UE** passes to **stage 2**.

2. **Initial random access procedure** (click 8.10)

   The **UE** acquires the **Uplink common configuration** and its **Uplink carrier synchronization**, makes itself known to the **NR node** and gets a **unique identifier** (the **C-RNTI**) within the **cell**. The **C-RNTI** is assigned to the **UE** by the **NR node**.

   A **random access procedure** (click 9) may be **also** carried out in **several circumstances** (e.g., **handover**, **beam failure recovery**, …) in **addition** to **initial access**.

   Upon completion of the **random access procedure**, the **network** is **aware** of the **presence** of the **UE** and the **UE** has its **unique identifier** (**C-RNTI**) within its **primary cell**. The **UE** can now **transmit** and **receive** over the **initial bandwidth part** of its **primary cell**. The **UE** has also started the **RRC connection establishment** by sending a `RRCSetupRequest` (click 5.8.5.10) as **Msg3** (click 9.3.4.2).

The **UE** passes to **stage 3**.

3. **UE configuration and RRC connection establishment** (click 8.11)

   The **UE** gets its **Uplink** and **Downlink dedicated configuration** (i.e., its **Non-SI parameters**, click 5.3.6) for **all** its **cells** and to enter the **RRC_CONNECTED** (click 5.8.2) state.

   Upon completion of the **UE configuration** and **RRC connection establishment**, the **UE** can **transmit** and **receive** in **all** its **bandwidth parts** (click 2.2.7), in **all** its **cells** (if **carrier aggregation**, click 10.2) and can be **connected** to **2 network nodes** (if **dual connectivity**, click 10.3).

   The **UE** has now successfully **completed** its **initial access**.

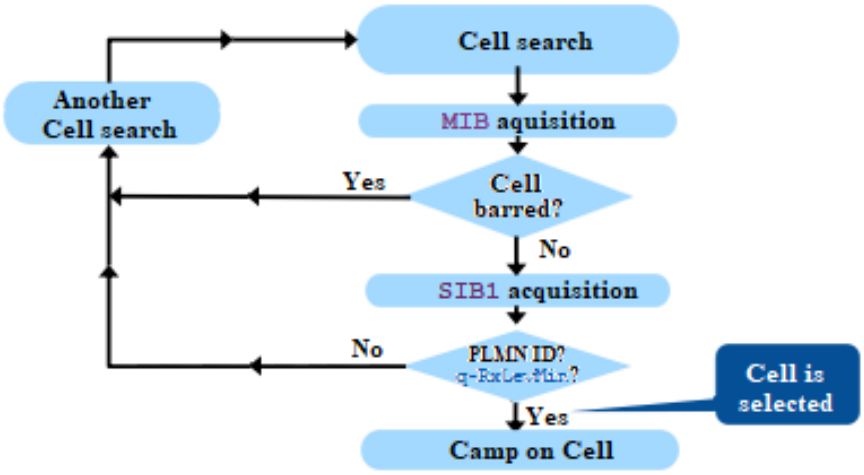
For the **evolution** of **bandwidth part usage** along **initial access**, click 2.2.7.4.

The **initial access procedure** is sometimes called the "**initial attach procedure**".

## 8.1 Cell selection procedure

**Cell selection** is the procedure the **UE** uses to **find** a **valid serving cell**. When the **cell selection procedure** is **successfully completed**, the **UE** is said to be "**camped on a cell**".

The **cell selection** procedure **usually** requires several repeated **attempts** until it is successfully completed. The **UE** performs **attempts** in **different frequencies** of the **synchronization raster** (click 6.1.2.1) until the procedure is **successfully completed**. The following figure shows an **overview** of the **cell selection** procedure, showing some events that may lead to an **unsuccessful cell selection attempt**. An **unsuccessful attempt** causes the **UE** to start a **new cell selection** attempt, doing another **cell search** in a **different frequency** from the **synchronization raster**:




**Each cell selection attempt** comprises the following **steps**:

1. **Cell search**: getting **PSS** and **SSS**
2. Getting **PBCH system information**, **beam selection** and `MIB`
3. Getting **CSS-Type 0** and **CORESET#0**
4. Getting the **PDCCH DCI messages**
5. Getting **PDSCH** and **System Information Block 1** (`SIB1`)
6. Getting the **CRB ranges**.

7. Getting the **initial bandwidth parts** and **carriers**.

An **overview** of **each** of these **steps**, within **each cell selection attempt**, is now presented:

1. **Cell search**: getting **PSS** and **SSS**

   **Cell search** consists of **finding** and **decoding** the **Primary Synchronization Signal** (**PSS**) and the **Secondary Synchronization Signal** (**SSS**). **Both** the **PSS** and the **SSS** span **127 contiguous subcarriers** over **1 symbol interval**.

   The **PSS** and the **SSS jointly** provide **frequency alignment**, **time synchronization** and **power measurements** (all of them for **Downlink**), plus the **physical cell identity** (**PCI**, click 3.5.3.2) of the **UE**'s **primary cell**.

   For more detail on **cell search**, click 8.2.

2. Getting **PBCH system information**, **beam selection** and **MIB**

   Locate and decode the **Physical Broadcast Channel** (**PBCH**). The **PSS/SSS** and **PBCH Block** (i.e., the **SSB**, click 6.1) spans **20 RBs** (**240 subcarriers**) over **4 symbol intervals**. **PBCH** contains the **PBCH system information** (click 5.3.2) and the **Master Information Block** (**MIB**, click 5.3.3). This information allows the **UE** to:

   - Perform **beam selection** (click 6.1.2.3)

     In case that the **NR node** is performing **Tx beam sweeping**, the **UE** has to perform **beam selection**.

   - Determine if the cell is **barred**

     This is indicated by **cellBarred** from **MIB** (click 8.12.3). If the cell is **barred**, the **UE** starts a **new cell search attempt** in another frequency from the synchronization raster.

   - Achieve **Downlink synchronization**

     The **UE** achieves **Downlink frame**, **subframe** and **slot synchronization**. This is achieved with **systemFrameNumber** from **MIB**, plus **PBCH system information** parameters **system-frame-number**, **half-frame-flag** and **ssb-time-domain-index** (click 6.4.1).

   - Allow **beam selection** (click 6.1.2.3)

     **Each beam** is identified with a **unique value** of the **PBCH system information** parameter **ssb-time-domain-index**.

   - Determine if intra-cell frequency reselection is enabled with **intraFreqReselection** from **MIB**.

   For more detail on **getting PBCH system information** and **MIB**, click 8.3.

3. Getting **CSS-Type 0** and **CORESET#0**

   **Common Search Space** (**CSS**) **Type 0** (click 6.8.6) provides the **multiplexing pattern** and the **PDCCH monitoring occasions**.

Getting **CORESET#0** location requires **all** of the following parameters:

- `systemFrameNumber`, `ssb-SubcarrierOffset` (click 8.7.1), `subCarrierSpacingCommon` and `pdcch-ConfigSIB1`, **all** of them from `MIB` (click 8.12.3).
- **system-frame-number**, **half-frame-flag** and **ssb-time-domain-index**, **all** of them from the **PBCH system information** (click 5.3.2).

Getting **CSS-Type 0** and **CORESET#0** requires `MIB`.

For more detail on **getting CSS-Type 0** and **CORESET#0** click 8.4.

4. Getting the **PDCCH DCI messages**

   Receive in **each** and **every PDDCH monitoring occasion** that contains **payload REs**, for a **possible incoming DCI message** (click 6.9).

   This **step ends** when the **UE** receives a **DCI message** that **schedules** `SIB1`.

   For more detail on **getting** the **PDCCH DCI messages**, click 8.5.

5. Getting **PDSCH** and **System Information Block 1** (`SIB1`)

   Decode the **physical Downlink shared channel** (**PDSCH**) (click 8.6) **payload REs** that are **scheduled** by the **DCI message** in the **previous step**. This requires knowing the location of the **PDSCH DM-RS**, which is indicated by `dmrs-TypeA-Position` from `MIB` (click 8.12.3).

   This allows the **UE** to get the **System Information Block 1** (`SIB1`, click 5.3.4). The **UE** checks if `SIB1` parameters `plmn-identityList` and `q-RxLevMin` are correct. In case that **either** (or **both**) is(are) **not correct**, the **UE** starts a **new cell search** attempt in **another frequency** from the **synchronization raster**.

   `SIB1` provides the **UE** with the **common parameters** for its **primary cell**, including the **Downlink** and **Uplink carriers** and **initial bandwidth parts**. It also provides the **UE** with the **common parameters** to **run** the **initial random access procedure** (click 8.10) in its **primary cell**.

   For more detail on **getting** the **PDSCH and System Information Block 1** (`SIB1`), click 8.6.

6. Getting the **CRB ranges**

   `SIB1` allows the **UE** to determine **Point A** (click 2.2.4.3) and the **Downlink** (or **TDD**) **CRB ranges** (click 2.2.4).

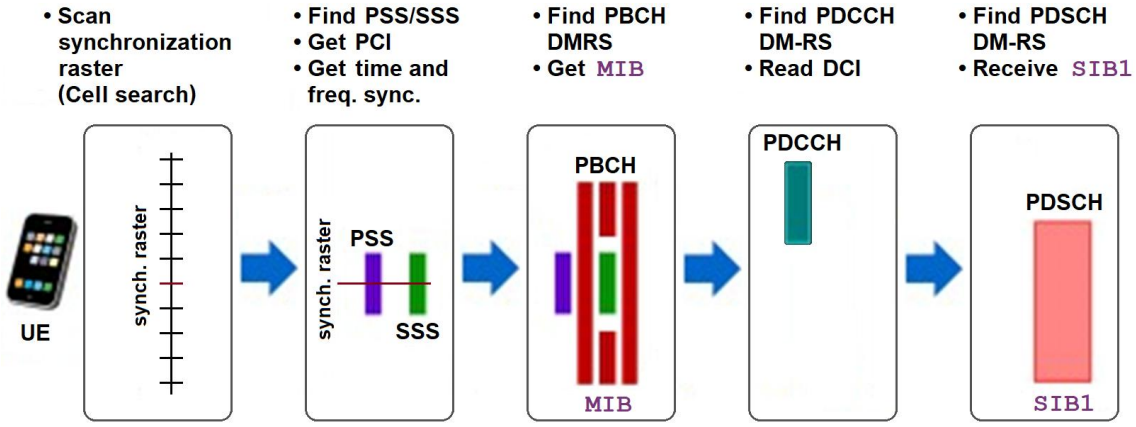
   For more detail on **getting** the **CRB ranges**, click 8.7.

7. Getting the **initial bandwidth parts** and **carriers**

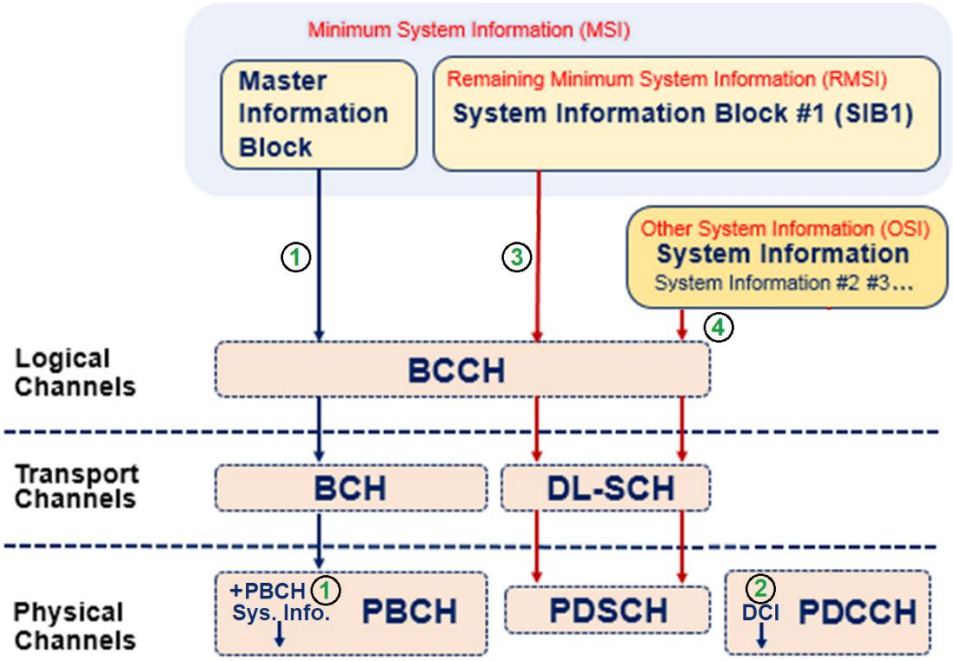
   `SIB1` allows the **UE** to determine the **Downlink** and **Uplink initial bandwidth parts** and the configuration of the **carriers**.

   For more detail on **getting** the **CRB ranges**, **initial bandwidth parts** and **carriers**, click 8.8.

The following figure summarizes the above **UE steps** in **1 cell selection attempt**:



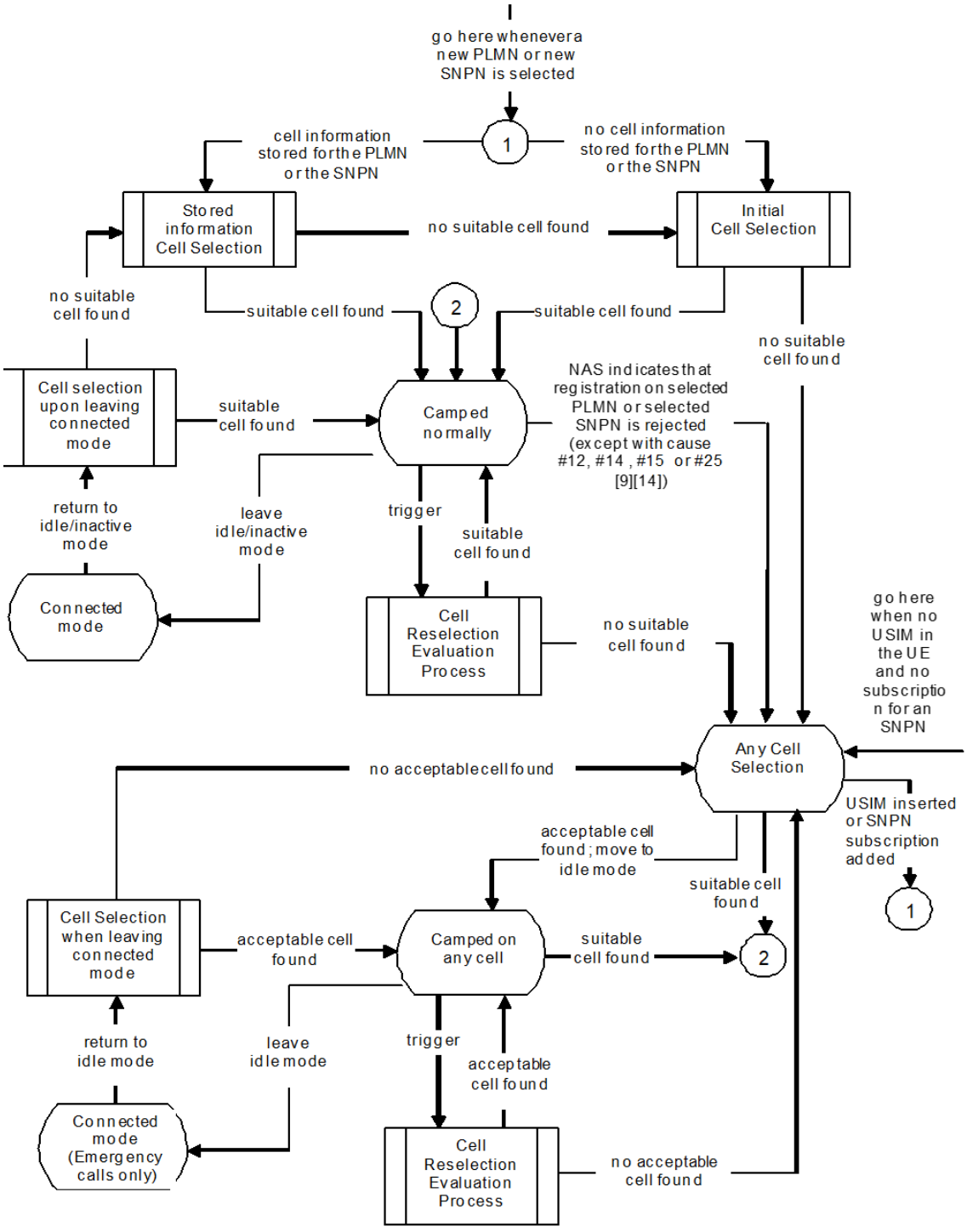

The following figure summarizes the flow of System Information sent by the **gNB** through the different channels (numbers indicate sequential steps):




Some **relevant remarks** on this figure are:

- The **PBCH system information** is also part of the **Minimum System Information**.
- The **delivery** of the **PCI** through **PSS/SSS** is **not shown**.

A fairly complete description of the **cell selection procedure** may be found in TS 38.304. The following is figure 5.2.2-1 of TS 38.304 that shows a **cell selection** state diagram:




Source web pages:

- https://www.linkedin.com/pulse/5g-cell-selection-process-bikas-singh/
- https://www.sharetechnote.com/html/5G/5G_Mib_Sib.html#MIB_SIB_Aquisition_Process

## 8.2 Cell search: getting the primary and secondary synchronization signals (PSS and SSS)

**Cell search** consists of **finding** and **decoding** the **Primary Synchronization Signal** (**PSS**) and the **Secondary Synchronization Signal** (**SSS**). **Both** the **PSS** and the **SSS** span **127 contiguous subcarriers** over **1 symbol interval**.

The **PSS** and the **SSS jointly** provide **frequency alignment**, **time synchronization** and **power measurements** (all of them for **Downlink**), plus the **physical cell identity** (**PCI**, click 3.5.3.2) of the **UE**'s **primary cell**.

The **UE** does the **cell search** procedure by **scanning** the **frequency locations** from the **5G synchronization raster** (click 6.1.2.1), until it finds a **primary synchronization signal** (**PSS**, click 6.1.2 and 6.19). The **UE scans** for **PSS** in **each reference frequency** of the **band synchronization raster**, with the **SSB subcarrier spacing** corresponding to that **band synchronization raster**. If a **band** has **more than 1 synchronization raster** (click 6.1.2.1.1) the **UE must scan** for **PSS**, for **each reference frequency** within **each** and **every synchronization raster** in that **band**.

The **UE scans each frequency location** from the corresponding **synchronization raster** for **20 ms**.

The **numerology** used to scan for the **SSB** (that includes the **PSS**) is known by the **UE**, because it is defined for **each band** (click 2.2.3.2). In case the band has **more than one** option for its **SSB** numerology, the **UE** will have to do the **scanning** for the **different options**.

After having detected the **primary synchronization signal** (**PSS**), the **UE** looks for the **secondary synchronization signal** (**SSS**) in **subsequent SSBs**, scanning over the **same 127 subcarriers**. The **SSS numerology** is the same as the one for **PSS**, that is already known.

The **physical cell identity** (**PCI**, click 3.5.3.2) is **jointly** provided by **PSS** (its **M-sequence cyclic shift** indicates the **PCI group** "$N_{ID}^{(1)}$"), and by **SSS** (its **specific Gold sequence** indicates the **PCI identity** "$N_{ID}^{(2)}$"), click 6.19.

The **SSB** may have a **different subcarrier spacing** than the **carrier**. A **large SSB subcarrier spacing** reduces the time duration of the **SSB**, which is relevant in the case the **NR node** uses **Tx beam sweeping** (click 6.1.2.3) over **many beams** with a corresponding **large quantity** of **time-multiplexed SSBs**.

Notice that the **synchronization reference frequency** is **not** centered in the **central subcarrier** of **PSS/SSS**, and rather it is **centered** in the **next-above-center subcarrier**, which is **PSS/SSS subcarrier** with **index 64**, that corresponds to **SSB subcarrier** with **index 120**.

The **cell search** procedure is based on the **SSB**, but for the specific case of **connected-state mobility** it can also be based on **channel state information reference signals** (**CSI-RS**) that have been explicitly configured to the **UE**.

The **SSB** (that includes **PSS/SSS**) is **broadcast** in **SSB-Cells** (click 3.2.1). For the **serving cells** that do **not** transmit **SSBs** (i.e., the **NSSB-Cells**), the **UE** acquires **time** and **frequency synchronization** with the serving cell based on **SSBs** from a **special cell**, or from another **secondary cell** of the **cell group** of the **serving cell** (if applicable) as described in TS 38.133.

For more detail on **cell search**, refer to clause 4.1 of TS 38.213.

Source web pages:

- https://www.linkedin.com/pulse/5g-cell-selection-process-bikas-singh/
- https://www.sharetechnote.com/html/5G/5G_PSS.html#:~:text=5G%2FNR%20%2D%20PSS,to%20the%20structure%20of%20SSB)

### 8.2.1 Example of getting PSS/SSS

This example is the **first one** in a **series** of **examples** showing the **bottom-up configuration** process of a **UE**, performed through the **initial access** procedure.

The **gNB configuration** used in this series of examples is the result of the series of examples showing the **top-down configuration** of the **gNB** (that starts in section 2.2.1.3 and finishes in section 6.1.2.2.3).

**NR node configuration**

From 2.2.1.3, this is b**and n257** in **FR2-1**, it is **TDD** and it spans from **26,500 MHz** to **29,500 MHz**. From 2.2.2.3, we have **one 100 MHz frequency block** from **27,650.08 MHz** to **27,750.08 MHz**. From 2.2.3.4, we have **one 100 MHz channel**, centered in the frequency block. The **channel NR-ARFCN** is **2,074,167**. The **channel** supports **subcarrier spacings** of **60 kHz** and **120 kHz**. From 2.2.3.6.3, the **transmission bandwidth** is **132 RBs** for **60 kHz SCS** and **66 RBs** for **120 kHz SCS**. From 2.2.4.2, subcarrier **396** from the **centered usable CRB range** is centered at **27,700.08 MHz** and its **subcarrier spacing** is **120 kHz**. From 2.2.4.6, **Point A** is placed at **27,652.56 MHz** (which is the **NR-ARFCN 2,073,375**). From 6.1.2.1.3, the **SSB subcarrier spacing** of **240 kHz**, the **SSB** is placed at **27,706.08 MHz**, the value of `subCarrierSpacingCommon` from `MIB` (click 8.12.3) is `scs30or120`, the value of `offsetToPointA` from `FrequencyInfoDL-SIB` (click 2.2.6.6) is `34` and "$k_{SSB}$" is **2**. From 6.1.2.2.3, the **SSB** temporal pattern is **case E**, an **SSB burst** is sent every **20 ms** and **each burst** contains **10 SSB instances** in the corresponding positions.

**UE configuration already acquired**

The **UE** has **just started** the **cell search**, and consequently, it has **not acquired any configuration information yet**.

**UE configuration acquired in this section**

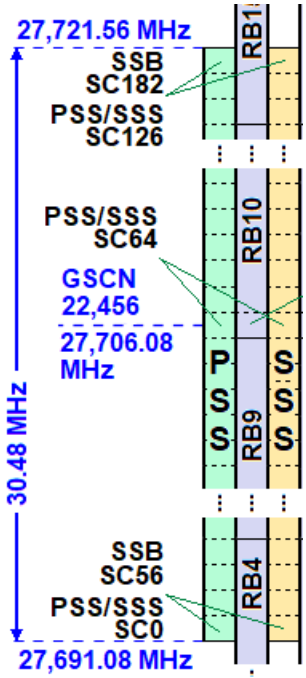
The **UE** starts the **cell search procedure** (click 8.2) by **scanning each reference frequency** in the **synchronization raster** (click 6.1.2.1). **Each frequency** is **scanned** for **20 ms**.

The **UE** detects a **burst** of **PSS signals** being sent in reference frequency **27,706.08 MHz** and **selects** the one with the **highest power**. It **also detects SSS 2 symbol intervals after each PSS**.

The **UE** now knows the **SSB subcarrier spacing**, which is **240 kHz**.

The **UE** now knows that there is a cell in band **n257** (because of the reference frequency where the **PSS/SSS** is located). It also knows the **Physical Cell Identity** (**PCI**, click 3.5.3.2), jointly conveyed by the **PSS** and **SSS** sequences. The **PCI** allows the **UE** to distinguish between incoming **PSS/SSS** sent by a given **gNB** (in a burst), from other possible incoming **PSS/SSS** sent by other **gNB**s.

The following figure shows the **PSS** and the **SSS** (**2 SSB symbol intervals** later) received by the **UE**:




Some **relevant remarks** on this figure are:

- **Subcarrier index 64** from **PSS/SSS** from a **Cell-defining SSB** (click 6.1.3) **must** be **centered** in a **synchronization frequency** (click 6.1.2.1). In this example, the said **synchronization frequency** is **27,706,08 MHz** (**GSCN 22,456**). Notice that the **central subcarrier** of **PSS/SSS** (that has **subcarrier index 63** within **PSS/SSS** and **119** within **SSB**) in **not centered** in the **synchronization raster** reference frequency.
- The **PSS/SSS bandwidth** is **always 127 subcarriers**, which for the **240 kHz SSB subcarrier spacing** in this example results in 127•.24 = **30.48 MHz**.
- The figure is drawn to **frequency scale**, where each **contiguous vertical pixel** is **15 kHz**. When there is a line between **2 frequency elements** (e.g., between 2 **RBs**), **half** of the line (**7.5 kHz**) belongs to **each frequency element**.

This example continues in 8.3.1.

## 8.3 Getting PBCH System Information, beam selection and MIB

The **frequency** and **time** location of **PBCH** in relation to **PSS** and **SSS** (which are already known) is **fixed**. The **PBCH numerology** is the same as the one for **PSS** and **SSS** (which are already known).

**PBCH** contains the **PBCH System Information** (click 5.3.2) and the **Master Information Block** (**MIB**, click 5.3.3). The **PBCH system information** and **MIB** are sent in **all** the **payload Resource Elements** of the **Physical Broadcast Channel** (**PBCH**, click 6.2).

The **UE** now has the **PBCH system information** and **MIB**, and it uses them as follows.

**Beam selection**

The **UE** does **beam selection** (click 6.1.2.3) based on **ssb-time-domain-index** (click 6.4.1) from **PBCH system information** (click 5.3.2). This parameter identifies the **beam** in which this **SSB** has been sent. This parameter has **2 bits** (up to **4** beams) in **FR1-1** bands, **3 bits** (up to **8** beams) in **FR1-2** bands and **6 bits** (up to **64** beams) in **FR2** bands. The **UE only** performs **beam selection** in case that the **gNB** is doing **Tx beam sweeping**.

**Cell Barred**

The **UE** knows if the **cell** is **barred** based on `cellBarred` (1 bit) from `MIB` (click 8.12.3). If the **cell** is **barred**, this **cell selection** attempt is **stopped**, and a **new cell selection attempt** is initiated in **another reference frequency** from the **synchronization raster**. A **cell** is usually **barred** because it is a **5G** cell in **Non-Standalone** mode, or because the **cell** is under **testing** or **maintenance**.

**Frame, subframe, slot and symbol synchronization**

The **UE** knows the **10-bit System Frame Number** of the current frame. The **6** most significant bits are from `systemFrameNumber` (from `MIB`, click 8.12.3) and the **4** least significant bits are from **PBCH system information** parameter **system-frame-number**. The **UE** also knows if the **SSB** is in the **first** or **second half-frame**, from **PBCH system information** parameter **half-frame-flag** (1 bit). The **UE** also knows the **half-frame symbol index** of the **first symbol** of **SSB** from **PBCH system information** parameter **ssb-time-domain-index**, and the **SSB burst pattern** stating its position (click 6.1.2.2.3). With this information, the **UE** achieves **Downlink frame**, **subframe**, **slot** and **symbol synchronization**. Recall that the **frame number** is **independent** of the **subcarrier spacing**, and that the **slot index** (within the **half-frame**) and the **symbol index** (within the **slot**) are computed using the **subcarrier spacing** from `subCarrierSpacingCommon` from `MIB` (click 8.12.3).

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_Mib_Sib.html#MIB_SIB_Aquisition_Process

### 8.3.1 Example of getting PBCH System Information and `MIB`

This example continues from 8.2.1.

**NR node configuration**

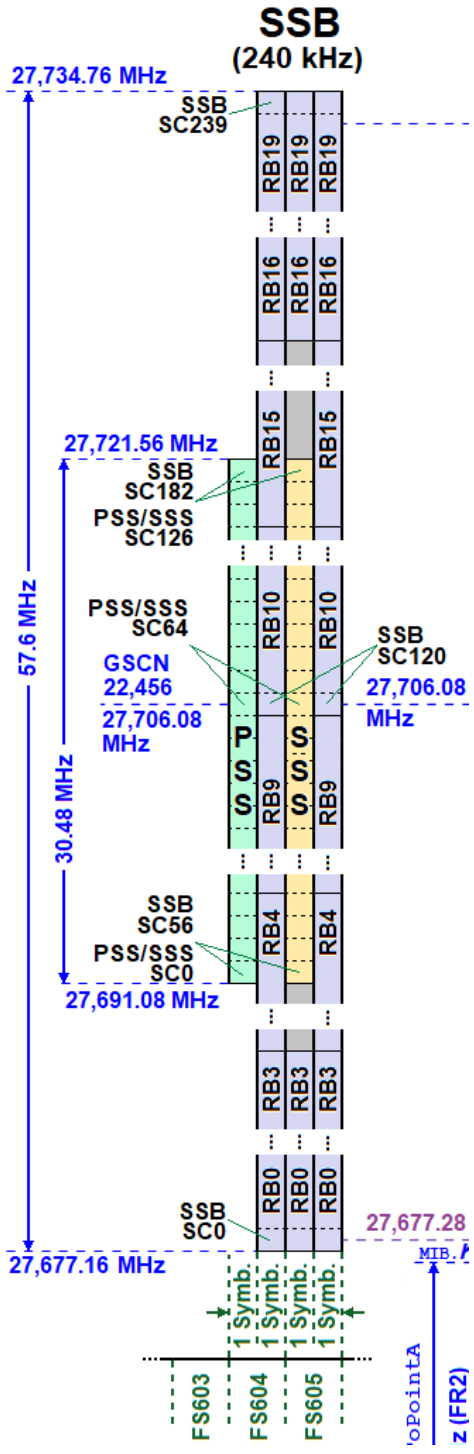
This is indicated in 8.2.1.

**UE configuration already acquired**

From 8.2.1, the **UE** knows the band (**n257**, in **FR2-1**), the location of **SSB** (**GSCN 22,456**, which is **27,706.08 MHz**), the **SSB** subcarrier spacing (**240 kHz**), the **PCI**, the location of **PBCH** (because it is in a fixed position in respect to **PSS/SSS**), and the location of **PBCH DM-RS** within **PBCH** (which is a funcion of **PCI**).

**UE configuration acquired in this section**

Based on the information the **UE** already knows, the following figure shows the **frequency** and **time location** of the **SSB**:




Some **relevant remarks** on this figure are:

- The **next-above-middle subcarrier** of **SSB**, which is **subcarrier index 120**, is **always** centered in the **scanned frequency** from the **synchronization raster**, which in this example is **27,706,08 MHz** (**GSCN 22,456**).

- The **SSB bandwidth** is **always 20 RBs**, which for the **240 kHz SSB subcarrier spacing** in this example results in 20•120•.24 = **57.6 MHz**.
- The **center frequency** of **SC0** from **SSB** is:

  $f_{SC0}^{SSB}$ = 27,706.08-120•.24 = **27,677.28 MHz**
- The **time location** (starting symbol) of the **SSB** is the **frame symbol index** number **604**, shown as "**FS604**". Notice that this frame symbol index is computed using the **PDCCH subcarrier spacing**, and **not** the **SSB subcarrier spacing**.
- The figure is drawn to **frequency scale**, where each **contiguous vertical pixel** is **15 kHz**. When there is a line between **two frequency elements** (e.g., between two **RBs**), **half** of the line (**7.5 kHz**) belongs to **each frequency element**.

The **UE** can therefore decode **PBCH** and get **MIB** and **PBCH System Information**. Notice that the **SSB always** uses the **normal cyclic prefix**. For this example, we consider that **MIB** values (click 8.12.3) are the following:

- **systemFrameNumber** (6 bits): **101011**

  It is the 6 most significant bits of **SFN**.
- **subCarrierSpacingCommon** (1 bit): it indicates **120 kHz**.
- **ssb-SubcarrierOffset** (click 8.7.1) (4 bits): **0010**

  With the values of this example, the value of this parameter (decimal **2**) is also the value of "$k_{SSB}$".
- **dmrs-TypeA-Position** (1 bit):

  Position of the **first DM-RS symbol** both for **PDSCH** and **PUSCH** when using **time-domain mapping** **typeA**. Indicates **slot symbol index** either **2** or **3**.
- **pdcch-ConfigSIB1** (8 bits): **00010011**
- **cellBarred** (1 bit): it indicates **not barred**
- **intraFreqReselection** (1 bit): not relevant for the example

For this example, we consider that the **PBCH system information** (click 5.3.2) parameter values are:

- **Only** in **FR1**, 1 most significant bit for **ssb-SubcarrierOffset** (click 8.7.1) from **MIB** (click 8.12.3).

  Since band **n257** is in **FR2**, this bit **not** used as the most significant bit for **ssb-SubcarrierOffset**, and it is **rather** used as one of the bits of the **PBCH system information** parameter **ssb-time-domain-index** (see next bullet).
- **ssb-time-domain-index** (2, 3 or 6 bits): **000111**

  Since band **n257** is in **FR2**, this parameter is **6 bits**. Its value indicates **SSB** index **7**. Recall that in section 8.3.1 we indicated that the **UE** had selected **SSB** with index **7**.
- **half-frame-flag** (1 bit): it indicates the **second half-frame**

- **system-frame-number** (4 bits): `0101`

  It is the **4** least significant bits of **SFN**.

The **SSB** is time located in **System Frame Number** "`1010110101`", in the **second** half-frame. Since the **SSB index** of the **SSB** selected by the **UE** is `7`, the corresponding **half-frame** symbol index is **44**. Since the **SSB** is in the **second** half-frame, and the subcarrier spacing of **PDCCH/PDSCH** is **120 kHz**, the **frame symbol index** where the **SSB** is located is $5 \cdot 14 \cdot 2^3 + 44 = \mathbf{604}$, as shown in the figure above.

This example continues in 8.4.1.

## 8.4 Getting CSS-Type 0 and CORESET#0

The **UE fist** determines if the **configuration** of **CSS-Type 0** is present in this **SSB**, depending on the value of "$k_{SSB}$" (click 8.7.2) as follows:

- In **FR1** bands, **CSS-Type 0 configuration** is present if "$k_{SSB} \leq 23$".
- In **FR2** bands, **CSS-Type 0 configuration** is present if "$k_{SSB} \leq 11$".

The value of "$k_{SSB}$" is obtained (click 8.7.2) out of `ssb-SubcarrierOffset` (click 8.7.1) from `MIB` (click 8.12.3).

In **Non-Standalone** (click 10.3) **5G deployments**, the `MIB` **does not contain** the **configuration** of **CSS-Type 0**, and therefore `SIB1` (click 5.3.4) **cannot** be sent through **5G NR**. In this case, the **5G** `SIB1` is received through **LTE procedures**.

If this **SSB contains** the **configuration** of **CSS-Type 0** and **CORESET#0**, the **UE** then:

1. Gets **CORESET#0 subcarrier spacing** (click 6.8.8.1).
2. Gets **CORESET#0 frequency scheduling** and **PDDCH candidate duration** (click 6.8.8.3).
3. Gets **CORESET#0 multiplexing Pattern** (click 6.8.8.4).
4. Gets **CSS-Type 0** (click 6.8.6)

The configuration in **steps 2** and **3** is obtained from `controlResourceSetZero`, while the configuration in **step 4** is obtained from `searchSpaceZero`. **Both** parameters are from `PDCCH-ConfigSIB1` (click 8.12.3.1), within `MIB` (click 8.12.3).

The **UE** needs the **configuration** of **CSS-Type 0** and of **CORESET#0** in order to receive **DCI 1_0 messages**. The **UE needs** these **DCI 1_0 messages** because they contain the **assignment** of **PDSCH REs** `SIB1` and **other SIBs and posSIBs** (click 5.3.5) are **broadcast**. **Other SIBs and posSIBs** are **broadcast** within **different instances** of the **RRC** `SystemInformation` (click 5.7.5) message.

### 8.4.1 Example of getting CORESET#0 (PDCCH) frequency location and size

This example continues from 8.3.1.

**NR node configuration**

This is indicated in 8.2.1.

**UE configuration already acquired**

From 8.2.1, the **UE** knows the band (**n257**, in **FR2-1**), the location of **SSB** (**GSCN 22,456**, which is **27,706.08 MHz**), the **SSB** subcarrier spacing (**240 kHz**), the **PCI**, the location of **PBCH** (because it is in a fixed position in respect to **PSS/SSS**), and the location of **PBCH DM-RS** within **PBCH** (which is a funcion of **PCI**). From 8.3.1, the **UE** knows the **center frequency** from **SC0** of **SSB** (**27,677.28 MHz**), and the value of the parameters from `MIB` and **PBCH System Information**.

**UE configuration acquired in this section**

As indicated in 6.8.8.1, **CORESET#0 always** uses **normal cyclic prefix**. Also indicated there, since band **n257** belongs to **FR2-1**, the value of `subCarrierSpacingCommon` (from `MIB`, 8.12.3) indicates that the subcarrier spacing of **PDCCH** (**CORESET#0**) and **PDCCH** is either **60 kHz** or **120 kHz**. In this example it indicates **120 kHz**.

As indicated in 8.4.3, **CORESET#0 size** and **frequency location** are indicated by the **4 most significant bits** of `pdcch-ConfigSIB1` (8 bits) from `MIB`. In this example, the value of these bits is **0001** (from 8.3.1). These 4 bits also indicate the multiplexing pattern used for **SSB**, **CORESET#0** and **PDSCH**. The **frequency location** of **CORESET#0** is provided relative to the location of **SSB**. The said **4 bits** are an **index** for tables **13-1** through **13-10A**, of TS 38.213. The specific table to be used depends on:

- Whether the band is for **licensed spectrum** or for **shared spectrum**.
- The minimum channel bandwidth in the band,
- The subcarrier spacing of **SBB** and **CORESET#0** (i.e., of **PDCCH**).

In this example, the subcarrier spacing of **SSB** is **240 kHz** and the one of **CORESET#0** is **120 kHz**, for **licensed spectrum**. The **UE** therefore uses **table 13.10**:

**Table 13-10: Set of resource blocks and slot symbols of CORESET for CSS-Type 0 search space when {SS/PBCH block, PDCCH} SCS is {240, 120} kHz**

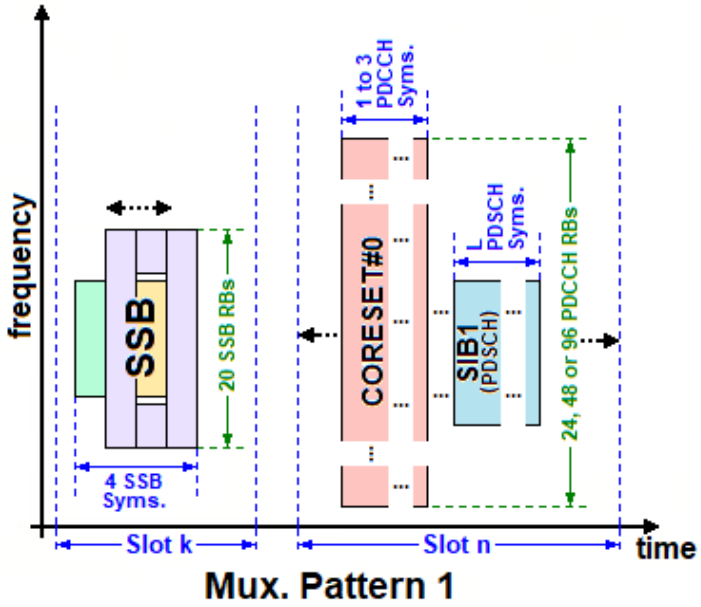
| Index | SSB and CORESET multiplexing pattern | Quantity of RBs $N_{RB}^{CORESET}$ | Quantity of Symbol intervals $N_{symb}^{CORESET}$ | Offset [RBs] |
|---|---|---|---|---|
| **0** | 1 | 48 | 1 | 0 |
| **1** | 1 | 48 | 1 | 8 |
| **2** | 1 | 48 | 2 | 0 |
| **3** | 1 | 48 | 2 | 8 |
| **4** | 2 | 24 | 1 | -41 if $k_{SSB} = 0$<br>-42 if $k_{SSB} > 0$ |
| **5** | 2 | 24 | 1 | 25 |
| **6** | 2 | 48 | 1 | -41 if $k_{SSB} = 0$<br>-42 if $k_{SSB} > 0$ |
| **7** | 2 | 48 | 1 | 49 |
| **8** | Reserved | | | |
| **9** | Reserved | | | |
| **10** | Reserved | | | |
| **11** | Reserved | | | |
| **12** | Reserved | | | |
| **13** | Reserved | | | |
| **14** | Reserved | | | |
| **15** | Reserved | | | |

We select the row with index **0001**, and from it we get:

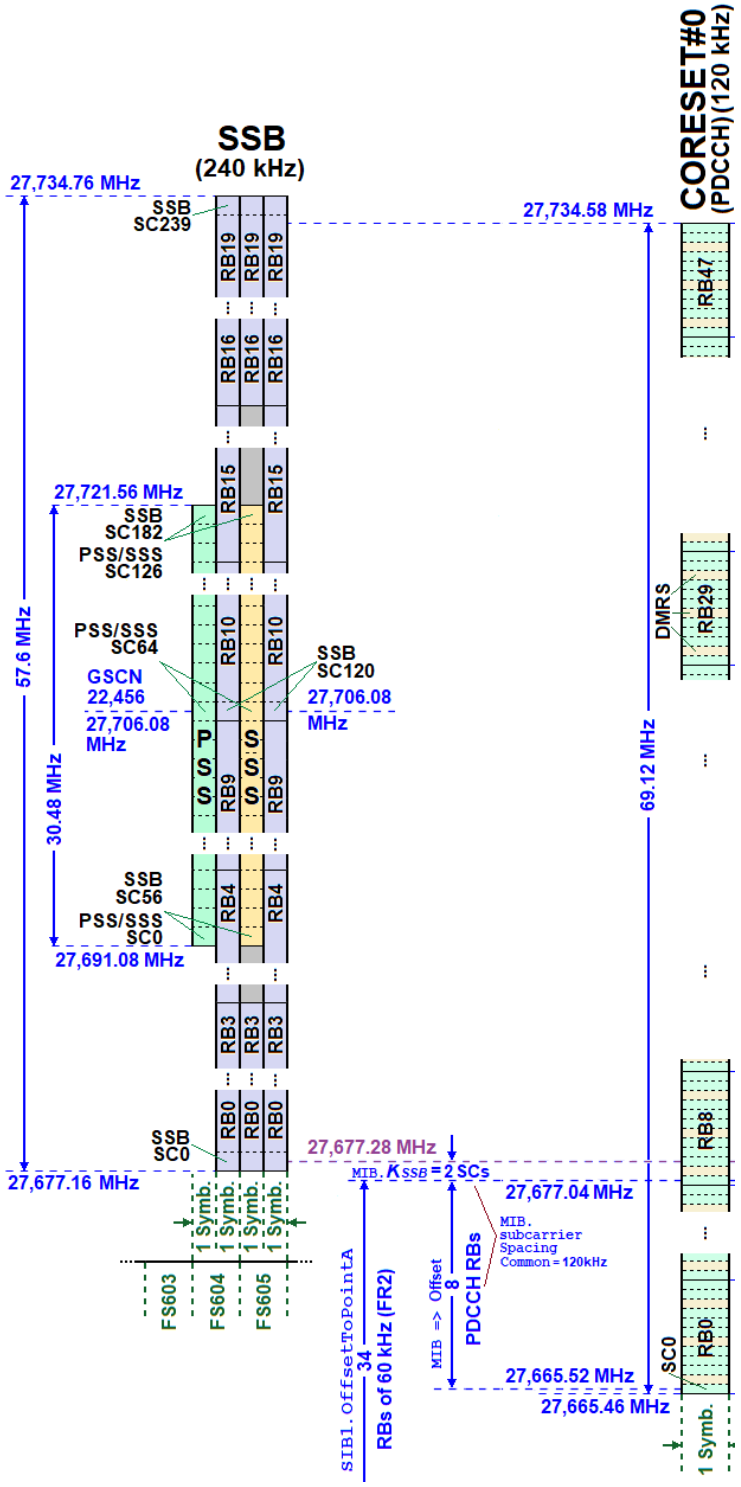
- Multiplexing Pattern: **1**
- Quantity of **RBs** of **CORESET#0**: **48**
- Quantity of **symbol intervals** of **CORESET#0**: **1**
- **Offset** in **RBs**: **8**
- The quantity of symbol intervals is irrelevant for this example.

The **UE** therefore uses **multiplexing Pattern 1**, where **CORESET#0** and **PDSCH** spans **48 RBs** with a **subcarrier spacing** of **120 kHz**, resulting in a total bandwidth of **69.12 MHz**. The value of **Offset** is **8**, measured in **RBs** of the **CORESET#0**, which in this example is 8•120 MHz = **960 MHz**. The **Offset** spans from the **lower edge** of **CORESET#0/PDSCH** to the **lower edge** of the **lowest CRB** that overlaps with the **center frequency** of **SSB subcarrier 0**.

The following figure shows, as a reminder, **multiplexing Pattern 1** (click 6.8.8.4.1):



The following figure shows all the elements the **UE** has learnt up to now, indicating their **absolute frequency location** and their **relative position** to one another:




Some **relevant remarks** on this figure are:

- The **CORESET#0 bandwidth** is (in this example) **48 RBs**, which for the **120 kHz CORESET#0 subcarrier spacing** in this example results in 48•12•.12 = **69.12 MHz**.
- The **symbol interval duration** of **CORESET#0** is **double** the one of **SSB**, because its **subcarrier spacing** is **half** the one of **SSB**.

- **Subcarrier spacing** of the **SSB** and of **CORESET#0/PDSCH** are **different**.
- The locations for **PDCCH** Demodulation Reference Signals (**DM-RS**), **always** in subcarriers **1**, **5** and **9** of **every PDCCH Resource Block**, are shown colored in mauve.
- Parameters `offsetToPointA` (click 8.7.3) from `FrequencyInfoDL-SIB` (click 2.2.6.6) and **Offset** (click 6.8.8.3) **must** be an **integer** number.
- Parameter `offsetToPointA` from `FrequencyInfoDL-SIB` is defined between **center of subcarriers**, while the value of **Offset** is defined between **edges of Resource Blocks**.
- The figure is drawn to **frequency scale**, where each **contiguous vertical pixel** is **15 kHz**. When there is a line between **two frequency elements** (e.g., between two **RBs**), **half** of the line (**7.5 kHz**) belongs to **each frequency element**.

This example continues in 8.4.2.

## 8.4.2 Example of getting CSS-Type 0

This example continues from 8.4.1.

**NR node configuration**

This is indicated in 8.2.1.

**UE configuration already acquired**

From 8.2.1, the **UE** knows the band (**n257**, in **FR2-1**), the location of **SSB** (**GSCN 22,456**, which is **27,706.08 MHz**), the **SSB** subcarrier spacing (**240 kHz**), the **PCI**, the location of **PBCH** (because it is in a fixed position in respect to **PSS/SSS**), and the location of **PBCH DM-RS** within **PBCH** (which is a funcion of **PCI**). From 8.3.1, the **UE** knows the **center frequency** from **SC0** of **SSB** (**27,677.28 MHz**), and the value of the parameters from `MIB` and **PBCH System Information**. From 8.4.1, the **UE** knows that **CORESET#0 subcarrier spacing** is **120 kHz**, its **frequency location**, its **bandwidth** is **48 RBs**, and it uses **multiplexing Pattern 1**.

**UE configuration acquired in this section**

The **UE** learns **common search space** (**CSS**) **Type 0** from `searchSpaceZero`, which is the **4 least** significant bits of the **8-bit** parameter `pdcch-ConfigSIB1` (click 8.12.3.1) from `MIB` (click 8.12.3). In this example, the said **4 bits** are `0011` (from 8.3.1). These **4 bits** are an **index** for **tables 13-11** through **13-15A**, of TS 38.213. The specific **table** to be used and how to use it depends on the **specific case**, as was detailed in 6.8.4.

Since for this example we are using multiplexing **Pattern 1** and **licensed spectrum**, the applicable **CORESET#0** case for time location is "**Case a**" (click 8.4.1). This case indicates that **CORESET#0** is present in **two slots** within every **other frame**.

Recall that **CORESET#0** subcarrier spacing is **120 kHz**, which implies "**μ=3**", Also, the band **n257** is in **FR2-1**. Therefore, within "**Case a**", in a **FR2-1** band with "**μ=3**", the **UE**

has to use **table 13-12** of TS 38.213. Since the **4** least significant bits of `pdcch-ConfigSIB1` are "`0011`" (index value `3`), the corresponding **entry** from **table 13-12** is:

| Index | *O* | Quantity of search spaces per slot | *M* | First symbol index |
|---|---|---|---|---|
| **3** | 2.5 | 2 | 0.5 | {0, if *i* is even}, {7, if *i* is odd} |

Recall from 8.3.1 that the value of the **PBCH System Information parameter** ssb-time-domain-index "***i***" is **7**.

The **UE** now determines if **PDCCH monitoring occasions** are present in **even** or **odd** frames, with the expression:

$floor((O \cdot 2^{\mu} + floor(i \cdot M)) / N_{slot}^{frame,\mu})$

$floor((2.5 \cdot 2 \cdot 3 + floor(7 \cdot 0.5)) / (10 \cdot 2^3))=0$, which is interpreted as **even**

Therefore, **PDCCH monitoring occasions** (of **CORESET#0**) are present in **even** frames.

The **UE** now determines the **frame slot index** "$n_0$" where the **first monitoring occasion** is present, with the expression:

$n_0 = (O \cdot 2^{\mu} + floor(i \cdot M)) \ mod \ N_{slot}^{frame,\mu}$

$n_0 = (2.5 \cdot 2 \cdot 3 + floor(7 \cdot 0.5)) \ mod \ (10 \cdot 2^3)=18$

Since "**μ=3**", the **frame slot index** of the **second monitoring occasion** is "$n_0$+1".

Since "***i***" is **odd**, from the table above we see that the **slot symbol index** of the **monitoring occasion** is **symbol 7**.

In summary, **two monitoring occasions** are present in every **even** frame, located in **frame slots 18** and **19**, their **slot symbol index** is **7**, and **PDCCH duration** is **1 symbol interval**.

This example continues in 8.6.3.

### 8.4.3 `MIB` parameter `pdcch-ConfigSIB1`

It is from `MIB` (click 8.12.3), which is **broadcast** by the **NR node** within **SSB**.

It is a **mandatory** and **common** parameter.

It is **represented** with **8 bits**.

Notice that the **suffix** of `pdcch-ConfigSIB1` (and of its **data type**) is "…`SIB1`" because its value is used to retrieve `SIB1`, and **not** because it is **contained** in `SIB1` (which is **not**).

Its **data type** is `PDCCH-ConfigSIB1`, which is a `SEQUENCE` that contains `controlResourceSetZero` and `searchSpaceZero`:

```
PDCCH-ConfigSIB1 ::= SEQUENCE {
  controlResourceSetZero ControlResourceSetZero,
  searchSpaceZero        SearchSpaceZero
}
```

where:

```
ControlResourceSetZero INTEGER (0..15)
SearchSpaceZero        INTEGER (0..15)
```

**Each** of these **2** parameters is **mandatory** and **common**, and it is **represented** with **4 bits**.

The **semantics** of `pdcch-ConfigSIB1` depends on the value of "$k_{SSB}$" (click 8.7.2), which is provided by `ssb-SubcarrierOffset` (click 8.7.1) from `MIB` (click 8.12.3). There are the following **3 cases**:

- ***If*** "$k_{SSB}$" indicates that **CSS-Type 0 configuration is provided** in **this SSB**,

  ***then***:

  - `searchSpaceZero` indicates the **PDCCH monitoring occasions**, as an **index** to the **corresponding table** among 13-11 to 13-15A of TS 38.213.

    Even though **search space zero** is **defined** within the **specific configuration** of the **Downlink initial bandwidth part** (**BWP#0**), it can **also** be **used** in **Downlink additional bandwidth parts** (i.e., in **BWPs other** than the **initial** one), **if** the conditions defined in clause 10 of TS 38.213 are satisfied.

  - `controlResourceSetZero` indicates the **frequency location**, the **monitoring occasion duration** and other characteristics of **CORESET#0** (click 8.4), as an **index** to the **corresponding table** among 13-1 to 13-10A of TS 38.213.

    Although **CORESET#0 may** be **configured only** in **Downlink initial bandwidth parts**, it can **also** be **used** in **Downlink additional bandwidth parts**, **if** the conditions defined in clause 10 of TS 38.213 are satisfied.

  Notice that `PDCCH-ConfigCommon` (click 6.12.4) contains **2 same-name parameters** (i.e., `searchSpaceZero` and `controlResourceSetZero`) with the **same data type** and the **same semantics** of this case.

- ***If*** "$k_{SSB}$" indicates that **CSS-Type 0 configuration is not provided** in **this SSB**, but it **is provided** in **another SSB**,

  ***then***,

  `pdcch-ConfigSIB1` provides the **GSCN** (click 6.1.2.1.2) **frequency location** of the **other SSB** that contains the **configuration** of **CORESET#0**. This **GSCN frequency location** is denoted "$N_{GSCN}^{SSB}$" and its value is:

  $$N_{GSCN}^{SSB} = N_{GSCN}^{Reference} + N_{GSCN}^{Size} \bullet N_{GSCN}^{Offset}$$

  Where:

  - "$N_{GSCN}^{Reference}$" is the **GSCN** of the current **SSB**.
  - "$N_{GSCN}^{Size}$" is **1** in **FR1** and **FR2-1** and it is **3** in **FR2-2**.

- "$N_{GSCN}^{Offset}$" is obtained by inputting "$k_{SSB}$" and the **integer** represented by the **8 bits** of `pdcch-ConfigSIB1` into table **13-16** for **FR1** or into table **13-17** for **FR2**, both of TS 38.213.

  For **FR1**, table **13-16** of TS 38.213 is:

| $k_{SSB}$ | `pdcch-ConfigSIB1` | $N_{GSCN}^{Offset}$ |
|---|---|---|
| **24** | **0, 1, …, 255** | 1, 2, …, 256 |
| **25** | **0, 1, …, 255** | 257, 258, …, 512 |
| **26** | **0, 1, …, 255** | 513, 514, …., 768 |
| **27** | **0, 1, …, 255** | -1, -2, …, -256 |
| **28** | **0, 1, …, 255** | -257, -258, …, -512 |
| **29** | **0, 1, …, 255** | -513, -514, …., -768 |
| **30** | **0, 1, …, 255** | Reserved |

  For **FR2**, table **13-17** of TS 38.213 is:

| $k_{SSB}$ | `pdcch-ConfigSIB1` | $N_{GSCN}^{Offset}$ |
|---|---|---|
| **12** | **0, 1, …, 255** | 1, 2, …, 256 |
| **13** | **0, 1, …, 255** | -1, -2, …, -256 |
| **14** | **0, 1, …, 255** | Reserved |

- ***If*** "$k_{SSB}$" indicates that **CSS-Type 0 configuration is not provided** in **this SSB**, and there is **no other SSB** that provides it within a **given GSCN** (click 6.1.2.1.2) **range**,

  ***then***, `controlResourceSetZero` and `searchSpaceZero` provide the said **GSCN range**, which is:

  from: $N_{GSCN}^{Reference}$ - `controlResourceSetZero`

  to: $N_{GSCN}^{Reference}$ + `searchSpaceZero`

  where "$N_{GSCN}^{Reference}$" is the **GSCN** of the current **SSB**.

For more detail on `pdcch-ConfigSIB1`, refer to clause 13 of TS 38.213, in particular its last part.

## 8.5 Getting the PDCCH DCI messages

Once the **UE** has obtained the **configuration** of **CSS-Type 0** and **CORESET#0** it demodulates incoming **CORESET#0 instances**, taking into account that **PDCCH DM-RS** are placed at **every PDCCH symbol interval** in **subcarriers 1**, **3** and **9** of **every PDCCH Resource Block**.

In order to get the broadcast **DCI 1_0 message** corresponding to `SIB1` location in **PDSCH**, the **UE** performs **blind decoding** (click 6.8.3.1) with **System Information RNTI** (**SI-RNTI**, click 6.9.10.1) over the **PDCCH monitoring occasions** of **CORESET#0** until it successfully decodes the said message. This **blind decoding** is done taking into account the **aggregation levels** and **maximum quantity** of **PDCCH candidates** from table 10.1-1 (click 6.8.6.2). Recall that **SI-RNTI** has a **fixed** value (**all 1s**) that is **known** to **all** the **UEs**.

## 8.6 Getting PDSCH and SIB1

The **RRC System Information Block 1** (**SIB1**, click 5.3.4) **message** is sent within a **RRC BCCH-DL-SCH-Message class** (click 5.8.5.2) which is **broadcast** by the **NR node** through **PDSCH** every **160 ms**. The **PDSCH location** containing the **SIB1 message** is indicated by a **DCI 1_0 message** masked with **SI-RNTI**. The **SIB1 message** contains a substantial quantity of **common parameters**. In order to get **SIB1**, the **UE** needs to successfully **locate** (click 8.6.1) the **PDSCH payload REs** and then **demodulate** and **decode** (click 4.12.16) them. The following sections detail how the **UE** determines the **location** and **MCS** of the **PDSCH payload REs**.

For the **SIB1** contents, click 5.7.4.

### 8.6.1 Location of PDSCH and its MCS

For the current specific case, which is a **DCI Format 1_0 message** with its **CRC** masked with **SI-RNTI** (click 6.9.5.2), located in a **Common Search Space** (**CSS**) **Type 0**, and scheduling a **PDSCH transmission** containing **SIB1**, the specific semantics of tis fields are as follows.

**DCI message field** "***Frequency domain resource assignment***" (Variable)

In the case presented in this section (a **DCI 1_0 message** masked with **SI-RNTI**, placed in a **CSS-Type 0** in **CORESET#0**) the semantics for this field is **frequency resource allocation DFA-Type 1** (without **dynamicSwitch**).

**DFA-Type 1 frequency-domain resource allocation** (click 6.14.3.3) indicates the **allocated frequency-domain resources** as a **start Virtual Resource Block** (**VRB**) and a **quantity of VRBs**, expressed as a **RIV**. In case **CORESET#0** is configured for the cell, the total **quantity** of **VRBs** "$N_{BWP,i}^{size}$" refers to the ones of **CORESET#0**, and **not** to the **active bandwidth part** (that in this case is **not** known yet to the **UE**). Therefore, for **cell search** the value of "$N_{BWP,i}^{size}$" is **24**, **48** or **96 PRBs**. Inputting these values into the formula from 6.14.3.3, the total size of the above frequency field, that contains the **RIV** value, is **9**, **11** or **13** bits, respectively.

The result is therefore the **start VRB** and the **quantity of VRBs** of **PDSCH**.

For more detail on **PDSCH frequency-domain resource assignment**, click 6.14.3.

**DCI message field** "***Time domain resource assignment***" (**4 bits**)

In the case presented in this section (a **DCI 1_0 message** masked with **SI-RNTI**, placed in a **CSS-Type 0** in **CORESET#0**) the "**PDSCH time-domain resource allocation**" to apply (from the table in 6.14.4) depends on the multiplexing **Pattern**, as follows (the referred tables are of TS.38.214).

- Multiplexing **Pattern 1**: **Default A for normal CP** (table 5.1.2.1.1-2).
- Multiplexing **Pattern 2**: **Default B** (table 5.1.2.1.1-4).
- Multiplexing **Pattern 3**: **Default C** (table 5.1.2.1.1-5).

Since the multiplexing **Pattern** was obtained in 8.4, the **UE** knows what table to apply. Inputting **both** the 4-bit value of this **DCI message field and** the value of **dmrs-TypeA-Position** from **MIB** (click 8.12.3) into the corresponding table provides:

- Slots offset "$K_o$": indicates the quantity of slots, from the one where the **DCI message** is located, where **PDSCH** is located.

- Start symbol "***S***": index of **first symbol** of **PDSCH** within the slot indicated by "***Ko***".
- Symbol length "***L***": **quantity of symbols** of **PDSCH**.

The table also provides the **PDSCH DM-RS** mapping `typeA` or `typeB`, that will be used in the next section 8.6.2.

For more detail on **PDSCH time-domain resource assignment**, click 6.14.4.

**DCI message field** "***VRB-to-PRB mapping***" (**1 bit**)

Value **0** implies **non-interleaved VRB-to-PRB** mapping, and value **1** implies **interleaved VRB-to-PRB** mapping.

If this field indicates **non-interleaved** mapping, each **VRB** is mapped to the **PRB** with its **same index**. Therefore, the **PDSCH Virtual REs** are the same as the **PDSCH Physical REs**.

If this field indicates **interleaved** mapping, for the case presented in this section (a **DCI 1_0 message** masked with **SI-RNTI**, placed in a **CSS-Type 0** in **CORESET#0**) the **VRB-to-PRB** mapping is as follows. The same **range of PRBs** of **CORESET#0** are **bundled** in **pairs** of **consecutive PRBs**[2]. **Each such bundle** is called a **PRB Bundle** (**PRBB**). The **PRBBs** are **indexed** from **0** (lowest frequency) to "$N_{bundle}$-1", where "$N_{bundle}$" is the quantity of bundles. The **VRBs** are also **bundled** in consecutive **pairs** called **VRB Bundles** (**VRBBs**). The **VRBBs** are also **indexed** from **0** to "$N_{bundle}$-1". The **highest index VRBB** is **mapped** to the **highest index PRBB**. For the other bundles, the **VRBB** with **index** "***j***" is mapped to **PRBB** with **index** "$f(j)$", where the function "$f()$" is defined as:

For **even** ***j***: $f(j) = 0, 1, 2, \ldots, C\text{-}1$
For **odd** ***j***: $f(j) = C, C+1, C+2, \ldots, N_{bundle}\text{-}2$
where $C = N_{bundle} / 2$

For more detail on **PDSCH VRB-to-PRB mapping**, click 4.12.15.

**DCI message field** "***Modulation and coding scheme (MCS)***" (**5 bits**)

This field is an **index** for one of the **MCS** tables of TS 38.214. In the present case (a **DCI 1_0 message** in a **CSS-Type 0** masked with **SI-RNTI**) the table to use is **5.1.3.1-1** (click 6.16.1). This table provides the **UE** with the **modulation order** ($Q_m$) of the **PDSCH payload REs** and the **target code rate** ($R$).

The modulation order $Q_m$ is the **quantity of bits** for **each** modulation **symbol** and allows to determine the modulation from the table in 11.2.1.

**DCI message field** "***Redundancy version***" (**2 bits**)

This field indicates the **specific LDPC redundancy bits** sent in **each** specific **HARQ** (**re**)**transmission**.

For more detail on "***Redundancy version***", click 4.5.6.6.

2 Note that the number of **PRBs** of **CORESET#0** is always **a multiple of four**, so it is **even** and "$N_{bundle}$" is also **even**.

**DCI message field "*System information indicator*" (1 bit)**

This field indicates if the **PDSCH transmission** scheduled by this **DCI 1_0 message** contains **either** `SIB1` (value "**0**") or **other SIBs or posSIBs** (value "**1**"). In the present case its value is "**0**".

### 8.6.2 Location of PDSCH payload and DM-RS REs

In order to successfully demodulate **PDSCH**, the **UE** also needs to know where the **PDSCH payload REs** and **PDSCH DM-RS REs** are, within the **PDSCH allocated REs**.

**Location of PDSCH DM-RS and PDSCH empty REs**

Regarding the **PDSCH DM-RS REs** and **PDSCH empty REs**, since the **UE** does not yet have the **PDSCH DM-RS** System Information, the following configuration applies:

- The **PDSCH DM-RS time-domain mapping** `typeA` or `typeB` was obtained from the **DCI 1_0 message** (as indicated in section 8.6.1).
- The **PDSCH DM-RS first symbol index** for `typeA` is indicated in parameter `dmrs-TypeA-Position` from `MIB` (click 8.12.3) or from `ServingCellConfigCommon` (click 3.9.6). For `typeB` it is always the **first symbol** of **PDSCH**.
- Parameter `maxLength` is **absent**. The **UE** therefore uses the default value, which is **single-symbol PDSCH DM-RS**.
- Parameter `dmrs-AdditionalPosition` is **absent**. The **UE** therefore uses the default value, which is "`pos2`". This implies that there are **3 PDSCH DM-RS symbols**, whose start symbol indexes are obtained from table **7.4.1.1.2-3** of TS 38.211. Notice that **PDSCH must** be wide enough to accommodate these **3 PDSCH DM-RS** symbols in their corresponding positions.
- Parameter `dmrs-Type` is **absent**. The **UE** therefore uses the default value, which is the **DM-RS frequency configuration DDM-Type 1**.

These **three** last **parameters** are from `DMRS-DownlinkConfig` (click 6.18.11).

- **PDSCH DM-RS subcarrier 0** is **subcarrier 0** from **CORESET#0**.

For more detail on the location of **PDSCH DM-RS REs** and **PDSCH empty REs**, click 6.14.5.

**Location of PDSCH payload REs**

The **PDSCH payload REs** are the **PDSCH allocated REs** that are **not**: **PDSCH DM-RS REs**, **PDSCH empty REs**, **PDSCH PT-RS REs**, **CSI-RS REs** and **PRS REs**.

**Location of PDSCH PT-RS REs**

Since the **UE** does **not** have **yet** the **PDSCH PT-RS** System Information, the following configuration applies:

- Parameter `phaseTrackingRS` from `DMRS-DownlinkConfig` (click 6.18.11) is **absent**. The **UE** therefore uses the default value, which is **no PDSCH PT-RS are present**.

For more detail on the location of **PDSCH PT-RS REs**, click 6.14.5.

### 8.6.3 Example of getting PDSCH and SIB1

This example continues from 8.4.2.

**NR node configuration**

This is indicated in 8.2.1.

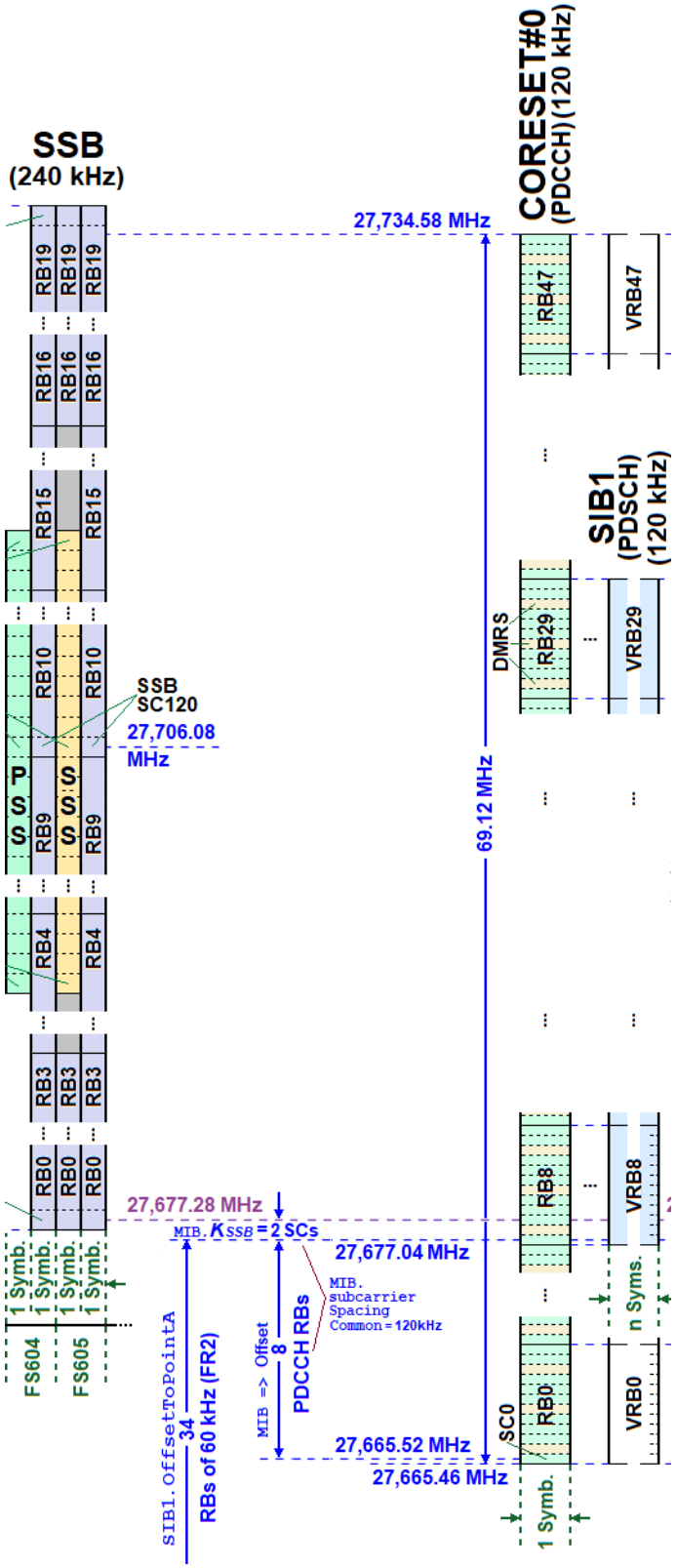
**UE configuration already acquired**

From 8.2.1, the **UE** knows the band (**n257**, in **FR2-1**), the location of **SSB** (**GSCN 22,456**, which is **27,706.08 MHz**), the **SSB** subcarrier spacing (**240 kHz**), the **PCI**, the location of **PBCH** (because it is in a fixed position in respect to **PSS/SSS**), and the location of **PBCH DM-RS** within **PBCH** (which is a funcion of **PCI**). From 8.3.1, the **UE** knows the **center frequency** from **SC0** of **SSB** (**27,677.28 MHz**), and the value of the parameters from **MIB** and **PBCH System Information**. From 8.4.1, the **UE** knows that **CORESET#0 subcarrier spacing** is **120 kHz**, its **frequency location**, its **bandwidth** is **48 RBs**, and it uses **multiplexing Pattern 1**. From 8.4.2, the **UE** knows the **search spaces for CORESET#0** and therefore it can decode **DCI 1_0 message** from **PDCCH**.

**UE configuration acquired in this section**

The **UE** decodes the **DCI 1_0 message**, and from its parameters it can locate the **PDSCH assignment** that contains **SIB1**. Since **multiplexing Pattern 1** is used, the **PDSCH assignment** is located **within**, and **referred to**, the same range of **RBs** as **CORESET#0**, that are already known to the **UE**. The **PDSCH assignment** is located in **n consecutive PDSCH symbols**, where the value of "**n**" is indicated in the **DCI 1_0 message**.

The following figure shows the **location** of **PDSCH** (same as **CORESET#0** in frequency, and with the **same subcarrier spacing**), and within **PDSCH** (in light blue), the location of `SIB1`.



Some **relevant remarks** on this figure are:

- The **RBs** of the **SSB may**, or **may not**, be **aligned** with the **CRBs**. In this example, they are **not aligned** (which is the most frequent case).
- The **RBs** of **CORESET#0** and of **PDSCH must** be **aligned** to the corresponding **CRB range**, although the **CRB ranges** are still not known to the **UE** (they will be known in the next step of this example).
- Although the **RBs** of **CORESET#0** and **PDSCH** are **aligned** to the corresponding **CRB range**, since the **CRB ranges** are still **not known** to the **UE**, the **RB indexing** for **CORESET#0** and for **PDSCH at this stage** is **specific**, and **independent** of the **CRB indexing**. This is why the **RBs** of **CORESET#0** and **PDSCH** are **indexed** from **0** to **47**.
- The **RBs** of **PDSCH** are labeled as **VRBs** (**Virtual Resource Blocks**), to highlight that interleaved **VRB-to-PRB mapping** (click 4.12.15) may be used.
- Although the figure shows that `SIB1` transmission spans from **PDSCH VRB8** to **VRB29**, this has **not** been actually **calculated**, and it may therefore not be a realistic range.
- The figure shows the **duration** of **1 symbol interval** for several physical elements to highlight that the **average symbol interval duration** is **inversely proportional** to the **subcarrier spacing**.
- The **duration** of **1 symbol** is the same for **PDCCH** (**CORESET#0**) and **PDSCH**, because both have the **same subcarrier spacing**.
- The **quantity** of **symbols** is different for **PDCCH** (**CORESET#0**) and **PDSCH**. **CORESET#0** only spans **1 symbol**, while **PDSCH** spans "**n**" **symbols** (the quantity scheduled to transmit `SIB1`).
- The figure is drawn to **frequency scale**, where **each contiguous vertical pixel** is **15 kHz**. When there is a line between **two frequency elements** (e.g., between two **RBs**), **half** of the line (**7.5 kHz**) belongs to **each frequency element**.

This example continues in 8.7.4.

## 8.7 Getting the CRB ranges

In case the **UE** is in a **SIB1-Cell** (i.e., a **cell** that **broadcasts** `SIB1`, click 5.3.4), the parameters from the **Master Information Block** (`MIB`) and from the **System Information Block 1** (`SIB1`) allow the **UE** to locate the **set** of **Downlink** (or **TDD**) **Common Resource Block ranges** (**CRB range set**, click 2.2.4.4).

The **Downlink** (or **TDD**) **CRB range set** is determined by the **Downlink** (or **TDD**) **Point A** (click 2.2.4.3).

The **Downlink** (or **TDD**) **Point A** is **jointly** obtained from "$k_{SSB}$" (click 8.7.2) and from `offsetToPointA` (click 8.7.3). Recall that:

- "$k_{SSB}$" is obtained from `ssb-SubcarrierOffset`, which is a parameter **within** `MIB` (click 5.3.3).
- `offsetToPointA` is a parameter **within** `SIB1` (click 5.3.4).

The expression for "$k_{SSB}$" is:

$$k_{SSB} = (f_{SC0}^{SSB} - (\textbf{Point A} + \texttt{offsetToPointA} \bullet 12 \bullet SCS_u)) / SCS_c$$

Which allows the **UE** to compute **Point A** as:

$$\textbf{Point A} = f_{SC0}^{SSB} - (\texttt{offsetToPointA} \bullet 12 \bullet SCS_u) - (k_{SSB} \bullet SCS_c)$$

where:

- The term "$\texttt{offsetToPointA} \bullet 12 \bullet SCS_u$" is the **frequency distance** between "$f_{SC0}^{SSB}$" and **PointA** expressed as an **integer quantity** of **RBs** with **subcarrier spacing** "$SCS_u$". Since this **frequency distance** is expressed as **integer quantity** of **RBs**, there may be a **gap** between this **frequency distance** and the **actual frequency distance**. This **gap** has a size from **0** to **11 subcarrier spacings** "$SCS_u$", which are the possible quantities **smaller** than **1 RB**.
- "$SCS_c$" is the **subcarrier spacing** of the **RBs** used to **calculate** the value of `offsetToPointA`. The value of "$SCS_c$" is indicated by `subCarrierSpacingCommon` from `MIB` (click 8.12.3).
- "$SCS_u$" is the **subcarrier spacing** of the **RBs** that are used for the **units** of `offsetToPointA`. The value of "$SCS_u$" is **15 kHz** in **FR1 bands** and **60 kHz** in **FR2 bands**. Consequently, the **units** of `offsetToPointA` are **12•15 kHz** in **FR1** and **12•60 kHz** in **FR2**.

The **UE** therefore knows **Point A**. Reference **Point A** is the **center frequency** of the **lowest subcarrier** (**SC0**) from the lowest Common Resource Block (**CRB0**) in the different **CRB ranges**. There is one **CRB range** for **each subcarrier spacing** being used. The **UE** knows **all** the **subcarrier spacings** being used from the different values of the `SIB1` parameters `subcarrierSpacing` (click 2.2.6.12), all them contained within `SIB1` parameter `scs-SpecificCarrierList` (click 2.2.6.4). The **UE** therefore knows the **Downlink** (or **TDD**) **CRB range set**.

In case the **UE** is **not** in a **PCell** (i.e., a cell that sends `SIB1`), the **UE** learns the position of **Point A** from **RRC** parameter `absoluteFrequencyPointA`.

### 8.7.1 `MIB` parameter `ssb-SubcarrierOffset`

It is from `MIB` (click 8.12.3), which is **broadcast** by the **NR node** within **SSB**.

It is a **mandatory** and **common** parameter.

Its **data type** is `INTEGER (0..15)`. Consequently, it is **represented** with **4 bits**. However, in **FR1 bands**, the value of the **4-bit** is **extended** with **1 additional most significant bit** that is taken from the **PBCH system information** (click 5.3.2).

It **indicates** the value of "$k_{SSB}$" (click 8.7.2). In a **particular case** (see further below) it **also provides** a **1-bit** value to derive the **QCL** relation between **SSBs**, with a **1-bit** value.

How `ssb-SubcarrierOffset` **indicates** the value of "$k_{SSB}$" depends **jointly** on the **frequency range** and on whether the **channel** is within **licensed spectrum** (click 2.2.2.1) or **not**:

- For [**licensed spectrum** in **FR1** and **FR2-1** bands] and for [**unlicensed spectrum** in **FR2-2** bands], `ssb-SubcarrierOffset` (with **1 additional bit** in **FR2**) **contains** "$k_{SSB}$".

- For [**unlicensed spectrum** in **FR1** bands], `ssb-SubcarrierOffset` **contains** $\overline{k}_{SSB}$, that allows to compute "$k_{SSB}$" as follows:

  ***If*** $\overline{k}_{SSB} \geq 24$

  - ***then*** $k_{SSB} = \overline{k}_{SSB}$
  - ***else*** $k_{SSB} = 2 \cdot floor( \overline{k}_{SSB} / 2 )$

    in this case, any **even** value of "$\overline{k}_{SSB}$" and the **same value** plus **1** result in the same value of "$k_{SSB}$". Consequently, the value of "$k_{SSB}$" does not depend on the **least significant bit** of `ssb-SubcarrierOffset`. For this reason, in this **specific case**, the value of the **least significant bit** of `ssb-SubcarrierOffset` is used for deriving the **QCL** relation between **SSBs**.

Because of the **bitlength** of `ssb-SubcarrierOffset` (see above), in **FR1 bands** the **range** of "$k_{SSB}$" is from `0` to `31` (**5 bits**) while in **FR2 bands** the **range** of "$k_{SSB}$" is from `0` to `15` (**4 bits**).

If `ssb-SubcarrierOffset` is **absent**, then "$k_{SSB}$" is derived from the **frequency difference** between **SSB** and **Point A**.

For more detail on how `ssb-SubcarrierOffset` is used to obtain "$k_{SSB}$", refer to clause 7.4.3.1 of TS 38.211.

For more detail on `ssb-SubcarrierOffset`, refer to clauses 4.1 and 13 of TS 38.213.

### 8.7.2 Variable "$k_{SSB}$"

The **value** of "$k_{SSB}$" indicates if the **configuration** of **common search space** (**CSS**) **Type 0** (click 8.4) is **provided** (**or not**) in the **SSB** whose `MIB` contains "$k_{SSB}$", as follows:

- In **FR1** bands where "$k_{SSB} \leq 23$", or,

  in **FR2** bands where "$k_{SSB} \leq 11$":

  **CSS-Type 0 configuration** is **provided** in this **SSB**, in parameter `pdcch-ConfigSIB1` from `MIB` (click 8.12.3).

- In **FR1** bands where "$24 \leq k_{SSB} \leq 29$", or,

  in **FR2** bands where "$12 \leq k_{SSB} \leq 13$":

  **CSS-Type 0 configuration** is **not provided** in this **SSB**, and the **UE** will look for **CSS-Type 0 configuration** in **another SSB** placed at a **specific GSCN jointly** indicated by "$k_{SSB}$" and `pdcch-ConfigSIB1` (click 8.12.3.1).

- In **FR1** bands where "$k_{SSB} = 31$", or,

  in **FR2** bands where "$k_{SSB} = 15$":

  **CSS-Type 0 configuration** is **not provided** in this **SSB**, and there is **no other SSB** providing it **within** a **specific GSCN range jointly** indicated by "$k_{SSB}$" and `pdcch-ConfigSIB1` (click 8.12.3.1). The **UE** should therefore **not search** for a **CSS-Type 0 configuration** in that **GSCN range**.

The variable name "$k_{SSB}$" stands for "***k*-stationary** Synchronization Signal and PBCH Block" or "***k*-stationary SSB**".

The value "$k_{SSB}$" is obtained from `ssb-SubcarrierOffset` (click 8.7.1) **within** `MIB` (click 5.3.3).

***If*** the **value** of "$k_{SSB}$" indicates that this **SSB** provides **CSS-Type 0**, ***then***, the value of "$k_{SSB}$" is the **quantity** of **common subcarrier spacings** "$SCS_c$" between [**Point A** + `offsetToPointA`•12•$SCS_u$] and [the **center frequency** of the **lowest subcarrier** of **SSB**]:

$$k_{SSB} = (f_{SC0}^{SSB} - (\textbf{Point A} + \texttt{offsetToPointA} \bullet \textbf{12} \bullet SCS_u)) / SCS_c$$

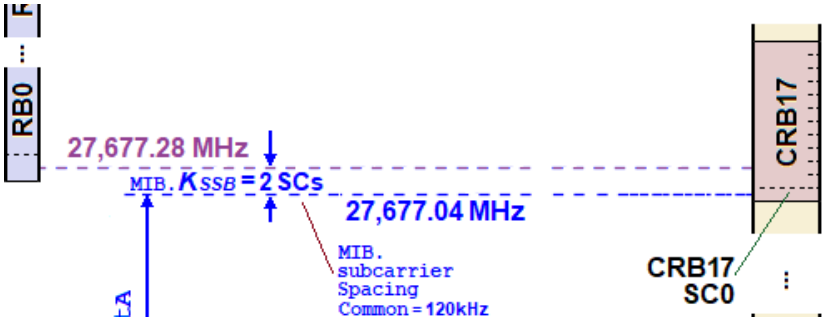
where:

- "$f_{SC0}^{SSB}$" is the **center frequency** of **subcarrier 0** of **SSB**.
- **Point A** is the frequency location of **Point A**.
- "$SCS_c$" is the **subcarrier spacing** of the **RBs** used to **calculate** the value of `offsetToPointA`. The value of "$SCS_c$" is indicated by `subCarrierSpacingCommon` from `MIB` (click 8.12.3).
- "$SCS_u$" is the **subcarrier spacing** of the **RBs** that are used for the **units** of `offsetToPointA`. The value of "$SCS_u$" is **15 kHz** in **FR1 bands** and **60 kHz** in **FR2 bands**. Consequently, the **units** of `offsetToPointA` are **12•15 kHz** in **FR1** and **12•60 kHz** in **FR2**.

The **above expression** for "$k_{SSB}$" **always** returns an **integer** number because:

- In **every band**, **every 2 consecutive channel frequencies** (click 2.2.3.3) and **every 2 consecutive synchronization frequencies** (click 6.1.2.1.1) are **spaced away** a multiple of **15 kHz** in **FR1** and of **30 kHz** in **FR2**.
- In **every band**, its **synchronization frequencies** are a **subset** of its **channel frequencies**.
- The possible values for "$SCS_c$" are **either** `15` **kHz** or `30` **kHz** in **FR1 bands**, and **either** `60` **kHz** or `120` **kHz** in **FR2-1 bands**.

The following figure shows one example where the value of "$k_{SSB}$" is **2**:




In this example, the subcarrier spacing of **SSB** is **240 kHz**. The subcarrier spacing used to **calculate** `offsetToPointA` is **120 kHz**, which in this case corresponds to value of the `SubcarrierSpacingCommon` from `MIB` (click 8.12.3). Therefore, we use the **120 kHz CRB range**. Because of the frequency we chose for **Point A** (27,652.56 MHz), the **highest frequency CRB** that **fits** up to the **center frequency** of **SC0** of **SSB** is **CRB16**. The **center frequency** of **SC0** of **SSB** is 27,677.28 MHz. The **center frequency** of **SC0** of **CRB17** is 27,677.04 MHz. The difference between both is 140 MHz. Dividing it by the subcarrier spacing used to **calculate** `offsetToPointA` we get "$k_{SSB}$ = **2**". Notice that this is the same example as the one used along section 8.1.

For an overall explanation on **UE configuration parameters** click 5. For a brief overview of **ASN.1**, click 5.9.

Source web pages:

- https://www.sqimway.com/nr_refA.php
- https://www.sqimway.com/nr_band.php
- https://5g-tools.com/5g-nr-gscn-calculator/

### 8.7.2.1 Definitions of "$k_{SSB}$" in the TS documents

There are several TS documents that describe the semantics of "$k_{SSB}$". Some of these descriptions are copied below, in case you want to go through them:

- The semantics of "$k_{SSB}$" from clause "6.2.2 Message definitions" (page 369) of TS 38.331:

  *"Corresponds to $k_{SSB}$ (see TS 38.213 [13]), which is the frequency domain offset between SSB and the overall resource block grid in number of subcarriers."*

- The semantics of "$k_{SSB}$" from clause 7.4.3.1 of TS 38.211:

  *"The quantity $k_{SSB}$ is the subcarrier offset from subcarrier 0 in common resource block NCRBSSB to subcarrier 0 of the SS/PBCH block, where NCRBSSB is obtained from the higher-layer parameter* **`offsetToPointA`***"*

- The semantics of "$k_{SSB}$" from the last part of clause "4.1 Cell search" of TS 38.213:

  *"Upon detection of a SS/PBCH block, the UE determines from MIB that a CORESET for Type0-PDCCH CSS set, as described in clause 13, is present if $k_{SSB}$ <24 [4, TS 38.211] for FR1 or if $k_{SSB}$ <12 for FR2. The UE determines from MIB that a CORESET for Type0-PDCCH CSS set is not present if $k_{SSB}$ >23 for FR1 or if $k_{SSB}$ >11 for FR2; the CORESET for Type0-PDCCH CSS set may be provided by PDCCH-ConfigCommon.*

  *For a serving cell without transmission of SS/PBCH blocks, a UE acquires time and frequency synchronization with the serving cell based on receptions of SS/PBCH blocks on the PCell, or on the PSCell, or on an SCell if applicable as described in [10, TS 38.133], of the cell group for the serving cell."*

- The semantics of "$k_{SSB}$" from clause 13 (page 213) of TS 38.213:

  *"If a UE detects a first SS/PBCH block and determines that a CORESET for Type0-PDCCH CSS set is not present, and for 24≤ $k_{SSB}$ ≤29 for FR1 or for 12≤ $k_{SSB}$ ≤13 for FR2, the UE may determine the nearest (in the corresponding frequency direction) global synchronization channel number (GSCN) of a second SS/PBCH block having a CORESET for an associated Type0-PDCCH CSS set as NGSCNReference+NGSCNSize•NGSCNOffset. NGSCNReference is the GSCN of the first SS/PBCH block, NGSCNSize=1 in FR1 and FR2-1, NGSCNSize=3 in FR2-2, and NGSCNOffset is a GSCN offset provided by Table 13-16 for FR1 and Table 13-17 for FR2. If the UE detects the second SS/PBCH block and the second SS/PBCH block does not*

*provide a CORESET for Type0-PDCCH CSS set, as described in clause 4.1, the UE may ignore the information related to GSCN of SS/PBCH block locations for performing cell search.*

*If a UE detects a SS/PBCH block and determines that a CORESET for Type0-PDCCH CSS set is not present, and for* $k_{SSB}$*=31 for FR1 or for* $k_{SSB}$*=15 for FR2, the UE determines that there is no SS/PBCH block having an associated Type0-PDCCH CSS set within a GSCN range [NGSCNReference −NGSCNStart, NGSCNReference +NGSCNEnd]. NGSCNStart and NGSCNEnd are respectively determined by* **controlResourceSetZero** *and* **searchSpaceZero** *in* **pdcch-ConfigSIB1**. *If the GSCN range is [NGSCNReference, NGSCNReference], the UE determines that there is no information for a second SS/PBCH block with a CORESET for an associated Type0-PDCCH CSS set on the detected SS/PBCH block.*

*If a UE does not detect any SS/PBCH block providing a CORESET for Type0-PDCCH CSS set, as described in clause 4.1, within a time period determined by the UE, the UE may ignore the information related to GSCN of SS/PBCH locations in performing cell search."*

### 8.7.3 SIB1 parameter offsetToPointA

It is from **FrequencyInfoDL-SIB** (click 2.2.6.6), which is **broadcast** (or **unicast**) by the **NR node** within **SIB1** (click 5.3.4).

It is a **mandatory** and **common** parameter.

Its **data type** is **INTEGER (0..2199)**.

It is the **largest integer quantity** of **RBs** (with the **common subcarrier spacing** "$SCS_c$"), that fits between **Point A** and the **center frequency** of the **lowest subcarrier** of **SSB** (click 6.1) expressed in **units** of **RBs** with **subcarrier spacing** "$SCS_u$". Therefore:

**offsetToPointA** = $int((f_{SC0}^{SSB}$ - **Point A**$) / (12 \cdot SCS_c)) \cdot (SCS_c / SCS_u)$

where:

- The **first term** "$int((f_{SC0}^{SSB}$…)" of this expression corresponds to the **textual definition** above "is the **largest integer quantity**…of **SSB**".
- The **second term** "$(SCS_c / SCS_u)$" of this expression corresponds to the **textual definition** above "expressed in **units** of RBs…", which is just a **scale conversion** factor. Since the value of "$SCS_u$" (see below) is the **minimum subcarrier spacing** in the corresponding **frequency range**, the value of "$(SCS_c / SCS_u)$" is always an **integer**.
- "$f_{SC0}^{SSB}$" is the **center frequency** of **subcarrier 0** of **SSB**.
- **Point A** is the frequency location of **Point A**.
- "$SCS_c$" is the **subcarrier spacing** of the **RBs** used to **calculate** the value of **offsetToPointA**. The value of "$SCS_c$" is indicated by **subCarrierSpacingCommon** from **MIB** (click 8.12.3).
- "$SCS_u$" is the **subcarrier spacing** of the **RBs** that are used for the **units** of **offsetToPointA**. The value of "$SCS_u$" is **15 kHz** in **FR1 bands** and **60 kHz** in

**FR2 bands**. Consequently, the **units** of `offsetToPointA` are **12•15 kHz** in **FR1** and **12•60 kHz** in **FR2**.

In TS 38.331 there is **only 1** parameter `offsetToPointA`, which is this one, that belongs to `SIB1`.

Notice that `offsetToPointA` is measured in **units** of **15 kHz RBs** in **FR1** and in **60 kHz RBs** in **FR2**. However, `offsetToCarrier` from `SCS-SpecificCarrier` (click 2.2.6.12) is **rather** measured in **units** of the **RBs** of the **carrier**.

For a description on how the **UE** finds out the location of **Point A**, click 8.4.

For your convenience, the definition of `offsetToPointA` and **Point A** from clause 4.4.4.2 of TS 38.211 is copied here:

"***4.4.4.2 Point A***

*Point A serves as a common reference point for resource block grids and is obtained from:*

- `offsetToPointA` *for a* PCell *downlink where* `offsetToPointA` *represents the frequency offset between point A and the lowest subcarrier of the lowest resource block, which overlaps with the SS/PBCH block used by the* ***UE*** *for initial cell selection, expressed in units of resource blocks assuming 15 kHz subcarrier spacing for FR1 and 60 kHz subcarrier spacing for FR2;*
  - *for operation licensed spectrum in FR1 and FR2-1, the lowest resource block has the subcarrier spacing provided by the higher layer parameter* `subCarrierSpacingCommon`*;*
  - *for operation for shared spectrum in FR1 or FR2, and for operation licensed spectrum in FR2-2, the lowest resource block has the subcarrier spacing same as the SS/PBCH block used by the* ***UE*** *for initial cell selection;*
- `absoluteFrequencyPointA` *for all other cases where* `absoluteFrequencyPointA` *represents the frequency-location of point A expressed as in ARFCN.*"

For an overall explanation on **UE configuration parameters** click 5. For a brief overview of **ASN.1**, click 5.9.

## 8.7.4 Example of getting the CRB ranges

This example continues from 8.6.3.

**NR node configuration**

This is indicated in 8.2.1.

**UE configuration already acquired**

From 8.2.1, the **UE** knows the band (**n257**, in **FR2-1**), the location of **SSB** (**GSCN 22,456**, which is **27,706.08 MHz**), the **SSB** subcarrier spacing (**240 kHz**), the **PCI**, the location of **PBCH** (because it is in a fixed position in respect to **PSS/SSS**), and the location of **PBCH DM-RS** within **PBCH** (which is a funcion of **PCI**). From 8.3.1, the **UE** knows the **center frequency** from **SC0** of **SSB** (**27,677.28 MHz**), and the value of the parameters from `MIB` and **PBCH System Information**. From 8.4.1, the **UE** knows

that **CORESET#0 subcarrier spacing** is **120 kHz**, its **frequency location**, its **bandwidth** is **48 RBs**, and it uses **multiplexing Pattern 1**. From 8.4.2, the **UE** knows the **search spaces for CORESET#0**, and therefore can decode **DCI message 1_0**, which provides the location of **SIB1** within **PDSCH**. From 8.6.3, the **UE** has decoded **SIB1**.

**UE configuration acquired in this section**

In order to get the **CRB ranges**, we **UE** needs to know **Point A** and the subcarrier spacings used in this channel.

From 8.7 , the expression to compute **Point A** is:

**Point A** = $f_{SC0}^{SSB}$ – (**offsetToPointA**•12•$SCS_u$ + $k_{SSB}$•$SCS_c$)

The value of "$k_{SSB}$" (click 8.7.2) is known to be **2**, out of parameter **ssb-SubcarrierOffset** (click 8.7.1) within **MIB** (click 8.3.1).

The **UE** knows that "$SCS_u$" is **60 kHz**, because we are in a **FR2** band.

The **UE** knows that "$SCS_c$" has the value of **subCarrierSpacingCommon** from **MIB**, because we are in a **FR2-1** band **licensed spectrum**. The value of "$SCS_c$" is therefore **120 kHz** (click 8.3.1).

The **UE** knows from **SIB1** (just obtained in this example section) the value of **offsetToPointA**. According to the **NR node** configuration of this example (click 8.2.1), **offsetToPointA** value is **34**.

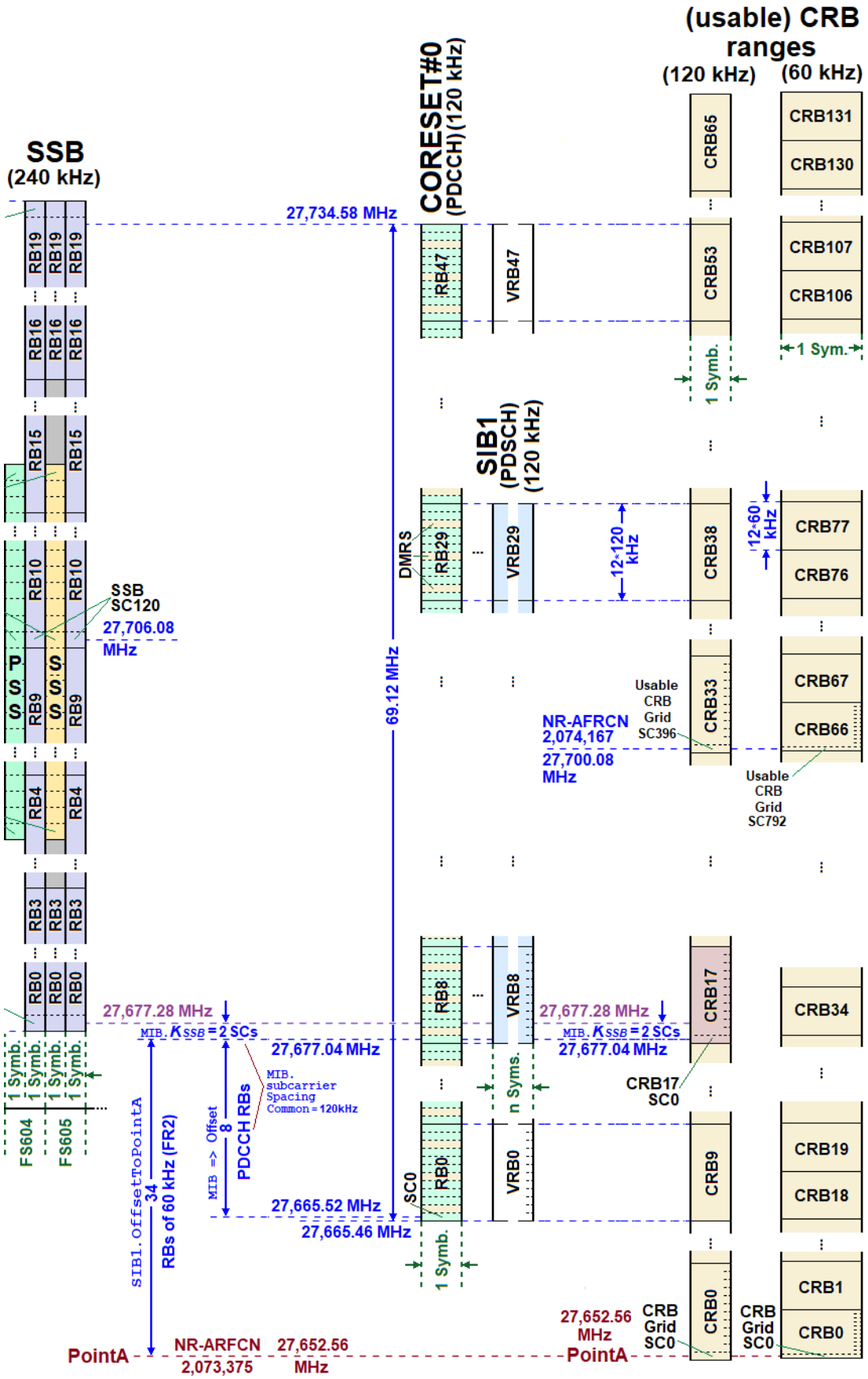
Therefore, for the values in this example:

**Point A** = 27,677.28 - (34 • 12 • .06 + 2 • .12) = **27,652.56 MHz**.

The **UE** knows that the **subcarrier spacings** of the **CRB ranges** are obtained from the different values of **SIB1** parameters **subcarrierSpacing**. According to the **NR node** configuration of this example (click 8.2.1), the channel supports data/shared subcarriers spacings of **60 kHz** and **120 kHz**. For this example, we choose to use **both** values of **SCS**.

The **UE** has therefore acquired **Point A**, and the **data/shared subcarrier spacings**, and therefore, it knows the **2 CRB ranges**.

The following figure shows the **frequency location** of **Point A**, and the two **CRB ranges**.



Some **relevant remarks** on this figure are:

- Within the **CRB range** whose **subcarrier spacing** is the one of `subCarrierSpacingCommon` from `MIB` (click 8.12.3), the **lowest Common Resource Block** that **overlaps** with the **center frequency** of **SSB subcarrier 0** is **CRB 34**. It has been highlighted with light maroon color, due to its importance.
- The **UE** is **not aware** of the **usable CRB ranges**: it is only aware of the **CRB ranges** and of the **carriers**. However, for completeness' sake, the figure shows the **channel middle frequency** (which belongs to the **channel raster**), that is the **center frequency** of the **next-above-middle subcarrier** of the **centered usable CRB range** (in this example, it is subcarrier **792** of the **CRB range** with **120 kHz subcarrier spacing**).
- **Both CRB ranges** have their **SC0** centered in **Point A.**
- Parameter `offsetToPointA` from `FrequencyInfoDL-SIB` (click 2.2.6.6) and variable "$k_{SSB}$" are shown, and both **must** be an **integer number**.
- Parameter `offsetToPointA` from `FrequencyInfoDL-SIB` (click 2.2.6.6) and variable "$k_{SSB}$" are defined between **center of subcarriers**, while the value of **Offset** is defined between **edges of Resource Blocks**.
- **Subcarrier spacing** of the **SSB**, of **CORESET#0/PDSCH** and of **one** of the two **control/shared CRB ranges**, are **different**.
- The figure shows the **duration** of **1 symbol interval** for several physical elements to highlight that the **average symbol interval duration** is **inversely proportional** to the **subcarrier spacing**.
- The figure is drawn to **frequency scale**, where each **contiguous vertical pixel** is **15 kHz**. When there is a line between **two frequency elements** (e.g., between two **RBs**), **half** of the line (**7.5 kHz**) belongs to **each frequency element**.

## 8.8 Getting the initial bandwidth parts and carriers

The **UE** gets the **initial bandwidth part** (click 2.2.7.1) and the **carriers** (click 2.2.4), both **Downlink** and **Uplink**, from the parameters from `SIB1`.

The `SIB1` parameters for the **Downlink carriers** and for the **Downlink initial bandwidth part** are described in 2.2.6.4.

The `SIB1` parameters for the **Uplink** and **SUL carriers**, and for the **Uplink** and **SUL initial bandwidth parts** are described in 2.2.6.2.

The `SIB1` parameters are described in 5.7.4.

This **step completes** the **cell selection procedure**, and the **UE** is said to have **camped** in a **cell**.

At this moment, the **UE** may **receive Downlink** within the **Downlink initial bandwidth part** of its **primary cell**. In spite of having the `SIB1` **Uplink** parameters, the still **cannot** transmit **Uplink data** because it does **not** have its **unique identifier** (the **C-RNTI**) in its **primary cell**. However, the **UE** has enough parameters to transmit through **PUSCH** in order to complete the **initial random access procedure**. In order to transmit **Uplink data**, the **UE** has to complete the **initial random access procedure** (click 8.10).

The **UE** can receive **PDCCH** and **PDSCH** information over the **Downlink initial bandwidth part** of its **primary cell**. The **UE cannot** transmit **PUCCH** information: notice that the **network** is still **unaware** of the **presence** of the **UE**. The **UE** can **transmit PUSCH** information, over the **Uplink initial bandwidth part**, but transmission can **only** be used for the **initial random access** procedure. The reason is that the **UE** still does **not** have its **C-RNTI** and the **UE cannot transmit PUCCH**.

## 8.9 PUCCH transmission before RRC connection establishment

In order to transmit **through PUCCH** before the **RRC connection establishment**, the **UE** gets its **PUCCH resource** for its **Uplink initial bandwidth part** out of **common parameters** from `SIB1` (click 5.3.4). The **PUCCH resource** is obtained by using `pucch-ResourceCommon` (from `PUCCH-ConfigCommon`, click 7.9.3) as an **index** to table 9.2.1-1 of TS 38.213:

| Index | PUCCH `format` | First symbol interval | Quantity of symbol intervals | PRB offset $RB_{BWP}^{offset}$ | Set of initial CS indexes |
|---|---|---|---|---|---|
| **0** | 0 | 12 | 2 | 0 | {0, 3} |
| **1** | 0 | 12 | 2 | 0 | {0, 4, 8} |
| **2** | 0 | 12 | 2 | 3 | {0, 4, 8} |
| **3** | 1 | 10 | 4 | 0 | {0, 6} |
| **4** | 1 | 10 | 4 | 0 | {0, 3, 6, 9} |
| **5** | 1 | 10 | 4 | 2 | {0, 3, 6, 9} |
| **6** | 1 | 10 | 4 | 4 | {0, 3, 6, 9} |
| **7** | 1 | 4 | 10 | 0 | {0, 6} |
| **8** | 1 | 4 | 10 | 0 | {0, 3, 6, 9} |
| **9** | 1 | 4 | 10 | 2 | {0, 3, 6, 9} |
| **10** | 1 | 4 | 10 | 4 | {0, 3, 6, 9} |
| **11** | 1 | 0 | 14 | 0 | {0, 6} |
| **12** | 1 | 0 | 14 | 0 | {0, 3, 6, 9} |
| **13** | 1 | 0 | 14 | 2 | {0, 3, 6, 9} |
| **14** | 1 | 0 | 14 | 4 | {0, 3, 6, 9} |
| **15** | 1 | 0 | 14 | $floor(N_{RB}^{BWP}/4)$ | |

Since these **PUCCH resources** do **not** have a **complete configuration** (e.g., they do **not** have an **instance** of `PUCCH-Resource` (click 7.9.7)), the **UE completes** its **required PUCCH configuration** as follows:

- **Quantity of resource blocks**

  The **quantity** "$M_{RB}^{PUCCH}$" of **PUCCH resource blocks** is **1**. However, if `nrofPRBs` from `PUCCH-ConfigCommon` (click 7.9.3) is **present** in a **FR2-2 band**, it contains the value of "$M_{RB}^{PUCCH}$".

- **Interlace configuration**

  Out of `useInterlacePUCCH-PUSCH-r16` from `BWP-UplinkCommon` (click 2.2.8.9). The **UE** expects that `useInterlacePUCCH-PUSCH-r16`, is **present** (for a serving cell) **either** in **all** **Uplink bandwidth parts** or in **none** of them.

- **Frequency hopping configuration**

  Out of `pucch-GroupHopping` and `intra-SlotFH-r17`, **both** from `PUCCH-ConfigCommon` (click 7.9.3).

For more detail on **PUCCH initial configuration**, refer to clause 9.2.1 of TS 38.213, up to and including table 9.2.1-1.

## 8.10 Initial random access procedure

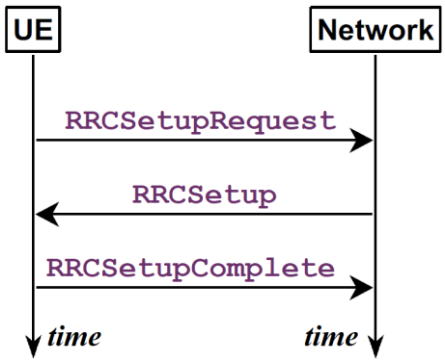
The **initial random access procedure** is the **RA procedure** (click 9) carried out during **initial access**. It **must** be a **CBRA procedure** (click 9.1.2) because during **initial access** the **UE** does **not have** its **C-RNTI** (click 6.9.10.2).

The **initial RA procedure** is done over **PRACH** and over the **Downlink** and **Uplink initial bandwidth parts** (click 2.2.7.2), whose configuration has been obtained from `SIB1`.

Within the **initial CBRA procedure**, the **UE** sends an `RRCSetupRequest` (click 5.8.5.10) **message** as its **Msg3** (click 9.3.4.2), in order to **initiate** the **RRC connection establishment**.

## 8.11 UE configuration and RRC connection establishment

The following figure shows the **RRC message exchange** for the **RRC connection establishment**.




The **UE** has already sent the `RRCSetupRequest` message as its **Msg3** (click 9.3.4.2), within the **initial random access procedure** (click 8.10).

It is then **only** required to exchange the `RRCSetup` (click 5.8.5.11) and `RRCSetupComplete` (click 5.8.5.12) messages.

The `RRCSetup` message typically conveys the **Non-SI** parameter group (click 5.3.6) that contains the **dedicated configuration** for the **UE**. **After security** setup the `RRCSetup` message also configures other **signaling radio bearers** (**SRBs**, click 4.5.2.1), with its parameter `radioBearerConfig`.

In some documents, the term "**RRC setup**" is used instead of "**RRC connection establishment**".

## 8.12 Initial access procedure parameters

The **cell search procedure** (click 8.2) does **not** have any configuration **parameter**, because it **must work without any UE configuration**.

The **initial access procedure** (click 8) is configured with:

- **Common** parameters that are **broadcast** in `MIB`, as described in section 8.12.1.

- **Common** parameters that are **broadcast** (or **unicast**) within `SIB1`, as described in section 8.12.2.

For an overall explanation on **UE configuration parameters** click 5. For a brief overview of **ASN.1**, click 5.9.

### 8.12.1 Initial access procedure parameters within `MIB`

The **initial access procedure** is configured with **common parameters** that are **broadcast** in `MIB` (click 5.3.3), which is sent within **SSB** (click 6.1), in the following case:

- In **the primary cell**, **during cell selection**, **after cell search**.

For the **initial access procedure** parameters within `MIB`, the **most relevant related data type** is:

- `PDCCH-ConfigSIB1` (click 8.12.3): common parameters

The following **data types**, whose **only parents** are `PDCCH-ConfigSIB1` (click 8.12.3) and `PDCCH-ConfigCommon` (click 6.12.4), are also relevant:

- `ControlResourceSetZero` (click 8.12.3): common parameters
- `SearchSpaceZero` (click 6.4.2): common parameters

The **ancestor tree** (click 5.9.8) of these **data types** up to the `BCCH-BCH-Message` **class** is:

```
                                BCCH-BCH-Message (click 5.8.5.1)
message                           BCCH-BCH-MessageType (click 5.8.5.1)
  mib                               MIB (click 8.12.3)
    pdcch-ConfigSIB1                  PDCCH-ConfigSIB1 (click 8.12.3)
      controlResourceSetZero            ControlResourceSetZero (click 8.12.3)
      searchSpaceZero                   SearchSpaceZero (click 8.12.3)
```

### 8.12.2 Initial access procedure parameters within `SIB1`

The **initial access procedure** is configured with **common parameters** within `SIB1` (click 5.3.4) in the following case:

- In **the primary cell**, **during initial access**, **after cell selection**.

For the **initial access procedure** parameters within `SIB1`, the **most relevant related data type** is:

- `PDCCH-ConfigCommon` (click 6.12.4): common parameters

The following **data types**, whose **only parents** are `PDCCH-ConfigSIB1` (click 8.12.3) and `PDCCH-ConfigCommon` (click 6.12.4), are also relevant:

- `ControlResourceSetZero` (click 8.12.3): common parameters
- `SearchSpaceZero` (click 8.12.3): common parameters

The **ancestor tree** (click 5.9.8) of this **most relevant related data type** up to `SIB1` is:

```
                                    SIB1 (click 5.7.4)
servingCellConfigCommon               ServingCellConfigCommonSIB (0,1) (click 3.9.3)
  downlinkConfigCommon                  DownlinkConfigCommonSIB (click 2.2.6.4)
    initialDownlinkBWP                    BWP-DownlinkCommon (click 2.2.8.6)
      pdcch-ConfigCommon                    PDCCH-ConfigCommon (click 6.12.3)
        controlResourceSetZero                ControlResourceSetZero (click 8.12.3)
        searchSpaceZero                       SearchSpaceZero (click 8.12.3)
```

## 8.12.3 `MIB` data type

It is **only instantiated** within `BCCH-BCH-MessageType` (click 5.8.5.1), within the `BCCH-BCH-Message` (click 5.8.5.1).

It is only **broadcast** (or **unicast**) within **SSB** (click 6.1).

For its **ancestor tree up to** the `BCCH-BCH-Message` **class**, click 8.12.1.

**All** its **parameters** are **common** (click 5.1.3).

**Each** of its **instances** provides the very essential configuration for the **initial access procedure during cell search**.

A `MIB` **instance** is represented with **23 bits** for its **encoding** and **transmission**. For a description of the **data representation** of a `MIB` **instance** click 6.4.2.

For a description of the `MIB` **parameter group**, click 5.3.3.

### 8.12.3.1 `MIB` parameters

**All** its **parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `systemFrameNumber BIT STRING (SIZE (6))`: mandatory, common

  It contains the **6 most significant bits** of the 10-bit **System Frame Number** (**SFN**). The **4 least significant bits** of the10-bit **SFN** are sent in **system-frame-number** (click 6.4.1) from **PBCH system information** (click 5.3.2).

  It is **represented** with **6 bits**.

- `subCarrierSpacingCommon ENUMERATED {scs15or60, scs30or120}`: mandatory, common

  Its **dual use** is as follows:

  - For **licensed spectrum** in **FR1** and **FR2-1** bands:

    It indicates **either** `15` **kHz** or `30` **kHz** in **FR1 bands**, while in **FR2-1 bands** it indicates **either** `60` **kHz** or `120` **kHz**.

    It indicates the **subcarrier spacing** for **CORESET#0** and for the corresponding **PDSCH Resource Elements**, for **Msg2** (click 9.3.4), **Msg4** (click 9.3.5) and **MsgB** (click 9.3.7) for **initial access**, for **paging** and for **broadcast SI-messages** (i.e., `SIB1` (click 5.3.4) and **other SIBs and posSIBs** (click 5.3.5)).

    It also indicates the **subcarrier spacing** of the **Resource Blocks** used to **calculate** the **value** of `offsetToPointA` from `SIB1` (click 5.3.4). Recall that the **units** of `offsetToPointA` are **different**.

    It also indicates the **subcarrier spacing** of the **units** of "$k_{SSB}$" (click 8.7.2).

- For **shared spectrum** in **FR1** bands (refer to TS 38.213) and in **FR2-2** bands:

  It is used to derive the **QCL** relation between **SSBs** as specified in clause 4.1 of TS 38.213.

  In this case, the **subcarrier spacing** for **all** the **elements indicated** in the **previous bullet** is the same as the **subcarrier spacing** of **SSB**.

This parameter is **represented** with **1 bit**.

- `ssb-SubcarrierOffset` **`INTEGER (0..15)`**: mandatory, common

  It **indicates** the value of "$k_{SSB}$" (click 8.7.2). In a **particular case** (click 8.7.1) it **also provides** a **1-bit** value to derive the **QCL** relation between **SSBs**, with a **1-bit** value.

  It has a **same-name** parameter from `CGI-InfoNR`.

  This parameter is somehow **relevant** and **complex**: this is why it is described in the **specific section** 8.7.1.

- `dmrs-TypeA-Position` **`ENUMERATED {pos2, pos3}`**: mandatory, common

  It is the **slot symbol index** of the **first DM-RS symbol** both for **PDSCH** and for **PUSCH** when using **time-domain mapping** `typeA` (click 7.11.5.1). Value `pos2` indicates **slot symbol index 2** and `pos3` indicates **3** (i.e., the **third or** the **fourth** symbol in the slot, respectively).

  It is used by the **UE** for its **primary cell**, unless it is **superseded** by its **same-name** parameter from `ServingCellConfigCommon` (click 3.9.6).

  It is **represented** with **1 bit**.

  For more detail on the position of **PDSCH-DM-RS**, refer to clause 7.4.1.1.1 of TS 38.211. For more detail on the position of **PUSCH-DM-RS**, refer to clause 6.4.1.1.3 of TS 38.211.

- `pdcch-ConfigSIB1`: mandatory, common

  It provides the configuration for **common search space** (**CSS**) **Type 0** and for **CORESET#0**.

  Its **data type** `PDCCH-ConfigSIB1` is a **`SEQUENCE`** of **2** other parameters, as follows:

```
PDCCH-ConfigSIB1 ::= SEQUENCE {
  controlResourceSetZero ControlResourceSetZero,
  searchSpaceZero        SearchSpaceZero
}
```

  where:

```
ControlResourceSetZero INTEGER (0..15)
SearchSpaceZero        INTEGER (0..15)
```

  Parameter `pdcch-ConfigSIB1` is somehow **relevant** and **complex**: this is why it is described in the **specific section** 8.4.3.

- `cellBarred` **`ENUMERATED {barred, notBarred}`**: mandatory, common

  It indicates if **UEs** are **allowed** (cell `notBarred`) or **not allowed** (cell `barred`) to access this cell. A cell may be barred, for example, because it is under maintenance or reconfiguration.

It is **represented** with **1 bit**.

- `intraFreqReselection ENUMERATED {allowed, notAllowed}`: mandatory, common

  It controls **cell selection/reselection** to **intra-frequency cells** when the highest ranked cell is **barred**, or **treated as barred**, by the **UE**. It is ignored by **IAB-MT**.

  It is **represented** with **1 bit**.

- `Spare BIT STRING (SIZE (1))`: mandatory, common

  Reserved for future use. It is **represented** with **1 bit**.

For more detail on these parameters, refer to the table "*MIB field descriptions*" of TS 38.331 and to clause 13 of TS 38.213.

Source web pages:

- https://wirelessbrew.com/5g-nr/5g-nr-master-information-block-mib/
- https://www.sharetechnote.com/html/5G/5G_PBCH.html
- http://howltestuffworks.blogspot.com/2019/10/5g-nr-pbch-and-master-information-block.html
- https://rfmw.em.keysight.com/wireless/helpfiles/n7631/Content/Main/Channel%20Setup%20DL.htm
- https://www.sharetechnote.com/html/5G/5G_SS_Block.html#SSB_Transmission_Pattern
- https://www.sharetechnote.com/html/5G/5G_FrameStructure.html#SS_PBCH_FrequencyDomainResourceAllocation

### 8.12.3.2 MIB ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
MIB ::= SEQUENCE {
  systemFrameNumber BIT STRING (SIZE (6)),
  subCarrierSpacingCommon ENUMERATED {scs15or60, scs30or120},
  ssb-SubcarrierOffset    INTEGER (0..15),
  dmrs-TypeA-Position     ENUMERATED {pos2, pos3},
  pdcch-ConfigSIB1 PDCCH-ConfigSIB1,
  cellBarred ENUMERATED {barred, notBarred},
  intraFreqReselection ENUMERATED {allowed, notAllowed},
  spare BIT STRING (SIZE (1))
}
```

where:

```
PDCCH-ConfigSIB1 ::= SEQUENCE {
  controlResourceSetZero ControlResourceSetZero,
  searchSpaceZero        SearchSpaceZero
}
```

and:

```
ControlResourceSetZero INTEGER (0..15)
SearchSpaceZero        INTEGER (0..15)
```

# 9. NR random access procedure

The **random access** (**RA**) **procedure** (sometimes called random access channel procedure, RACH procedure or PRACH procedure) allows:

- The **UE** to achieve **Uplink synchronization** (**frame number**, **timing advance**, …)
- The **NR node** to learn what **beam** has been **selected** by the **UE** (**only** if the **NR node** uses **Tx beam sweeping**, click 6.1.2.3).
- The **UE** to get a **unique identity** (**C-RNTI**, click 6.9.10.2) within the **cell** (**only** for the case of **initial access**, click 8).
- The **UE** to get **System Information** (**SI**, click 5.2), for the specific case of **on-demand SI** (click 5.3.5).

An **RA procedure** may be **triggered** by a number of different **events** (click 9.4), such as **initial access**.

Before starting the **random access procedure**, the **UE** has to perform **resource selection** (click 9.2) to determine **different variables** that are required for the **RA procedure**.

According to the **quantity** of **steps**, the **RA procedures** can be divided into **4-step RA procedures** (click 9.1.1) and **2-step RA procedures**.

According to the **possibility**, or **not**, of **contention**, the **RA procedures** can be divided into **contention-based RA** (**CBRA**) **procedures** (click 9.1.2) and **contention-free** (**CBRA**) **procedures**.

The **combination** of the **quantity** of **steps** and the **existence** or **not** of **contention** leads to **4 different random access procedures** (plus a **fallback variant**). For more detail on the **4 different RA procedures**, click 9.1.

The **random access** (**RA**) **procedure** is sometimes called the "**random access channel procedure**", the "**RACH procedure**" or the "**PRACH procedure**".

For more detail on the **random access procedure**, refer to:

- Clause 8 of TS 38.213.
- Clause 5.1 of TS 38.321.
- Clauses 5.3.4 and 9.2.6 of TS 38.300.

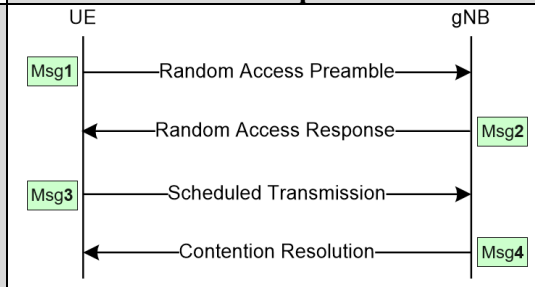
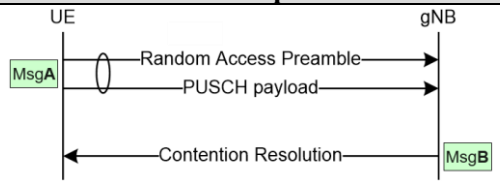
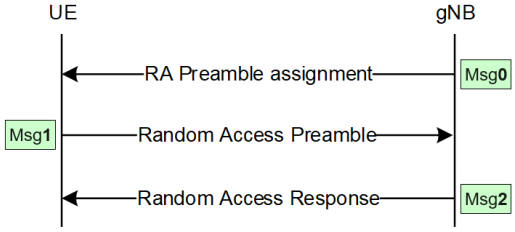
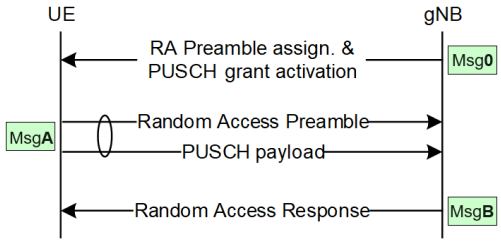
Source web pages:

- https://www.sharetechnote.com/html/5G/5G_RACH.html
- http://howltestuffworks.blogspot.com/2019/09/5g-nr-random access-procedure.html
- http://howltestuffworks.blogspot.com/2020/04/5g-nr-2-step-random access-procedure.html

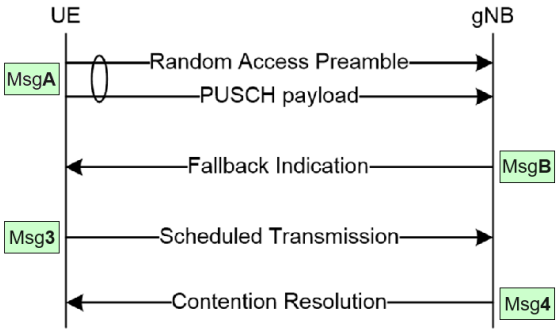
## 9.1 The 4 RA procedures (and 1 variant)

The **combination** of the **quantity** of **steps** (**4-step** or **2-step**, click 9.1.1) and the **possibility** or **not** of **contention** (**CBRA** or **CFRA**, click 9.1.2) results in **4 different**

**random access procedures**. The following table shows the **4 different RA procedures**, including for **each** of them its **time diagram**:

| | 4-step | 2-step |
|---|---|---|
| **Contention-Based (CBRA)** |  (click 9.1.1) |  (click 9.1.5) |
| **Contention-Free (CFRA)** |  (click 9.1.4) |  (click 9.1.6) |

There is also a **variant** of the **2-step CBRA procedure** (click 9.1.5) that is triggered by the **NR node** by sending a **MsgB** (click 9.3.7) with a **Fallback Indication**, (e.g., because the previous **MsgB** was **lost**) as shown in the following time diagram:




Notice that the **RA preamble** (click 7.3.1) is sent in the **four cases** of **RA procedure**, including in the **fallback** variant.

The **UE determines** which of the **4 different random access procedures** has to use based on its configured parameters:

- ***If* both `cfra` and `cfra-TwoStep-r16` are absent**, ***then***, the **UE** uses a **CBRA procedure**. The **UE** uses an **RSRP threshold** to select between the **4-step CBRA** and the **2-step CBRA**.
- ***If* `cfra` is present**, ***then***, the **UE** uses **4-step CFRA**.
- ***If* `cfra-TwoStep-r16` is present**, ***then***, the **UE** uses **2-step CFRA**.

Parameters `cfra` and `cfra-TwoStep-r16` are **mutually exclusive** for any given **bandwidth part** (**BWP**). **Both** `cfra` and `cfra-TwoStep-r16` are from `RACH-ConfigDedicated`, click 9.5.10.

For the **specific case** of an **RA procedure** for **beam failure recovery**, a **CFRA procedure** is used if an `BeamFailureRecoveryConfig` (click 9.5.4) has been setup within `BWP-UplinkDedicated` (click 2.2.8.10). Otherwise, a **CBRA procedure** is used.

For the **configuration parameters** of the **4 types** of **RA procedure**, click 9.5.

**Each** of the **4 RA procedures** has some **specific characteristics** depending on its **triggering event** (click 9.4.2).

### 9.1.1 The 4-step and 2-step RA procedures

The **4-step RA procedures** require **4** (or **3**) **message exchanges** (i.e., "**steps**") between the **UE** and the **NR node**, while the **2-step RA procedures** require **2** (or **3**) **message exchanges**.

The **4-step RA procedures** were defined in the **first release** (**R15**) of **5G**. The main reason to introduce the **2-step RA procedures** in **R16** was to **reduce** the overall **latency** of the **4-step RA procedures**.

The **4-step RA procedures** are:

- **4-step contention-based random access** (**4-step CBRA**), click 9.1.3. This **RA procedure** has **4 steps**.
- **4-step contention-free random access** (**4-step CFRA**), click 9.1.4. This **RA procedure** has **3 steps**.

The **2-step RA procedures** are:

- **2-step contention-based random access** (**2-step CBRA**), click 9.1.5. This **RA procedure** has **2 steps**. The **2-step CBRA** includes a **fallback variant**.
- **2-step contention-free random access** (**2-step CFRA**), click 9.1.6. This **RA procedure** has **3 steps**.

The **2-step RA procedures** have the following **advantages** over the **4-step** ones:

- **Reduce** the quantity of **signaling steps** of the **RA procedure** (**4 steps** to **2 steps**), which **reduces latency**.
- **Reduce** the **latency** between elements of the **RA procedure** (**time between transmission slots**).
- Allowing **PUSCH transmission** at the **very first step** (which **reduces latency**).
- **Reduce** the control **signaling overhead**.
- If it is used in **unlicensed spectrum**, it can **reduce** the **quantity** of **listen before talk** (**LBT**) attempts.

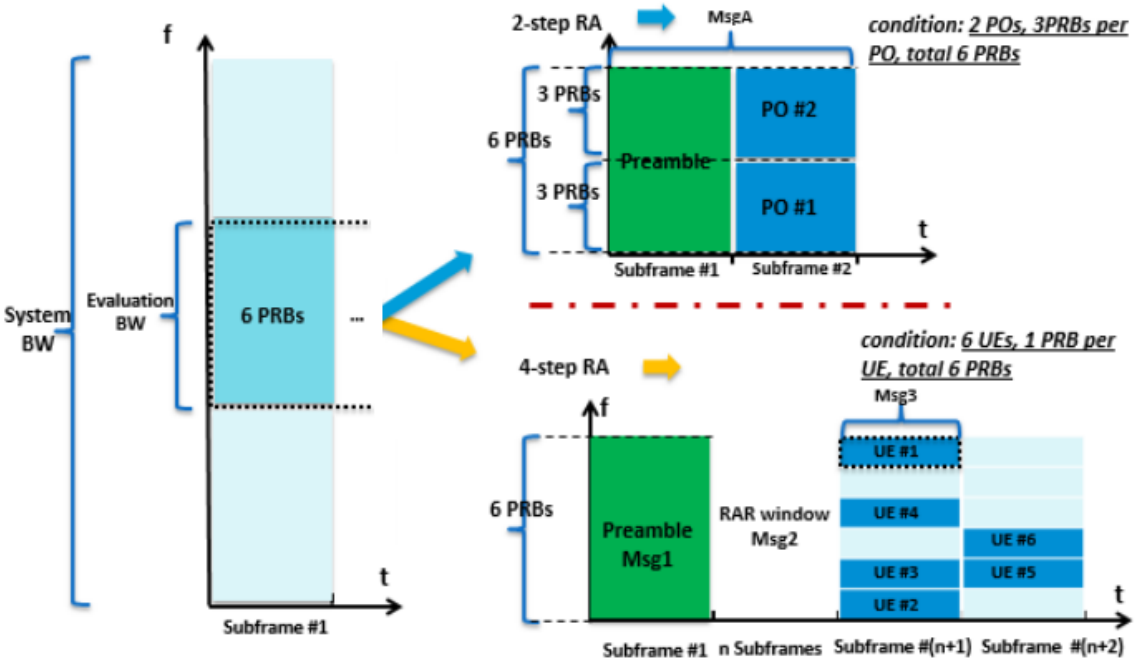
The **2-step RA procedures** have the following **disadvantages** over the **4-step** ones:

- A **demodulation performance degradation** observed without time offset compensation at the **NR node**, especially for **mid-range cells** or **wide area cells**.
- In the case that **all** the **RA preambles** (click 7.3.1) from multiple **UEs** trying to perform the **initial access** are **mapped** to the same **PUSCH physical resources**, the **different RA preambles overlap**, which might result in **unsuccessful decoding**. There is therefore a **trade-off** between the **collision probability** of the

**PUSCH part** of **MsgA** (click 9.3.6) and the **resource overhead** for **2 Scheduling Requests**.

**Timing of 4-step and 2-step RA procedures**

The following figure compares the **timing** of **4-step RA procedures** and of **2-step RA procedures**:




Some **relevant remarks** on this figure are:

- The **latency** of the **2-step procedures** is **much lower** than the one of the **4-step procedures**.
- **Msg1** (click 9.3.6) **only** contains the **RA preamble**.
- **MsgA** (click 9.3.6) contains **both** the **RA preamble** and a **PUSCH payload**.
- **Msg3** (click 9.3.4) **only** contains a **PUSCH payload**.
- **Each dark blue rectangle** of the **2-step RA procedure** represents a **PUSCH transmission occasion** (**PO**). **Each such PO** is defined in the **PUSCH configured grant**. **Each UE** may send its **PUSCH payload** (within **MsgA**) in its corresponding **POs**.
- **Each dark blue rectangle** of the **4-step RA procedure** represents a **PUSCH assignment**. **Each such PUSCH assignment** corresponds to a **different RA preamble sequence** received by the **NR node**. The **RA UEs in each contending group** will transmit their **Msg3** in the **corresponding PUSCH assignment**.

In TS 38.213, the **4-step RA procedures** are sometimes called "**Type-1**" **RA procedures** and the **2-step RA procedures** are sometimes called "**Type-2**" **procedures**.

When **carrier aggregation** is configured, a **2-step RA procedure** is **only** performed on the **PCell**.

Source web pages:

- https://howltestuffworks.blogspot.com/2020/04/5g-nr-2-step-random-access-procedure.html?utm_source=chatgpt.com

### 9.1.2 The contention-based (CBRA) and the contention-free (CFRA) RA procedures

**CBRA procedures might** have **contention** and are designed to **solve it**, while **CFRA procedures cannot** have **contention**.

**Contention** occurs when **more than 1 UE** sends **either** its **scheduled transmission** (**Msg3**, click 9.3.4) or its **PUSCH payload** (**within MsgB**, click 9.3.6) over the **same PUSCH resources**. ***If*** **1 transmission** is **correctly received** by the **NR node** (because it **dominated** all the other transmissions, radio-wise), ***then***, the **sender UE wins** the **contention** and **successfully completes** its **CBRA procedure**. **Each** and **every sender UE** that did **not win** the **contention** has to **restart** its **RA procedure**.

**Starting peer**

- **CBRA procedures** are **started by** the **UE**, sending **either Msg1** or **MsgA**.
- **CFRA procedures** are **started by** the **NR node**, sending **Msg0** (click 9.3.1). Consequently, **CFRA procedures cannot** be used for the **UE Initial Access**. Regarding the **2-step CFRA procedure**, it is **only** supported for **handover**.

**Selection of CBRA or CFRA**

- **CBRA procedures** can **always** be used.
- **CFRA procedures** can **only** be used if the **UE** already has a **C-RNTI** and certain conditions hold (click 9.4.2).
- If the **UE** already **has** a **C-RNTI**, a **CFRA procedure** will **most likely** be **used**. However, in **some scenarios** (e.g., **Uplink desynchronizatio**n or **handover failure recovery**) a **CBRA procedure** is **used** even if the **UE** already **has** a **C-RNTI**.
- For the actual **determination** of **which RA procedure** to **use**, click 9.1.

The **CBRA procedures** are:

- **4-step contention-based random access** (**4-step CBRA**), click 9.1.3. This procedure has **4 steps**.
- **2-step contention-based random access** (**2-step CBRA**), click 9.1.5. This procedure has **2 steps**. The **2-step CBRA** includes a **fallback variant**.

The **CFRA procedures** are:

- **2-step contention-free random access** (**2-step CFRA**), click 9.1.6. This procedure has **3 steps**.
- **4-step contention-free random access** (**4-step CFRA**), click 9.1.4. This procedure has **3 steps**.

A **CFRA procedure** on an **SCell** can only be **initiated** by the **NR node** to establish **timing advance** for a **secondary TAG**. In this case:

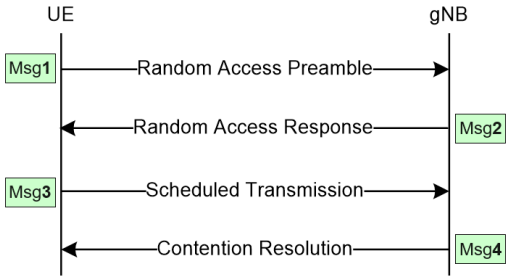
- The **CFRA procedure** is initiated by the **NR node** with a **PDCCH Order** (**Msg0**) that is **sent** on an **activated SCell** of the **secondary TAG**,
- The **RA preamble transmission** (**step 1**) takes place on the **SCell**,
- The **random access response** (**RAR**) **transmission** (**step 2**) takes place on the **PCell**.

When **carrier aggregation** is configured:

- The **contention resolution** (**step 4**) of a **CBRA** procedure can be **cross-scheduled** by the **PCell**.
- The **3 steps** of a **CFRA procedure** that was **started** on the **PCell** remain on the **PCell**.

### 9.1.3 The 4-step CBRA procedure

The **time diagram** of the **4-step CBRA procedure** is:




The **UE** performs **resource selection** (click 9.2) for the **RA procedure**.

Afterwards, the **4 major message exchanges** (“**steps**”) of the **4-step CBRA procedure** are:

1. **Each RA UE sends** its **Msg1** (**RA preamble**, click 9.3.2): see subsection 9.1.3.1.
2. **Each remaining UE receives** a **correct Msg2** (**RA response**, click 9.3.3): see subsection 9.1.3.2.
3. **Each remaining UE sends** its **UE-specific Msg3** (**Scheduled transmission**, click 9.3.4): see subsection 9.1.3.3.
4. **Each successful UE receives** its **UE-specific Msg4** (**Contention resolution**, click 9.3.5): see subsection 9.1.3.4.

When **carrier aggregation** is configured, the **first 3 steps** of a **4-step CBRA procedure always** occur on the **PCell**.

For a list of **cases** where a **CBRA procedure** may be used, click 9.4.2.

For more detail on the **parameters** for the **4-step CBRA procedure**, click 9.5.

#### 9.1.3.1 Step 1: each RA UE sends its Msg1 (RA preamble)

The **actions** (not necessarily sequential) within **step 1** of the **4-step CBRA procedure** are:

**a)** **One** (or **more**) **UEs transmit(s) Msg1** in a **given PRACH occasion**.

**One** (or **more**) **UEs transmit(s) Msg1** (**RA preamble**, click 9.3.2) through **PRACH** (click 7.1) in a **given PRACH occasion** (click 7.2.2.1). We will call **all these UEs** the "**RA UEs**".

**Each RA UE independently selected** its **PRACH occasion** based on the **PRACH scheduling** (click 9.2), but they **all** happened to **transmit** in the **same PRACH occasion**.

**Each RA UE** sends an **RA preamble sequence randomly selected** (click 9.2.1) by itself.

**b)** **Each** and **every RA UE computes** its **RA-RNTI**.

**Each** and **every RA UE computes** (click 9.1.7) the **same value** of **RA-RNTI**. They compute the **same value** because **all** the **RA UEs transmitted** their **RA preamble** in the **same PRACH occasion**, and the **RA-RNTI** is **only computed** from the **variables** of the **PRACH occasion**.

**c)** **Each RA UE waits** for the **reception** of **1 correct Msg2**

**Each RA UE waits** for the **reception** of **1 correct Msg2** (**random access response**, click 9.3.3). The **correct Msg2** is the one that **fulfills both**:

- Its **PDSCH resources** have been **assigned** by **1 DCI Format 1_0 message** (sent through **PDCCH**) with the **last 16 bits** of the **CRC masked** with the **RA-RNTI**. Notice that **RA-RNTI** is used by **every RA UE**, **even** if a given **RA UE** already **has** a **C-RNTI**.
- The **Msg2** (received through **PDSCH**) contains in its **6-bit field** "**Random Access Preamble Identifier** (**RAPID**)" the **index** of the **RA preamble sequence** that the **RA UE sent** in **action a**) above.

**Each RA UE waits** for the **correct Msg2 up to** a **certain quantity** of **slots**. This **quantity** of **slots** that **each RA UE waits** is called the **RA response window**. **Each** and **every RA UE** gets the **value** of its **RA response window** from **`ra-ResponseWindow`**.

This **parameter** is from an **instance** of **`RACH-ConfigGeneric`** (see at the end of this subsection).

**d)** The **NR node receives 1** (or **more**) **RA preambles**(**s**) from the **RA UEs**

It is **likely** (but **not guaranteed**) that the **NR node correctly receives some** (or even **all**) of the **different RA preamble sequences** that were sent by the **RA UEs**. This is possible because:

- The **RA UEs** that sent the **same RA preamble sequence** will cause the **NR node** to receive that **RA preamble sequence** with **more power** and **some distortion**. This makes it **possible** (but **not guaranteed**) that the **NR node correctly decodes** this **RA preamble sequence** from the **superposed signals** that were sent by several **UEs**.

- The **RA preamble sequences** that are **different** are **orthogonal** (click 11.20.4). Their **orthogonality** makes it **possible** (but **not guaranteed**) that the **NR node correctly decodes some** (or **even all**) of them, **in spite** of the **superposition** of their **signals**.

**e)** The **NR node produces 1** (or **more**) **Msg2**

The **NR node** produces **1 Msg2** for **each different RA preamble sequence** that it **received** in this **PRACH occasion**. **Each Msg2** includes the **index** of **1** of the **RA preamble sequences** that were received by the **NR node**. **Each Msg2** also contains a **PUSCH assignment** and **other** configuration **variables** that are **intended** for the **1** (or **more**) **UE(s)** that **sent** the corresponding **RA preamble sequence**.

The **NR node** can **integrate 1** (or **more**) **Msg2** into **1 random access response** (**RAR**) **MAC PDU** (click 4.5.5.7). The **result** is therefore that the **NR node** sends **1** (or **more**) **RAR MAC PDU(s)** that **jointly contain all** the **Msg2**.

**f)** The **NR node produces 1** (or **more**) **DCI Format 1_0 message(s)**

The **NR node** produces **1 DCI Format 1_0 message** for **each RAR MAC PDU** produced in the **previous action**. **Each** such **DCI Format 1_0 message assigns** the **PDSCH resources** for the **UE reception** of **1 RAR MAC PDU**.

**g)** The **NR node sends all** the **DCI Format 1_0 message(s)**

The **NR node** sends through **PDCCH each** and **every DCI Format 1_0 message** that was **produced** in the **previous action**. The **last 16 bits** of the **CRC** of **each** and **every DCI Format 1_0 message** are masked with the **RA-RNTI**. There is **only 1 RA-RNTI** because its value is **computed** from the **PRACH occasion**, and **all** the processing described in this **4-step CBRA section** corresponds to **only 1 PRACH occasion** (see **action a**) above). The **DCI messages** are **sent** through the **SpCell**, within a **CSS-Type1**.

**h)** The **NR node sends all** the **Msg2**

The **NR node** sends through **PDSCH each** and **every RAR MAC PDU** that was **produced** in **action e**) above. **Each RAR MAC PDU** is sent in the resources that are **assigned** by **1** of the **DCI Format 1_0 messages** that the **NR node** sent in the **previous action**.

Although **Msg2** is sent through **PDSCH**, the **HARQ ACKs** (click 4.5.6.7) are **not** sent in this case, but the **HARQ FEC** might be used.

**i)** **Every RA UE receives all** the **DCI Format 1_0 message(s)**

Since **every RA UE computed** the **same RA-RNTI**, **every RA UE receives** (**except** for **channel impairments**) and **processes all** the **DCI Format 1_0 message(s)** sent by the **NR node** in **action g**) above.

**j)** **Every RA UE receives all** the **Msg2**

Since **every RA UE receive** and **process all** the **DCI Format 1_0 message(s)** (see the previous **action**), **every RA UE receives** (**except** for **channel impairments**) **all** the **RAR MAC PDUs**. Consequently, **every RA UE receives** (**except** for **channel impairments**) **all** the **Msg2** sent by the **NR node** in **action h**) above.

**k) Each RA UE discards all** the **Msg2** for **other RA UEs**

**Each RA UE discards all** the **received Msg2** (**within RAR MAC PDUs** scheduled with the **DCI Format 1_0 message** masked with **RA-RNTI**) that do **not** contain the **index** of the **RA preamble sequence** that the **RA UE** sent in **action a**) above. Consequently, **each RA UE** in **keeps only 1 Msg2** (**except** for **channel impairments**, which could imply **0 Msg2 kept**).

When **an RA UE** receives its **expected Msg2** (the one with its **RA preamble sequence index**), the **RA UE** does **not** send a **HARQ ACK** (click 4.5.6.7) upon reception of **each Msg2**. However, the **NR node** learns that **every RA UE** from has received its **expected Msg2** when **none** of them **retransmits** its **Msg1** (see **action l**) **below**). For more detail, refer to clause 8.2 of TS 38.213.

For **each different Msg2 produced** by the **NR node** (i.e., for **each RA preamble sequence** that it has received), there are **1** (or **more**) **RA UE**(**s**) that receives it (i.e., the **RA UEs** that **sent** the **same RA preamble sequence**). **Each set** of **RA UEs** that **receives** the **same Msg2** (i.e., the **same PUSCH assignment**) is **1 RA contending group**. **All** the **RA UEs** in **each RA contending group** have sent the **same RA preamble sequence** in the **same PRACH occasion**. This is why, **all** of **them** receive the **same PUSCH assignment**.

When a **RA contending group** contains **more than 1 RA UE**, **all** of these **RA UEs transmit** their **Msg3** over the **same PUSCH resources**. Consequently, **all** of their **transmissions** will **superimpose** (i.e., **collide**) and **contention occurs**. **Contention** is **solved** by the **NR node** in **action c**) of **step 3** (click 9.1.3.3).

**l)** The **wait** by the **RA UE** for **Msg2** is **over**

The **wait** (from **action c**) **above**) by **each RA UE** to receive **1 correct Msg2** (through **PDSCH**) is **over**. **Each RA UE ends** its **waiting** with **1 among** the **3** following **outcomes**:

- The **correct Msg2** has **not** been **received** within the **RA response window** (see **action c**) above).

  Then, the **UE retransmits** its **Msg1** with **transmission power** increased in `powerRampingStep`, and goes **back** to **action b**) above.

- The **correct Msg2** has **not** been **received** after `preambleTransMax` **retransmissions**.

  Then, the **UE gives up** accessing the **network** in **this cell**.

- The **correct Msg2** has been **received** within the **RA Response window**.

  Then, the **RA UE continues** in **step 2** (click 9.1.3.2). The **RA UEs** that **continue** in **step 2** are called the "**remaining UEs**".

The **2 parameters** just referenced are from an **instance** of `RACH-ConfigGeneric` (see at the end of this subsection).

The reasons why an **RA UE might not receive** the **correct Msg2** are:

- The **NR node** has **not received** the **RA preamble sequence sent** by this **RA UE**, **neither** from the **RA preamble** sent by this **UE**, **nor** from the **RA preamble** sent by **any other RA UE** in its **RA contending group**.

- The **NR node received** the **RA preamble sequence sent** by this **RA UE**, but the **RA UE** has **not received** the corresponding **Msg2** because the **Msg2**'s **scheduling DCI message** was **lost** due to **channel impairments**.
- The **NR node received** the **RA preamble sequence sent** by this **RA UE**, and the **RA UE received** the **DCI message** that **schedules** the **Msg2 reception**, but the **UE** has **not received** the corresponding **Msg2**, because it was **lost** due to **channel impairments**.

Regardless of the **cause**, if the **RA UE** does **not receive** the **correct Msg2**, the **RA UE** will **react** as indicated at the **beginning** of **this action**.

In the **4-step CBRA procedure**, the **instance** of `RACH-ConfigGeneric` (click 9.5.5) that is used depends on the triggering event:

- ***If*** this **4-step CBRA procedure** is **not** for **beam failure recovery**, ***then***,

  the **said instance** is the **son** of an **instance** of `RACH-ConfigCommon` (click 9.5.3).
- ***Else***, the **said instance** is the **son** of an **instance** of `BeamFailureRecoveryConfig` (click 9.5.4).

For more detail on the **parameters** for the **4-step CBRA procedure**, click 9.5.

#### 9.1.3.2 Step 2: each remaining UE receives a correct Msg2 (RA response)

**All** the **remaining UEs** in **each RA group** have **received** (through **PDSCH**) **1 correct Msg2** (**random access response** (**RAR**), click 9.3.3) within the **RA response window**. **Recall** that:

- The "**remaining UEs**" were defined in **action l**) from **step 1** (click 9.1.3.1).
- The **correct Msg2** and the **RA response window** were defined in **action c**) from **step 1**.

The **correct Msg2** received by **each remaining UE** contains:

- A **TC-RNTI** (i.e., a t**emporary C-RNTI**) for the **remaining UE**.
- A **PUSCH assignment** for the **remaining UE**'s **transmission** of its **Msg3**.
- The **backoff time** that the **remaining UE must wait** before retrying the **4-step CBRA procedure**.

#### 9.1.3.3 Step 3: each remaining UE sends its UE-specific Msg3 (Scheduled transmission)

The **actions** (not necessarily sequential) within **step 3** of the **4-step CBRA procedure** are:

**a) Each remaining UE sends its UE-specific Msg3**

**Each remaining UE sends its UE-specific Msg3** (**scheduled transmission**, click 9.3.4), using the **PUSCH assignment** received in **Msg2** (see **step 2**, click 9.1.3.2). **Each UE-specific Msg3** contains a **unique identity** (click 9.3.4) that is included by its **sending remaining UE**. This **unique identity** is **required** by the **NR node** to solve the **contention** (see **action c**) below) in **each RA contending group** that contains **more than 1 remaining UE**.

**Recall** from **step 2** (click 9.1.3.2) that **all** the **remaining UEs** in **each RA contending group** have received the **same PUSCH assignment**, within the **same received Msg2** (**random access response**, click 8.2). Consequently, when an **RA contending group** has **more than 1 remaining UE**, **each** and **every** of its **remaining UEs transmits** its **UE-specific Msg3** in the **same PUSCH resources**. Therefore, the transmissions by **all** the **remaining UEs** in **each RA contending group** will **overlap** (i.e., they **collide**).

Although **Msg3** is sent through **PUSCH**, **HARQ ACKs** (click 4.5.6.7) are **not** sent in this case, but the **HARQ FEC** might be used.

**b) Each remaining UE waits** for its **Msg4**

**Each remaining UE waits** for the **reception** of its **UE-specific Msg4** (**contention resolution**, click 9.3.5).

Its **UE-specific Msg4** is the one that **contains** the **identity** that the **UE sent** within its **Msg3** from **action a**) above.

**Each remaining UE waits** for `ra-ContentionResolutionTimer` from `RACH-ConfigCommon` (click 9.5.3).

**c)** The **NR node sends 1 Msg4** for **each received Msg3**

In case that **more than 1 remaining UE**, from a **given RA contending group**, **transmits** their **UE-specific Msg3**, they will **collide** (see **action a**) above). In spite of this, it is **likely** (but **not guaranteed**) that the **UE** will be able to **receive 1** of the **transmitted UE-specific Msg3**.

For **each received UE-specific Msg3** there are **2 possible cases**, depending on whether the **remaining UE** that sent the **UE-specific Msg3 has**, or **not**, a **C-RNTI**:

- The **remaining UE has** a **C-RNTI**, ***then***:

  The **NR node sends 1 UE-specific Msg4** as a **DCI message** with the **last 16 bits** of its **CRC masked** with the **UE**'s **C-RNTI**. Notice that in this case, this **remaining UE included** its **C-RNTI** as its **identity** within its **Msg3** from **action a**) above.

- The **remaining UE** does **not** have a **C-RNTI**, ***then***:

  1. The **NR node** sends (through **PDCCH**) **1 UE-specific DCI message** with the **last 16 bits** of the **CRC masked** with the **TC-RNTI** (i.e., the **temporary C-RNTI**) that was **included** by the **Msg2** received in **step 2** (click 9.1.3.2). This **DCI message assigns** the **PDSCH reception** of the **UE-specific Msg4**.
  2. The **NR node** sends **1 UE-specific Msg4** as a "**UE Contention Resolution Identity**" **MAC CE**. **This Msg4** contains the **UE identity** that the **remaining UE sent** within its **Msg3** in **action a**) above.

**d)** The **wait** for **Msg4** by **each remaining UE** is **over**

The **wait** (from **action b**) above) by **each remaining UE** to receive **its UE-specific Msg4** is **over**. The **action** by the **remaining UE** depends on whether **its**

**UE-specific Msg4** has been **received** within `ra-ContentionResolutionTimer` or **not**:

- **Its UE-specific Msg4** has **not** been **received**

  Then, the **remaining UE waits** for the **backoff time received** in **Msg2** from **step 2** (click 9.1.3.2) and **retries** the **4-step CBRA procedure**. Therefore, the **remaining UE** goes **back** to **action a**) from **step 1** (click 9.1.3.1).

- **Its UE-specific Msg4** has been re**ceived**

  Then, the **remaining UE continues** in **step 4** (click 9.1.3.4). The **remaining UEs** that **continue** in **step 4** are called the "**successful UEs**".

The reasons why a **remaining UE might not receive** its **UE-specific Msg4** are:

- The **NR node** did **not receive** the **Msg3** from the **remaining UE** due to **channel impairments**. Consequently, the **NR node** did **not** send the **UE-specific Msg4**.
- The **NR node received** the **Msg3** from the **remaining UE**, but **either** the **DCI message** that **scheduled** the **reception** of its **UE-specific Msg4**, or the **UE-specific Msg4** itself, was **lost** due to **channel impairments**. This case **only** applies when the **remaining UE** does **not** have **C-RNTI**.
- The **NR node received** the **Msg3** from the **remaining UE**, but the **DCI message** that conveys the **UE-specific Msg4** was **lost** due to **channel impairments**. This case **only** applies when the **remaining UE** has a **C-RNTI**.

Regardless of the **cause**, if the **remaining UE** did **not receive** its **UE-specific Msg4**, the **remaining UE** will **react** as indicated at the **beginning** of **this action**.

### 9.1.3.4 Step 4: each successful UE receives its UE-specific Msg4 (Contention resolution)

In case that a **successful UE received** its **UE-specific Msg4** (**contention resolution**, click 9.3.5) as a **MAC PDU**, the **UE** sends a **HARQ ACK** (click 4.5.6.7).

When a **successful UE** has **received** its **UE-specific Msg4**, there are **2 possible cases**:

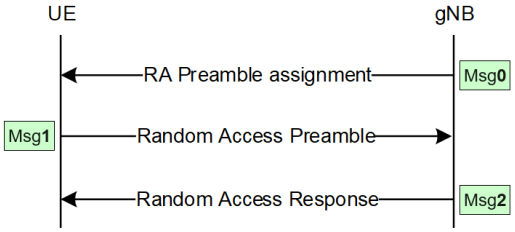
- ***T***he **successful UE has** a **C-RNTI**, ***then***, it **keeps** its **C-RNTI** and **discards** the **received TC-RNTI** (i.e., the **temporary C-RNTI**).
- ***Else***, the **successful UE promotes** its **TC-RNTI** to its **C-RNTI**.

In **either** case, the **successful UE** has **successfully completed** its **4-step CBRA procedure**.

In case that this **RA procedure** was for **initial access** (click 8) the **UE-specific Msg4** is an `RRCConectionSetup` (**DL-CCCH**, click 5.8.5.3) **message**.

### 9.1.4 The 4-step CFRA procedure

Its **time diagram** is:




The **3 major message exchanges** ("**steps**") of the **4-step CFRA procedure** are:

1. The **UE receives Msg0** (**RA preamble assignment**, click 9.3.1): see subsection 9.1.4.1.
2. The **UE sends Msg1** (**RA preamble**, click 9.3.2): see subsection 9.1.4.2.
3. The **UE receives** the **correct Msg2** (**RA response**, click 9.3.2): see subsection 9.1.4.3

For more detail on the **parameters** for the **4-step CFRA procedure**, click 9.5.

For a list of **cases** where a **CFRA procedure** may be used, click 9.4.2.

#### 9.1.4.1 Step 1: the UE receives Msg0 (RA preamble assignment)

The **actions** (not necessarily sequential) within this **step 1** of the **4-step CFRA procedure** are:

**a)** The **UE receives** the **Msg0** (**RA preamble assignment**, click 9.3.1) that was **sent** by the **NR node**. **Msg0** is **usually** sent as a **DCI Format 1_0 message** (click 6.9.5), but it can **also** be sent as a **RRC message** through **PDSCH**. In the **4-step CFRA procedure**, **Msg0** does **not activate** a **configured grant** for **PUSCH**.

**b)** The **UE** performs **resource selection** (click 9.2) for the **RA procedure**.

**c)** The **UE** passes to **step 2** (click 9.1.4.2).

#### 9.1.4.2 Step 2: the UE sends Msg1 (RA preamble)

The **actions** (not necessarily sequential) within this **step 2** of the **4-step CFRA procedure** are:

**a)** The **UE** then sends the **RA preamble sequence** that was indicated by the **NR node** in the **Msg0** from the **previous step**.

**b)** The **UE** then **waits** for the reception of a **correct Msg2** (click 9.3.3).

The **correct Msg2** is the one that contains in its **Random Access Identifier** (**RAPID**) **field** the **index** of the **RA preamble sequence** that was sent by the **UE** in the **previous step**.

The **UE waits** along the **period** `ra-responseWindow` from `RACH-ConfigGeneric` (see at the end of this subsection).

**c)** The **wait** from the **previous action** may **end** with **1 among** the following **3 outcomes**:

- The **UE** does **not receive** the **correct Msg2** within the said **period**.

  Then, the **UE retransmits Msg1**.

- The **UE** does **not receive** the **correct Msg2** after a certain **quantity** of **retransmissions**.

  Then, the procedure is **aborted**.

- The **UE receives** the **correct Msg2**.

  Then, the **UE passes** to the **next step**.

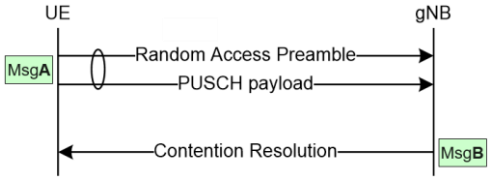
In the **4-step CFRA procedure**, the **instance** of `RACH-ConfigGeneric` (click 9.5.5) is the **son** of an **instance** of `CFRA` (click 9.5.8). This **instance** of `CFRA` is the **son** of an **instance** of `RACH-ConfigDedicated` (click 9.5.10).

#### 9.1.4.3 Step 3: the UE receives the correct Msg2 (RA response)

The **UE** has **successfully completed** its **4-step CFRA procedure**.

### 9.1.5 The 2-step CBRA procedure

Its **time diagram** is:




The **2 major message exchanges** ("**steps**") of the **2-step CBRA procedure** are:

1. **Each RA UE sends** its **MsgA** (**RA preamble** and **PUSCH payload**, click 9.3.6): see subsection 9.1.5.1.
2. **Each UE receives** a **contention resolution MsgB** (click 9.3.7): see subsection 9.1.5.2.

For more detail on the **parameters** for the **2-step CBRA procedure**, click 9.5.

#### 9.1.5.1 Step 1: each RA UE sends its MsgA

The **actions** (not necessarily sequential) within this **step 1** of the **2-step CBRA procedure** are:

**a)** **One** (or **more**) **UEs transmit**(**s**) the **RA preamble** of **MsgA**

**One** (or **more**) **UEs transmit**(**s**) the **RA preamble** (click 9.3.2) through **PRACH** (click 7.1) in a **given PRACH occasion** (click 7.2.2.1). We will call **all these UEs** the "**RA UEs**".

**Each RA UE independently selected** its **PRACH occasion** based on the **PRACH scheduling** (click 9.2), but they **all** happened to **transmit** in the **same PRACH occasion**.

**Each RA UE** sends an **RA preamble sequence randomly selected** (click 9.2.1) by itself.

**b)** **Each** and **every RA UE computes** its **RA-RNTI**.

**Each** and **every RA UE computes** (click 9.1.7) the **same value** of **RA-RNTI**. They compute the **same value** because **all** the **RA UEs transmitted** their **RA preamble** in the **same PRACH occasion**, and the **RA RNTI** is **only computed** from the **variables** of the **PRACH occasion**.

**c)** **Each** and **every RA UE** sends its **PUSCH payload** of **MsgA**

**Each RA UE** transmits its **PUSCH payload** in the **same frequency-time block** as its **RA preamble** from **action a**) above. **Each** and **every PUSCH payload** includes a **unique identity** for the **UE** that sent it. This **unique identity** is **required** by the **NR node** to solve the **contention** (see **action e**) below) in **each RA contending group** that contains **more than 1 RA UE**.

**Each PUSCH payload** is sent in a **PUSCH transmission occasion** that belongs to the **configured grant** that the **RA UE** has been **configured** by the **NR node**.

**Each set** of **RA UEs** that **uses** the **same PUSCH transmission occasion** to transmit is **PUSCH payload** is **1 RA contending group**.

In case that **more than 1 RA UE** uses the **same PUSCH occasion** (**PO**) from its **configured grant**, they will **transmit** over the **same PUSCH resources**.

When a **RA contending group** contains **more than 1 RA UE**, **all** of these **RA UEs transmit** their **Msg3** over the **same PUSCH resources**. Consequently, **all** of their **transmissions** will **superimpose** (i.e., **collide**) and **contention occurs**. **Contention** is **solved** by the **NR node** (see **action e**) below).

Although **Msg3** is sent through **PUSCH**, **HARQ ACKs** (click 4.5.6.7) are **not** sent in this case, but the **HARQ FEC** might be used.

**d)** **Each RA UE waits** for the reception of a **MsgB** (click 9.3.7). **Each RA UE waits** for **MsgB up to** a **certain quantity** of **slots**. This **quantity** of **slots** that **each RA UE waits** is called the **RA response window**. **Each** and **every RA UE** gets the **value** of its **RA response window** from **either** `msgB-ResponseWindow-r16` or `msgB-ResponseWindow-v1700`, **both** of them from `RACH-ConfigGenericTwoStepRA-r16` (click 9.5.9).

In the **2-step CBRA procedure**, **MsgB** can be **either** a **Contention Resolution** or a **Fallback Indication**.

**e)** The **NR node sends 1 MsgB** for **each received MsgA**

It is **likely** (but **not guaranteed**) that the **NR node correctly receives some** (or even **all**) of the **different RA preamble sequences** that were sent by the **RA UEs** (see **action d**) from 9.1.3.1).

In case that **more than 1 remaining UE**, from a **given RA contending group**, **transmits** their **PUSCH payload**, they will **collide** (see **action c**) above). In spite of this, it is **likely** (but **not guaranteed**) that the **UE** will be able to **receive 1** of the **transmitted PUSCH payloads**.

In case that the **NR node** receives an **RA preamble** but it does **not** receive the **corresponding PUSCH payload**, it **either** sends to the **UE** a **MsgB** with a **fallback indication**, or **does nothing** (i.e., the **NR node** expects the **UE** to **restart** its **RA procedure**).

For **each received complete MsgA**, the **NR node sends 1 contention resolution MsgB**.

**Several MsgBs** may be sent within **1 MSGB MAC PDU**, for efficiency reasons.

**Each MSGB MAC PDU** is **scheduled** with **1 DCI Format 1_0 message** (click 6.9.5).

The **DCI Format 1_0 messages** are sent through **PDCCH**, within a **CSS type1** in the **SpCell**.

**f)** The **wait** from **action d)** above may **end** with **1 among** the **4** following **outcomes**:

- **MsgB** is **not received** within the **RA response window**.

  Then, the **UE retransmits** its **MsgA** with transmission **power increased** and goes back to **action b)** above.

- **MsgB** is **not received** after a certain quantity of **retransmissions**:

  Then, the **UE** aborts the **RA procedure**.

- **MsgB** is **received** within the **RA response window**, and it is a **Fallback indication**.

  Then, the **RA procedure** continues in **step 3** (click 9.1.3.3) of the **4-step CBRA procedure** (click 9.1.3). In this case, the **UE** sends its **Msg3** in the **Uplink assignment** contained in the **received MsgB**. Notice that the **Fallback variant** of the **2-step CBRA** procedure turn into a **4 step procedure**.

- **MsgB** is **received** within the **RA response window**, and it is a **Contention Resolution**.

  Then, **UE continues** in **step 2** (click 9.1.5.2).

In the **2-step CBRA procedure**, the **instance** of `RACH-ConfigGenericTwoStepRA-r16` (click 9.5.9) is the **son** of an **instance** of `RACH-ConfigCommonTwoStepRA-r16` (click 9.5.8). This **instance** of `RACH-ConfigCommonTwoStepRA-r16` is the **son** of an **instance** of `MsgA-ConfigCommon-r16` (click 9.5.6).

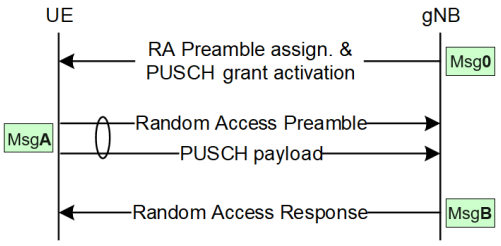
For more detail on the **parameters** for the **2-step CBRA procedure**, click 9.5.

### 9.1.5.2 Step 2: each successful UE receives a contention resolution MsgB

**Each successful UE** has **successfully completed** its **2-step CBRA procedure**.

## 9.1.6 The 2-step CFRA procedure

Its **time diagram** is:



The **3 major message exchanges** (“**steps**”) of the **2-step CFRA procedure** are:

1. The **UE receives Msg0** (**RA preamble assignment**, click 9.3.1): see subsection 9.1.6.1.
2. The **UE sends MsgA** (**RA preamble** and **PUSCH payload**, click 9.3.6): see subsection 9.1.6.29.1.4.2.
3. The **UE receives MsgB** (**RA response**, click 9.3.7): see subsection 9.1.6.3.

For more detail on the **parameters** for the **2-step CFRA procedure**, click 9.5.

For a list of **cases** where a **CFRA procedure** may be used, click 9.4.2.

### 9.1.6.1 Step 1: the UE receives Msg0 (RA preamble assignment)

The **actions** (not necessarily sequential) within this **step 1** of the **2-step CFRA procedure** are:

**a)** The **UE receives** the **Msg0** (**RA preamble assignment**, click 9.3.1) that was **sent** by the **NR node**. **Msg0** is **usually** sent as a **DCI Format 1_0 message** (click 6.9.5), but it can **also** be sent as a **RRC message** through **PDSCH**.

**b)** The **UE activates** a **configured grant** for **PUSCH**.

**c)** The **UE** performs **resource selection** (click 9.2) for the **RA procedure**.

**d)** The **UE** passes to **step 2** (click 9.1.6.2).

### 9.1.6.2 Step 2: the UE sends MsgA (RA preamble and PUSCH payload)

The **actions** (not necessarily sequential) within this **step 2** of the **2-step CFRA procedure** are:

**a)** The **UE sends MsgA** (**RA preamble** and **PUSCH payload**, click 9.3.6). The **RA preamble sequence** is the one **indicated** by the **NR node** in **Msg0**, in **action a**) of the **previous step**. The **PUSCH payload** is **transmitted** using the **configured grant** that was **activated** with the said **Msg0**.

**b)** The **UE waits** for the reception of a **MsgB** (click 9.3.7). **The UE waits** for **MsgB up to** a **certain quantity** of **slots**. This **quantity** of **slots** that the **UE waits** is called the **RA response window**. The **UE** gets the **value** of its **RA response window** from **either** `msgB-ResponseWindow-r16` or `msgB-ResponseWindow-v1700`, **both** of them from `RACH-ConfigGenericTwoStepRA-r16` (click 9.5.9).

**c)** The **wait** from the previous **action** may **end** with **1 among** the **4** following **outcomes**:

- **MsgB** is **not received** within the **RA response window**.

  Then, the **UE retransmits** its **MsgA** with transmission **power increased** and goes back to **action b**) above.

- **MsgB** is **not received** after a certain quantity of **retransmissions**:

  Then, the **UE** aborts the **RA procedure**.

- **MsgB** is **received** as a **RA response**

  Then, **UE continues** in **step 3** (click 9.1.6.3).

#### 9.1.6.3 Step 3: the UE receives MsgB (RA response)

The **UE** has **successfully completed** its **2-step CFRA procedure**.

### 9.1.7 Computation of RA-RNTI

The **value** of the **RA-RNTI** (click 6.9.10) is **computed** from the **PRACH time occasion** (click 7.2.2) of the **transmission** of the **RA preamble** (7.3.1). **Both** the **UE** and the **NR node compute** the **RA-RNTI independently**, but over the **same variables** (the **PRACH occasion**). In this way, the **UE** and the **NR node** can **share** the same **RA-RNTI value** even though this value has **not** been explicitly **conveyed**.

The computation of **RA-RNTI** is:

$$\textbf{RA-RNTI} = 1 + s_id + 14 \times t_id + 14 \times 80 \times f_id + 14 \times 80 \times 8 \times ul_bnd$$

where:

- "*s_id*" is the **index** (within the **slot**) of the **first symbol interval** of the **RA preamble transmission** "$0 \leq s_id < 14$".
- "*t_id*" is a **slot index** (within the **frame**). Its **range** is from **0** to **79**. Its semantics depends on the value of "**μ**":
  - For "$\mu \in \{0, 1, 2, 3\}$",

    "*t_id*" is the **first slot** of the **RA preamble transmission** in a **system frame**. The **subcarrier spacing** to determine "*t_id*" is based on the value of "**μ**" specified in clause 5.3.2 in TS 38.211
  - For "$\mu \in \{5, 6\}$"

    "*t_id*" is the **slot index** of the **120 kHz slot** in a **system frame** that contains the **RA preamble transmission**.
- "*f_id*" is the **index** of the **RA preamble transmission** in the **frequency domain**. Its **range** is from **0** to **8**.
- "*ul_bnd*" depends on the **type** of **band** where the **RA preamble** was **transmitted**: **0** if sent in an **FDD** band (sometimes called "**normal Uplink**"), **1** if sent in a **SUL** band.

The **resulting RA-RNTI** can have the **same value** as an **already assigned C-RNTI**, because they share the **same range of values**. However, this is **not** a **problem** because the **procedures** and **control data** allow to **avoid confusing** one with the other.

For more detail on the **RA-RNTI calculation**, refer to clause 5.1.3 of TS 38.321.

## 9.2 RA resource selection

**Before** the initiation of a **random access procedure**, the **UE** needs to **select** the following **resource**s for the **RA procedure**:

- A value of **ssb-time-domain-index**

  The **UE** selects this by **either**: **a**) **UE beam selection** (click 6.1.2.3); **or b**) an **indication** received from the **NR node**.

  The **UE performs beam selection only** when the **NR node** uses **Tx beam sweeping**.

The **indication** sent by the **NR node** to the **UE** can be **either Msg0** (click 9.3.1), or a **specific PDCCH Order** (click 9.2.6).

- An **RA preamble sequence**

  The **UE** selects this by **either**: **a**) a **random decision**, **or b**) an **indication** sent by the **NR node** to the **UE**.

  A **random decision** is used in **CBRA procedures** (click 9.1.2). The **random decision** is made by assigning an **equal probability** to **each** of the **RA preamble sequences** within the **set** that **corresponds** to the **value** of the **ssb-time-domain-index** (click 6.1.2.3).

  An **indication** is used in **CFRA procedures** (click 9.1.2). The **indication** sent by the **NR node** can be **either Msg0** (click 9.3.1), or a **specific PDCCH Order** (click 9.2.6).

- The **PRACH transmission occasion** (i.e., the **PRACH frequency-time resources**)

  This depends on the **value** of the **ssb-time-domain-index** (click 6.1.2.3).

- The **target** received **power** for the **RA preamble**

  The **UE** selects a **value** for the **first RA procedure attempt**, and the value is **ramped up** in **every successive retry** due to a **RA procedure failure**.

The **resource selection** has some **variations** depending on the **type** of **RA procedure**, on the **RA procedure indication message** and/or on the **RA procedure triggering event**. Some of these **variations** are:

- **RA resource selection** for **CFRA procedures** (click 9.2.2)
- **RA resource selection** for **beam failure recovery** (click 9.2.3)
- **RA resource selection** for **SI request** (click 9.2.4)
- **RA resource selection** for `ReconfigurationWithSync` (click 3.9.8) (click 9.2.5)
- **RA resource selection** for **PDCCH Order** (click 9.2.6)

### 9.2.1 RA preamble sequence, PRACH occasion and beam sweeping

When the **NR node** uses **Tx beam sweeping** (click 6.1.2.3), the **NR node** needs to know **what** is the **beam** that has been **selected** by the **UE**, in order to use the **said beam** when transmitting to the **UE**. The **NR node** learns the **ssb-time-domain-index value** of the **beam** that was **selected** by the **UE** from **both**:

- The **PRACH occasion** (i.e., the **frequency-time PRACH resources**) that the **UE used** to transmit its **RA preamble** (click 7.3.1).

  This is why **each PRACH transmission occasion** is **mapped** to a **certain quantity** of **ssb-time-domain-index values** (i.e., to a **certain quantity** of **beams**).

- The **RA preamble sequence** sent by the **UE**.

  This is why **each ssb-time-domain-index value** is mapped to **1 set** that contains a certain **quantity** of **RA preamble sequences**, where the **different sets** may have **non-empty intersection** (i.e., the **sets** may "**overlap**").

The following example may clarify this:

- The **cell** has **8 beams**. **Each** of them is **identified** with **1 value** from **0** to **7** of **ssb-time-domain-index**.

  There is **1 PRACH occasion** for **every 4 beams**. Consequently, the **CHOICE** within **ssb-perRACH-OccasionAndCB-PreamblesPerSSB** (from **RACH-ConfigCommon**, click 9.5.3) is **four**. The **maximum quantity** of **RA preamble sequences** per **beam** is therefore **16** (**64** divided by **4**). Let us use **this maximum quantity**, which implies that the value of "**four**" is **16**.

- **Each PRACH occasion** serves **4 beams**, where **each beam** uses **1** among **16 RA preamble sequences**. The mapping could be:

| ssb-time-domain-index | PRACH Occasion | RA preamble sequences |
|---|---|---|
| 0 | A | 0 to 15 |
| 1 | A | 16 to 31 |
| 2 | A | 32 to 47 |
| 3 | A | 48 to 63 |
| 4 | B | 0 to 15 |
| 5 | B | 16 to 31 |
| 6 | B | 32 to 47 |
| 7 | B | 48 to 63 |

- It is then possible for the **NR node** to determine the **selected beam** (i.e., the value of **ssb-time-domain-index**) from **both** the **PRACH occasion** and the **RA preamble sequence**.

The **relationship** between the **ssb-time-domain-index** of the **selected beam**, the **PRACH occasions** and the **RA preamble sequences** is based on **RRC parameters**. The **parameters** that are used depend on the **type of RA procedure** where the **PRACH transmission** takes place:

- For a **4-step procedure** (click 9.1.1)

  The parameter is **ssb-perRACH-OccasionAndCB-PreamblesPerSSB** from **RACH-ConfigCommon** (click 9.5.3).

- For a **2-step procedure** (click 9.1.1)

  The parameter is **msgA-CB-PreamblesPerSSB-PerSharedRO-r16**. The **PRACH transmission** can be on a **subset** of **PRACH occasions** associated with a given **ssb-time-domain-index** within an **SSB-RO mapping cycle** for a **UE** that is provided with a **PRACH mask index** by **msgA-SSB-SharedRO-MaskIndex**. **Both parameters** are from **RACH-ConfigCommonTwoStepRA-r16** (click 9.5.8).

For more detail on the relationship between **PRACH scheduling** and **beam sweeping**, refer to clause 8 of TS 38.213.

### 9.2.2 RA resource selection for CBRA/CFRA procedures

When the **NR node** uses **Tx beam sweeping** (click 6.1.2.3), the **UE** decides on **1 ssb-time-domain-index** (i.e., **1 beam**) based on **Reference Signal Received Power** (**RSRP**) measurement of the **SSBs received** on **different beams** during **cell search**.

***If*** the **UE** receives **SSBs** where the **Reference Signal Received Power** (**RSRP**) of the **Synchronization Signal** (**SS**) is above `rsrp-ThresholdSSB`, ***then***, the **UE** selects **1** of these **SSBs**. ***Else***, the **UE** selects **any SSB**.

### 9.2.3 RA resource selection for beam failure recovery (BFR)

For **beam failure recovery**, a **CFRA procedure** is used if an **instance** of `BeamFailureRecoveryConfig` (click 9.5.4) is **present**. Otherwise, **CBRA** is used.

If **CFRA** is used, the **UE** may be configured with a **set** of a **candidate beams** via `candidateBeamRSList` from `BeamFailureRecoveryConfig` (click 9.5.4) (**SSB** and/or **CSI-RS**) and their resources (Resource ID, Preamble ID, **PRACH occasion**, etc…).

- If at least one of the candidate beam's **RSRP** is above a configured threshold (`rsrp-ThresholdSSB` from `BeamFailureRecoveryConfig` (click 9.5.4)) the **UE** selects an **SSB** or **CSI-RS.**
- Once a **Downlink** beam is identified, the preamble would be transmitted on the associated **Uplink** beam.
- The **UE** uses the corresponding (click 9.5) `ra-PreambleIndexo` (if **present**) that corresponds to the selected **SSB** or **CSI-RS**.
- If **CSI-RS** resource is configured with `ra-PreambleIndex`, the **UE** uses the `ra-PreambleIndex` corresponding to the **SSB** in `candidateBeamRSList` which is quasi-collocated with the selected **CSI-RS**.

By transmitting an **RA preamble** on the **dedicated RA resources** (click 9.2) corresponding to the selected **CSI-RS/SSB**, the **UE not only** informs the network that it has **detected beam failure** but **also** indicates the **selected beam**.

### 9.2.4 RA resource selection for SI request (On-demand SI)

For **UEs** in **RRC_IDLE** or in **RRC_INACTIVE** states, a request for **other SIBs** and **posSIBs** (`RRCSsytemInformation` message, click 5.8.5.7) triggers an **RA procedure**. If the network configures the **UE** with **RA resources** (click 9.2) for **SI request**, **CFRA** is used, otherwise **CBRA** is used:

- The network may configure the **UE** with dedicated **RA resources** for **SI request** with `si-RequestConfig` from `SI-SchedulingInfo` (click 5.5.3.1) and possibly also with `si-RequestConfigSUL` (for **SUL**). In this case **Msg1** (9.3.2) is used to indicate the requested **other SIBs** and **posSIBs**. The **data type** of the two said parameters is `SI-RequestConfig` (click 5.5.3.4). Parameter `si-SchedulingInfo` from `SIB1` is an instance of data type `SI-SchedulingInfo` (click 5.5.3.1)
- **UE** selects an **RA preamble** corresponding to the selected **SSB**, from the **RA preamble**(s) determined according to `ra-PreambleStartIndex` from `SI-RequestConfig` (click 5.5.3.4).

- When **Msg1** is used, the **minimum granularity** of the request is **one RRC** `SystemInformation` **message** (i.e. **one** set of **SIBs**).
- **1 RA preamble** and/or **1 PRACH resource** can be used to request **multiple SI messages**.
- The **NR node** acknowledges the request in **Msg2** (click 9.3.3).

When **CBRA** is used for **SI request** via **Msg3** (click 9.3.4), the **NR node** acknowledges the request in **Msg4** (click 9.3.5).

### 9.2.5 RA resource selection for reconfiguration with sync

For the procedure of `ReconfigurationWithSync` (click 3.9.8) (e.g., **handover**), the **RA** configuration to be used is provided by the **NR node** within an **instance** of `RACH-ConfigDedicated` (click 9.5.10), within `ReconfigurationWithSync`. Its **optional** parameter `cfra` provides the required parameters for **CFRA** to a given **target cell**. If `cfra` is **absent**, the **UE** performs **CBRA**.

For **CFRA**, the **UE** is configured with **SSB** or **CSI-RS RA resources** (click 9.2).

If the configured **CFRA RA resources** are for **SSB**, the **UE** selects an **SSB** with **SS-RSRP** above `rsrp-ThresholdSSB` amongst the associated **SSBs** and then picks the corresponding **RA resources**.

Similarly, if the configured **CFRA RA resources** are for **CSI-RS**, the **UE** selects a **CSI-RS** with **CSI-RSRP** above `rsrp-ThresholdCSI-RS` amongst the associated **CSI-RSs** and then picks the corresponding **RA resources**.

### 9.2.6 RA resource selection for PDCCH Order

A **DCI Format 1_0 message** with its **CRC masked** with **C-RNTI** (click 6.9.5.1) and its field "***Frequency domain resource assignment***" set to **all 1**'s indicates that the received **DCI message** is for an **RA procedure** initiated by a **PDCCH Order**. This **DCI message** contains the following fields:

- ***Random Access Preamble Index*** (6 bits)
- ***UL/SUL indicator*** (1 bit)

  If the **UE** is configured with **SUL** in the `ServingCellConfig` (click 3.9.7) instance of this **cell**, this **field** indicates the **Uplink carrier** in the **cell** to transmit the **PRACH**, otherwise, this field is reserved.
- ***SS/PBCH index*** (6 bits).

  If the value of the **field** "***Random Access Preamble Index***" (see above) is **not all zeros**, this field indicates the value for the **ssb-time-domain-index** that the **UE must** use to determine the **PRACH occasion** for the **PRACH** transmission. Otherwise, this field is reserved.
- ***PRACH Mask index*** (4 bits)

  If the value of the **field** "***Random Access Preamble Index***" (see above) is **not all zeros**, this field indicates the **PRACH occasion** associated with the value of **ssb-time-domain-index** **instance** indicated by the **field** "***SS/PBCH index***" (see above) for the **PRACH** transmission. Otherwise, this field is reserved.

***If*** the "***Random Access Preamble Index***" **field** in the **PDCCH Order** is **not all zeros**,

***then***, the **UE** performs **CFRA** on the **SSB** signaled by **PDCCH Order**.

***Else,*** (i.e., "***Random Access Preamble Index***" **is all zeros**) the **UE** performs **CBRA**.

Note: The **RA** procedure on an **SCell** shall **only** be initiated by a **PDCCH Order** with the **field** "***Random Access Preamble Index***" **different** from **all zeros**.

## 9.3 Messages for RA procedures

This section describes **each** of the **messages** that are used in the **different random access procedures**.

### 9.3.1 Msg0 (Downlink, RA preamble assignment with(out) PUSCH grant activation)

**Msg0** is **sent** by the **NR Node** to **1 UE**, in order to signal the **UE** to **start** a **contention-free random access** (**CFRA**, click 9.1.2) **procedure**.

**Msg0** is **only** sent, and is **always** sent, in **CFRA procedures**.

**Msg0** contains a **PRACH assignment** (click 9.1) and an **indication** for the **UE** to use a **UE-specific RA preamble sequence** (click 7.3.3). In order to **send Msg0**, the **NR node must know** the **UE** to which it is **indicating** the **RA preamble sequence** to **use**, and consequently, there is **no contention**.

For the case of **2-step CFRA** (click 9.1.6), the **reception** of **Msg0 activates** a **PUSCH grant**, that the **UE** uses as soon as possible to send its **MsgA** (click 9.3.6).

**Msg0** is **sent** by the **NR node** as **either** a **DCI message** or an **RRC message**, depending on **different factors**:

- In **most cases**,

  **Msg0** is **sent** as a **DCI Format 1_0 message** (click 6.9.5) through **PDCCH**. For more detail, click 9.3.1.1.

- In **some cases**, in particular for **dedicated access** (e.g., **handover** or network slicing),

  **Msg0** is **sent** as an `RRCReconfiguration` (or similar, click 9.4.2) **RRC message** through **PDSCH**. For more detail, click 9.3.1.2.

For **random access** in a **cell** configured with **SUL**, the **NR node** can **explicitly signal** which **carrier** to use (**UL** or **SUL**). Otherwise, the **UE** selects the **SUL carrier if and only if** the measured **quality** of the **Downlink** is **lower** than a **broadcast threshold**. The **UE** performs **carrier selection before** deciding between **2-step** and **4-step RA procedures**. The **RSRP threshold** for **selecting** between **2-step** and **4-step RA** can be configured separately for **UL** and **SUL**. Once started, **all Uplink transmissions** of the **RA procedure** remain on the **selected carrier**.

The network can associate a **set** of **RA resources** (click 9.2) with feature(s) applicable to a **random access procedure**: **Network Slicing** (clause 16.3 of TS 38.300), **RedCap** (clause 16.13), **small data transmission** (**SDT**, clause 18), and **NR coverage enhancement** (clause 19). A set of **RA resources** associated with a **feature** is only valid for **random access procedures** applicable to at least that feature; and a **set** of **RA resources** associated with several features is only valid for **random access procedures** having at least all of these features. The **UE** selects the set(s) of applicable **RA resources**

**after** having selected the **Uplink carrier** (i.e., **UL** or **SUL**) and the **BWP**, and **before** determining which of the **4 RA procedures** (click 9.1) has to be used.

#### 9.3.1.1 Msg0 sent as a DCI message

**Msg0** is **sent** by the **NR node** (through **PDCCH**) as a **DCI Format 1_0 message** with the **last 16 bits** of its **CRC masked** by **C-RNTI**, **CS-RNTI** or **MCS-C-RNTI** (click 6.9.5.1) and containing the **4** following **fields**:

- "***Random Access Preamble Index***" (6 bits)

  It is the **index** of the **RA preamble sequence** (click 7.3.3) that the **UE** should **transmit** within its **RA preamble** (click 7.3.1).

- "***SS/PBCH Index***" (6 bits)

  ***If*** the value of the "***Random Access Preamble index***" field is **not all zeros**, ***then***, this field indicates the value of the **ssb-time-domain-index** (click 9.2.1) that shall be **used** to **determine** the **PRACH occasion** for the **PRACH transmission**. ***Else***, this field is **reserved**.

- "***PRACH Mask Index***" (4 bits)

  ***If*** the value of the "***Random Access Preamble index***" field is **not all zeros**, ***then***, this field indicates the **PRACH occasion** associated with the **SS/PBCH** indicated in the field "***SS/PBCH index***" for the **PRACH transmission**, according to clause 5.1.1 of TS 38.321. ***Else***, this field is **reserved**.

- "***UL/SUL indicator***" (**1 bit**)

  ***If*** [[the value of the "***Random Access Preamble index***" field is **not all zeros**] and [the **UE** is configured with **supplementaryUplink** in **ServingCellConfig** (click 3.9.7) in the **cell**]], ***then***, this field indicates which **Uplink carrier** in the **cell** to transmit the **PRACH** according to Table 7.3.1.1.1-1 of TS 38.212. ***Else***, this field is **reserved**.

The **first 3 fields only appear** in the **DCI Format 1_0 message** when it is masked with **either** of the **said RNTIs** (see above).

#### 9.3.1.2 Msg0 sent as an RRC message

**Msg0** is **sent** by the **NR node** (through **PDSCH**) as an **RRCReconfiguration** (click 5.8.5.4), (or similar, click 9.4.2) **RRC message**.

The **RRC message** contains an **indication** for the **UE** to use a **UE-specific RA preamble sequence** (click 7.3.3) for its **Msg1** (click 9.3.2), other **RA preamble** (click 7.3.1) configuration, **timing parameters**, and a **PRACH occasion** for the **RA transmission**.

This case is used for **UE-specific configurations**, such as **handover** or **network slicing**.

Source web page:

- wirelessbrew.com

### 9.3.2 Msg1 (Uplink, Random Access preamble)

**Msg1** is **sent** by **each RA UE** (click 9.1.3.1) to the **NR node**, in order to:

- Convey to the **NR node** the **index** of the **RA preamble sequence** (click 7.3.3) from the **UE** to the **NR node**.

- Convey to the **NR node** the **RA-RNTI**.
- Convey to the **NR node which beam** has been **selected** by the **UE** (**only** if the **NR node** uses **Tx beam sweeping**).
- Allow the **NR node** to determine the **timing advance** required by the **sending UE**.
- Signal the **NR node** the **presence** of the **UE** in the **serving cell** (**only** in **initial access**).

**Msg1** is **only** sent, and is **always** sent, in the **4-step** (click 9.1.1) **RA procedures**.

**Msg1** is **sent** by the **UE** (through **PRACH**) as the **RA preamble** (click 7.3.1). The **RA preamble sequence** within the **RA preamble must belong** to the **cell**'s **set** of **64 RA preamble sequences**. The **PRACH occasion** (i.e., the **frequency-time resources**) of the **RA preamble transmission** are determined by the **UE** according to the **PRACH scheduling** (click 7.1).

The information is **conveyed** by the **UE** to the **NR node** in the following way:

- The **index** of the **RA preamble sequence** is conveyed by the **RA preamble sequence itself**.
- The **RA-RNTI** is **computed** (click 9.1.7) by the **NR node** from the **PRACH occasion** where the **RA preamble** has been transmitted. The **NR node** uses the **RA-RNTI** for the **PDSCH scheduling** of **Msg2** (click 9.3.3).
- The **beam** that has been **selected** by the **UE** is **jointly conveyed** by the **index** of the **RA preamble sequence** and by the **PRACH occasion** where the **RA preamble** has been **transmitted**.

Source web pages:

- https://www.nrexplained.com/ra_msg1

### 9.3.3 Msg2 (Downlink, Random Access Response)

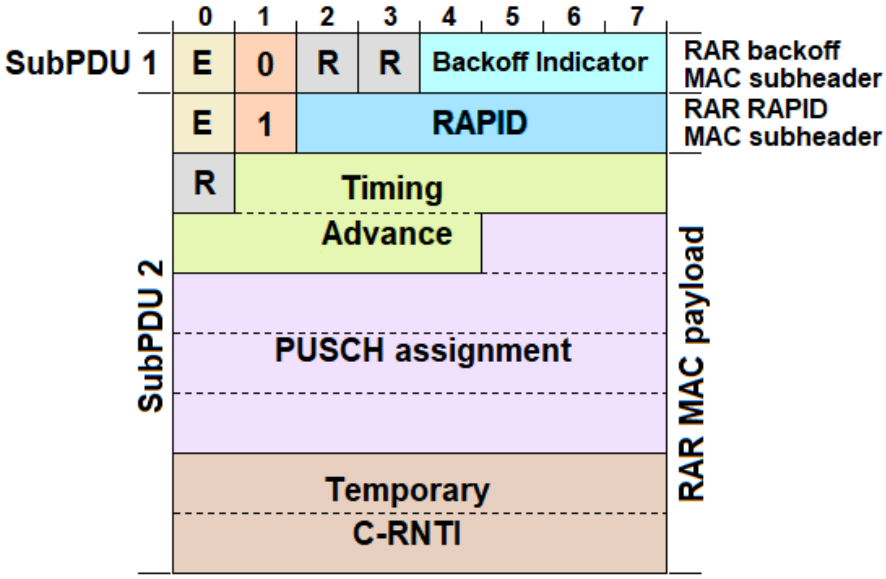
**Msg2** is **sent** by the **NR node** to **1** (or **more**) **RA UE**(**s**) (click 9.1.3.1), in order to:

- Provide the **RA UE**(**s**) with a **PUSCH assignment** to allow it to transmit its **Msg3** (**scheduled transmission**, click 9.3.4).
- Provide the **RA UE**(**s**) with a **TC-RNTI** (i.e., a t**emporary C-RNTI,** click 6.9.10.2).
- Provide the **RA UE**(**s**) with a **Timing Advance** (click 2.3.6.1).
- Provide the **RA UEs** with a **backoff time** (click 4.5.5.7).

**Msg2** is **only** sent, and is **always** sent, in the **4-step** (click 9.1.1) **RA procedures**.

**Msg2** is sent by the **NR node** (through **PDSCH**) within a "**Random access response** (**RAR**)" **MAC PDU** (click 4.5.5.7). The **NR node** can **integrate 1** (or **more**) **Msg2** into **1 random access response** (**RAR**) **MAC PDU** (click 4.5.5.7). The **result** is therefore that the **NR node** sends **1** (or **more**) **RAR MAC PDU**(**s**) that **jointly contain all** the **Msg2**.

**Each Msg2** usually contains **1 RAR backoff indicator MAC subheader** (click 4.5.5.7.1), **1 RAR RAPID MAC subheader** (click 4.5.5.7.1) and **1 RAR MAC payload** (click 4.5.5.7.3). The following figure shows **1 such Msg2**:




## 9.3.4 Msg3 (Uplink, Scheduled Transmission)

**Msg3** is sent by **each remaining UE** (click 9.1.3.3) to the **NR node**, in order to provide the **NR node** with a **unique identity** for **each remaining UE**. This is **required** for **contention resolution** (click 9.1.3.3).

**Msg3** is **only** sent, and is **always** sent, in the **4-step CBRA procedure** (click 9.1.3), and in the **fallback variant** (click 9.1.5) of the **2-step CBRA procedure** (click 9.1.5).

**Msg3** is sent by the **UE** (through **PUSCH**) using the **PUSCH assignment** received in **Msg2** (**random access response**, click 9.3.3).

**Msg3** is **sent** by the **UE** as **either** a **MAC PDU** that contains a **C-RNTI MAC CE** or as an **RRC message**, depending on whether the **UE** already **had**, or **not**, a **C-RNTI** at the **initiation** of the **RA procedure**:

- ***If*** the **UE** already **had** a **C-RNTI**, ***then***,

  **Msg3** is **sent** by the **UE** as a **MAC PDU** that contains a **C-RNTI MAC Control Element**: click 9.3.4.1

- ***If*** the **UE** did **not have** a **C-RNTI** (e.g., during **initial access**), **then**,

  **Msg3** is **sent** by the **UE** as an **RRC message**: click 9.3.4.2.

### 9.3.4.1 Msg3 sent as a MAC PDU

Two relevant cases are:

- **Msg3** is **sent** by the **UE** (through **PUSCH**) as a **C-RNTI MAC CE** (click 9.3.4.1.1), within a given **MAC PDU**.
- **Msg3** is **sent** by the **UE** (through **PUSCH**) as a **MAC PDU** that contains `RRCSystemInfoRequest` (click 9.3.4.1.2)

  This is **only** sent for the case of **on-demand System Information** (click 5.3.5).

#### 9.3.4.1.1 Msg3 sent as a C-RNTI MAC CE

**Msg3** is sent by the **UE** (through **PUSCH**) as a **MAC PDU** (click 4.5.5.1) that contains a **C-RNTI MAC Control Element** (**CE**, click 4.5.5.3).

The **C-RNTI MAC CE** is identified by a **MAC subheader** (click 4.5.5.2) with the **logical channel identifier** (**LCID**). It has a fix**ed size** and consists of a **single 16-bit field**. This **field** contains the **UE identity**, which is the **UE**'s **C-RNTI**.

In case the **UE** also needs to transmit **Uplink data** and there is a **Scheduling Request** (**SR**) **problem** (e.g., **SR not configured**, **SR failure** or **Uplink Out-of-Sync**), the **UE** may also include in **Msg3** a "**Buffer Status Report**" **MAC Control Element** to indicate **buffer status** to get appropriate **PUSCH resources** from the **NR node**.

For more detail on a **C-RNTI MAC CE**, refer to clause 6.1.3.2 of TS 38.321.

For more detail on a "**Buffer Status Report**" **MAC CE**, refer to clause 6.1.3.1 of TS 38.321.

For more detail on **MAC PDUs** and **CEs**, refer to clause 6 of TS 38.321.

#### 9.3.4.1.2 Msg3 sent as a MAC PDU with a `RRCSystemInfoRequest`

**Msg3** is sent by the **UE** (through **PUSCH**) as a **MAC PDU** (click 4.5.5.1) that contains a **MAC SDU** (click 4.5.5) that contains a `RRCSystemInfoRequest` (click 5.8.5.14) message (**UL-CCCH**, click 5.8.5.7). The **UE identity** that is required for **contention resolution** can be sent in the **MAC subheader**, because the `RRCSystemInfoRequest` message does **not** contain an **identity parameter**.

For more detail on **MAC PDUs** and **CEs**, refer to clause 6 of TS 38.321.

### 9.3.4.2 Msg3 sent as an RRC message

**Msg3** is sent by the **UE** (through **PUSCH**) as an **RRC message** (a **CCCH SDU**) over **signaling radio bearer 0** (**SRB0**, click 4.5.2.1). For the case of **CBRA procedures**, the **RRC message** has to contain a **UE identity** for the **contention resolution**. The **RRC message** that is sent, and the **parameter** that contains the **UE identity** (for **CBRA procedures**), is as follows:

- ***If*** the **UE** is transitioning from **RRC_IDLE** to **RRC_CONNECTED** (click 5.8.2), ***then***, the **UE** sends **Msg3** as a `RRCSetupRequest` (click 5.8.5.10) message (**UL-CCCH** class, click 5.8.5.7). The **UE identity** is sent in `ue-Identity` from `RRCSetupRequest-IEs` (click 5.8.5.10). Its **data type** is `InitialUE-Identity`, which is a **39-bit string**.
- ***If*** the **UE** is transitioning from **RRC_INACTIVE** to **RRC_CONNECTED** (click 5.8.2), ***then***, the **UE** sends **Msg3** as a `RRCResumeRequest` message (**UL-CCCH** class, click 5.8.5.7) or a `RRCResumeRequest1` message (**UL-CCCH-1** class, click 5.8.5.8). For the **UE identity** in the `RRCResumeRequest` message, see the previous bullet. For the `RRCResumeRequest1` message, the **UE identity** is sent in `resumeIdentity`, from `RRCResumeRequest1-IEs`. Its **data type** is `I-RNTI-Value`, which is a **40-bit string**.
- ***If*** the **UE** is in **RRC Connection Re-establishment** procedure, ***then***, the **UE** sends **Msg3** as a `RRCReestablishmentRequest` (**UL-CCCH** class, click 5.8.5.7) message. The **UE identity** is sent in `ue-Identity` from `RRCReestablishmentRequest-IEs`. Its **data type** is `ReestabUE-`

`Identity`, which is a **`SEQUENCE`** that contains a **C-RNTI** (click 6.9.10.2), a **Physical Cell Id** (click 3.5.3.2) and a **shortMAC-I**.

- ***If*** [the **RA procedure** was initiated by the **MAC sublayer** itself or by the **RRC sublayer**] and [the **RA procedure** is **CBRA** during **synchronous RRC reconfiguration** (e.g., **handover**)], ***then***, the **UE** sends **Msg3** as a `RRCReconfigurationComplete` (**UL-DCCH** class, click 5.8.5.9) message. This particular case of **Msg3** does not include the **UE identity** is sent in the **MAC subheader**.

### 9.3.5 Msg4 (Downlink, Contention Resolution)

**Msg4 resolves** the **possible contention** in the **4-step CBRA procedure** (click 9.1.3). When a **UE** receives its **UE-specific Msg4**, it has **successfully completed** the **CBRA procedure**.

**Msg4** is **only** sent, and is **always** sent, in the **4-step CBRA procedure** (click 9.1.3), and in the **fallback variant** (click 9.1.5) of the **2-step CBRA procedure** (click 9.1.5).

**Msg4** is **sent** by the **NR node** in **1 among 2** different ways, depending on whether the **UE** already **had**, or **not**, a **C-RNTI** at the **initiation** of the **RA procedure**:

- ***If*** the **UE** already **had** a **C-RNTI**, ***then***,

  **Msg4** is **sent** by the **NR node** (through **PDCCH**) as a **DCI message** (click 9.3.5.1).

- ***If*** the **UE** did **not have** a **C-RNTI** (e.g., during **initial access**), ***then***,

  **Msg4** is **sent** by the **NR node** (through **PDSCH**) as a **MAC PDU** that contains a "**UE Contention Resolution Identity**" **MAC CE** (click 9.3.5.2).

#### 9.3.5.1 Msg4 sent as a DCI message

**Msg4** is **sent** by the **NR node** (through **PDCCH**) as a **DCI message** (**either** an **Uplink** or **Downlink assignment**) with the **last 16 bits** of its **CRC masked** with the **UE**'s **C-RNTI**.

#### 9.3.5.2 Msg4 sent as a "UE Contention Resolution Identity" MAC CE

**Msg4** is **sent** by the **NR node** (through **PDSCH**) as a **MAC PDU** that contains a "**UE Contention Resolution Identity**" **MAC CE**. The "**UE Contention Resolution Identity**" **MAC CE** is identified by a **MAC subheader** with **Logical Channel ID 62** and consists of a **single 48-bit field** "**UE Contention Resolution Identity**". In this **field**, the **NR node copies** the **first 48-bits** of the **Uplink CCCH SDU** that it received in **Msg3** (click 9.3.4). This includes the **identity** information that the **UE** conveyed within its **Msg3**.

For more detail on a "**UE Contention Resolution Identity**" **MAC CE**, refer to clause 6.1.3.3 of TS 38.321.

For more detail on **MAC PDUs** and **CEs**, refer to clause 6 of TS 38.321.

### 9.3.6 MsgA (Uplink, RA preamble and PUSCH payload)

**MsgA** consists of **an RA preamble** and a **PUSCH payload** that are **jointly transmitted** in the **same frequency-time resource block**.

**MsgA** is **roughly equivalent** to **Msg1** (**RA preamble**, click 9.3.2) **plus Msg3** (**scheduled transmission**, click 9.3.4).

**MsgA** is **only** sent, and is **always** sent, in the **2-step RA procedures** (click 9.1.1).

**MsgA** is sent by the **UE** by sending the **RA preamble** through **PRACH** and the **scheduled transmission** through **PUSCH**.

**MsgA** is **transform-precoded** (click 4.12.12) when `msgA-TransformPrecoder-r16` from `MsgA-PUSCH-Config-r16` (click 9.5.7) is **set** to `enabled`.

For the **scheduling** of the **PUSCH payload**, click 9.3.6.1.

For the **contents** of the **PUSCH payload**, click 9.3.6.2.

### 9.3.6.1 PUSCH payload scheduling

The **PUSCH payload** within **MsgA** (**RA preamble** and **PUSCH payload**, click 9.3.6) is **transmitted at least "n" symbol intervals after** the **RA preamble** transmission, where **n** depends on the **subcarrier spacing** of the **active Uplink bandwidth part**. If **SCS** is 15 or 30 kHz, "***n***=2"; if SCS is 60 or 120 kHz, "***n***=4". The actual **time offset** between the **PUSCH transmission** and the **start** of **each PRACH slot** is configured with `msgA-PUSCH-TimeDomainOffset-r16`. The **first PUSCH occasion** (**PO**) in **each slot** is defined by a **start-and-length indicator value** (**SLIV**) which is configured with `startSymbolAndLengthMsgAPO-r16`. **Both** parameters are from `MsgA-PUSCH-Resource-r16`.

For the case of **2-step CFRA**, the **reception** of **Msg0** (click 9.3.1) **activates** a **configured grant** for **PUSCH**, that the **UE** uses to send its **PUSCH payload** (as part of **MsgA**). This **PUSCH grant** is **configured** with `msgA-CFRA-PUSCH-r16` from `CFRA-TwoStep-r16` (click 9.5.12), within `RACH-ConfigDedicated` (click 9.5.10). Recall that `RACH-ConfigDedicated` is a descendant of `ReconfigurationWithSync` (click 3.9.8).

### 9.3.6.2 PUSCH payload contents

The **PUSCH payload** within **MsgA** (**RA preamble** and **PUSCH payload**, click 9.3.6) is sent as a **SCH MAC PDU** (click 4.5.5.5).

The **MAC PDU** contains a **unique UE identity** for the purpose of **contention resolution**. It is typically a **randomly generated identifier** (e.g., a **TC-RNTI** (i.e., a **temporary C-RNTI**) or a **MAC** layer **identifier**).

The **MAC PDU** can **also** contain **some** of the following **optional information**:

- **Scheduling Request** (**SR**)

  If configured, the **UE** can include an **SR** to indicate to the **NR node** that the **UE** has **Uplink data** for transmission.

- **Buffer Status Report** (**BSR**)

  The **UE** can include a **BFR** to help the **NR node allocate resources**.

- **RRC message**

  **Only** allowed for specific use cases (e.g., `RRCRequest`, `SIRequest`).

- **Other L2 signaling** (**optional**)

  The **UE may** include **MAC CE** (**Control Elements**) for **other purposes** (e.g., **power headroom reports**).

For more detail on **MAC PDUs** and **CEs**, refer to clause 6 of TS 38.321.

### 9.3.7 MsgB (Downlink, 3 variants)

**MsgB** is **scheduled** with a **DCI Format 1_0 message** (click 6.9.5), placed in a **PDCCH CSS-Type 1**. If the **destination UE** has a **C-RNTI** (**CFRA**, click 9.1.4 and 9.1.6), it is used to **mask** the **last 16 bits** of the **CRC** of the **DCI message**. Otherwise, a **MSGB-RNTI** is used. The **MSGB-RNTI value** is **independently calculated** from the **RA-RNTI** (click 9.1.7) by **both** the **UE** and the **NR node**:

$$\textbf{MSGB-RNTI} = \textbf{RA-RNTI} + 14 \times 80 \times 8 \times 2$$

**MsgB** is sent by the **NR node** (through **PDSCH**) within a **MSGB MAC PDU** (click 4.5.5.8).

**MsgB** has the following **3 variants**,

- **MsgB**: **random access response variant** (click 9.3.7.1)

  It belongs to the **2-step CFRA procedure** (click 9.1.6).

- **MsgB**: **fallback indication variant** (click 9.3.7.2)

  It belongs to the **2-step CBRA procedure** (click 9.1.5).

- **MsgB**: **contention resolution variant** (click 9.3.7.3)

  It belongs to the **2-step CBRA procedure**.

**Each** of these **3 MsgB variants** has **different semantics** and is sent as a **different combination** of **MSGB MAC subPDUs**, all which is presented in the **following subsections**.

For more detail on the **MSGB MAC PDU**, refer to clauses 6.1.5a, 6.2.2a and 6.2.3a of TS 38.221.

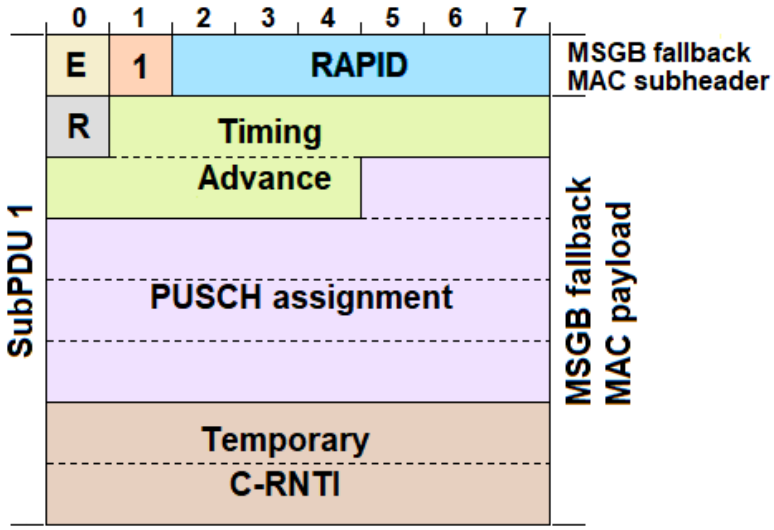
For more detail on **MAC PDUs** and **CEs**, refer to clause 6 of TS 38.321.

#### 9.3.7.1 MsgB: random access response variant

It belongs to the **2-step CFRA procedure** (click 9.1.6).

It has a similar functionality to **Msg2** (**random access response**, click 9.3.3) from the **4-step CBRA procedure** (click 9.1.3), but with **1 difference**: **MsgB terminates the 2-step CFRA procedure**, and therefore, there is **no** need to convey the **backoff** information.

It may be sent as **1 MSGB fallback MAC subheader** (click 4.5.5.7.2) and **1 MSGB fallback MAC payload** (click 4.5.5.7.3), as shown in the following figure:



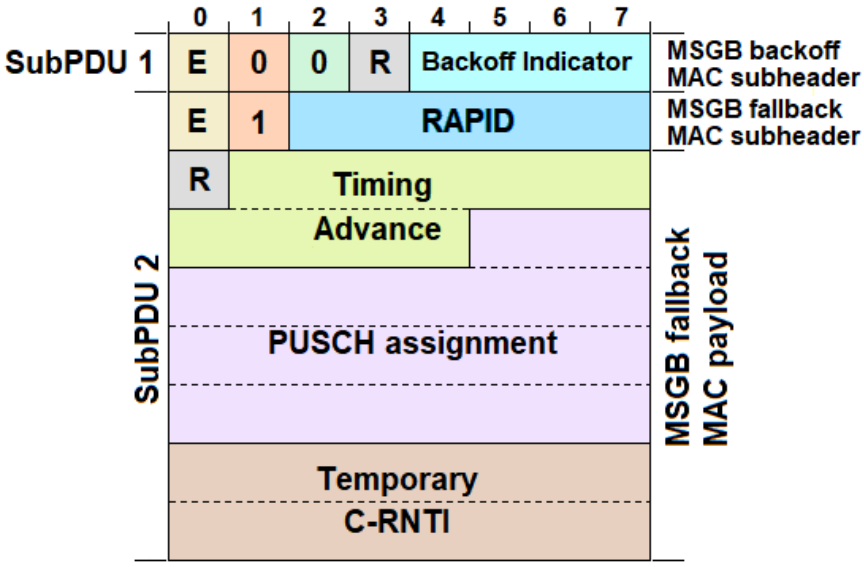

### 9.3.7.2 MsgB: fallback indication variant

It belongs to the **2-step CBRA procedure** (click 9.1.5).

It activates the **Fallback variant** that turns the **2-step CBRA** procedure into a **4-step CBRA** procedure (click 9.1.3).

It has a very similar functionality to **Msg2** (**random access response**, click 9.3.3) from the **4-step CBRA procedure** (click 9.1.3), and this is why the information conveyed in both messages is extremely similar.

It may be sent as **1 MSGB backoff MAC subheader** (click 4.5.5.8.3), **1 MSGB fallback MAC subheader** (click 4.5.5.8.1) and **1 MSGB fallback MAC payload** (click 4.5.5.7.3), as shown in the following figure:



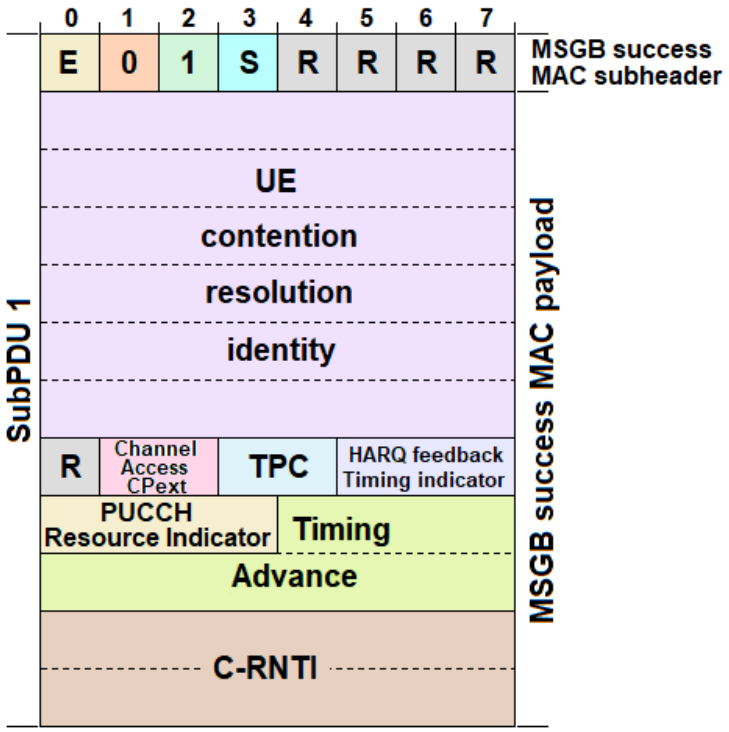

### 9.3.7.3 MsgB: contention resolution variant

It belongs to the **2-step CBRA procedure** (click 9.1.5).

It **successfully completes** the **2-step CBRA procedure**.

It has a similar functionality to **Msg4** (**contention resolution**, click 9.3.5) from the **4-step CBRA procedure** (click 9.1.3).

It may be sent as **1 MSGB success MAC subheader** (click 4.5.5.8.2) and **1 MSGB success MAC payload** (click 4.5.5.8.5), as shown in the following figure:




## 9.4 Triggering and characteristics of a RA procedure

A **random access procedure** can be **triggered** from the **physical layer**, from the **MAC sublayer** or from the **RRC protocol entity** (click 9.4.1).

A **random access procedure** can be **triggered** by **different events** which implies some **specific characteristics** (click 9.4.2)

### 9.4.1 Triggering entities of RA procedure

A **random access procedure** can be **triggered** by the **3** following **entities**:

- **Physical layer**

  The **UE** is instructed to **start** a **RA procedure** by a **PDCCH Order** from the **NR node**. The **PDCCH Order** is a **DCI message Format 1_0 message**. The **UE starts** the **RA procedure** using the values received in the **DCI fields** "***Random Access Preamble index***", "***SSB Index***", "***PRACH Mask Index***" and "***UL/SUL Indicator***".

- **MAC sublayer**

  The **UE randomly** selects a **RA preamble sequence** (click 7.3.17.3.3) that is sent within **Msg**1 as the **first step** of a **CBRA procedure** (click 9.1.2).

- **RRC protocol entity**

  The **RRC protocol entity** launches an **RA procedure** due to various possible causes, such as: **initial access**, **reconstruction**, **handover**, **RRC_INACTIVE**

transition to **RRC_CONNECTED** state, **request** for **other SI**, **RRC request** during **synchronous reconfiguration**, etc.

### 9.4.2 Triggering events and characteristics of RA procedure

A **random access procedure** can be **triggered** by **either** of the following **events** (from clause 9.2.6 of TS 38.300):

- **Initial access** from **RRC_IDLE**.
- **RRC connection resume**: transition from **RRC_INACTIVE** to **RRC_CONNECTED**.
- **RRC connection re-establishment**.
- **RRC synchronous reconfiguration** (for **handover** or **not** for **handover**).
- **Downlink** or **Uplink data** arrival while **UE** is **out** of **sync**. More in detail, **Downlink** or **Uplink data** arrival during **RRC_CONNECTED** or during **RRC_INACTIVE** while **small data transmission** (**SDT**) **procedure** (see clause 18.0 of TS 38.300) is **ongoing**, when **Uplink synchronization status** is "**non-synchronized**".
- **Uplink data arrival without PUCCH SR resources**. More in detail, the **NR node** receives **Uplink data** from a **UE** that is in **either RRC_CONNECTED** or during **RRC_INACTIVE**, while **small data transmission** (**SDT**) **procedure** is **ongoing**, but there are **no PUCCH resources** available for **scheduling request**.
- **On-demand system information**: **UE request** for **other SIBs** and **posSIBs** (see clause 7.3 of TS 38.300).
- **Beam failure recovery**.
- **Scheduling request failure**.
- **Time alignment** for a **new time alignment group** (**TAG**). More in detail, **establish** the **time alignment** for a **new Scell** that **belongs** to a **new TAG**.
- Consistent **Uplink listen before talk** (**LBT**) **failure** on **special cell**.
- **Small data transmission** (**SDT**) in **RRC_INACTIVE** (see clause 18 of TS 38.300).
- **Positioning purpose** during **RRC_CONNECTED** requiring **random access procedure**, e.g., when **timing advance** is needed for **UE positioning**.

A **random access procedure** has some **specific characteristics** depending on the **triggering event** (and possibly also on **other variables**).

The following table summarizes the **specific characteristics** of a **random access procedure**, depending on its **triggering event** (and on a few cases, also on other variables):

| Triggering event | CFRA | Characteristics |
|---|---|---|
| **Initial access** | No | **UE**: select a **RA preamble sequence** corresponding to the **selected beam**. Send **Msg3** (**scheduled transmission**, click 9.3.4) as `RRCSetupRequest` (click 5.8.5.10). |
| **RRC connection resume** | No | **UE**: send **Msg3** as `RRCResumeRequest` (click 5.8.5.7). |

| Triggering event | CFRA | Characteristics |
|---|---|---|
| **RRC connection re-establishment** | No | **UE**: send **Msg3** as `RRCReestablishmentRequest` (click 5.8.5.7). |
| **RRC synchronous reconfiguration: handover** | Yes | **NR node**: provide **RA preamble sequence index** using `RACH-ConfigDedicated` within `RRCReconfiguration` (click 5.8.5.4)<br>**UE**: send **Msg3** as `RRCReconfigurationComplete` (click 5.8.5.9). |
| **RRC synchronous reconfiguration: not handover** | Yes | **NR node**: trigger **synchronous reconfiguration** by including `reconfigurationWithSync` within `RRCReconfiguration`.<br>**UE**: **RA procedure** is **either contention-based** (if `ra-PreambleIndex` is **absent**) or **contention-free** (if `ra-PreambleIndex` is **present**), within `reconfigurationWithSync`. |
| **Downlink data arrival while UE is out of sync** | Yes | **NR node**: send a **PDCCH Order** using **DCI Format 1_0**, **either with** an **RA preamble sequence index** (**Msg0**, click 9.3.1, **contention-free**) or **without** an **RA preamble sequence index** (**contention-based**).<br>**UE**: for **contention resolution**, **Msg3** includes a **C-RNTI MAC CE**. |
| **Uplink data arrival while UE is out of sync** | No | **UE**: for **contention resolution**, **Msg3** includes a **C-RNTI MAC CE**. |
| **Uplink data arrival without PUCCH SR resources** | No | **UE**: for **contention resolution Msg3** includes a **C-RNTI MAC CE**. **Msg3** might also include a "**Buffer Status Report**" **MAC CE** to request a **PUSCH assignment**. |
| **On-demand system information** | Yes: (**2-step only**) | **If contention-based** (**CBRA**):<br>**NR node**: do not include `si-RequestConfig` in `SIB1` (click 5.3.4).<br>**UE**: send **Msg3** as `RRCSystemInfoRequest` (click 5.8.5.7).<br>**If contention-free** (**CFRA**):<br>**NR node**: include `si-RequestConfig` in `SIB1`.<br>**UE**: transmit a specific **RA preamble sequence** in **Msg1** (**random access preamble**, click 9.3.2) to request a specific **SI**, then monitor **PDCCH** for **SI** after **Msg2** (**random access response**, click 9.3.3). |
| **Beam failure recovery** | Yes | **If contention-based** (**CBRA**):<br>**UE**: select a **RA preamble sequence** corresponding to the **selected beam** for **recovery**.<br>**If contention-free** (**CFRA**):<br>**NR node**: provide the **UE** with **1 RA preamble sequence index** for **each beam** that is available for recovery. |
| **Scheduling request failure** | No | **UE**: **start** a **RA procedure** if the **NR node** does **not** provide a **PUSCH** assignment after `sr-TransMax` (4 to 64) **Scheduling Requests**. |
| **Time alignment for a new time alignment group (TAG)** | Yes | Used to **initialize Timing Advance** of a newly added **SCell** that belongs to a **new Timing Advance Group** (TAG).<br>**UE**: **RA procedure** is **either contention-based** (if `ra-PreambleIndex` is **absent**) or **contention-free** (if `ra-PreambleIndex` is **present**), within `reconfigurationWithSync`. |

Some **relevant remarks** on this table are:

- The **first column** indicates the **triggering event**.
- The **second column** indicates for the corresponding case if it is **possible**, or **not**, to use a **contention-free** (**CFRA**) **procedure.** Note that a **contention-based** (**CBRA**) **procedure** can **always** be used.
- The **third column** indicates some specific **characteristics** for **each case**.

Source web pages:

- https://www.nrexplained.com/ra

## 9.5 Random access procedure parameters

The **random access procedure** is configured with:

- **Common parameters** that are **broadcast** (or **unicast**) within `SIB1`, as described in section 9.5.1.
- **Common** and **dedicated** parameters that are **unicast** within the **non-SI parameter group**, as described in section 9.5.2.

**Random access** configuration does **not** have **any parameter** within `MIB` (click 5.3.3).

Possibly existing **random access parameters** within **other SIBs and posSIBs** are **not addressed** in this book.

For an overall explanation on **UE configuration parameters** click 5. For a brief overview of **ASN.1**, click 5.9.

### 9.5.1 Random access parameters within `SIB1`

The **random access procedure** is configured with **common parameters** that are **broadcast** (or **unicast**) within `SIB1` (click 5.3.4) in the following case:

- In **UE**'s **primary cell**, for **Downlink** and **Uplink initial bandwidth parts**, during **initial access**.

For **random access** parameters within `SIB1`, the **most relevant related data types** for **4-step CBRA** (click 9.1.3) are:

- `RACH-ConfigCommon` (click 9.5.3): common parameters
- `RACH-ConfigGeneric` (click 9.5.5): instantiated as common

For **random access** parameters within `SIB1`, the **most relevant related data types** for **2-step CBRA** (click 9.1.5) are:

- `MsgA-ConfigCommon-r16` (click 8.5.4): common parameters
- `RACH-ConfigCommonTwoStepRA-r16` (click 9.5.8): common parameters
- `RACH-ConfigGenericTwoStepRA-r16` (click 9.5.9): instantiated as common

Notice that `SIB1` does **not** contain **parameters** for **CFRA procedures** (click 9.1.2). The reason is that `SIB1` provides the **minimum required configuration** for the **initial random access procedure** (click 8.10), which is **always** a **CBRA procedure**. It **cannot** be a **CFRA procedure** because **during initial access** the **UE** does **not have** its **C-RNTI**.

The **ancestor tree** (click 5.9.8) of all these **most relevant related data types** up to **SIB1** is:

```
                                    SIB1 (click 5.7.4)
servingCellConfigCommon          ServingCellConfigCommonSIB (0,1) (click 3.9.3)
  uplinkConfigCommon               UplinkConfigCommonSIB (click 2.2.6.5)
    initialUplinkBWP                 BWP-UplinkCommon (click 2.2.8.9)
      rach-ConfigCommon                RACH-ConfigCommon (click 9.5.3)
        rach-ConfigGeneric               RACH-ConfigGeneric (click 9.5.5)
      msgA-ConfigCommon-r16            MsgA-ConfigCommon-r16 (click 8.5.4)
        rach-ConfigCommonTwoStepRA-r16
                                           RACH-ConfigCommonTwoStepRA-r16 (9.5.8)
          rach-ConfigGenericTwoStepRA-r16
                                             RACH-ConfigGenericTwoStepRA-r16
                                                                    (click 9.5.9)
  supplementaryUplink              UplinkConfigCommonSIB (click 2.2.6.5)
    initialUplinkBWP                 BWP-UplinkCommon (click 2.2.8.9)
      rach-ConfigCommon                RACH-ConfigCommon (click 9.5.3)
        rach-ConfigGeneric               RACH-ConfigGeneric (click 9.5.5)
      msgA-ConfigCommon-r16            MsgA-ConfigCommon-r16 (click 8.5.4)
        rach-ConfigCommonTwoStepRA-r16
                                           RACH-ConfigCommonTwoStepRA-r16 (9.5.8)
          rach-ConfigGenericTwoStepRA-r16
                                             RACH-ConfigGenericTwoStepRA-r16
                                                                    (click 9.5.9)
  si-SchedulingInfo              SI-SchedulingInfo (click 5.5.3.1)
    si-RequestConfig               SI-RequestConfig (click 5.5.3.4)
      rach-OccassionsSI              SEQUENCE
        rach-ConfigSI                  RACH-ConfigGeneric (click 9.5.5)
        ssb-perRACH-Occasion           ENUMERATED
      si-RequestResources            SI-RequestResources (1..32) (click 5.5.3.5)
    si-RequestConfigSUL            SI-RequestConfig (click 5.5.3.4)
      rach-OccassionsSI              SEQUENCE
        rach-ConfigSI                  RACH-ConfigGeneric (click 9.5.5)
        ssb-perRACH-Occasion           ENUMERATED
      si-RequestResources            SI-RequestResources (1..32) (click 5.5.3.5)
```

Some **relevant remarks** on this ancestor tree are:

- The value **32** in parameter **si-RequestResources** comes from the constant **maxSI-Message**, which is the **maximum quantity** of **SI messages** in a **SI request**.
- The **last branch** (the one of **si-SchedulingInfo**) contains the **random access parameters** for the specific case of **on-demand System Information** (click 5.5.2).

For the **ancestor trees** of **SIB1** **up to** the **top**, click 5.3.4.

The **initial random access procedure** requires, in **addition** to its **specific parameters** indicated above, the parameters for **PDCCH** (click 6.12.2), for **PDSCH** (click 6.18.2) and for **PUSCH** (click 7.15.2).

## 9.5.2 Random access parameters within the non-SI parameter group

The **random access procedure** is configured with **common** and **dedicated parameters** that are **unicast** in the **Non-SI parameter group** (click 5.3.6) in the following cases:

- In **the primary-secondary cell** and in **secondary cells**.
- In **the primary cell**, if configuration changes are done **after initial access**.

For **random access** parameters within the **Non-SI parameter group**, the **most relevant related data types** for **4-step CBRA** (click 9.1.3) are:

- `RACH-ConfigCommon` (click 9.5.3): common parameters
- `RACH-ConfigGeneric` (click 9.5.5): instantiated as dedicated
- ***If*** the **RA procedure** is for **beam failure recovery** (**BFR**),
  - ***then***, `BeamFailureRecoveryConfig` (click 9.5.4): dedicated parameters
  - ***Else***, `RACH-ConfigCommon` (click 9.5.3): common parameters

For **random access** parameters within the **Non-SI parameter group**, the **most relevant related data types** for **2-step CBRA** (click 9.1.5) are:

- `MsgA-ConfigCommon-r16` (click 9.5.6): common parameters
- `RACH-ConfigCommonTwoStepRA-r16` (click 9.5.8): common parameters
- `RACH-ConfigGenericTwoStepRA-r16` (click 9.5.9): instantiated as dedicated

For **random access** parameters within the **Non-SI parameter group**, the **most relevant related data types** for **4-step CFRA** (click 9.1.4) are:

- `RACH-ConfigDedicated` (click 9.5.10): dedicated parameters
- `CFRA` (click 9.5.8): dedicated parameters
- `RACH-ConfigGeneric` (click 9.5.5): instantiated as dedicated

For **random access** parameters within the **Non-SI parameter group**, the **most relevant related data types** for **2-step CFRA** (click 9.1.6) are:

- `RACH-ConfigDedicated` (click 9.5.10): dedicated parameters
- `CFRA-TwoStep-r16` (click 9.5.12): dedicated parameters
- `RACH-ConfigGeneric TwoStepRA-r16` (click 9.5.9): instantiated as dedicated

The **ancestor tree** (click 5.9.8) of these **most relevant related data types** up to `CellGroupConfig` is:

```
                                CellGroupConfig (click 3.7.3)
spCellConfig                      SpCellConfig (click 3.9.5)
  reconfigurationWithSync           ReconfigurationWithSync (0,1) (click 3.9.8)
    spCellConfigCommon                ServingCellConfigCommon (click 3.9.6)
      uplinkConfigCommon                UplinkConfigCommon (click 2.2.6.9)
        initialUplinkBWP                  BWP-UplinkCommon (click 2.2.8.9)
          rach-ConfigCommon                 RACH-ConfigCommon (click 9.5.3)
            rach-ConfigGeneric                RACH-ConfigGeneric (click 9.5.5)
          msgA-ConfigCommon-r16             MsgA-ConfigCommon-r16 (click 8.5.4)
            rach-ConfigCommonTwoStepRA-r16
                                              RACH-ConfigCommonTwoStepRA-r16 (9.5.8)
              rach-ConfigGenericTwoStepRA-r16
                                                RACH-ConfigGenericTwoStepRA-r16
                                                                    (click 9.5.9)
```

```
    supplementaryUplinkConfig  UplinkConfigCommon (click 2.2.6.9)
      initialUplinkBWP           BWP-UplinkCommon (click 2.2.6.9)
        rach-ConfigCommon          RACH-ConfigCommon (click 9.5.3)
          rach-ConfigGeneric         RACH-ConfigGeneric (click 9.5.5)
        msgA-ConfigCommon-r16    MsgA-ConfigCommon-r16 (click 8.5.4)
          rach-ConfigCommonTwoStepRA-r16
                                     RACH-ConfigCommonTwoStepRA-r16 (9.5.8)
            rach-ConfigGenericTwoStepRA-r16
                                       RACH-ConfigGenericTwoStepRA-r16
                                                                (click 9.5.9)
begin of parameters for CFRA procedures
  uplink                   RACH-ConfigDedicated (0,1) (click 9.5.10)
    cfra                     CFRA (click 9.5.8)
      rach-ConfigGeneric       RACH-ConfigGeneric (click 9.5.5)
    cfra-TwoStep-r16         CFRA-TwoStep-r16 (click 9.5.12)
      rach-ConfigGenericTwoStepRA-r16
                               RACH-ConfigGenericTwoStepRA-r16
                                                    (click 9.5.9)
  supplementaryUplink      RACH-ConfigDedicated (1,0) (click 9.5.10)
    cfra                     CFRA (click 9.5.8)
      rach-ConfigGeneric       RACH-ConfigGeneric (click 9.5.5)
    cfra-TwoStep-r16         CFRA-TwoStep-r16 (click 9.5.12)
      rach-ConfigGenericTwoStepRA-r16
                               RACH-ConfigGenericTwoStepRA-r16 (9.5.9)
end of parameters for CFRA procedures
spCellConfigDedicated      ServingCellConfig (click 3.9.7)
  uplinkConfig               UplinkConfig (click 2.2.8.42.2.8.3)
    initialUplinkBWP           BWP-UplinkDedicated (click 2.2.8.10)
      beamFailureRecoveryConfig  BeamFailureRecoveryConfig (click 9.5.4)
        rach-ConfigBFR             RACH-ConfigGeneric (click 9.5.5)
    uplinkBWP-ToReleaseList    BWP-Id (0..4) (click 2.2.8.13)
    uplinkBWP-ToAddModList     BWP-Uplink (0..4) (click 2.2.8.8)
      bwp-Common                 BWP-UplinkCommon (click 2.2.6.9)
        rach-ConfigCommon          RACH-ConfigCommon (click 9.5.3)
          rach-ConfigGeneric         RACH-ConfigGeneric (click 9.5.5)
        msgA-ConfigCommon-r16    MsgA-ConfigCommon-r16 (click 8.5.4)
          rach-ConfigCommonTwoStepRA-r16
                                     RACH-ConfigCommonTwoStepRA-r16 (9.5.8)
            rach-ConfigGenericTwoStepRA-r16
                                       RACH-ConfigGenericTwoStepRA-r16
                                                                (click 9.5.9)
      bwp-Dedicated              BWP-UplinkDedicated (click 2.2.8.102.2.8.7)
        beamFailureRecoveryConfig  BeamFailureRecoveryConfig (click 9.5.4)
          rach-ConfigBFR             RACH-ConfigGeneric (click 9.5.5)
  supplementaryUplink        UplinkConfig (click 2.2.8.3)
    initialUplinkBWP           BWP-UplinkDedicated (click 2.2.8.10)
      pusch-Config               PUSCH-Config (0,1) (click 7.15.5)
    uplinkBWP-ToReleaseList    BWP-Id (0..4) (click 2.2.8.13)
    uplinkBWP-ToAddModList     BWP-Uplink (0..4) (click 2.2.8.8)
      bwp-Common                 BWP-UplinkCommon (click 2.2.6.9)
        rach-ConfigCommon          RACH-ConfigCommon (click 9.5.3)
          rach-ConfigGeneric         RACH-ConfigGeneric (click 9.5.5)
        msgA-ConfigCommon-r16    MsgA-ConfigCommon-r16 (click 8.5.4)
          rach-ConfigCommonTwoStepRA-r16
                                     RACH-ConfigCommonTwoStepRA-r16 (9.5.8)
            rach-ConfigGenericTwoStepRA-r16
                                       RACH-ConfigGenericTwoStepRA-r16
                                                                (click 9.5.9)
      bwp-Dedicated              BWP-UplinkDedicated (click 2.2.8.102.2.8.7)
        beamFailureRecoveryConfig  BeamFailureRecoveryConfig (click 9.5.4)
          rach-ConfigBFR             RACH-ConfigGeneric (click 9.5.5)
```

```
sCellToAddModList              SCellConfig (0..maxNrofSCells) (click 3.9.4)
  sCellConfigCommon              ServingCellConfigCommon (0,1) (click 3.9.6)
    uplinkConfigCommon             UplinkConfigCommon (click 2.2.6.9)
      initialUplinkBWP               BWP-UplinkCommon (click 2.2.8.9)
        rach-ConfigCommon              RACH-ConfigCommon (click 9.5.3)
          rach-ConfigGeneric             RACH-ConfigGeneric (click 9.5.5)
        msgA-ConfigCommon-r16        MsgA-ConfigCommon-r16 (click 8.5.4)
          rach-ConfigCommonTwoStepRA-r16
                                         RACH-ConfigCommonTwoStepRA-r16 (9.5.8)
            rach-ConfigGenericTwoStepRA-r16
                                           RACH-ConfigGenericTwoStepRA-r16 (9.5.9)
    supplementaryUplinkConfig    UplinkConfigCommon (click 2.2.6.9)
      initialUplinkBWP               BWP-UplinkCommon (click 2.2.8.9)
        rach-ConfigCommon              RACH-ConfigCommon (click 9.5.3)
          rach-ConfigGeneric             RACH-ConfigGeneric (click 9.5.5)
        msgA-ConfigCommon-r16        MsgA-ConfigCommon-r16 (click 8.5.4)
          rach-ConfigCommonTwoStepRA-r16
                                         RACH-ConfigCommonTwoStepRA-r16 (9.5.8)
            rach-ConfigGenericTwoStepRA-r16
                                           RACH-ConfigGenericTwoStepRA-r16 (9.5.9)
  sCellConfigDedicated           ServingCellConfig (click 3.9.7)
    uplinkConfig                   UplinkConfig (click 2.2.8.42.2.8.3)
      initialUplinkBWP               BWP-UplinkDedicated (click 2.2.8.10)
        beamFailureRecoveryConfig    BeamFailureRecoveryConfig (click 9.5.4)
          rach-ConfigBFR                 RACH-ConfigGeneric (click 9.5.5)
      uplinkBWP-ToReleaseList        BWP-Id (0..4) (click 2.2.8.13)
      uplinkBWP-ToAddModList         BWP-Uplink (0..4) (click 2.2.8.8)
        bwp-Common                     BWP-UplinkCommon (click 2.2.6.9)
          rach-ConfigCommon              RACH-ConfigCommon (click 9.5.3)
            rach-ConfigGeneric             RACH-ConfigGeneric (click 9.5.5)
          msgA-ConfigCommon-r16        MsgA-ConfigCommon-r16 (click 8.5.4)
            rach-ConfigCommonTwoStepRA-r16
                                           RACH-ConfigCommonTwoStepRA-r16 (9.5.8)
              rach-ConfigGenericTwoStepRA-r16
                                             RACH-ConfigGenericTwoStepRA-r16
                                                                    (click 9.5.9)
        bwp-Dedicated                  BWP-UplinkDedicated (click 2.2.8.102.2.8.7)
          beamFailureRecoveryConfig    BeamFailureRecoveryConfig (click 9.5.4)
            rach-ConfigBFR                 RACH-ConfigGeneric (click 9.5.5)
    supplementaryUplink            UplinkConfig (click 2.2.8.3)
      initialUplinkBWP               BWP-UplinkDedicated (click 2.2.8.10)
        beamFailureRecoveryConfig    BeamFailureRecoveryConfig (click 9.5.4)
          rach-ConfigBFR                 RACH-ConfigGeneric (click 9.5.5)
      uplinkBWP-ToReleaseList        BWP-Id (0..4) (click 2.2.8.13)
      uplinkBWP-ToAddModList         BWP-Uplink (0..4) (click 2.2.8.8)
        bwp-Common                     BWP-UplinkCommon (click 2.2.6.9)
          rach-ConfigCommon              RACH-ConfigCommon (click 9.5.3)
            rach-ConfigGeneric             RACH-ConfigGeneric (click 9.5.5)
          msgA-ConfigCommon-r16        MsgA-ConfigCommon-r16 (click 8.5.4)
            rach-ConfigCommonTwoStepRA-r16
                                           RACH-ConfigCommonTwoStepRA-r16 (9.5.8)
              rach-ConfigGenericTwoStepRA-r16
                                             RACH-ConfigGenericTwoStepRA-r16
                                                                    (click 9.5.9)
        bwp-Dedicated                  BWP-UplinkDedicated (click 2.2.8.102.2.8.7)
          beamFailureRecoveryConfig    BeamFailureRecoveryConfig (click 9.5.4)
            rach-ConfigBFR                 RACH-ConfigGeneric (click 9.5.5)
```

Some **relevant remarks** on this ancestor tree are:

- Within the **parameters** for **CFRA procedures**, there is a **CHOICE** between **uplink** and **supplementaryUplink**. This is why the **first** one is tagged as "**(0,1)**", while the **second** one is tagged as "**(1,0)**".

For an overall explanation on **UE configuration parameters** click 5. For a brief overview of **ASN.1**, click 5.9.

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_RACH.html
- https://www.5g-networks.net/5g-technology/5g-nr-rach-preamble-types-long-and-short-preambles/
- https://www.blacktechnoguys.com/2020/02/prach-basic-1.html
- https://www.blacktechnoguys.com/2020/02/prach-basic-2.html
- https://www.blacktechnoguys.com/2020/03/prach-basic-3.html

### 9.5.3 RACH-ConfigCommon data type

It can be **both broadcast** (or **unicast**) within **SIB1** (click 5.3.4) and **unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to SIB1**, click 9.5.1.

For its **ancestor tree up to CellGroupConfig** (click 3.7.3), click 9.5.2.

Its **2 parent data types** are **BWP-UplinkCommon** (click 2.2.8.9) and **AdditionalRACH-Config-r17** (whose ancestor tree leads **only** to **BWP-UplinkCommon**).

**All** its parameters are **common** (click 5.1.3).

**Each** of its **instances** contains **most** of the **common parameters** for **PRACH** (click 7.1) and for the **4-step random access procedures** (click 9.1.1), including the case of **beam failure recovery**.

#### 9.5.3.1 RACH-ConfigCommon parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- **msg1-SubcarrierSpacing** **SubcarrierSpacing** (click 2.2.10.5.1): optional, common

  It provides the **PRACH subcarrier spacing** for **short** preamble **Formats** (**A1** to **C2**) for the **4-step RA procedures**. Notice that for **long** preamble **Formats** (**0** to **3**), the **PRACH subcarrier spacing** is determined by the **preamble Format itself**. The allowed values of **PRACH subcarrier spacing** depend on the combination of **Frequency Range** and **preamble format** and **sequence length** (see table in 7.3.1).

- `prach-RootSequenceIndex` **`CHOICE`**: mandatory, common
- `prach-RootSequenceIndex-r16` **`CHOICE`**: optional, common

  **Either** of them provides the **PRACH sequence length** "$L_{RA}$". Their **data types** and **possible values** are:

```
prach-RootSequenceIndex CHOICE {
  l839 INTEGER (0..837),
  l139 INTEGER (0..137)
},
prach-RootSequenceIndex-r16 CHOICE {
  l571 INTEGER (0..569),
  l11151 INTEGER (0..1149)
} OPTIONAL                                                    -- Need
R
```

  The **UE** obtains the **ZC** sequence index "$u$" from the **integer value** in the corresponding parameter above. This **value** is an **index** for one of the tables **6.3.3.1-3** to **6.3.3.1-4B**, of TS 38.211. The said tables provide the actual value of "$u$". The table to use is the one corresponding to the value of the sequence length "$L_{RA}$".

- `rach-ConfigGeneric` `RACH-ConfigGeneric` (click 9.5.5): mandatory, common

  It contains **many common parameters** for the **4-step RA procedures** (click 9.1.1), including the case of **beam failure recovery**.

- `restrictedSetConfig` **`ENUMERATED`** `{unrestrictedSet, restrictedSetTypeA, restrictedSetTypeB }`: mandatory, common

  If it is **present**, it provides the **type** of **set** of **RA preamble sequences**.

  If it is **absent**, the **common parameter** `msgA-RestrictedSetConfig-r16` from `RACH-ConfigCommonTwoStepRA-r16` (click 9.5.8) is rather used.

- `ssb-perRACH-OccasionAndCB-PreamblesPerSSB` **`CHOICE`**: optional, common

  It applies to the **4-step RA procedures**. Its meaning is **twofold**:

  - The **selected parameter** from the **`CHOICE`** indicates the **quantity** of **different values** of **ssb-time-domain-index** per **RACH occasion** (which matches the **first part** of the parameter name "`ssb-perRACH-Occasion`").

    Selecting `oneEighth` indicates **1 value** of **ssb-time-domain-index** associated with **8 RACH occasions**, selecting `oneFourth` indicates to **1 value** of **ssb-time-domain-index** associated with **4 RACH occasions**, selecting `four` indicates **4 values** of **ssb-time-domain-index** associated with **1 RACH occasion**, and so on.

  - **Each** **`ENUMERATED`** indicates the **quantity** of **RA preamble sequences** that can be used for **each possible value** of **ssb-time-domain-index** (which matches the **second part** of the parameter name "`CB-PreamblesPerSSB`").

  The **total quantity** of **contention based** (**CB**) **RA preambles** that would be **required** in a **RACH occasion** is obtained from their **product**:

  `CB-preambles-per-SSB` • `ssb-per-rach-occasion`

- `totalNumberOfRA-Preambles INTEGER (1..63)`: optional, common

  It is the **total quantity** of **random access** (**RA**) **preamble sequences** that can be used for the **4-step RA procedures** (click 9.1.1), excluding the **RA preamble sequences** used for **other purposes** (e.g., for **SI request**). If it is **absent**, **all 64 RA preamble sequences** can be used for the **RA procedure**.

  Its value **must** be a **multiple** of the **quantity** of **SSBs** per **RACH occasion** (see above `ssb-perRACH-OccasionAndCB-PreamblesPerSSB`).

Notice that **data type** `RACH-ConfigCommonTwoStepRA-r16` (click 9.5.8) contains **very similar parameters**, with **very similar semantics**, but **actually different**, to the **parameters** of **this data type**.

For more detail on these parameters, refer to the table "*RACH-ConfigCommon field descriptions*" of TS 38.331.

### 9.5.3.2 `RACH-ConfigCommon` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
RACH-ConfigCommon ::= SEQUENCE {
  rach-ConfigGeneric RACH-ConfigGeneric,
  totalNumberOfRA-Preambles INTEGER (1..63) OPTIONAL,                -- Need S
  ssb-perRACH-OccasionAndCB-PreamblesPerSSB CHOICE {
    oneEighth ENUMERATED {n4,n8,n12,n16,n20,n24,n28,n32,
                          n36,n40,n44,n48,n52,n56,n60,n64},
    oneFourth ENUMERATED {n4,n8,n12,n16,n20,n24,n28,n32,
                          n36,n40,n44,n48,n52,n56,n60,n64},
    oneHalf   ENUMERATED {n4,n8,n12,n16,n20,n24,n28,n32,
                          n36,n40,n44,n48,n52,n56,n60,n64},
    one       ENUMERATED {n4,n8,n12,n16,n20,n24,n28,n32,
                          n36,n40,n44,n48,n52,n56,n60,n64},
    two       ENUMERATED {n4,n8,n12,n16,n20,n24,n28,n32},
    four    INTEGER (1..16),
    eight   INTEGER (1..8),
    sixteen INTEGER (1..4)
  } OPTIONAL,                                                      -- Need M
  groupBconfigured SEQUENCE {
    ra-Msg3SizeGroupA ENUMERATED {b56, b144, b208, b256, b282, b480, b640,
         b800, b1000, b72, spare6, spare5, spare4, spare3, spare2, spare1},
    messagePowerOffsetGroupB ENUMERATED {minusinfinity, dB0, dB5, dB8,
                                         dB10, dB12, dB15, dB18},
    numberOfRA-PreamblesGroupA INTEGER (1..64)
  } OPTIONAL, -- Need R
  ra-ContentionResolutionTimer ENUMERATED {sf8, sf16, sf24, sf32, sf40,
                                           sf48, sf56, sf64},
  rsrp-ThresholdSSB     RSRP-Range OPTIONAL,                       -- Need R
  rsrp-ThresholdSSB-SUL RSRP-Range OPTIONAL,                     -- Cond SUL
  prach-RootSequenceIndex CHOICE {
    l839 INTEGER (0..837),
    l139 INTEGER (0..137)
  },
  msg1-SubcarrierSpacing SubcarrierSpacing OPTIONAL,           -- Cond L139
  restrictedSetConfig ENUMERATED {unrestrictedSet, restrictedSetTypeA,
                                  restrictedSetTypeB},
  msg3-transformPrecoder ENUMERATED {enabled} OPTIONAL,            -- Need R
  ...,
  [[
```

```
    ra-PrioritizationForAccessIdentity-r16 SEQUENCE {
      ra-Prioritization-r16 RA-Prioritization,
      ra-PrioritizationForAI-r16 BIT STRING (SIZE (2))
    } OPTIONAL,                                          -- Cond InitialBWP-Only
    prach-RootSequenceIndex-r16 CHOICE {
      l571 INTEGER (0..569),
      l1151 INTEGER (0..1149)
    } OPTIONAL                                                    -- Need R
    ]],
    [[
    ra-PrioritizationForSlicing-r17 RA-PrioritizationForSlicing-r17
                                    OPTIONAL,            -- Cond InitialBWP-Only
    featureCombinationPreamblesList-r17 SEQUENCE
          (SIZE(1..maxFeatureCombPreamblesPerRACHResource-r17))
          OF FeatureCombinationPreambles-r17 OPTIONAL    -- Cond AdditionalRACH
    ]]
  }
```

### 9.5.4 `BeamFailureRecoveryConfig` data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 9.5.2.

Its **only parent data type** is `BWP-UplinkDedicated` (click 2.2.8.10).

**All** its parameters are **dedicated** (click 5.1.3).

**Each** of its **instances** contains **many dedicated parameters** for **PRACH** (click 7.1) and for the **random access procedures** in the **specific case** of **beam failure recovery**.

#### 9.5.4.1 `BeamFailureRecoveryConfig` parameters

**All** of its **parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `beamFailureRecoveryTimer` **`SEQUENCE`**: optional, dedicated

  It is the **beam failure recovery timer**. Upon **expiration** of this **period**, the **UE** does **not** use **CFRA** for **BFR**. Its possible values are `10`, `20`, `40`, `80`, `100`, 150 and `200` **ms**.

- `candidateBeamRSList` **`SEQUENCE`**: optional, dedicated

- `candidateBeamRSListExt-v1610` **`SetupRelease{`**`CandidateBeamRSListExt-r16`**`}`**: optional, dedicated

  These **two parameters jointly** contain the **Set** of **reference signals** (**CSI-RS** and/or **SSB**) that identify the **candidate beams** for **recovery** and the **associated RA parameters**. This **set** includes **all** the elements of `candidateBeamRSList` and **all** the elements of `candidateBeamRSListExt-v1610`. The **UE** maintains `candidateBeamRSList` and `candidateBeamRSListExt-v1610` **separately**: Receiving `candidateBeamRSListExt-v1610` to **release only releases** the **entries** that were configured by `candidateBeamRSListExt-v1610`. Receiving `candidateBeamRSListExt-v1610` to **setup only replaces** the **entries** that were **configured** by `candidateBeamRSListExt-v1610` with the **newly signaled entries**. The network configures these reference signals to be within the **linked Downlink bandwith part** (i.e., within the **Downlink BWP** with the same `bwp-Id`) of the **Uplink BWP** in which the `BeamFailureRecoveryConfig` **instance** is **present**.

- `msg1-SubcarrierSpacing SubcarrierSpacing` (click 2.2.10.5.1): optional, dedicated

  It is the **subcarrier spacing** for **contention-free beam failure recovery** (refer to clause 5.3.2 of TS 38.211). Its possible values depend on the **frequency range** as follows:

  - **FR1**: **15** or **30 kHz**
  - **FR2-1**: **60** or **120 kHz**
  - **FR2-2**: **120**, **480**, or **960 kHz**

- `rsrp-ThresholdSSB RSRP-Range`: optional, dedicated

  It is the **L1-RSRP threshold** used for determining if a **candidate beam** may be **used** by the **UE** to **attempt contention-free random access** (**CFRA**) for **beam failure recovery** (refer to clause 6 of TS 38.213).

- `ra-prioritization RA-Prioritization`: optional, dedicated

  It contains the **parameters** for **prioritized random access procedure** for **beam failure recovery** (refer to clause 5.1.1 of TS 38.321).

- `ra-PrioritizationTwoStep-r16 RA-Prioritization`: optional, dedicated

  It contains the **parameters** for **prioritized 2-step random access procedure** for **beam failure recovery** (refer to clause 5.1.1 of TS 38.321).

- `ra-ssb-OccasionMaskIndex INTEGER (0..15)`: optional, dedicated

  It is the explicitly signaled **PRACH Mask Index** for **random access resource selection** in TS 38.321. The **mask** is **valid** for **all** the corresponding **SSB resources**.

  It has **3 same-name parameters**, with the **same data type** and **similar semantics**, from `CFRA` (click 9.5.11), from `CFRA-TwoStep-r16` (click 9.5.12) and from `SI-RequestResources` (click 5.5.3.5).

- `rach-ConfigBFR RACH-ConfigGeneric` (click 9.5.5): optional, dedicated

  It contains the **random access parameters** for **beam failure recovery**.

- `recoverySearchSpaceId SearchSpaceId` (click 6.12.9): optional, dedicated

  It contains the **identifier** of the **search space** for **beam failure recovery random access response** (**RAR**). The network configures this **search space** to be within the **linked Downlink bandwidth part** (i.e., within the **Downlink BWP** with the **same** `bwp-Id`) of the **Uplink BWP** in which the `BeamFailureRecoveryConfig` **instance** is **present**. The **CORESET associated** with the **recovery search space cannot** be **associated** with **another search space**. Network **always** configures the **UE** with this **parameter** when **contention-free random access resources** for **beam failure recovery** are **configured**.

- `rootSequenceIndex-BFR INTEGER (0..137)`: optional, dedicated

  It is the **PRACH root sequence index** for **beam failure recovery** (refer to clause 6.3.3.1 of TS 38.211).

- `spCell-BFR-CBRA-r16` **ENUMERATED** `{true}`: optional, dedicated

  It indicates that **UE** is configured to send a **MAC PDU** with a "**Beam failure recovery**" **MAC control element** (**CE**) for **SpCell beam failure recovery**, as specified in TS 38.321.

- `ssb-perRACH-Occasion` **ENUMERATED**: optional, dedicated

  It indicates the **quantity** of **SSBs per RACH occasion** for a **contention-free RA procedure** for **beam failure recovery** (refer to clause 8.1 of TS 38.213). Its possible values are `1/8`, `1/4`, `1/2`, `1`, `2`, `4`, `8` and `16`.

  Notice that **data type** `SI-RequestConfig` (click 5.5.3.4) has the **same-name parameter** `ssb-perRACH-Occasion`.

For more detail on these parameters, refer to the table "*BeamFailureRecoveryConfig field descriptions*" of TS 38.331.

### 9.5.4.2 `BeamFailureRecoveryConfig` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
BeamFailureRecoveryConfig ::= SEQUENCE {
  rootSequenceIndex-BFR INTEGER (0..137) OPTIONAL,                  -- Need M
  rach-ConfigBFR RACH-ConfigGeneric OPTIONAL,                       -- Need M
  rsrp-ThresholdSSB RSRP-Range OPTIONAL,                            -- Need M
  candidateBeamRSList SEQUENCE (SIZE(1..maxNrofCandidateBeams))
                      OF PRACH-ResourceDedicatedBFR OPTIONAL,       -- Need M
  ssb-perRACH-Occasion ENUMERATED {oneEighth, oneFourth, oneHalf,
                       one, two, four, eight, sixteen} OPTIONAL,    -- Need M
  ra-ssb-OccasionMaskIndex INTEGER (0..15) OPTIONAL,                -- Need M
  recoverySearchSpaceId SearchSpaceId OPTIONAL,                     -- Need R
  ra-Prioritization RA-Prioritization OPTIONAL,                     -- Need R
  beamFailureRecoveryTimer ENUMERATED {ms10, ms20, ms40, ms60,
                           ms80, ms100, ms150, ms200} OPTIONAL,     -- Need M
  ...,
  [[
  msg1-SubcarrierSpacing SubcarrierSpacing OPTIONAL                 -- Need M
  ]],
  [[
  ra-PrioritizationTwoStep-r16 RA-Prioritization OPTIONAL,          -- Need R
  candidateBeamRSListExt-v1610
            SetupRelease{CandidateBeamRSListExt-r16} OPTIONAL       -- Need M
  ]],
  [[
  spCell-BFR-CBRA-r16 ENUMERATED {true} OPTIONAL                    -- Need R
  ]]
}
```

## 9.5.5 `RACH-ConfigGeneric` data type

It can be **both broadcast** (or **unicast**) within `SIB1` (click 5.3.4) and **unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `SIB1`, click 9.5.1.

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 9.5.2.

Its **3 parent data types** are `RACH-ConfigCommon` (click 9.5.3), `BeamFailureRecoveryConfig` (click 9.5.4), `CFRA` (click 9.5.11) and `SI-RequestConfig` (click 5.5.3.4).

**Each** of its **instances** contains **either all common parameters or all dedicated parameters** (click 5.1.3), depending on where it is **instantiated**.

**Each** of its **instances** contains **many parameters** for **PRACH** (click 7.1) and for the **4-step RA procedures** (click 9.1.1), including the case of **beam failure recovery**.

### 9.5.5.1 `RACH-ConfigGeneric` parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `msg1-FDM` **ENUMERATED** `{one, two, four, eight}`: mandatory, common **or** dedicated

  It is the **quantity** of **frequency transmission occasions** (in 1 **PRACH time occasion**) for the **RA preamble** in **4-step RA procedures** (click 9.1.1). Its possible values are `1`, `2`, `4` or `8`.

  The **UE ignores** it if `msgA-RO-FDM-r16` from `RACH-ConfigGenericTwoStepRA-r16` (click 9.5.9) is **present**.

- `msg1-FrequencyStart` **INTEGER** `(0..maxNrofPhysicalResourceBlocks-1)`: mandatory, common **or** dedicated

  It provides the **frequency transmission occasions** (through **PRACH**) for the **RA preamble** in **4-step RA procedures** (click 9.1.1).

  The **UE ignores** it if **either** `msgA-RO-FrequencyStart-r16` from `RACH-ConfigGenericTwoStepRA-r16` is **present**.

  The constant `maxNrofPhysicalResourceBlocks-1` is `274`, which is the **maximum quantity**, minus **1**, of **PRBs** in **each** of the **UE**'s **bandwidth parts**.

- `powerRampingStep` **ENUMERATED** `{dB0, dB2, dB4, dB6}`: mandatory, common **or** dedicated

  It is the **power ramping step** for **RA retransmissions** (refer to clause 5.1.3 of TS 38.321).

- `prach-ConfigurationFrameOffset-IAB-r16` **INTEGER** `(0..63)`: optional, common **or** dedicated

  It is the **frame offset** defined in TS 38.211 and applicable to **IAB-MTs**, altering the **ROs frame** defined in the baseline configuration indicated by `prach-ConfigurationIndex-v1610`.

- `prach-ConfigurationIndex` **INTEGER** `(0..255)`: mandatory, common **or** dedicated

  It provides the **PRACH sequence Format** by entering its value in the corresponding table 6.3.3.2-2 (**FR1** and **paired spectrum**), 6.3.3.2-3 (**FR1** and **TDD**) or 6.3.3.2-4 (**FR2** and **TDD**) of TS 38.211.

  The **UE ignores** it if `msgA-PRACH-ConfigurationIndex-r16` from `RACH-ConfigGenericTwoStepRA-r16` (click 9.5.9) is **present**.

- `prach-ConfigurationPeriodScaling-IAB-r16` **ENUMERATED** : optional, common **or** dedicated

  It is the **scaling factor** defined in TS 38.211 and applicable to **IAB-MTs**, extending the **periodicity** of the **PRACH occasions** baseline configuration indicated by `prach-ConfigurationIndex-v1610`.

- `prach-ConfigurationSOffset-IAB-r16` **INTEGER** `(0..39)`: optional, common **or** dedicated

  It is the **subframe/slot offset** defined in TS 38.211 and applicable to **IAB-MTs**, altering the **ROs subframe** or **slot** defined in the baseline configuration indicated by `prach-ConfigurationIndex-v1610`.

- `preambleTransMax` **ENUMERATED** `{n3, n4, n5, n6, n7, n8, n10, n20, n50, n100, n200}`: mandatory, common **or** dedicated

  It is the **maximum quantity** of **RA preamble consecutive transmissions** before declaring a **failure** (refer to clauses 5.1.4 and 5.1.5 of TS 38.321).

- `ra-ResponseWindow` **ENUMERATED** `{sl1, sl2, sl4, sl8, sl10, sl20, sl40, sl80}`: mandatory, common **or** dedicated

- `ra-ResponseWindow-v1610` **ENUMERATED** `{sl60, sl160}`: optional, common **or** dedicated

- `ra-ResponseWindow-v1700` **ENUMERATED** `{sl240, sl320, sl640, sl960, sl1280, sl1920, sl2560}`: optional, common **or** dedicated

  **Each** of these parameters is the **window length** for **Msg2** (**random access response**, click 9.3.3) in **quantity** of **slots**. The network configures a value **lower than or equal** to **10 ms** when **Msg2** is transmitted using **licensed spectrum** and a value **lower than or equal** to **40 ms** when **Msg2** is transmitted using **shared spectrum** channel access (refer to clause 5.1.4 of TS 38.321).

  The **UE ignores** these parameters if included within `SCellConfig` (click 3.9.4).

  The **UE ignores** `ra-ResponseWindow` if **either** `ra-ResponseWindow-v1610` or `ra-ResponseWindow-v1700` (see below) is **present**.

  Parameter `ra-ResponseWindow-v1700` is applicable to **subcarrier spacing** of **480 kHz** and **960 kHz**.

- `zeroCorrelationZoneConfig` **INTEGER** `(0..15)`: mandatory, common **or** dedicated

  It provides (together with other variables) the **set** of values "$C_v$".

  The **UE ignores** it if `msgA-ZeroCorrelationZoneConfig-r16` from `RACH-ConfigGenericTwoStepRA-r16` (click 9.5.9) is **present**.

For more detail on these parameters, refer to the table "*RACH-ConfigGeneric field descriptions*" of TS 38.331.

#### 9.5.5.2 `RACH-ConfigGeneric` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
RACH-ConfigGeneric ::= SEQUENCE {
  prach-ConfigurationIndex INTEGER (0..255),
  msg1-FDM ENUMERATED {one, two, four, eight},
  msg1-FrequencyStart INTEGER (0..maxNrofPhysicalResourceBlocks-1),
  zeroCorrelationZoneConfig INTEGER(0..15),
  preambleReceivedTargetPower INTEGER (-202..-60),
  preambleTransMax ENUMERATED {n3, n4, n5, n6, n7, n8, n10, n20, n50, n100,
      n200},
  powerRampingStep ENUMERATED {dB0, dB2, dB4, dB6},
  ra-ResponseWindow ENUMERATED {sl1, sl2, sl4, sl8, sl10, sl20, sl40, sl80},
  ...,
  [[
  prach-ConfigurationPeriodScaling-IAB-r16 ENUMERATED
        {scf1,scf2,scf4,scf8,scf16,scf32,scf64} OPTIONAL,          -- Need R
  prach-ConfigurationFrameOffset-IAB-r16 INTEGER (0..63) OPTIONAL, -- Need R
  prach-ConfigurationSOffset-IAB-r16 INTEGER (0..39) OPTIONAL,     -- Need R
  ra-ResponseWindow-v1610 ENUMERATED {sl160, sl160} OPTIONAL,      -- Need R
  prach-ConfigurationIndex-v1610 INTEGER (256..262) OPTIONAL       -- Need R
  ]],
  [[
  ra-ResponseWindow-v1700 ENUMERATED {sl240, sl320, sl640, sl960, sl1280,
      sl1920, sl2560} OPTIONAL                                     -- Need R
  ]]
}
```

Where `maxNrofPhysicalResourceBlocks-1` is `274`, which is the **maximum quantity**, minus **1**, of **PRBs** in **each** of the **UE**'s **bandwidth parts**.

### 9.5.6 `MsgA-ConfigCommon-r16` data type

It can be **both broadcast** (or **unicast**) within `SIB1` (click 5.3.4) and **unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `SIB1`, click 9.5.1.

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 9.5.2.

Its **2 parent data types** are `BWP-UplinkCommon` (click 2.2.8.9) and `AdditionalRACH-Config-r17`.

**All** its **parameters are common**.

**Each** of its **instances** contains **most** of the **common parameters** for **PRACH** (click 7.1) and for the **2-step random access procedures** (click 9.1.1), including the case of **beam failure recovery**. This includes **parameters** for the transmission of the **PUSCH payload** within **MsgA** (click 9.3.6).

#### 9.5.6.1 `MsgA-ConfigCommon-r16` parameters

Its **two parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `msgA-PUSCH-Config-r16` `msgA-PUSCH-Config-r16` (click 9.5.7): optional, common

  It contains the **common parameters** (for a given **Uplink bandwidth part**) for the **transmission** of the **PUSCH payload** of **MsgA**, in a **2-step CBRA procedure** (click 9.1.5). If it is **absent** in a **given Uplink bandwidth part**, the **UE** shall use the **MsgA PUSCH configuration** of the **initial Uplink bandwidth part**.

- `rach-ConfigCommonTwoStepRA-r16` `RACH-ConfigCommonTwoStepRA-r16` (click 9.5.8): mandatory, common

  It contains the **common parameters** (for a given **Uplink bandwidth part**) for the **2-step RA procedures** (**both CBRA** and **CFRA**), **including** the ones for **2-step CBRA** for **beam failure recovery**.

#### 9.5.6.2 `MsgA-ConfigCommon-r16` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
MsgA-ConfigCommon-r16 ::= SEQUENCE {
  rach-ConfigCommonTwoStepRA-r16 RACH-ConfigCommonTwoStepRA-r16,
  msgA-PUSCH-Config-r16 MsgA-PUSCH-Config-r16
                                          OPTIONAL --Cond InitialBWPConfig
}
```

### 9.5.7 `MsgA-PUSCH-Config-r16` data type

It can be **both broadcast** (or **unicast**) within `SIB1` (click 5.3.4) and **unicast** within the **Non-SI parameter group** (click 5.3.6).

For the **ancestor tree** of its **parent data type up to** `SIB1`, click 9.5.1.

For the **ancestor tree** of its **parent data type up to** `CellGroupConfig` (click 3.7.3), click 9.5.2.

Its **2 parent data types** are `MsgA-ConfigCommon-r16` (click 9.5.6) and `FeatureCombinationPreambles-r17`.

**All** its parameters are **dedicated** (click 5.1.3).

**Each** of its **instances** contains **most** of the **common parameters** for the transmission of the **PUSCH payload** of **MsgA** (click 9.3.6) in the **2-step random access procedures** (click 9.1.1).

#### 9.5.7.1 `MsgA-PUSCH-Config-r16` parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `msgA-PUSCH-ResourceGroupA-r16` `MsgA-PUSCH-Resource-r16`: optional, common

  It contains the **PUSCH resources** for the **UE**'s **transmission** of **MsgA** (click 9.3.6) using **RA preambles group A**.

  If it is **absent** for the **selected Uplink bandwidth part**, the **UE** uses use **either**:

  - The **MsgA PUSCH configuration** for **group A** of the **initial Uplink bandwidth part**.
  - For **RedCap UEs**, the **RedCap-specific initial Uplink bandwidth part** (if configured).

- `msgA-PUSCH-ResourceGroupA-r16` `MsgA-PUSCH-Resource-r16`: optional, common

  The same as the **previous parameter** but referred to **RA preamble group B**.

- **msgA-TransformPrecoder-r16** **ENUMERATED** {**enabled**, **disabled**}: optional, common

  It **enables** or **disables** the **transform-precoder** (click 4.12.12) for **MsgA transmission** (refer to clause 6.1.3 of TS 38.214).

#### 9.5.7.2 MsgA-PUSCH-Config-r16 ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
MsgA-PUSCH-Config-r16 ::= SEQUENCE {
  msgA-PUSCH-ResourceGroupA-r16 MsgA-PUSCH-Resource-r16
                                OPTIONAL,           -- Cond InitialBWPConfig
  msgA-PUSCH-ResourceGroupB-r16 MsgA-PUSCH-Resource-r16
                                OPTIONAL,           -- Cond GroupBConfigured
  msgA-TransformPrecoder-r16 ENUMERATED {enabled, disabled}
                             OPTIONAL,                                -- Need R
  msgA-DataScramblingIndex-r16 INTEGER (0..1023) OPTIONAL,            -- Need S
  msgA-DeltaPreamble-r16 INTEGER (-1..6) OPTIONAL                     -- Need R
}
```

### 9.5.8 RACH-ConfigCommonTwoStepRA-r16 data type

It can be **both broadcast** (or **unicast**) within **SIB1** (click 5.3.4) and **unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to SIB1**, click 9.5.1.

For its **ancestor tree up to CellGroupConfig** (click 3.7.3), click 9.5.2.

Its **only parent data type** is **MsgA-ConfigCommon-r16** (click 8.5.4).

**All** its parameters are **common** (click 5.1.3).

**Each** of its **instances** contains **most** of the **common parameters** for **PRACH** (click 7.1) and for the **2-step random access procedures** (click 9.1.1), including the case of **beam failure recovery**.

#### 9.5.8.1 RACH-ConfigCommonTwoStepRA-r16 parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- **msgA-SubcarrierSpacing-r16** **SubcarrierSpacing** (click 2.2.10.5.1): optional, common

  It provides the **PRACH subcarrier spacing** for **short** preamble **Formats** (**A1** to **C2**) for the **2-step RA procedures**. Notice that for **long** preamble **Formats** (**0** to **3**), the **PRACH subcarrier spacing** is determined by the **preamble Format itself**. The allowed values of **PRACH subcarrier spacing** depend on the combination of **Frequency Range** and **preamble Format** and **sequence length** (see table in 7.3.2).

  In case it is **absent**, ***if*** it is a **2-step only bandwidth part**,

  - ***then***, the **UE** applies the **subcarrier spacing** derived from **msgA-PRACH-ConfigurationIndex** from **RACH-ConfigGenericTwoStepRA-r16** (click 9.5.9), based on tables 6.3.3.1-**1** and -**2** plus tables 6.3.3.2-**2** and -**3** from TS 38.211).
  - ***Else***, the **UE** applies the **same subcarrier spacing** as **Msg1** (click 9.3.2), derived from **RACH-ConfigCommon** (click 9.5.3).

The value also applies to a **2-step contention-free random access procedure** type (`RACH-ConfigDedicated`, click 9.5.10).

- `rach-ConfigGenericTwoStepRA-r16` `RACH-ConfigGenericTwoStepRA-r16` (click 9.5.9): mandatory, common

  It contains **many common parameters** for the **2-step RA procedures** (click 9.1.1), including the case of **beam failure recovery**.

- `msgA-SSB-perRACH-OccasionAndCB-PreamblesPerSSB` **`CHOICE`**: optional, common

  It applies to the **2-step RA procedures**. Its meaning is **twofold**:

  - The **selected parameter** from the **`CHOICE`** indicates the **quantity** of **different values** of **ssb-time-domain-index** per **RACH occasion** (which matches the **first part** of the parameter name "`SSB-perRACH-Occasion`").

    Selecting `oneEighth` indicates **1 value** of **ssb-time-domain-index** associated with **8 RACH occasions**, selecting `oneFourth` indicates to **1 value** of **ssb-time-domain-index** associated with **4 RACH occasions**, selecting `four` indicates **4 values** of **ssb-time-domain-index** associated with **1 RACH occasion**, and so on.

  - **Each `ENUMERATED`** indicates the **quantity** of **RA preamble sequences** that can be used for **each possible value** of **ssb-time-domain-index** (which matches the **second part** of the parameter name "`CB-PreamblesPerSSB`").

  The **total quantity** of **contention based** (**CB**) **RA preambles** that would be **required** in a **RACH occasion** is obtained from their **product**:

  `CB-preambles-per-SSB` • `SSB-per-rach-occasion`

- `MsgA-totalNumberOfRA-Preambles-r16` **`INTEGER`** **`(1..63)`**: optional, common

  It is the **total quantity** of **random access** (**RA**) **preamble sequences** that can be used for the **2-step RA procedures** (click 9.1.1), when the **2-step RA procedures** and the **4-step RA procedures** do **not have shared random access occasions**. If it is **absent**, and the **2-step RA procedures** and the **4-step RA procedures** do **not have shared random access occasions**, **all 64 RA preamble sequences** can be used for the **2-step RA procedure**.

  Its value **must** be a **multiple** of the **quantity** of **SSBs** per **RACH occasion** (see above `msgA-SSB-perRACH-OccasionAndCB-PreamblesPerSSB`).

Notice that **data type** `RACH-ConfigCommon` (click 9.5.3) contains **very similar parameters**, with **very similar semantics**, but **actually different**, to the **parameters** of **this data type**.

For more detail on these parameters, refer to the table "*RACH-ConfigCommonTwoStepRA field descriptions*" of TS 38.331.

### 9.5.8.2 RACH-ConfigCommonTwoStepRA-r16 ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
RACH-ConfigCommonTwoStepRA-r16 ::= SEQUENCE {
  rach-ConfigGenericTwoStepRA-r16 RACH-ConfigGenericTwoStepRA-r16,
  msgA-TotalNumberOfRA-Preambles-r16 INTEGER (1..63) OPTIONAL,      -- Need S
  msgA-SSB-PerRACH-OccasionAndCB-PreamblesPerSSB-r16 CHOICE {
    oneEighth ENUMERATED {n4,n8,n12,n16,n20,n24,n28,n32,
                          n36,n40,n44,n48,n52,n56,n60,n64},
    oneFourth ENUMERATED {n4,n8,n12,n16,n20,n24,n28,n32,
                          n36,n40,n44,n48,n52,n56,n60,n64},
    oneHalf   ENUMERATED {n4,n8,n12,n16,n20,n24,n28,n32,
                          n36,n40,n44,n48,n52,n56,n60,n64},
    one       ENUMERATED {n4,n8,n12,n16,n20,n24,n28,n32,
                          n36,n40,n44,n48,n52,n56,n60,n64},
    two       ENUMERATED {n4,n8,n12,n16,n20,n24,n28,n32},
    four    INTEGER (1..16),
    eight   INTEGER (1..8),
    sixteen INTEGER (1..4)
  } OPTIONAL,                                                  -- Cond 2StepOnly
  msgA-CB-PreamblesPerSSB-PerSharedRO-r16 INTEGER (1..60)
                                          OPTIONAL,          -- Cond SharedRO
  msgA-SSB-SharedRO-MaskIndex-r16 INTEGER (1..15) OPTIONAL,        -- Need S
  groupB-ConfiguredTwoStepRA-r16 GroupB-ConfiguredTwoStepRA-r16
                                 OPTIONAL,                         -- Need S
  msgA-PRACH-RootSequenceIndex-r16 CHOICE {
    l839  INTEGER (0..837),
    l139  INTEGER (0..137),
    l571  INTEGER (0..569),
    l1151 INTEGER (0..1149)
  } OPTIONAL,                                                  -- Cond 2StepOnly
  msgA-TransMax-r16 ENUMERATED {n1, n2, n4, n6, n8, n10,
                                n20, n50, n100, n200} OPTIONAL,    -- Need R
  msgA-RSRP-Threshold-r16    RSRP-Range OPTIONAL,          -- Cond 2Step4Step
  msgA-RSRP-ThresholdSSB-r16 RSRP-Range OPTIONAL,                  -- Need R
  msgA-SubcarrierSpacing-r16 SubcarrierSpacing
                             OPTIONAL,                -- Cond 2StepOnlyL139
  msgA-RestrictedSetConfig-r16 ENUMERATED {unrestrictedSet,
         restrictedSetTypeA, restrictedSetTypeB} OPTIONAL, -- Cond 2StepOnly
  ra-PrioritizationForAccessIdentityTwoStep-r16 SEQUENCE {
    ra-Prioritization-r16 RA-Prioritization,
    ra-PrioritizationForAI-r16 BIT STRING (SIZE (2))
  } OPTIONAL,                                          -- Cond InitialBWP-Only
  ra-ContentionResolutionTimer-r16 ENUMERATED {sf8, sf16, sf24,
           sf32, sf40, sf48, sf56, sf64} OPTIONAL,         -- Cond 2StepOnly
  ...,
  [[
  ra-PrioritizationForSlicingTwoStep-r17 RA-PrioritizationForSlicing-r17
                                         OPTIONAL,   -- Cond InitialBWP-Only
  featureCombinationPreamblesList-r17 SEQUENCE
      (SIZE(1..maxFeatureCombPreamblesPerRACHResource-r17))
      OF FeatureCombinationPreambles-r17 OPTIONAL      -- Cond AdditionalRACH
  ]]
}
```

Where `maxFeatureCombPreamblesPerRACHResource-r17` is `256`, which is the **maximum quantity**, of **feature combination preambles**.

## 9.5.9 RACH-ConfigGenericTwoStepRA-r16 data type

It can be **both broadcast** (or **unicast**) within `SIB1` (click 5.3.4) and **unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `SIB1`, click 9.5.1.

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 9.5.2.

Its **2 parent data types** are `RACH-ConfigCommonTwoStepRA-r16` (click 9.5.8) and `CFRA-TwoStep-r16` (click 9.5.12).

It contains **either all common parameters or all dedicated parameters** (click 5.1.3).

**Each** of its **instances** contains **many parameters** for **PRACH** (click 7.1) and for the **2-step RA procedures** (click 9.1.1), including the case of **beam failure recovery**.

#### 9.5.9.1 `RACH-ConfigGenericTwoStepRA-r16` parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `msgA-PreamblePowerRampingStep-r16` `ENUMERATED {dB0, dB2, dB4, dB6}`: optional, common **or** dedicated

  It is the **power ramping step** for **MsgA PRACH**, in **2-step RA procedures**.

  If it is **absent** within the **instance** of `RACH-ConfigCommonTwoStepRA-r16` (click 9.5.8), within an **instance** of `AdditionalRACH-Config-r17`, the **UE** shall use `powerRampingStep` from `RACH-ConfigGeneric` (click 9.5.5) within the **instance** of `RACH-ConfigCommon` (click 9.5.3) that is within the **same instance** of `AdditionalRACH-Config-r17`.

  If the field is **absent** in **other cases**, the **UE** shall use the value of `powerRampingStep` from `RACH-ConfigGeneric` within the configured bandwidth part (refer to clause 5.1.3 of TS 38.321).

  It may **only** be **present** when **either**:

  - **No 4-step type RA** is **configured** in the **bandwidth part**.
  - **Separate random access occasions** (**ROs**) with **4-step RA procedures**.

  It is **absent** if `RACH-ConfigGenericTwoStepRA-r16` (click 9.5.9) is included in an **instance** of `CFRA-TwoStep-r16` (click 9.5.12), within `RACH-ConfigDedicated` (click 9.5.10). In this case, the **UE** uses `msgAPreamblePowerRampingStep` from `RACH-ConfigGenericTwoStepRA-r16` configured for **CBRA**.

- `msgA-RO-FDM-r16` `ENUMERATED {one, two, four, eight}`: optional, common **or** dedicated

  It is the **quantity** of **frequency transmission occasions** (in 1 **PRACH time occasion**) for the **RA preamble** in **2-step RA procedures** (click 9.1.1). Its possible values are `1`, `2`, `4` or `8`.

  If it is **absent** within an instance of `RACH-ConfigCommonTwoStepRA-r16` that is directly configured in a **bandwidth part** (i.e., **not** within `AdditionalRACH-Config-r17`), the **UE** shall use `msg1-FDM` from the `RACH-ConfigGeneric` **instance** from the **corresponding bandwidth part**.

  If it is **absent** within an **instance** of `RACH-ConfigCommonTwoStepRA-r16`, that is within an **instance** of `AdditionalRACH-Config-r17`, the **UE** shall use the value of `msg1-FDM` that is within the `RACH-ConfigCommon` **instance**, that is within the same **instance** of `AdditionalRACH-Config-r17` (refer to clause 6.3.3.2 of TS 38.211).

It may **only** be **present** if **no 4-step RA procedure** is configured in the BWP or in the case of **separate ROs** with **4-step RA procedures**.

- `msgA-RO-FrequencyStart-r16 INTEGER (0..maxNrofPhysicalResourceBlocks-1)`: optional, common **or** dedicated

  It provides the **frequency transmission occasions** (through **PRACH**) for the **RA preamble** in **2-step RA procedures** (click 9.1.1).

  If it is **absent** within an instance of `RACH-ConfigCommonTwoStepRA-r16` that is directly configured in a **bandwidth part** (i.e., **not** within `AdditionalRACH-Config-r17`), the **UE** shall use `msg1-FrequencyStart` from the `RACH-ConfigGeneric` **instance** from the **corresponding bandwidth part**.

  If it is **absent** within an **instance** of `RACH-ConfigCommonTwoStepRA-r16`, that is within an **instance** of `AdditionalRACH-Config-r17`, the **UE** shall use the value of `msg1-FrequencyStart` that is within the `RACH-ConfigCommon` **instance**, that is within the same **instance** of `AdditionalRACH-Config-r17` (refer to clauses 5.3.2 and 6.3.3.2 of TS 38.211).

  It may **only** be **present** if **no 4-step RA procedure** is configured in the BWP or in the case of **separate ROs** with **4-step RA procedures**.

  The constant `maxNrofPhysicalResourceBlocks-1` is `274`, which is the **maximum quantity**, minus **1**, of **PRBs** in **each** of the **UE**'s **bandwidth parts**.

- `msgA-PRACH-ConfigurationIndex-r16 INTEGER (0..262)`: optional, common **or** dedicated

  It provides the **PRACH sequence Format** for **2-step RA procedures** by entering its value in the corresponding table 6.3.3.2-2 (**FR1** and **paired spectrum**), 6.3.3.2-3 (**FR1** and **TDD**) or 6.3.3.2-4 (**FR2** and **TDD**) of TS 38.211.

  If it is **absent** within an instance of `RACH-ConfigCommonTwoStepRA-r16` that is directly configured in a **bandwidth part** (i.e., **not** within `AdditionalRACH-Config-r17`), the **UE** shall use `prach-ConfigurationIndex` from the `RACH-ConfigGeneric` **instance** from the **corresponding bandwidth part**.

  If it is **absent** within an **instance** of `RACH-ConfigCommonTwoStepRA-r16`, that is within an **instance** of `AdditionalRACH-Config-r17`, the **UE** shall use the value of `prach-ConfigurationIndex` that is within the `RACH-ConfigCommon` **instance**, that is within the same **instance** of `AdditionalRACH-Config-r17`. If the value is **from** `256` **to** `262`, the parameter `prach-ConfigurationIndex-v1610` should be considered **configured** (refer to clause 6.3.3.2 of TS 38.211).

  It may **only** be **present** if **no 4-step RA procedure** is configured in the BWP or in the case of **separate ROs** with **4-step RA procedures**.

- `msgB-ResponseWindow-r16 ENUMERATED {sl240, sl640, sl960, sl1280, sl1920, sl2560}`: optional, common **or** dedicated
- `msgB-ResponseWindow-v1700 ENUMERATED {sl240, sl320, sl640, sl960, sl1280, sl1920, sl2560}`: optional, common **or** dedicated

  **Each** of these parameters is the **window length** for **MsgB** (click 9.3.7) in **quantity** of **slots**.

  The network configures a value **lower than or equal to 40ms** (refer to clause 5.1.1 of TS 38.321).

  These **2 parameters** are **mutually exclusive**. If **both** are **absent**, the **UE** uses the value of `msgB-ResponseWindow-r16` from `RACH-ConfigGenericTwoStepRA-r16` (click 9.5.9) configured for **CBRA**.

- `msgA-zeroCorrelationZoneConfig-r16 INTEGER (0..15)`: optional, common **or** dedicated

  It is the value of "*Ncs*" for **msgA RA preamble** (refer to table 6.3.3.1-5 of TS 38.211).

  If it is **absent** within the **instance** of `RACH-ConfigCommonTwoStepRA-r16` (click 9.5.8), within an **instance** of `AdditionalRACH-Config-r17`, the **UE** shall use `zeroCorrelationZoneConfig` from `RACH-ConfigGeneric` (click 9.5.5) within the **instance** of `RACH-ConfigCommon` (click 9.5.3) that is within the **same instance** of `AdditionalRACH-Config-r17`.

  If the field is **absent** in **other cases**, the **UE** shall use the value of `zeroCorrelationZoneConfig` from `RACH-ConfigGeneric` within the configured bandwidth part (refer to clause 5.1.3 of TS 38.321).

  It may **only** be **present** when **either**:

  - **No 4-step type RA** is **configured** in the **bandwidth part**.
  - **Separate random access occasions** (**ROs**) with **4-step RA procedures**.

For more detail on these parameters, refer to the table "*RACH-ConfigGenericTwoStepRA field descriptions*" of TS 38.331.

### 9.5.9.2 `RACH-ConfigGenericTwoStepRA-r16` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
RACH-ConfigGenericTwoStepRA-r16 ::= SEQUENCE {
  msgA-PRACH-ConfigurationIndex-r16 INTEGER (0..262)
                                    OPTIONAL,              -- Cond 2StepOnly
  msgA-RO-FDM-r16 ENUMERATED {one, two, four, eight}
                  OPTIONAL,                                -- Cond 2StepOnly
  msgA-RO-FrequencyStart-r16 INTEGER (0..maxNrofPhysicalResourceBlocks-1)
                             OPTIONAL,                     -- Cond 2StepOnly
  msgA-ZeroCorrelationZoneConfig-r16 INTEGER (0..15)
                                     OPTIONAL,           -- Cond 2StepOnly
  msgA-PreamblePowerRampingStep-r16 ENUMERATED {dB0, dB2, dB4, dB6}
                                    OPTIONAL,        -- Cond 2StepOnlyNoCFRA
  msgA-PreambleReceivedTargetPower-r16 INTEGER (-202..-60)
                                       OPTIONAL,     -- Cond 2StepOnlyNoCFRA
  msgB-ResponseWindow-r16 ENUMERATED {sl1, sl2, sl4, sl8, sl10, sl20,
                          sl40, sl80, sl160, sl320} OPTIONAL, -- Cond NoCFRA
  preambleTransMax-r16 ENUMERATED {n3, n4, n5, n6, n7, n8, n10, n20,
                       n50, n100, n200} OPTIONAL,    -- Cond 2StepOnlyNoCFRA
```

```
    ...,
    [[
    msgB-ResponseWindow-v1700 ENUMERATED {sl240, sl640, sl960, sl1280,
                              sl1920, sl2560} OPTIONAL          -- Cond NoCFRA2
    ]]
}
```

Where `maxNrofPhysicalResourceBlocks-1` is `274`, which is the **maximum quantity**, minus **1**, of **PRBs** in **each** of the **UE**'s **bandwidth parts**.

### 9.5.10 `RACH-ConfigDedicated` data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 6.12.3.

Its **only parent data type** is `ReconfigurationWithSync` (click 3.9.8).

**All** its **parameters** are **dedicated** (click 5.1.3).

**Each** of its **instances** contains **many dedicated parameters** for **PRACH** (click 7.1) and for the **4 types** of **random access procedures** (click 9.1).

#### 9.5.10.1 `RACH-ConfigDedicated` parameters

Its **four parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `cfra CFRA` (click 9.5.8): optional, dedicated

  If **present**, the **UE** uses for **random access** (click 9) the **4-step contention-free random access** (**4-step CFRA**, click 9.1.4) **procedure**. It contains the **parameters** for the **4-step CFRA procedure**.

- `cfra-TwoStep-r16 CFRA-TwoStep-r16` (click 9.5.12): optional, dedicated

  If **present**, the **UE** uses for **random access** the **2-step contention-free random access procedure** (**2-step CFRA**, click 9.1.6). It contains the **parameters** for the **2-step CFRA procedure**. It can **only** be **present** if `msgA-ConfigCommon-r16` from `BWP-UplinkCommon` (click 2.2.8.9) is configured on the **bandwidth part**.

Parameters `cfra` and `cfra-TwoStep` are **mutually exclusive**. If **both** of them are **absent**, the **UE** uses for **random access** a **contention-based random access** (**CBRA**, click 9.1.2) **procedure**.

- `ra-prioritization RA-Prioritization`: optional, dedicated

  It contains the parameters for the **prioritized random access procedure** to a **given target cell** (refer to clause 5.1.1 of TS 38.321).

- `ra-PrioritizationTwoStep-r16 RA-Prioritization`: optional, dedicated

  It contains the parameters for the **2-step prioritized random access procedure** to a **given target cell** (refer to clause 5.1.1 of TS 38.321).

#### 9.5.10.2 RACH-ConfigDedicated ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
RACH-ConfigDedicated ::= SEQUENCE {
  cfra CFRA OPTIONAL,                                    -- Need S
  ra-Prioritization RA-Prioritization OPTIONAL,          -- Need N
  ...,
  [[
  ra-PrioritizationTwoStep-r16 RA-Prioritization OPTIONAL,   -- Need N
  cfra-TwoStep-r16 CFRA-TwoStep-r16 OPTIONAL             -- Need S
  ]]
}
```

### 9.5.11 CFRA data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 9.5.2.

Its **only parent data type** is `RACH-ConfigDedicated` (click 9.5.10).

**All** its parameters are **dedicated** (click 5.1.3).

**Each** of its **instances** contains **many dedicated parameters** for **PRACH** (click 7.1) and for the **4-step contention-based random access (CFRA) procedure** (click 9.1.4).

#### 9.5.11.1 CFRA parameters

**Some** of its **relevant parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `occasions` **SEQUENCE**: optional, dedicated

  It contains the **random access occasions** for **contention-free random access (CFRA)** . If it is **absent**, the **UE** uses the **random access occasions** configured in the `RACH-ConfigCommon` (click 9.5.3) **instance** of the **first active Uplink bandwidth part**. Its **SEQUENCE** contains `rach-ConfigGeneric` and `ra-ssb-OccasionMaskIndex` (see below).

- `ra-ssb-OccasionMaskIndex` **INTEGER** `(0..15)`: mandatory (within **CHOICE**), dedicated

  It is the explicitly signaled **PRACH Mask Index** for **random access resource selection** in TS 38.321. The **mask** is valid for **all SSB resources** signaled in `ssb-ResourceList` within the **same data type instance**.

  It has **3 same-name parameters**, with the **same data type** and **similar semantics**, from `BeamFailureRecoveryConfig` (click 9.5.4), from `CFRA-TwoStep-r16` (click 9.5.12) and from `SI-RequestResources` (click 5.5.3.5).

- `resources` **CHOICE**: mandatory, dedicated

  It is a **CHOICE** between `ssb` (see below) and `csirs`.

- `rach-ConfigGeneric` `RACH-ConfigGeneric` (click 9.5.5): mandatory (within **OPTIONAL**), dedicated

  It contains the configuration of the **contention-free random access occasions** for **CFRA**. The **UE** shall **ignore** the values of `preambleReceivedTargetPower`, `preambleTransMax`, `powerRampingStep` and `ra-ResponseWindow` within

this **parameter** and use the corresponding values provided in the corresponding `RACH-ConfigCommon` (click 9.5.3) **instance**.

- `ssb` `SEQUENCE`: mandatory (within `CHOICE`), dedicated

  Its `SEQUENCE` contains `ssbResourceList` (see below) and `ra-ssb-OccasionMaskIndex` (see above).

- `ssb-perRACH-Occasion` `ENUMERATED`: optional, dedicated

  It is the **quantity** of **SSBs** per **RACH occasion**. Its possible values are `1/8`, `1/4`, `1/2`, `1`, `2`, `4`, `8` and `16`.

- `ssb-ResourceList` `SEQUENCE OF`: mandatory, dedicated

  It is a **list** of **SSB resources**, **each** with **data type** `CFRA-SSB-Resource`.

- `totalNumberOfRA-Preambles` `INTEGER (1..63)`: optional, dedicated

  It is the **total quantity** of **preambles** used for **contention-free random access** (**CFRA**) in the **RACH resources** defined in **CFRA**, **excluding** preambles used for **other purposes** (e.g., for **SI request**). If it is **absent** and `occasions` is **present**, the **UE** may assume that **all** the **64 preambles** are for **RA**. Its value should be **consistent** with the value of `ssbperRACH-Occasion` (see above), if **present**, (i.e., it should be a **multiple** of the **quantity** of **SSBs per RACH occasion**).

For more detail on these parameters, refer to the table "*CFRA field descriptions*" of TS 38.331.

### 9.5.11.2 `CFRA` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
CFRA ::= SEQUENCE {
  occasions SEQUENCE {
    rach-ConfigGeneric RACH-ConfigGeneric,
    ssb-perRACH-Occasion ENUMERATED {oneEighth, oneFourth,
        oneHalf, one, two, four, eight, sixteen} OPTIONAL  -- Cond Mandatory
  } OPTIONAL,                                              -- Need S
  resources CHOICE {
    ssb SEQUENCE {
      ssb-ResourceList SEQUENCE (SIZE(1..maxRA-SSB-Resources))
                       OF CFRA-SSB-Resource,
      ra-ssb-OccasionMaskIndex INTEGER (0..15)
    },
    csirs SEQUENCE {
      csirs-ResourceList SEQUENCE (SIZE(1..maxRA-CSIRS-Resources))
                         OF CFRA-CSIRS-Resource,
      rsrp-ThresholdCSI-RS RSRP-Range
    }
  },
  ...,
  [[
  totalNumberOfRA-Preambles INTEGER (1..63) OPTIONAL       -- Cond Occasions
  ]]
}
```

## 9.5.12 `CFRA-TwoStep-r16` data type

It is **only unicast** within the **Non-SI parameter group** (click 5.3.6).

For its **ancestor tree up to** `CellGroupConfig` (click 3.7.3), click 6.12.3.

Its **only parent data type** is `RACH-ConfigDedicated` (click 9.5.10).

**All** its parameters are **dedicated** (click 5.1.3).

**Each** of its **instances** contains **many dedicated parameters** for **PRACH** (click 7.1) and for the **2-step contention-based random access** (**CFRA**) **procedure** (click 9.1.6).

#### 9.5.12.1 `CFRA-TwoStep-r16` parameters

Its **eight parameters** (indicating for **each** of them its **data type** and **semantics**) are:

- `msgA-CFRA-PUSCH-r16` `MsgA-PUSCH-Resource-r16`: mandatory, dedicated

  It defines the **PUSCH grant** for the transmission of **MsgA** (click 9.3.6) in the **2-step CFRA procedure**.

- `msgA-TransMax-r16` **`ENUMERATED`**: optional, dedicated

  It is the **maximum quantity** of **MsgA preamble transmissions** before **switching** to **4-step random access** (click 9.1). For more detail, refer to clause 5.1.1 of TS 38.321.

  This **parameter only** applies when **2-step** and **4-step RA procedures** are **both configured** and **switching** to **4-step random access** is supported.

  If it **absent**, **switching** from a **2-step random access procedure** to a **4-step random access procedure** is **not allowed**.

- `occasionsTwoStepRA-16` **`SEQUENCE`**: optional, dedicated

  It defines the **RA occasions** for the **2-step CFRA** (click 9.1.6) **procedure**. If it is **absent**, the **UE** uses the **RA occasions** configured in the **instance** of `RACH-ConfigCommonTwoStepRA-r16` (click 9.5.8) from the **first active Uplink bandwidth part**.

  Its **data type** is a **`SEQUENCE`** that contains `rach-ConfigGenericTwoStepRA-r16` and `ssb-PerRACH-OccasionTwoStepRA-r16` (see below).

- `ra-ssb-OccasionMaskIndex` **`INTEGER`** `(0..15)`: mandatory, dedicated

  It is the explicitly signaled **PRACH Mask Index** for **random access resource selection**. The **mask** is **valid** for **all** the **SSB resources signaled** in `ssb-ResourceList` within the **same data type instance**.

  It has **3 same-name parameters**, with the **same data type** and **similar semantics**, from `BeamFailureRecoveryConfig` (click 9.5.4), from `CFRA` (click 9.5.11), from `CFRA-TwoStep-r16` (click 9.5.12) and from `SI-RequestResources` (click 5.5.3.5).

- `rach-ConfigGenericTwoStepRA-r16` `RACH-ConfigGenericTwoStepRA-r16` (click 9.5.9): mandatory (within optional), dedicated

  It contains the **contention free random access occasions** for the **2-step CFRA procedure**.

- `resourcesTwoStep-r16` **`SEQUENCE`**: mandatory, dedicated

  Its **data type** is a **`SEQUENCE`** that contains `ra-ssb-OccasionMaskIndex` (see above) and `ssb-ResourceList` (see below).

- `ssb-ResourceList SEQUENCE OF`: mandatory, dedicated

  It is a **list** of **SSB resources**, **each** with **data type** `CFRA-SSB-Resource`.

- `ssb-PerRACH-OccasionTwoStepRA-r16 ENUMERATED`: mandatory (within optional), dedicated

  It is the **quantity** of **SSBs** per **RACH occasion** for the **2-step random access procedures**.

### 9.5.12.2 `CFRA-TwoStep-r16` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
CFRA-TwoStep-r16 ::= SEQUENCE {
  occasionsTwoStepRA-r16 SEQUENCE {
    rach-ConfigGenericTwoStepRA-r16 RACH-ConfigGenericTwoStepRA-r16,
    ssb-PerRACH-OccasionTwoStepRA-r16
          ENUMERATED {oneEighth, oneFourth, oneHalf, one,
                      two, four, eight, sixteen}
  } OPTIONAL,                                                    -- Need S
  msgA-CFRA-PUSCH-r16 MsgA-PUSCH-Resource-r16,
  msgA-TransMax-r16 ENUMERATED {n1, n2, n4, n6, n8, n10,
                                n20, n50, n100, n200} OPTIONAL,  -- Need S
  resourcesTwoStep-r16 SEQUENCE {
    ssb-ResourceList SEQUENCE (SIZE(1..maxRA-SSB-Resources))
                     OF CFRA-SSB-Resource,
    ra-ssb-OccasionMaskIndex INTEGER (0..15)
  },
  ...
}
```

## 9.5.13 `RA-InformationCommon-r16` data type

Its **instances** do **not** contain **UE** configuration **parameters**.

It is used by the **UE** to **report** to the **NR node** relevant information in respect to the **random access procedure** (click 9).

Its **3 parent data types** are `RA-Report-r16` (**random access** report), `RLF-Report-r16` (**radio link failure** report) and `SuccessHO-Report-r17` (**successful handover** report).

Its **ancestor tree up to** the **top** (`UL-DCCH-Message`) is:

```
                          UL-DCCH-Message (click 5.8.5.9)
message                     UL-DCCH-MessageType
  c1                          CHOICE
    messageClassExtension       CHOICE
      c2                          CHOICE
        ueInformationResponse-r16 UEInformationResponse-r16
          criticalExtensions          CHOICE
            ueInformationResponse-r16 UEInformationResponse-r16-IEs
              ra-ReportList-r16           RA-ReportList-r16 (0..1)
                                            RA-Report-r16 (1..maxRAReport-r16)
                  ra-InformationCommon-r16  RA-InformationCommon-r16 (0..1)
            nonCriticalExtension        UEInformationResponse-v1700-IEs (0..1)
              successHO-Report-r17        SuccessHO-Report-r17 (0..1)
                ra-InformationCommon-r17  RA-InformationCommon-r16 (0..1)
```

It also has **3 ancestor trees up to** the **top**, corresponding to **3 UE variables** (click 5.6.2), as follows:

```
                          VarRA-Report-r16
  ra-ReportList-r16         RA-ReportList-r16 (0..1)
                              RA-Report-r16 (1.. maxRAReport-r16)
      ra-InformationCommon-r16  RA-InformationCommon-r16 (0..1)

                          VarRLF-Report-r16
  rlf-Report-r16            RLF-Report-r16
    ra-InformationCommon-r16  RA-InformationCommon-r16 (0..1)

                          VarSuccessHO-Report-r17-IEs
  successHO-Report-r17      SuccessHO-Report-r17 (0..1)
    ra-InformationCommon-r17  RA-InformationCommon-r16 (0..1)
```

### 9.5.13.1 `RA-InformationCommon-r16` parameters

Its **parameters** may be seen in its **ASN.1 specification**.

### 9.5.13.2 `RA-InformationCommon-r16` ASN.1 specification

Its **complete ASN.1** (click 5.9) specification is:

```
RA-InformationCommon-r16 ::= SEQUENCE {
  absoluteFrequencyPointA-r16 ARFCN-ValueNR,
  locationAndBandwidth-r16 INTEGER (0..37949),
  subcarrierSpacing-r16 SubcarrierSpacing,
  msg1-FrequencyStart-r16
                  INTEGER (0..maxNrofPhysicalResourceBlocks-1) OPTIONAL,
  msg1-FrequencyStartCFRA-r16
                  INTEGER (0..maxNrofPhysicalResourceBlocks-1) OPTIONAL,
  msg1-SubcarrierSpacing-r16     SubcarrierSpacing OPTIONAL,
  msg1-SubcarrierSpacingCFRA-r16 SubcarrierSpacing OPTIONAL,
  msg1-FDM-r16     ENUMERATED {one, two, four, eight} OPTIONAL,
  msg1-FDMCFRA-r16 ENUMERATED {one, two, four, eight} OPTIONAL,
  perRAInfoList-r16   PerRAInfoList-r16,
  ...,
  [[
  perRAInfoList-v1660 PerRAInfoList-v1660 OPTIONAL
  ]],
  [[
  msg1-SCS-From-prach-ConfigurationIndex-r16
                  ENUMERATED {kHz1dot25, kHz5, spare2, spare1} OPTIONAL
  ]],
  [[
  msg1-SCS-From-prach-ConfigurationIndexCFRA-r16
                  ENUMERATED {kHz1dot25, kHz5, spare2, spare1} OPTIONAL
  ]],
  [[
  msgA-RO-FrequencyStart-r17
                   INTEGER (0..maxNrofPhysicalResourceBlocks-1) OPTIONAL,
  msgA-RO-FrequencyStartCFRA-r17
                   INTEGER (0..maxNrofPhysicalResourceBlocks-1) OPTIONAL,
  msgA-SubcarrierSpacing-r17 SubcarrierSpacing OPTIONAL,
  msgA-RO-FDM-r17     ENUMERATED {one, two, four, eight} OPTIONAL,
  msgA-RO-FDMCFRA-r17 ENUMERATED {one, two, four, eight} OPTIONAL,
  msgA-SCS-From-prach-ConfigurationIndex-r17
                   ENUMERATED {kHz1dot25, kHz5, spare2, spare1} OPTIONAL,
  msgA-TransMax-r17 ENUMERATED {n1, n2, n4, n6, n8,
                                n10, n20, n50, n100, n200} OPTIONAL,
  msgA-MCS-r17 INTEGER (0..15) OPTIONAL,
  nrofPRBs-PerMsgA-PO-r17 INTEGER (1..32) OPTIONAL,
  msgA-PUSCH-TimeDomainAllocation-r17
                            INTEGER (1..maxNrofUL-Allocations) OPTIONAL,
  frequencyStartMsgA-PUSCH-r17
                   INTEGER (0..maxNrofPhysicalResourceBlocks-1) OPTIONAL,
```

```
    nrofMsgA-PO-FDM-r17 ENUMERATED {one, two, four, eight} OPTIONAL,
    dlPathlossRSRP-r17 RSRP-Range OPTIONAL,
    intendedSIBs-r17 SEQUENCE (SIZE (1..maxSIB)) OF SIB-Type-r17 OPTIONAL,
    ssbsForSI-Acquisition-r17 SEQUENCE (SIZE (1..maxNrofSSBs-r16))
                              OF SSB-Index OPTIONAL,
    msgA-PUSCH-PayloadSize-r17 BIT STRING (SIZE (5)) OPTIONAL,
    onDemandSISuccess-r17 ENUMERATED {true} OPTIONAL
    ]]
  }
```

# 10. NR procedures and techniques

## 10.1 Bandwidth adaptation (BA)

**Bandwidth adaptation** (**BA**) consists of dynamically changing the **Downlink bandwidth** and/or the **Uplink** (or **SUL**) **bandwidth** to better suit the **characteristics** and **needs** of the **different UEs** camped in the **cell**.

**Bandwidth adaptation** is **implemented** through the introduction of **bandwidth parts** (click 2.2.7).

Having designed **5G carriers** and **UE bandwidth parts** to support **different locations**, a variable quantity of **Resource Blocks** and **different subcarrier spacings**, enables **5G** to handle signals with a **similar structure** but **very different characteristics**. This empowers **5G** with **enormous flexibility** to adapt to **very diverse** terminals and communication scenarios, spanning very different **bandwidths**.

**Device-side receive Bandwidth Adaptation** consists of a **UE** capability to use a relatively modest bandwidth for **monitoring** control channels and **receiving** low/medium data rates, and on some cases, also dynamically open up a wideband receiver when needed, to support very high data rates. This has the great advantages of allowing for **simpler/cheaper UEs** that cannot handle the total channel bandwidth and of **reducing power consumption** in the **UE**.

Contrary to this **flexibility** introduced in **5G**, in **LTE all** devices **must** support the **full channel bandwidth**, with **only one subcarrier spacing**, making **LTE** a much more rigid technology than **5G**.

For an **overview** of **RRC bandwidth adaptation**, refer to clauses 6.10 and 7.8 of TS 38.300.

## 10.2 Carrier aggregation (CA)

**Carrier aggregation** is the capability to **transmit** and **receive** in **parallel** over **several carriers** and **aggregate** (inverse multiplexing) their data flows into **one single data flow**. A **UE without carrier aggregation capability** can **receive** in **only 1 Downlink carrier** and **transmit** in **only 1 Uplink carrier**, where **both carriers** correspond to **only 1 serving cell**.

**Each aggregated carrier** is called a **component carrier** (**CC**). **Each component carrier** belongs to **1** and **only 1 serving cell**.

**Carrier aggregation** implies **very difficult radio problems**, because it requires that the **UEs** can **transmit/receive multiple carriers** in **parallel**.

**Carrier aggregation cannot** be performed between **5G carriers** and **LTE carriers**. However, it is possible to perform **dual connectivity** (click 10.3) between **5G** and **LTE**, which **adds** the **capacity** provided by the **5G** (**aggregated**) **carrier**(**s**) and by the **LTE** (**aggregated**) **carrier**(**s**). **Carrier aggregation** and **dual connectivity** are **complementary** technologies, and on many cases they are **jointly used**.

**Carrier aggregation**, **dual connectivity** and their **joint usage** result in **different cell types** (click 3.2.2).

**Carrier aggregation** is usually **presented** as a way to **boost** the **UEs' system capacity**, aggregating **many carriers**, **each** of which has a **system capacity** close to the **5G**

**technology capacity**. However, in practice it is more frequently used to **aggregate carriers** from **fragmented operator spectrum**, **each** of which have a **system capacity** well below the **5G technology capacity**, in order to provide the **UEs** with a **decent system capacity**.

For the **carrier aggregation configuration**, click 10.2.1.1.

For the **carrier aggregation control**, click 10.2.1.1.

For the **carrier aggregation architecture**, click 10.2.4.

For the **carrier aggregation objectives**, click 10.2.5.

For **carrier aggregation** and **time synchronization**, click 10.2.6.

For the **UE capabilities** in respect to **carrier aggregation**, click 4.9.1.

For more detail on **carrier aggregation**, refer to clauses 4.5 of TS 38.211 and 5.4, 6.7, 7.7 of TS 38.300.

Source web pages:

- https://www.nokia.com/about-us/newsroom/articles/5g-carrier-aggregation-explained/
- https://www.ericsson.com/en/blog/2021/6/what-why-how-5g-carrier-aggregation

## 10.2.1 Carrier aggregation control and configuration

The **NR node** performs the **carrier aggregation control** (click 10.2.1.1) and **configuration** (click 10.2.1.1).

### 10.2.1.1 Carrier aggregation control

The **NR node** can configure **each UE** with a **complete cell group** (click 3.2.2) and it can **also dynamically add** or **remove secondary cells** from the **UE**'s **cell group**.

The **reconfiguration**, **addition** and **removal** of **secondary cells** is performed with **RRC**. At **intra-NR handover** and during **connection resume** from **RRC_INACTIVE** (click 5.8.2) the network can also **add**, **remove**, **keep**, or **reconfigure secondary cells** for usage with the target **special cell**.

When **adding** a **new secondary cell**, **dedicated RRC signaling** is used for **sending all** required **system information** of the **secondary cell** (i.e. while in **RRC_CONNECTED**) and the **UE does not need** to acquire **broadcast system information** directly from the **secondary cells**. For the configuration **parameters** of a **cell**, click 3.9.

The **configuration** of a **whole cell group** can also be conveyed via **RRC**. For the configuration **parameters** of a **cell group**, click 3.7.

### 10.2.1.2 Carrier aggregation configuration

The different **component carriers** usually have **different bandwidths** and **binary capacities**. When **supplementary Uplink** (**SUL**, click 10.4) is used, the **SUL carrier** may be **aggregated** as **one more component carrier**.

The **quantity** of **Uplink** (or **SUL**, if configured) **component carriers must** be **lower** than, **or equal** to, the **quantity** of **Downlink component carriers**. The **carrier aggregation configuration** in **Downlink** and in **Uplink** (or **SUL**, if configured) is usually different. The **carrier aggregation configuration** can be **changed** in **every slot**.

For **Downlink**, the **maximum quantity** of **component carriers** is **16** and the **maximum aggregated bandwidth** is **3.2 GHz** .

For **Uplink** (with **SUL**, if configured) the **maximum quantity** of **component carriers** is **16** and the **maximum aggregated bandwidth** is **1.6 GHz** .

For **each carrier aggregation configuration type** (click 10.2.1.2.1), the TS 38.101-*x* documents indicate **all** the **allowed carrier aggregation configurations**, in respect to:

- **CA operating bands**
- **CA bandwidth classes**
- **CA configuration** (**overall**)

The following table indicates the specific **TS document table** that **contains each instance** of **allowed configurations**:

| Table content | CA configuration type | | FR1 table(s) of TS 38.101-2 | FR2 table(s) of TS 38.101-2 |
|---|---|---|---|---|
| **CA operating bands** | **Intra-band CA** | **Contiguous** | 5.2A1-1 | 5.2A1-1 |
| | | **Non-contiguous** | 5.2A1-2 | |
| | **Inter-band CA** | **2 band combs.** | 5.2A2.1-1 | 5.2A2-1 |
| | | **3 band combs.** | 5.2A2.2-1 | |
| | | **4 band combs.** | 5.2A2.3-1 | |
| | | **5 band combs.** | 5.2A2.4-1 | |
| | **Inter-FR CA** | **2 band combs.** | 5.2A.1-1 of TS 38.101-3 | |
| **CA bandwidth classes (min and max BW, CC# and fallback group)** (10.2.1.2.2) | | | 5.3A.5-1 | 5.3A.4-1 |
| **CA configuration: Bands, bandwidth classes, channels and bandwidth combination sets (BCS)**<br><br>**For FR2, also fallback groups** | **Intra-band CA** | **Contiguous** | 5.5A.1-1 | 5.5A.1-1 |
| | | **Non-contiguous** | 5.5A.2-1 | 5.5A.2-1 (1 class) |
| | | | | 5.5A.2-2 (n classes) |
| | | **Mixed** | 5.5A.2-2 | N/A |
| | **Inter-band CA** | **2 band combs.** | 5.5A.3.1-1a, b, c, d, e, f, g, h, i, j, k, l, m, n | 5.5A.3-1 |
| | | **3 band combs.** | 5.5A.3.2-1 | N/A |
| | | **4 band combs.** | 5.5A.3.3-1 | |
| | | **5 band combs.** | 5.5A.3.4-1 | |
| | **Inter-FR CA** | **2 band combs.** | 5.2A.1-1 of TS 38.101-3 | |

For the allowed **configurations** (bands, channels, component carriers, bandwidth, power, …) **5G carrier aggregation**, refer to clauses 5.2A, 5.3A, 5.4A, 5.5A, 6.2A, 6.3A, 6.4A, 6.5A, 7.3A, 7.4A, 7.5A, 7.6A 7.7A, 7.8A, 7.9A and 7.10A of TS 38.101-**1**, **-2** and **-5**.

For more detail on the different cases of **carrier aggregation configuration**, refer to clause 5.3A5 of TS 38.101-**1** (for FR1) and clause 5.3A4 of TS 38.101-**1** (for **FR1**).

#### 10.2.1.2.1 Carrier aggregation configuration types

There are **3** main **types** of **carrier aggregation configuration**:

- **Intra-band carrier aggregation configuration**

  **All** the **component carriers** belong to the **same band**.

  This main type can be divided into **two types**:

  - **Intra-band contiguous carrier aggregation configuration**

    **All** the **component carriers** within the **band** are **contiguous** in frequency.

  - **Intra-band non-contiguous carrier aggregation configuration**

    **Not all** the **component carriers** within the **band** are **contiguous** in frequency.

- **Inter-band carrier aggregation configuration**

  **All** the **component carriers** belong to **different bands** of the **same frequency range**.

- **Inter-FR carrier aggregation configuration**

  The **component carriers** belong to **bands** from **both FR1** and **FR2**.

#### 10.2.1.2.2 Carrier aggregation bandwidth class

**Each carrier aggregation bandwidth class** defines **its**:

- **Identifier**

  **Each carrier aggregation bandwidth class** is **identified** by **either** a **capital letter** (e.g., "**C**", "**H**") or the **letter** "**R**" **followed** by a **number** (e.g., "**R7**" or "**R12**").

- **Minimum total aggregated bandwidth**.
- **Maximum total aggregated bandwidth**.
- **Quantity** of **contiguous components carriers**.
- **Fallback group**(**s**).

**Each NR frequency range** has its specific definition of **carrier aggregation bandwidth classes**. The **identifiers** for the **classes** in **FR1** and in **FR2 overlap**, which may create some **confusion**. **Every 2 classes**, **one** from **FR1** and the **other** from **FR2**, that have the **same class identifier** (e.g., **both** have the identifier "**B**" or "**G**") are **two completely different classes**.

The **carrier aggregation bandwidth classes** are **defined** in tables 5.3A.5-1 (for **FR1**) and 5.3A.4-1 (for **FR2**) of (respectively) TS 38.101-**1** and -**2**.

In **FR1**, there are **14 carrier aggregation bandwidth classes**. Their **quantity** of **contiguous components carriers** ranges from **1** to **7**. Their **aggregated bandwidth** ranges from **20 MHz** to "$4\times BW_{Channel,max}$" (**400 MHz**). The value of "$BW_{Channel,max}$" is the **maximum channel bandwidth** among **all FR1 bands** in a **release**.

In **FR2**, there are **33 carrier aggregation bandwidth classes**. Their **quantity** of **contiguous components carriers** ranges from **1** to **12**. Their **aggregated bandwidth** ranges from **400 MHz** to **2.4 GHz**.

### 10.2.2 Signaling in carrier aggregation

When **carrier aggregation** (**CA**) is configured, the **signaling** of the **different cells** (click 3.2.2) in the **UE**'s **cell group** is conveyed in the following way:

- **RRC signaling** (click 5)

  There is **only 1 RRC connection** between the **NR node** and the **UE**. This connection is **conveyed** through **PDSCH** and **PUSCH** of the **special cell**. The **secondary cells** do **not** convey **RRC signaling** (aside from possibly **system information**, click 5.1.2).

  This implies that the **semi-persistent scheduling** (click 4.11.4.3) **signaling** is conveyed through the **RRC connection** of the **special cell**, for **all** the **cells** in the **UE**'s **cell group**.

- **MAC PDU signaling** (click 4.5.5)

  **Each cell** in the **UE**'s **cell group** conveys its **MAC PDU signaling** between the **2 MAC peer entities** (**1** at the **NR node** and **1** at the **UE**) in the **cell**. The **PDSCH** and the **PUSCH** of **each cell** in the **UE**'s **cell group** convey the corresponding **exchange** of **MAC PDUs**.

- **HARQ signaling** (click 4.5.5)

  By **default**, **each cell** in the **UE**'s **cell group** conveys its **HARQ signaling** between the **2 HARQ peer entities** (**1** at the **NR node** and **1** at the **UE**) in the **cell**. The **PDCCH** and the **PUCCH/PUSCH** of **each cell** in the **UE**'s **cell group** convey the corresponding **exchange** of **HARQ signaling**.

  If **explicitly configured**, the **HARQ signaling** for **all** the **cells** in the **UE**'s **cell group** can be conveyed through the **special cell**. The **PDCCH** and the **PUCCH/PUSCH** of the **special cell** convey the **HARQ signaling** for **all** the **cells** in the **UE**'s **cell group**. This might require specific **timing** configuration, e.g., with the **7** parameters "`dl-DataToUL-ACK`…" from `PUCCH-Config` (click 7.9.4).

- **Dynamic scheduling signaling** (click 4.11.4.3)

  By **default**, **each cell** in the **UE**'s **cell group** conveys its **dynamic scheduling signaling** from the **NR node** to the **UE**. The **PDCCH** of **each cell** in the **UE**'s **cell group** conveys the **DCI messages** (click 6.9) for the **UE**'s **scheduling** in the **cell**.

  If **explicitly configured**, the **PDSCH** and **PUSCH** dynamic **scheduling signaling** of a **given cell** in the **UE**'s **cell group** may be conveyed through the **PDCCH** of **another cell**. This is called "**cross-carrier scheduling**" (click 10.2.3).

- **C-RNTI** (click 6.9.10.2)

  The **UE only** has **1 C-RNTI** that was **assigned** through the **special cell**. This **single C-RNTI** is used in **all** the **cells** in the **UE**'s **cell group**.

At **RRC** connection establishment/re-establishment/handover, the **special cell** conveys the **NAS** mobility information, and at **RRC connection re-establishment/handover**, **special cell** conveys the **security input**. **This cell** is referred to as the **special cell** (**SpCell**).

### 10.2.3 Cross-carrier scheduling

**Cross-carrier scheduling** can be **individually** configured (or **not**) for **each cell** in the **UE**'s **cell group**. When **cross-carrier scheduling** is configured in a **cell**, it can **either** indicate that the **cell** is **scheduled** by **another cell**, or that the **cell schedules another cell**.

**Cross-carrier scheduling** is configured with `crossCarrierSchedulingConfig` from `ServingCellConfig` (click 3.9.7). Its **data type** is `CrossCarrierSchedulingConfig`. Parameter `schedulingCellinfo` from `CrossCarrierSchedulingConfig` is a `CHOICE` between `own` and `other`, with the following semantics:

- When `own` is **configured**

  **This cell** is **cross-carrier scheduled** by **another serving cell**.

- When `other` is **configured**

  **This cell cross-carrier schedules 1**, or **several**, **serving cell**(s). Parameter `other` contains the **list** of **cell index**(**es**) (**each** being an **instance** of `ServCellIndex`, click 3.9.9) that identify the **cells** that are **cross-carried** by this **cell**.

  If `other` is **configured** for a **SpCell** (i.e., the **SpCell** is **cross-carrier scheduled** by a **secondary cell**), the **SpCell** can be **additionally scheduled** by the **PDCCH** on the **SpCell**.

When **cross-carrier scheduling** is configured, **each** received **DCI message schedules 1 carrier** among **several ones**. It is then required to **identify** the **carrier** to which the **scheduling DCI message** should be **applied**. This is done with the **DCI message** (click 6.9) **field** "***Carrier indicator***", whose **bitlength ranges** from **0** to **3**. **Bitlength 0** corresponds to the case of **no carrier aggregation**. The following table shows, for **PDSCH** and **PUSCH assignments**, the **DCI message Formats** where the said **field** is **present**, what is **assigned** by **each** of these **DCI Formats** and what is the **field**'s **bitlength** in **each** of these **DCI Formats**:

| **DCI message field "*Carrier indicator*"** | | |
|---|---|---|
| **Assigns** | **DCI Format** | **Bitlength** |
| **PUSCH Unicast** | Format 0_1 (6.9.3) | 0, 3 |
| | Format 0_2 (6.9.4) | 0, 3 |
| **PDSCH Unicast** | Format 1_1 (6.9.6) | 0 to 3 |
| | Format 1_2 (6.9.7) | 0 to 3 |

The "***Carrier indicator***" field contains the **index** (within the **UE**'s `SEQUENCE OF` **cells** of the corresponding **cell group**) of the **cell** to which the **enclosing DCI message applies**.

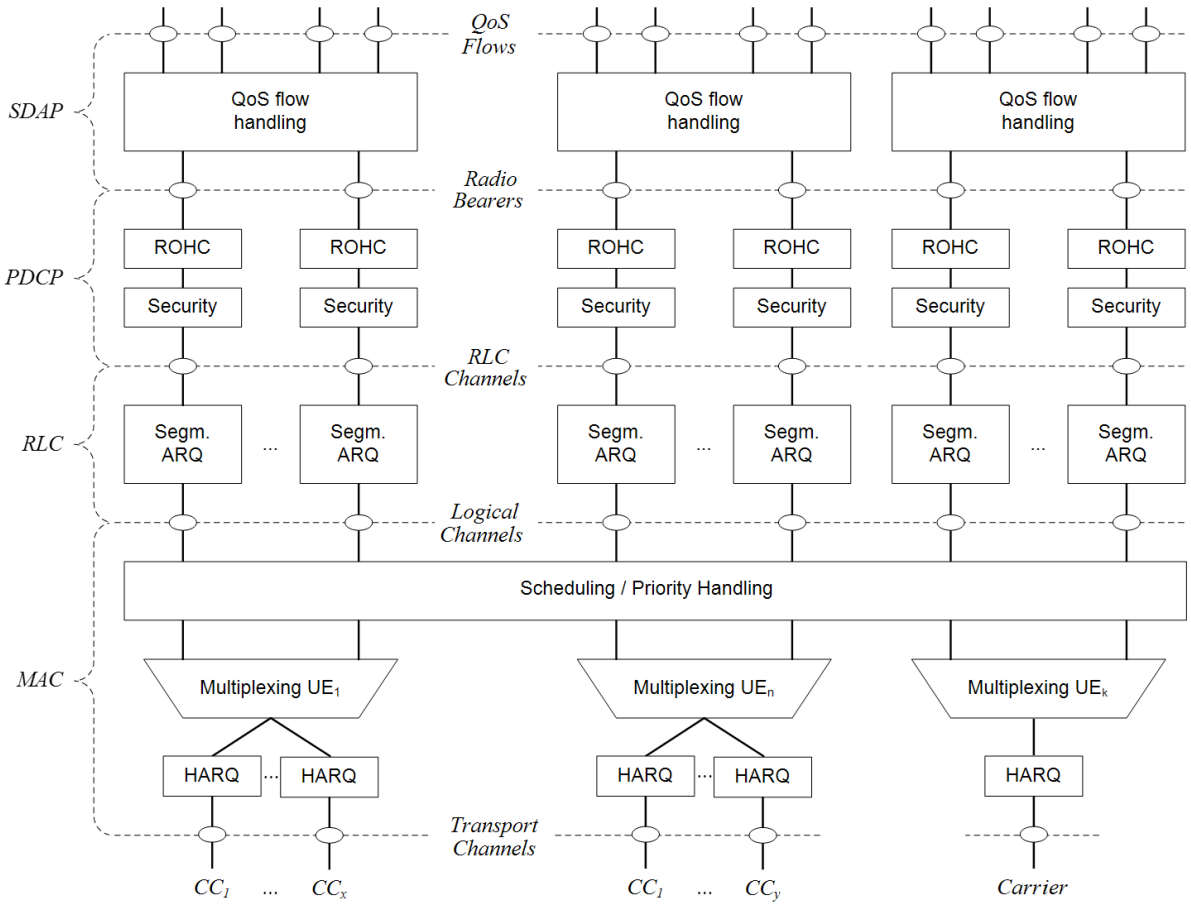
### 10.2.4 Carrier aggregation architecture

**Carrier aggregation** is performed **within** the **Mux/Demux function** of the **MAC sublayer**. For **each cell group**, there is **only 1 pair** of **Mux/Demux functions**: **1** at the **NR node** and **1** at the **UE**. **Each** of these **Mux/Demux entity** is installed over **all** the **HARQ protocol entities** (click 4.5.6.1) of the corresponding **cell group**. **Each carrier** has **its** own **HARQ protocol entity** and **its** own **physical layer**. The **HARQ protocol entity** and the **physical layer** of **each carrier** operate **almost independently** of the others, **almost** as when there is **no carrier aggregation**.

In **transmission**, the **Mux/Demux entity numbers each MAC PDU** (click 4.5.5.1) within the **ones** that **belong** to **each** of the **RLC protocol entities** (click 4.5.3). The **Mux/Demux function** then hands **each numbered MAC PDU**, as a **transport block** (click 4.12.1.1) to **1** of the **HARQ protocol entities** of the **corresponding cell group**. Consequently, **all** the **retransmissions** of **each** and **every transport block** remain in the **same carrier** of its **first transmission**.

In **reception**, the **Mux/Demux entity re-sequences** the **MAC PDUs** that correspond to **each RLC protocol entity** and delivers them (in sequence) to the corresponding **RLC protocol entity**.

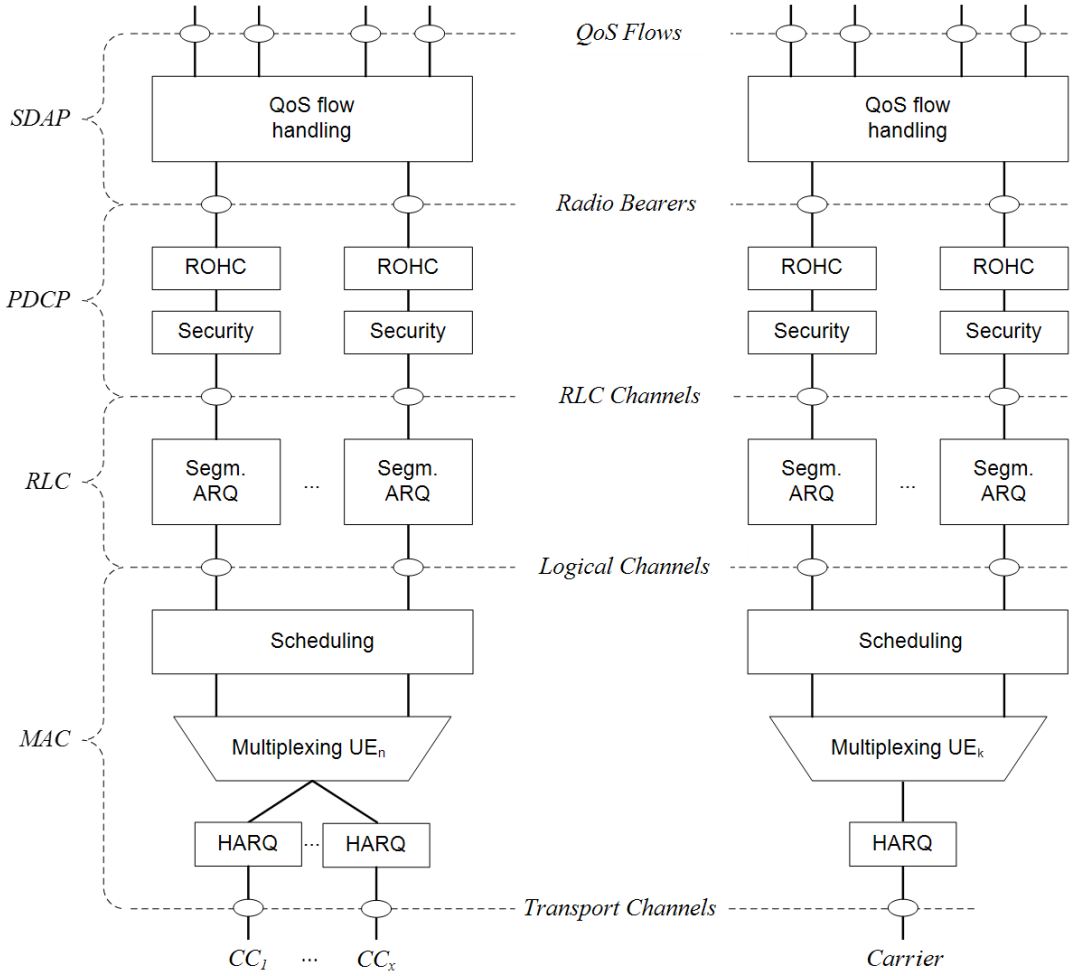
The following figure shows the **Downlink Tx architecture** at the **NR node**.




Some **relevant remarks** on this figure are:

- **Each** of the **2 leftmost protocol architectures** is the **peer** of **1 UE** with **carrier aggregation**, as may be seen because of the **multiple component carriers** (**$CC_i$**) that **each** has at its **bottom**. These **2 protocol architectures** are the **peers** of **$UE_1$** and of **$UE_n$**.
- The **rightmost protocol architecture** is the **peer** of **1 UE without carrier aggregation**, as may be seen because of the **single carrier** that it has at its **bottom**. This **protocol architecture** is the **peer** of **$UE_k$**.
- There is a **single scheduling/priority handling entity** for the **whole NR node**.

The following figure shows the **Uplink Tx architecture** at the **UE**:




Some **relevant remarks** on this figure are:

- The **protocol architecture** on the **left** corresponds to **$UE_n$**, that operates **with carrier aggregation**, as may be seen because of the **multiple component carriers** (**$CC_i$**) at its **bottom**.
- The **protocol architecture** on the **right** corresponds to **$UE_n$**, that operates **without carrier aggregation**, as may be seen because of the **single carrier** at its **bottom**.
- There is a **single scheduling entity** for **each UE**.

The **UE technological capacity** (click 11.1.2) of **carrier aggregation** is sometimes expressed in terms of the **maximum added bandwidth** of **all** the **aggregated radio channels** (click 2.2.3). This may create the **wrong** impression that **carrier aggregation** works by sending a **single carrier signal** that is **analogically split** among the different **aggregated radio channels**. As it has been presented in this subsection, this is **not** the way **carrier aggregation** works.

## 10.2.5 Carrier aggregation objectives

**Carrier aggregation** is sometimes presented as if its **only objective** is to **aggregate maximum-throughput carriers** (i.e., over **maximum bandwidth channels**) in order to achieve a **much higher throughput than** what can be obtained with **only 1 carrier**. The said **objective** is **correct**, but it is **not** the **only one**.

However, **carrier aggregation** is **also** used **very frequently** to **aggregate low-throughput carriers** (i.e., over **low bandwidth channels**) to **add-up** their **throughput** for the benefit of the user. The cause of an operator having **several low bandwidth channels** instead of a **single high-bandwidth channe**l (that would **not** require **carrier aggregation**) is the frequency allocation process (click 2.2.2.1) and channel trading.

**Spectrum reallocation** to **concentrate** the **operator**'s **licensed bandwidth** into a **small quantity** of **high-bandwidth channels** would **ideally** be a better solution than **carrier aggregation**. However, **spectrum reallocation** are **very complex** under the **administrative**, **legal** and **business** dimension. Consequently, **carrier aggregation** remains an **excellent solution** for **fragmented spectrum**.

### 10.2.6 Carrier aggregation and time synchronization

**Carrier aggregation** can only be done between **slots** that are **synchronized**, that is, the **start** and **end** of the **slots** being aggregated **must** be exactly the **same**. In **R15** it was also required that the **subcarrier spacing** of the **slots** being aggregated be exactly the **same.** However, **since R16** it is allowed to have different **subcarrier spacings**, by supporting **cross-carrier scheduling** (click 10.2.3) with **different subcarrier spacings** on the **scheduling** and **scheduled** carriers.

When **carrier aggregation** is deployed in the **RAN**, it is required that:

- **Either frame timing** and **system frame number** are **aligned** across **cells** that can be **aggregated**,
- **Or**, **every CA UE** is configured with an **offset** (in **multiples** of **1 slot**) between its **PCell/PSCell** and **each** and **every** of its **SCells** where **carrier aggregation** is performed.

A **UE** with **single timing advance capability** for **carrier aggregation** can **simultaneously receive** on **multiple Downlink component carriers** and/or **simultaneously transmit** on **multiple Uplink component carriers**, provided that **all** the **serving cells** that host the **component carriers** have the **same timing advance** (i.e., **all** the said **serving cells** belong to the **same timing advance group** (**TAG**)).

A **UE** with **multiple timing advance capability** for **carrier aggregation** can **simultaneously receive** on **multiple Downlink component carriers** and/or **simultaneously transmit** on **multiple Uplink component carriers**, even if **some** or **all** the **serving cells** that host the **component carriers** have **different timing advances** (i.e., **some** or **all** of the said **serving cells** belong to **different timing advance groups** (**TAGs**)). The **NG-RAN** ensures that **each TAG** contains **at least 1 serving cell**.

## 10.3 Dual connectivity (DC)

**Dual connectivity** consists of the **UE** being **simultaneously connected** (i.e., the **UE** is in **RRC_CONNECTED**, click 5.8.2) to **2 different RAN nodes**, and **aggregating** the **user data flows from/to both nodes**.

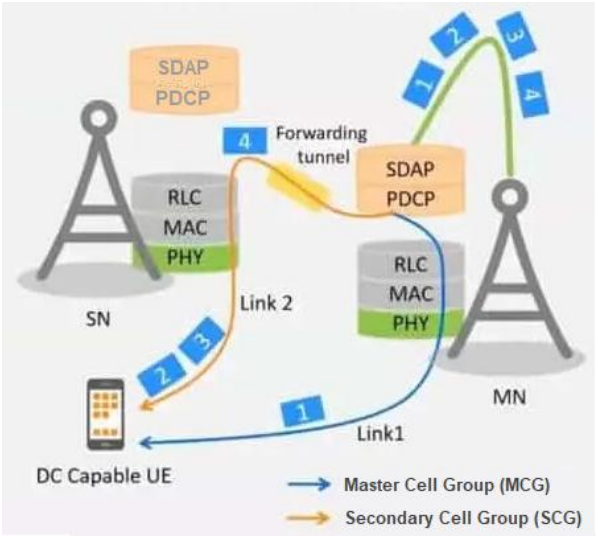
The **UE** is configured to utilize **radio resources** provided by **2 different schedulers**, located in **2 different RAN nodes**. **One** of the **RAN nodes** is **NR node** and the **other** provides **either 5G-NR** access or **LTE** (**E-UTRA**) access.

The **aggregation/disaggregation** of the data flows from both **nodes** is done at the **Packet Data Convergence Protocol** (**PDCP**, click 4.5.2), which is above the **Media Access Control** (**MAC**) **sublayer** and above the **Radio Link Control** (**RLC**) layer. Therefore,

**each** of the **NR nodes** will handle the **channels**, the **MAC** and the **RLC** in a **totally independent way**, and **aggregation/disaggregation** of the data flows will be done at the **PDCP** layer.

The **UE** may initiate **2 redundant PDU Sessions** over the **5G network**. The **5G system** (i.e., **RAN** plus **Core**) sets up the **user plane paths** of the **2 redundant PDU sessions** to be **disjoint**. When **PDU session setup** or **modification** is **initiated**, the **RAN** can configure **dual connectivity** in **1 NR node** or **2 NR nodes** for the **2 redundant PDU sessions** to **ensure** the **disjoint user plane paths** according to the **redundancy information** received from the **5G Core**. The **RAN** shall ensure that the **resources** of the **data radio bearers** for the **two redundant PDU sessions** are **isolated**. If the **RAN cannot** satisfy the **disjoint user plane requirement**, the **redundant PDU sessions** may be **kept** or **not kept** according to the **RAN** local **configuration**. The **redundancy information** is **transferred** to **the target NR node** in case of **handover**.

The following figure shows the involved protocols in **dual connectivity**:




**Dual connectivity** is **only allowed** on **some bands**, on **some band combinations**, and with **some carrier aggregation configurations**. For the allowed **configurations** (bands, channels, power, …) of **dual connectivity**, refer to clauses 5.2B, 5.5B, 6.2B, 6.3B, 6.4B, 6.5B, 7.3B, 7.4B, 7.5B, 7.6B, 7.7B, 7.8B and 7.9B of TS 38.101**-1**, **-2** and **-5**.

The design of **5G** and of the latest versions of **LTE** allows for **dual connectivity**, but **each UE model** will have its **specific restrictions** (click 4.9.1).

Conceptually, **dual connectivity** is **not** completely an **air interface** (**AI**) element, because it requires the involvement of at **higher-layer protocols** and has a significant impact on **signaling** and control.

For more detail on **multi-radio dual connectivity**, refer to:

- Clauses 4.5, 6.8 and 16.1.6.1 from TS 38.300.
- Clauses 5.1.3, 5.3.5.10, 5.7.1a, 5.7.2a and 5.7.3.3 of TS 38.331.
- For detailed information on **Conditional PSCell Addition** (**CPA**), **Conditional PSCell Change** (**CPC**) or the usage of **2 MAC entities**, refer to TS 37.340 V17.7.0.

Source web pages:

- https://www.gizchina.com/2022/10/23/5g-dual-connectivity-what-is-it-all-about/
- https://www.5gworldpro.com/blog/2021/11/07/what-is-mcg-scg-pcell-scell-pscell-in-5g-nsa/
- https://www.blacktechnoguys.com/2019/06/type-of-dual-connectivity.html

### 10.3.1 Types of RAN nodes in dual connectivity

In **dual connectivity**, there are **2** different types of **nodes**:

- **Master node**

  It manages the **control** and **user plane connections** to the **Core** and it serves as **anchor point** for the **UE**.

- **Secondary node**

  It contributes **additional RAN transmission resources** to the **UE**. It does **not** manage the **control** and **user plane connections** to the **Core** and it does **not** serve as **anchor point** for the **UE**.

Notice that **both nodes** may be part of the **same physical equipment**. In case they are part of **two** different physical **equipment**, they can be placed at the **same physical location** or at **different physical locations**.

### 10.3.2 Types of dual connectivity

There are **two** main **types** of **5G dual connectivity**, depending on whether **one** or **two radios** are required:

- **New Radio dual connectivity** (**NR-DC**)

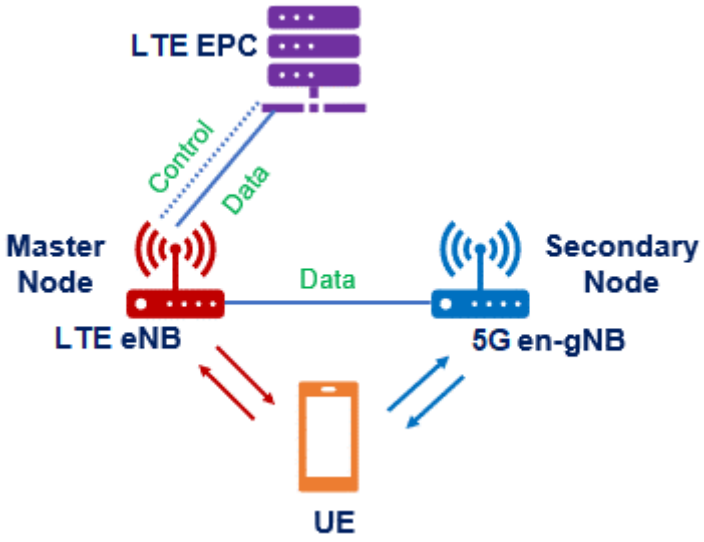
  It is when the **UE** is connected to **two NR nodes**. In this case the **UE** only requires **one radio**.

  The **master NR node** is connected to the **5G core** (**5GC**) via the **NG interface** and it is connected with the **secondary NR node** via the **Xn** interface. The **secondary NR node** may also connected to the **5GC** via the **NG-U** interface **New radio dual connectivity** (**NR-DC**) is sometimes called "**next generation dual connectivity** (**NG-DC**)".

- **Multi-radio dual connectivity** (**MR-DC**)

  It is when the **UE** is connected to **one 5G NR node** and to **one LTE eNB** (or some of their respective **variants**). In this case the **UE** requires **two different radios**: a **NR radio** and an **LTE radio**. In this case, the **2 NR nodes** are connected via a **non-ideal backhaul**.

The following figure shows an example of **multi-radio dual connectivity** with its different involved subsystems:




Notice that this example is for a **5G deployment Non-Standalone (NSA) option 3** (see below).

Within **multi-radio dual connectivity** (**MR-DC**), there are **three variants**, related to the **5G deployment options**:

- **E-UTRA and NR dual connectivity** (**EN-DC**)

  It is specific of **NSA deployment option 3** and it works with an **LTE E-UTRA connected** to an **LTE evolved packet core** (**EPC**). The **UE** is connected to one **eNB** (**master node**) and to one **en-gNB** (**secondary node**). The **eNB** (**master node**) is connected to the **EPC** via the **S1** interface and is connected to the **en-gNB** (**secondary node**) via the **X2** interface. The **en-gNB** may also be connected to the **EPC** via the **S1-U** interface and to other **en-gNBs** via the **X2-U** interface.

- **NG-RAN E-UTRA and NR dual connectivity** (**(NG)EN-DC**)

  It is specific of **NSA deployment option 7** and it works with a **5GC**. The **UE** is connected to one **ng-eNB** (**master node**) and to one **gNB** (**secondary node**). The **ng-eNB** (**master node**) is connected to the **5GC** and is connected to the **en-gNB** (**secondary node**) via the **Xn** interface.

- **NR and E-UTRA dual connectivity** (**NE-DC**)

  It is specific of **NSA deployment option 4** and it works with a **5GC**. The **UE** is connected to one **gNB** (**master node**) and to one **ng-eNB** (**secondary node**). The **gNB** (**master node**) is connected to the **5GC** and is connected to the **ng-eNB** (**secondary node**) via the **Xn** interface.

### 10.3.3 Types of cell groups in dual connectivity

In **dual connectivity** there are **two types** of **cell groups**:

- **Master Cell Group** (**MCG**)

  It consists of the **serving cells associated** with the **master node**. It always includes the **primary cell** (**PCell**). The **primary cell** is the one where the **UE**

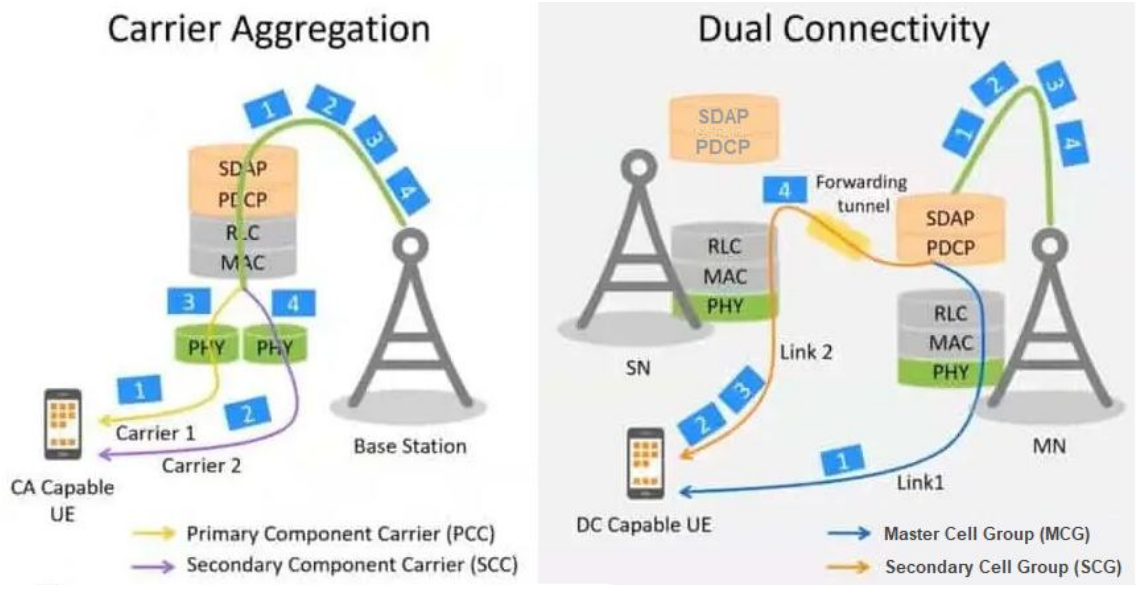
performed its **initial access** (click 8). In case **carrier aggregation** is performed, there will **also** be one or more **secondary cells** (**SCells**).

- **Secondary Cell Group** (**SCG**)

  It consists of the **serving cells associated** with the **secondary node**. It always includes the **primary-secondary cell** (**PSCell**). In case **carrier aggregation** is performed, there will **also** be one or more **secondary cells** (**SCells**).

### 10.3.4 Dual connectivity and carrier aggregation

As already introduced in the previous subsection, **dual connectivity** and **carrier aggregation** (click 10.2) are **complementary** technologies, and, in most cases, they are **jointly used**. For the different **types** of **cells** arising from the **joint** usage of **dual connectivity** and **carrier aggregation**, click 3.2.2. Regarding their technical characteristics, **carrier aggregation** is more efficient and involves a greater degree of integration than **dual connectivity**. The following figure compares some relevant characteristics of **dual connectivity** and **carrier aggregation**:




The **UE** has **1 MAC entity** for **each** of its **2 cell groups** (click 3.2.2), which is the same as having **1 MAC entity** for **each** of its **2 NR nodes**. **Each** of the **2 MAC entities** performs **carrier aggregation** within the corresponding **cell group**. The **aggregation** of **flows** from **both RAN nodes** is performed at a **higher layer**.

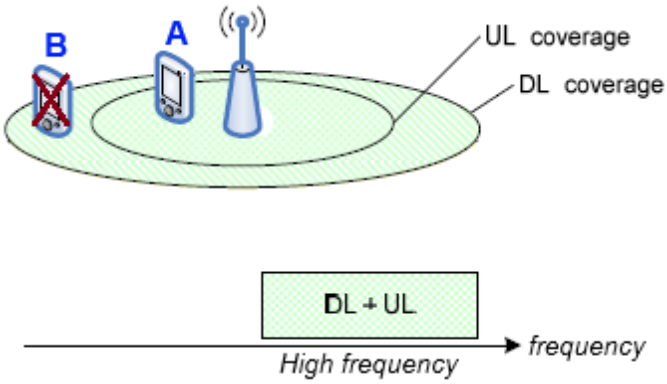
## 10.4 Supplementary Uplink (SUL)

**SUL overview**

**Supplementary Uplink extends** the **coverage** of a **cell** that operates in **high frequency channel(s)** (e.g., **FR1-2** or higher) by **adding** to the **cell** an **alternative low frequency** (**FR1-1**) **Uplink channel** (the **SUL channel**).

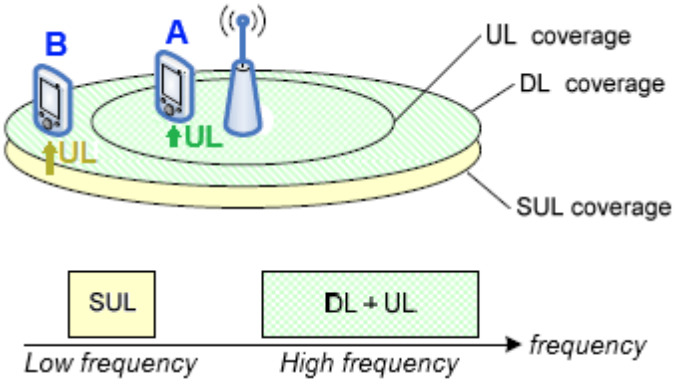
The rationale is as follows. In **higher frequencies**, the **Uplink cell coverage** is usually **smaller** than the **Downlink coverage** because the **UE transmit** capabilities (power, antennas, …) are **lower** than the ones of the **NR node**. Since a **UE** requires **both Downlink** and **Uplink** communication, the **cell area** is **limited** to the **smaller** of both, which is the **Uplink area**.

The following figure shows the problem. In this cell, the **UE** "**A**" **can** communicate, because it has **both Downlink** and **Uplink coverage**. However, the **UE** "**B**" **cannot** communicate, because it has **Downlink coverage**, but **not Uplink coverage**.




This can be solved by **adding** to the **cell** an **alternative lower frequency Uplink channel** that has a **coverage** area **similar** to the **Downlink area**. This allows the **UEs within** the **Downlink coverage**, but **outside** the **Uplink coverage**, to **communicate** using the **SUL carrier(s) instead of** the **Uplink carrier(s)**. **SUL** can be used in a **cell** with **1 pair** of **FDD channels** and also in a **cell** with **1 main TDD channel**. Regardless of the case, the **SUL channel** is **only** used to **transmit Uplink**, with its **specific Uplink carriers** and its **specific Uplink bandwidth parts**.

The following figure shows the **SUL** solution. **Both UEs can** now communicate. **Both UEs** share the **Downlink channel**, **UE** "**A**" **only** uses the **Uplink channel**, and **UE** "**B**" **only** uses the **SUL channel**.




The **SUL** bands are **n80**, **n81**, **n82**, **n83**, **n84**, **n86**, **n89**, **n95**, **n97**, **n98** and **n99**, **all** of them within **FR1-1** (click 2.2.13.1).

One limitation of **SUL** is that **most SUL bands collide** with **LTE bands**. This can trigger **SUL-LTE coexistence problems** (refer to TR 37.872 and TR 37.716).

**SUL Procedures**

**Each UE** can use **either** the **Uplink carrier(s) or** the **Uplink carrier(s)** but **not both**.

**During initial access** (click 8), the **UE** selects the **SUL carrier** if and only if the measured quality of the **Downlink** is **lower** than a broadcast **threshold**. Once started, **all** the **UE**'s **transmissions** (**PRACH** and others) of the **random access procedure** (click 9) remain on the **selected carrier**.

In the specific case that a **UE** is configured with **two Uplink** carriers for a **cell** and the **UE** detects a **PDCCH Order**, the **UE** uses the **Uplink/SUL** indicator field value from

the detected **PDCCH Order** to determine the **Uplink carrier** for the corresponding **PRACH transmission**.

**After initial access**, the **NR node controls** the usage by **each UE** of **either** the **Uplink** or the **SUL carrier(s)**. The **NR node** indicates **each UE** which **PUSCH** should use (**either** the **Uplink** or the **SUL**) using the conventional **Uplink scheduling** (click 4.11.4.1) with **DCI 0_1 messages** (click 6.9.3). The **NR node** indicates **each UE** which **PUCCH** to use (**either** the **Uplink** or the **SUL**) through **RRC configuration**.

For **supplementary Uplink** parameters, see the **carrier parameters** (click 2.2.6) and the **bandwidth part parameters** (click 2.2.8).

For the allowed **configurations** (bands, channels, power, …) of **supplementary Uplink**, refer to clauses 5.2C, 5.5C, 6.2C, 6.3C, 6.4C, 7.3C and 7.6C of TS 38.101-**1**, -**2** and -**5**.

**SUL DCI signaling**

The **3 DCI Format 0_x messages** (click 6.9) **may** contain a **1-bit** "***UL/SUL indicator***" field that indicates if the **PUSCH assignment** corresponds to "**UL**" (field=**0**) or to "**SUL**" (field=**1**).

This field exists for **UEs** configured with `supplementaryUplink` in `ServingCellConfig` (click 3.9.7) and the **bitlength** of **DCI Format 1_0 before padding** is **larger** than the **bitlength** of **DCI Format 0_0 before padding**. The **semantics** of this field is:

- ***If*** [[the "***UL/SUL indicator***" is **present** in **DCI Format 0_0**] and [`pusch-Config` from `BWP-UplinkDedicated` (click 2.2.8.10) is **absent** on **both** the **UL** and the **SUL** configuration]], ***then***, the **UE ignores** the "***UL/SUL indicator***" field in **DCI Format 0_0**, and the corresponding **PUSCH** scheduled by the **DCI Format 0_0** is for the **UL** or **SUL** for which `pucch-Config` is **present**.
- ***If*** [[the "***UL/SUL indicator***" is **absent** in **DCI Format 0_0**] and [`pucch-Config` is **present**], ***then***, the corresponding **PUSCH** scheduled by the **DCI Format 0_0** is for the **UL** or **SUL** for which `pucch-Config` is **present**.
- **If** [[the "***UL/SUL indicator***" is **absent** in **DCI Format 0_0**] and [`pucch-Config` is **absent**]], ***then***, the corresponding **PUSCH** scheduled by the **DCI Format 0_0** is for the **uplink** on which the **latest PRACH** is **transmitted**.

For more detail on **supplementary Uplink**, refer to:

- Clauses 5.4.2, 6.9, 9.2.6 and 10.3 of TS 38.300.
- Clause 8 of TS 38.213.
- Clause 5.1.1 of TS 38.321.
- Clause 7.3.1.1.1 of TS 38.214.

Source web pages:

- https://www.sharetechnote.com/html/5G/5G_SUL.html

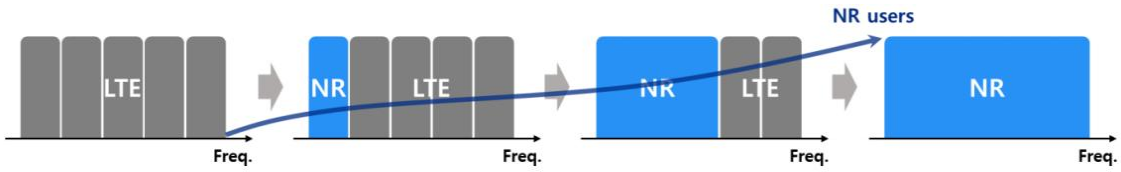
## 10.5 Supplementary Downlink (SDL)

Although **3GPP** has defined **Supplementary Downlink** bands, there is **almost no reference** to **SDL** in **R17 RRC configuration parameters** (click 5), nor in most **R17 TS documents**.

## 10.6 5G/LTE static frequency-domain sharing

**Static frequency-domain sharing** consists of providing **5G** and **LTE** service in **different channels** of the **same FDD band**.

For example, one operator has a licensed **frequency block pair** of **25+25 MHz** in band **n71**. The operator **initially** configured the **frequency block pair** with a **25+25 MHz LTE channel pair**. When the operator wanted to start providing **5G** service, the operator reconfigured the **frequency block pair** with one **20+20 MHz channel pair** for **LTE** and one **5+5 MHz channel pair** for **5G**. As **5G** traffic increases, the operator may change the channel configuration, **successively** increasing the bandwidth of the **5G channel pair**, while decreasing the one of the **LTE channel pair**. In this example, the evolution of the **LTE/5G** channel bandwidths could be: **25/0**, **20/5**, **15/10**, **10/15**, **5/20** and **0/25 MHz**. The following figure illustrates some steps of the configuration evolution of one of the frequency blocks in the pair:



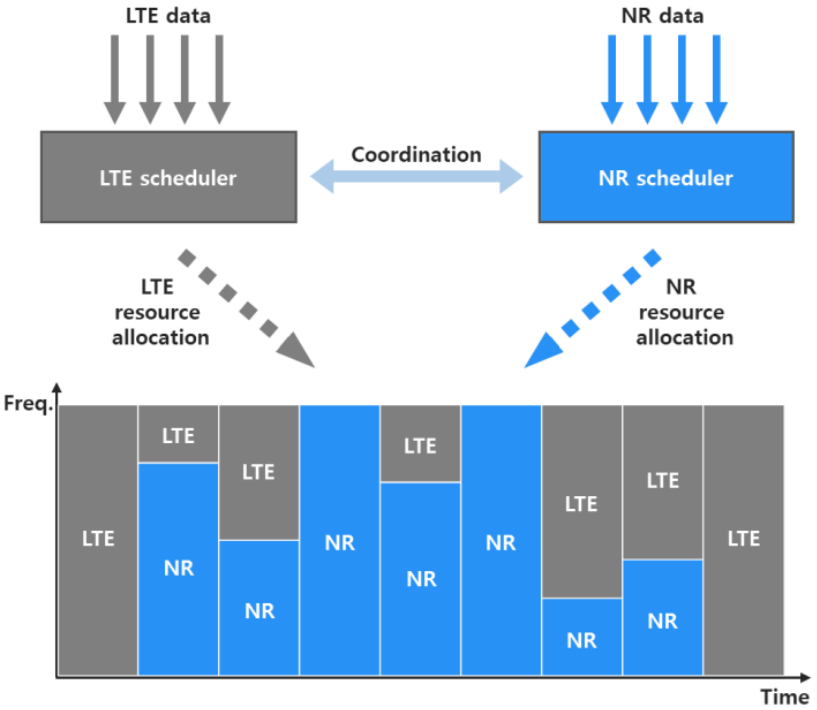

**Static frequency-domain sharing** cannot adapt to the **actual** traffic load split between **LTE** and **5G** in the area, and **each** of its **successive 5G/LTE** channel reconfigurations is non-trivial.

For these reasons, a much better technique to provide both **5G** and **LTE** service over the same **frequency block** is "**Dynamic Spectrum Sharing**" (**DSS**, click 10.7). **DSS** allows **5G** and **LTE** to **share** the **same channel**, changing dynamically the transmission resources allocated to each of them, depending on the **5G/LTE** traffic in the cell. This is done in **FDD** bands, by sending in the **same channel**, **alternating** at **different time intervals**, the signals of a **5G carrier and** of an **LTE carrier**.

## 10.7 5G/LTE dynamic spectrum sharing (DSS)

**Dynamic spectrum sharing** (**DSS**) consists of allowing **5G** and **LTE** to **share the same carrier**. To achieve this, the **5G** and **LTE schedulers coordinate**, exchanging **traffic status** and/or **resource sharing status**, as well as **dynamically assigning** available

**resources** in a **synchronized manner**. The following figure illustrates the way **DSS** works:




As you may see, the way to allocate resources between **5G** and **LTE** is by assigning each of them a given **quantity** of **subcarrier groups** in **each subframe**. It is therefore required that both **5G** and **LTE** use the **same subcarrier spacing**, this is, **15 kHz**.

**DSS** provides a very convenient **migration** path from **LTE** to **5G** by allowing the **gNB** to **dynamically** allocate the resources between **LTE** and **5G**, depending on the characteristics and **traffic demands** of its active **UEs**. This is a great advantage over **static frequency-domain sharing** (click 10.6). However, **DSS** results in overheads that **reduce** the total **throughput** of the carrier, in respect to **static frequency-domain sharing**.

Since **DSS** requires that **5G** uses a **subcarrier spacing** of **15 kHz** (to be the same as in **LTE**), it can only be used in **FR1** bands. Moreover, it can only be used in **band-channel** combinations compatible with a **15 kHz** subcarrier spacing (click 2.2.3.2).

**DSS** was defined in **5G/LTE-R15** and further enhanced in **5G/LTE-R16** and **5G/LTE-R17**. As of December 2022, **DSS** was only defined for some **FDD** bands, and not for **TDD** bands.

In order to design **dynamic spectrum sharing**, it has been required to **introduce** in **5G** an **optional 7.5 kHz shift** of its **Uplink subcarrier grid** (click 2.2.4.8). This **7.5 kHz shift** is **required** for **Uplink transmission compatibility** between **LTE** and **5G**.

As a final remark, **DSS** is sometimes called "**LTE-NR spectrum sharing**".

Source web page:

- https://howltestuffworks.blogspot.com/2020/11/5g-nr-dss-dynamic-spectrum-sharing.html

## 10.8 Multicast-broadcast services (MBS)

**Multicast-broadcast services** (**MBS**) provide **both multicast services** (click 10.8) and **broadcast services** (click 10.8) to **5G UEs**.

The **MBS design** is quite **generic**, mostly based on existing **unicast functionalities**. This allows to **provide MBS** in **most pre-existing UEs**, with **only** a **software update**. And the **same** applies to the **RAN** and **Core**.

**MBS** were introduced in **R17**.

**Relevant characteristics of 5G multicast services**

- A **multicast session** can **only** be **received** by the **UE** if it uses its **Uplink**, which **requires** that it is in the **RRC_CONNECTED** (click 5.8.2) state, as it happens with **unicast services**.
- The **network knows each** and **every UE** that is **receiving** the **multicast session**. This allows the **network** to **optimize** the **service mode** and the **used resources** to **each specific situation**.
- **Multicast services** support **most service guarantees** provided by **unicast services**, including **reliability**, **coverage**, **latency**, and **mobility**.
- The **UE must request** to **join** a **multicast session** and be **admitted** to it. A **multicast session** therefore has a **known set** of **UEs**, with **admission control** and **security** being controlled by the network.
- In **multicast services** the **network layer** can **support seamless mobility**.

**Relevant characteristics of 5G broadcast services**

- The **UE** may **receive broadcast services without** it having to use its **Uplink**: this is **always possible**, as long as the **UE** is within **coverage**.
- The **network** does **not know how many UEs** are **receiving** the **broadcast services**, which can **sometimes** be **wasteful** in terms of resources.
- The **network** does **not know** which **UEs** are **receiving** the **broadcast services**, and this **prevents** the **network** from **adapting** the transmission to the **receiving UEs**. The **broadcast transmission** is therefore **blindly configured**, **except** for some possible **adaptation** performed at the **application layer**.
- **Broadcast services** do **not** support **many service guarantees** provided by **unicast services**.
- The **reliability** and **efficiency** in **broadcast services**, are **lower** than the ones of **multicast services**. The reason is that in **broadcast services** the **UEs** do **not** use the **Uplink**, and consequently, **cannot** send **communication feedback** to the **NR node**.
- The **UE** does **not** issue any **request** to **receive** the **broadcast services**: it just **listens** to the **broadcast transmission**. This is why **admission control** and **security cannot** be provided.
- In **broadcast services**, the **network layer cannot support seamless mobility**, and it is **rather** the **application layer** that handles **mobility** between **broadcast areas**.

**Multicast and broadcast service sessions**

The **5G Core** may provide **IP multicast** to **connected UEs** over **MBS sessions**. The **MBS sessions** can be **established** and **released**, as required by **each** specific **service**. The **MBS sessions** are the **MBS equivalent** to the **protocol data unit** (**PDU**) **sessions** of **unicast services**.

As it happens with **PDU sessions**, **each MBS session** has some associated **Quality of Service** (**QoS**) **requirements**. In the **RAN**, **each MBS session** is conveyed to the corresponding **UEs** over **1** (or more) **Multicast Radio Bearers** (**MRBs**, click 4.5.2.1).

**MBS frequency resource**

A **UE** is **not** expected to **receive PDSCH** or **PDCCH** associated with **MBS transmissions** scheduled with **G-RNTI**, **G-CS-RNTI** or **MCCH-RNTI outside** the **common MBS frequency resource**.

A **common MBS frequency resource** is a **range** of **CRBs** within a **bandwidth part**. The **starting location** "$N_{MBS,i}^{start,\mu}$" of the **common MBS frequency resource** $i$ is defined relative to **point A** and its **size** is denoted by "$N_{MBS,i}^{size,\mu}$". The **RBs** in a **common MBS frequency resource** are **numbered** in the **same** way as the **bandwidth part resource blocks** with "$N_{BWP,i}^{start,\mu}$" and "$N_{BWP,i}^{size,\mu}$" replaced (respectively) by "$N_{MBS,i}^{start,\mu}$" and "$N_{MBS,i}^{size,\mu}$".

For more detail on **MBS frequency resource**, refer to clauses 4.4.4.4 and 4.4.6 of TS 38.211.

Source web pages:

- https://www.ericsson.com/en/blog/2022/12/multicast-broadcast-group-communication

## 10.9 Small data transmission (SDT)

The **small data transmission** (**SDT**) procedure allows a **very fast** (i.e., **very low latency**) **transmission** of **small** amount of **data** by a **UE**. **SDT** is implemented by **reducing** the **signaling overhead** and by using the **mini-slot scheduling** (click 4.11.4.6) procedure.

Instead of a **full** `RRCResume` (click 5.8.3.1) and a conventional **full-slot scheduling** (click 4.11.4.1), **SDT** allows the **UE** to transmit data after a **RRC lightweight connection resume** or even while in the **RRC_INACTIVE** (click 5.8.2) state.

**SDT reduces signaling overhead**, **saves power** and, in particular, it allows a **very low latency**. For these reasons, **SDT** is particularly useful for the **URLLC** and **mMTC** use cases (click 10.10).

## 10.10 Use cases

**5G** is designed to support **extremely diverse performance requirements** in order to comply with the **requirements** of a **broad range** of **professional customers** (the so-called "**verticals**"), in addition to the conventional **individual customers**.

To this end, **5G** has defined a few so-called **use cases**. **Each use case** represents a **certain configuration** of the **5G system** that delivers the specific performance characteristics that a certain profile of **professional user** demands. Three very relevant **5G use cases** are:

- **URLLC**: **ultra-reliable low latency communications**

  It provides **very low latency** (e.g., a few milliseconds) and **extremely high reliability** (e.g., **five nines**: 99.999% packet delivery probability). It is targeted to providing a **wireless connection** to extremely sensitive **production machinery** in **industrial environments**. This environment typically has **hard-real time** requirements.

  The **small data transmission** (**SDT**, click 10.9) service and the **mini-slot scheduling** (click 4.11.4.6) are valuable **5G** capacities for **URLLC**.

- **mMTC**: **massive machine-type communications**

  It provides **communication** with **very low energy consumption** with **low complexity** and **low throughput**, to an **extremely large quantity** of **simple UEs**. This is targeted to **massive internet** of **things** (**IoT communications**), connecting **hundreds** of **billions** of **sensors**, **actuators**, **meters**, **smart tags** and similar **devices** to the **5G network**.

  The **small data transmission** (**SDT**, click 10.9) service and the **mini-slot scheduling** (click 4.11.4.6) are valuable **5G** capacities for **mMTC**.

- **eMBB**: **enhanced mobile broadband**

  It provides a **mobile broadband service** with significant **enhancements** over the **legacy LTE service**. It targets both a **very high UE-in-cell capacity** (click 4.9.3.3) and a **very high cell area capacity** (click 3.10.4) to support **areas densely populated** by **UEs where each** of them demands a **high throughput**. Some examples of such scenarios are **airports**, **railway stations**, **business districts** and **sport stadiums**.

## 10.11 Handover

The **handover** procedure is performed when a **UE controlled** by a given **NR node passes** to be **controlled** by a **neighbor NR node.**

**Handover** is typically **signaled** between **NR nodes** using **inter-node RRC messages** (click 5.8.6) such as:

- `HandoverCommand`

  It is used to transfer the **handover command** as generated by the **target NR node**. For its **parameters**, click 3.7.2.

  **Direction**: **target gNB** to **source gNB/source RAN**.

- `HandoverPreparationInformation`

  It is used to **assist inter-RAT** or **intra-RAT handovers**, particularly when the **handover** involves **another network node** (such as an **NG-RAN** node, **eNB**, or **gNB**) that is **not** the **UE**'s **serving cell**. For its **parameters**, click 3.6.3.

**Handover** typically requires **UE reconfiguration**, that is **signaled** with the `RRCReconfiguration` (click 5.8.5.4) **RRC message**.

**Handover** typically requires the **UE** to run a **contention-free random access procedure** (click 9.1.2), in order to acquire **Uplink synchronization** and other **physical layer adjustments**.

## 10.12 Processing of OFDM, OFDMA and DFT-s-OFDM in 5G

In **5G**, the **Downlink frequency-time multiplexing** is done with "**Cyclic-Prefix Orthogonal Frequency Division Multiplexing**" (**CP-OFDM**, click 11.14), same as in **LTE**. For an overview of **OFDM** processing, click 10.12.1.

The **Uplink frequency-time multiple access** is done with "**Cyclic Prefix Discrete Fourier Transform spread Orthogonal Frequency Division Multiplexing**" (**CP-DFT-s-OFDM**, click 11.16), same as in **LTE**. For an overview of **DFT-s-OFDM** processing, click 10.12.3.

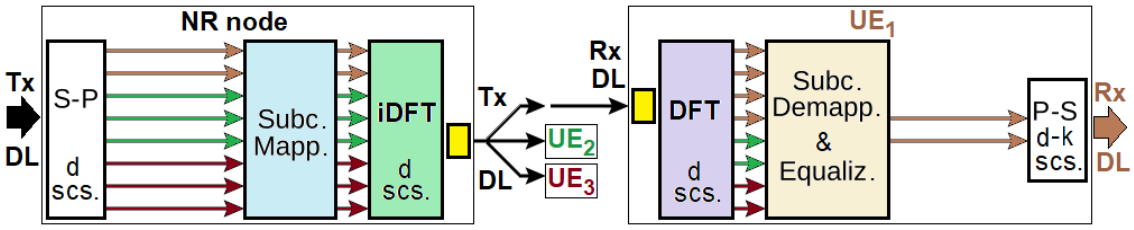
For **Uplink**, **5G allows** to **optionally disable** the **DFT precoding** of **CP-DFT-s-OFDM**, which results in "**Cyclic-Prefix Orthogonal Frequency-Division Multiple Access**" (**CP-OFDMA**, click 11.15). **LTE** does **not** allow for this option. For an overview of **OFDMA** processing, click 10.12.2.

Source document:

- https://support.ixiacom.com/sites/default/files/resources/whitepaper/sc-fdma-indd.pdf

### 10.12.1 OFDM processing

The **processing** of a **5G OFDM** (click 11.14) **signal** at the **transmitter** (**NR node**) and at the **receiver** (**UE1**) is shown in the following figure:

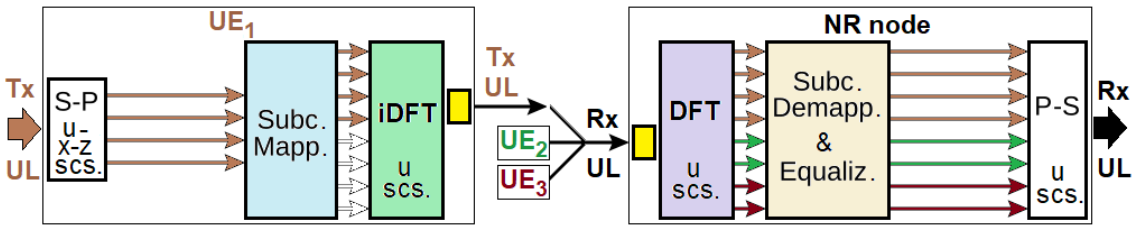



Some **relevant comments** on this figure are:

- **Every UE** receives the **Downlink OFDM signal** from the **NR node** that contains the **subcarriers** for **all** the destination **UEs**.
- **Each destination UE demultiplexes** the **Downlink OFDM signal** to extract its **corresponding subcarriers**.
- For the **labels** and **symbols** in the figure, click 10.12.5.

## 10.12.2 OFDMA processing

The **processing** of a **5G OFDMA** (click 11.15) **signal** at the **transmitter** (**UE$_1$**) and at the **receiver** (**NR node**) is shown in the following figure:

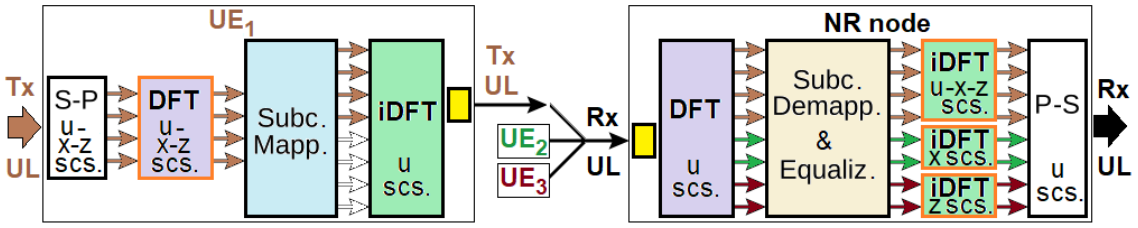



Some **relevant comments** on this figure are:

- The **OFDMA processing** is basically the **same** (**both** at the **transmitter** and at the **receiver**) as the **OFDM processing** (click 10.12.1).
- The main difference between **OFDM** and **OFDMA** processing is that in **OFDM every receiver** (every **UE**) receives the **Downlink OFDM signal** from the **transmitter** (the **NR node**) that contains the **subcarriers** for **all** the **receivers** (the **UEs**), while in **OFDMA each transmitter** (each **UE**) transmits its **specific OFDMA signal**, and **the receiver** (the **NR node**) receives the **superposition** of **all** the **transmitted OFDMA signals**.
- For the **labels** and **symbols** in the figure, click 10.12.5.

## 10.12.3 DFT-s-OFDM processing

The **processing** of a **5G DFT-s-OFDM** (click 11.16) **signal** at the **transmitter** (**UE$_1$**) and at the **receiver** (**NR node**) is shown in the following figure:

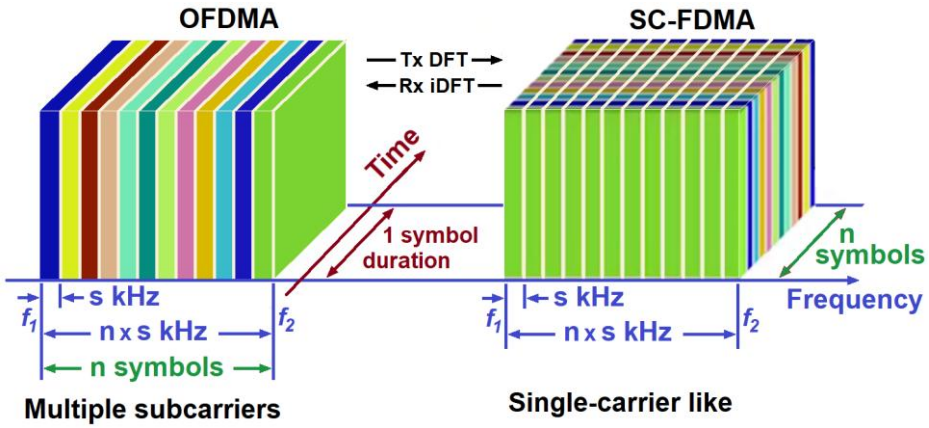



Some **relevant comments** on this figure are:

- The **processing** for **DFT-s-OFDM** is the **OFDMA processing** (click 10.12.2) with an **additional DFT precoder** at **each transmitter** (**each UE**) and **additional iDFT decoders** at the **receiver** (the **NR node**).
- The **additional Tx DFT precoder** and **Rx iDFT decoders** are **highlighted** in the figure with an **orange border**.
- For **each transmitter** (each **UE**) the **receiver** (the **NR node**) **requires 1 specific iDFT**.
- For the **labels** and **symbols** in the figure, click 10.12.5.

The effect of the **additional Tx DFT** precoding is to transform the **time domain signal** into the **frequency domain**. This may be seen as transforming **symbols** into **subcarriers**,

or in other words, as **spreading each symbol** into **all** the **subcarriers**. The following figure illustrates this:

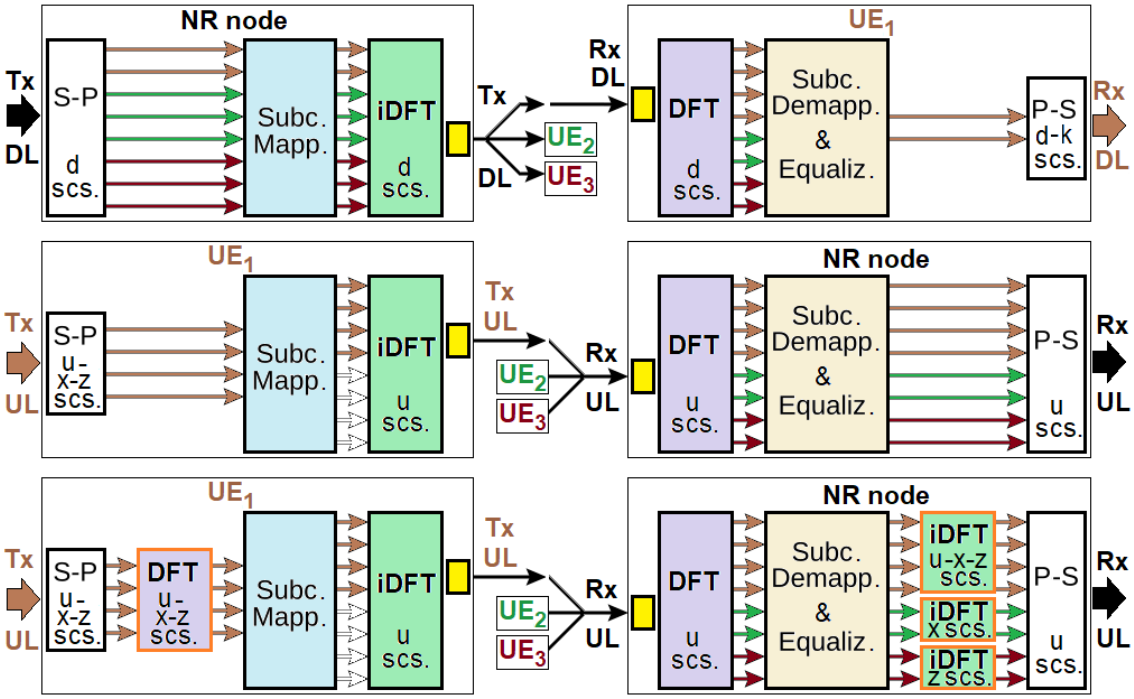



Some **relevant comments** on this figure are:

- **Each colored rectangle** on the **left** represents **1 symbol** within **1 subcarrier** over **1 symbol duration**, as corresponds to an **OFDMA** signal.
- **Each colored rectangle** on the **right** represents **1 symbol** across **all** the **subcarriers** over a **fraction** of **1 symbol duration**, as corresponds to an **SC-FDMA** signal.
- Applying the **DFT** to the **signal** on the **left** may be understood as **rotating** the "**signal cube**" **90º clockwise** (looking from the top of the cube), which transforms **time** into **frequency**.
- Applying the **iDFT** to the **signal** on the **right** may be understood as **rotating** the "**signal cube**" **90º counterclockwise** (looking from the top of the cube).
- **Before** the **DFT**, **each symbol** is **transmitted** in **parallel**, within its own **subcarrier**, along the **whole time** of a **symbol duration**.
- **After** the **DFT**, **each symbol** is **transmitted sequentially**, across **all** the **subcarriers**, along a **time fraction** of a **symbol duration**. Therefore, **all** the **subcarriers jointly** contain **only one symbol** at any given moment, which resembles a **single-carrier signal**.
- The **DFT** is applied in **transmission** and the **iDFT** is applied in **reception**.
- The example in the figure uses **localized subcarrier mapping** (click 11.17).
- "**s**" represents the **subcarrier spacing**.

## 10.12.4 Summary of (SC)OFDM(A) processing differences

The following figure shows the processing of **OFDM**, **OFDMA** and **SC-FDMA**:




Some **relevant comments** on this figure are:

- The **OFDMA processing** is basically the **same** (**both** at the **transmitter** and at the **receiver**) as the **OFDM processing** (click 10.12.1).
- The **SC-FDMA processing** is the **OFDMA processing** (click 10.12.2) with an **additional DFT precoder** at **each transmitter** (**each UE**) and **additional iDFT decoders** at the **receiver** (the **NR node**).
- The **additional Tx DFT precoder** and **Rx iDFT decoders** are **highlighted** in the figure with an **orange border**.
- For the **labels** and **symbols** in the figure, click 10.12.5.

## 10.12.5 Labels on the figures for (SC)OFDM(A) processing

- **Each thin black arrow** represents a **passband radio signal** in the **air**.
- **Each small yellow box** represents the **cyclic prefix adder/remover** (click 11.13) and the **RF functional block** (click 4.5.8).
- The **thin colored arrows** represent **subcarriers**.
- The **thin white arrows** represent **null subcarriers** (**zero Fourier coefficient**).
- **Each thick arrow** represents a **Tx** or **Rx symbol stream**. The **black ones** represent the **complete Tx or Rx symbol stream** at the **NR node**. The **brown ones** represent the **specific Tx or Rx symbol stream** of **$UE_1$**.
- "**d**" is the **total quantity** of **Downlink subcarriers** in the **signal**.
- "**k**" is the **quantity** of **Downlink subcarriers** assigned to **UEs other** than **$UE_1$**.
- "**u**" is the **total quantity** of **Uplink subcarriers** in the **signal**.

- "**x**" is the **quantity** of **Uplink subcarriers** assigned to **UE2**.
- "**z**" is the **quantity** of **subcarriers** assigned to **UE3**.
- "**Subc.**" and "**scs.**" stand for **subcarriers**.
- "**Mapp.**", "**Demapp.**" and "**Equaliz.**" stand (respectively) for **Mapping**, **Demapping** and **Equalization**.
- **iDFT** and **DFT** stand (respectively) for **inverse discrete Fourier transform** and **discrete Fourier transform**.
- **S-P** stands for **serial-to-parallel**, while **P-S** stands for **parallel-to-serial**. The additional conversions for **SC-FDMA** has **not** been represented.
- **DL** and **UL** stand (respectively) for **Downlink** and **Uplink**.

## 10.13 Coordinated Multipoint (CoMP)

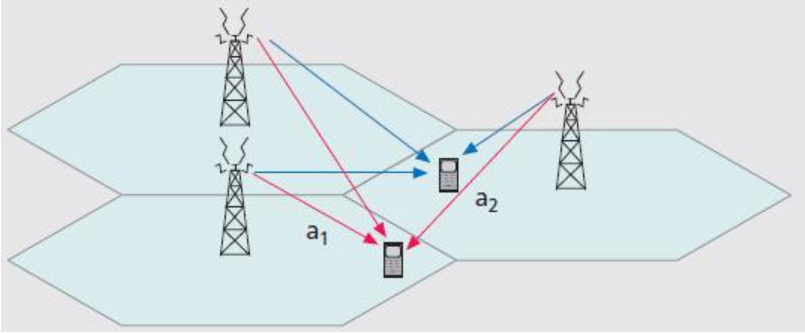
**CoMP** is a **diversity technique** (click 11.8) where **multiple antenna sites cooperate** to improve the **signal reception** for **UEs**, in particular at the **cell edges** where **interference** and **weak signals** are **common**. The **primary goal** is to **enhance signal reliability** and **mitigate interference**.

Types of **CoMP** implementations:

- **Joint Transmission (JT):**

  The **same data** is **simultaneously transmitted** to the **UE** from **multiple neighbor antenna sites**, improving signal strength through **constructive interference**.

  The following figure illustrates this type:




- **Dynamic Point Selection (DPS):**

  The **best** among the **neighbor antenna sites** is **dynamically selected** for **each UE** based on **channel conditions**, which **minimizes interference**.

- **Coordinated Scheduling/Beamforming (CS/CB):**

  The **beamforming** and/or the **scheduling** from **neighbor antenna sites** is **coordinated** to reduce **inter-cell interference**.

## 10.14 Integrated Access and Backhaul (IAB)

**Integrated access** and **backhaul** (**IAB**) allows for **multi-hop backhauling**. **IAB** may **either** use the **same channels** employed for **UE access** or a **specific** and **dedicated channels**.

**Integrated access** and **backhaul mobile termination** (**IAB-MT**) is a **subsystem** of an **IAB node**. In a substantial number of aspects regarding the **5G Downlink** and **Uplink** (but **not Sidelink**) specifications, the **IAB-MT** subsystem behaves as a **UE**.

For more detail on **integrated access and backhaul**, refer to clause 4.7 of TS 38.300.

## 10.15 5G deployment options: Standalone (SA) and Non-Standalone (NSA)

Operators do **not deploy** the **5G infrastructure** in **one-shot**, and rather they defined some **evolution** from their **LTE infrastructure** to the **5G infrastructure**. **3GPP** standardized several **5G deployment options**, to facilitate this **evolution** from the **LTE** infrastructure to the **5G infrastructure**. **Each operator** selected its **deployment option** depending on his allocated spectrum, the geographic area it serves (terrain and user density), the capabilities of its equipment, and business factors (such as cashflow, equipment renewal lifecycle, etc).

The **5G deployment options** are categorized as **Standalone** (**SA**) or **Non-Standalone** (**NSA**), as follows:

- **Standalone options** (**SA**)

  **Only one radio technology** (**either LTE** or **5G**) is required for **UE** network access. **Both control** and **user** planes go through the same **NR node**. **Inter-RAT** handover is needed for service continuity when the **UE moves** between **LTE** coverage and **5G** coverage.

  **Standalone** includes **option 2** (**5GC** with **gNB**) and **option 5** (**5GC** with **ng-eNB**). There is also **option 1**, but this is just the legacy native **LTE** (**EPC** with **eNB**).

- **Non-Standalone options** (**NSA**)

  **Both radio technologies** (**LTE** and **5G**) are required for **UE** network access using **multi-radio dual connectivity**. The **control** plane goes through the **master** node whereas **user** plane is **split** between the **master** node and a **secondary** node. There is a tight interworking between **LTE RAN** and **5G RAN**.

  **Non-Standalone** includes **option 3** (**EPC** plus **master eNB** and **secondary en-gNB**), **option 4** (**5GC** plus **master gNB** and **secondary ng-eNB**), and **option 7** (**5GC** plus **master ng-eNB** and **secondary gNB**).

For a description of these different **node types**, click 4.2.

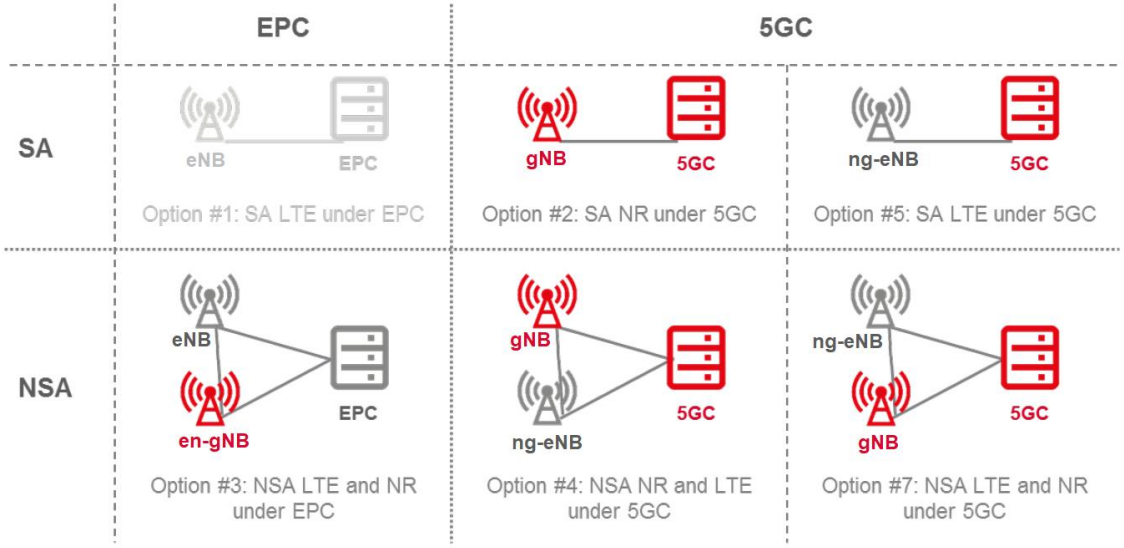
The next table summarizes the **5G deployment** options corresponding to **NSA** and **SA**:

| Deployment option | Mode | Core | Master RAT | Secondary RAT | 3GPP term | 3GPP release and date |
|---|---|---|---|---|---|---|
| **Option 1** | SA | EPC | LTE | - | LTE | R8, 2006 |
| **Option 2** | SA | 5GC | NR | - | NR | R15, Jun18 |
| **Option 3** | NSA | EPC | LTE | NR | EN-DC | R15, Dec17 |
| **Option 4** | NSA | 5GC | NR | eLTE | NE-DC | R15, Mar19 |
| **Option 5** | SA | 5GC | eLTE | - | eLTE | R15, Jun18 |
| **Option 7** | NSA | 5GC | eLTE | NR | (NG)EN-DC | R15, Mar19 |

Some **relevant remarks** on this table are:

- The term **eLTE** stands for "**evolved LTE**", and refers to an **LTE RAN** whose **nodes** can connect to the **5G Core** (**5GC**).
- The term **EN-DC** stands for "**E-UTRA and NR dual connectivity**".
- The term **NE-DC** stands for "**NR and E-UTRA dual connectivity**".
- The term **(NG)EN-DC** stands for "**NG-RAN E-UTRA and NR dual connectivity**" (click 10.3).

The following figure shows the **5G deployment options** corresponding to **NSA** and **SA**:




Some **relevant remarks** on this figure are:

- **Gray** "waves" represent the **LTE radio** (**E-UTRA**) while **red** "waves" represent the **5G** (**NR**) **radio**.
- The **LTE Evolved Packet Core** (**EPC**) for **option 3** is **5G enabled**.

Within **option 3**, there are **three variants** called **option 3** (plainly), **option 3a** and **option 3x**. These **three option 3 variants share** the following **characteristics**:

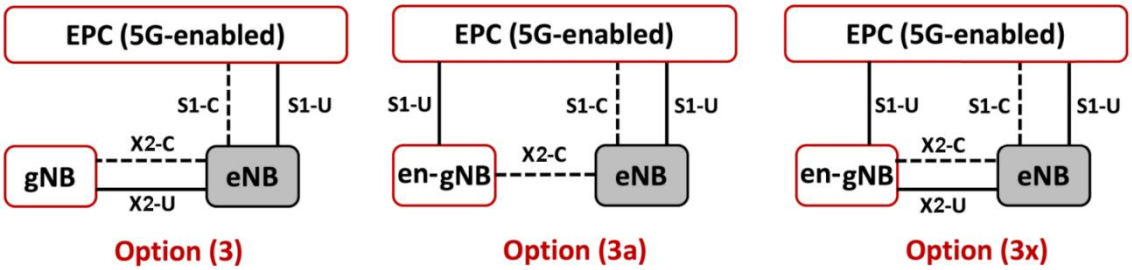
- The control plane between the **EPC** and the **eNB** is defined by the **S1-C** interface.
- The control plane between the **eNB** and the **gNB** is defined by the **X2-C** interface (sometimes called the **Xx** interface)
- There is **no** direct **control traffic** between **EPC** and **gNB**.

The differences between these **three option 3 variants** are in how the **user** plane **traffic** is **routed**:

- In **option 3**, the **user** plane **traffic** is **split by** the **eNB**, specifically by its **PDCP** sublayer. Some **user** plane **traffic** is sent from/to the **eNB** to/from the **gNB** across the **X2-U** interface.
- In **option 3a**, the **user** plane **traffic** is **split by** the **EPC**. Some user plane traffic is sent from/to the **EPC** to/from the **en-gNB** across the direct **S1-U** interface.

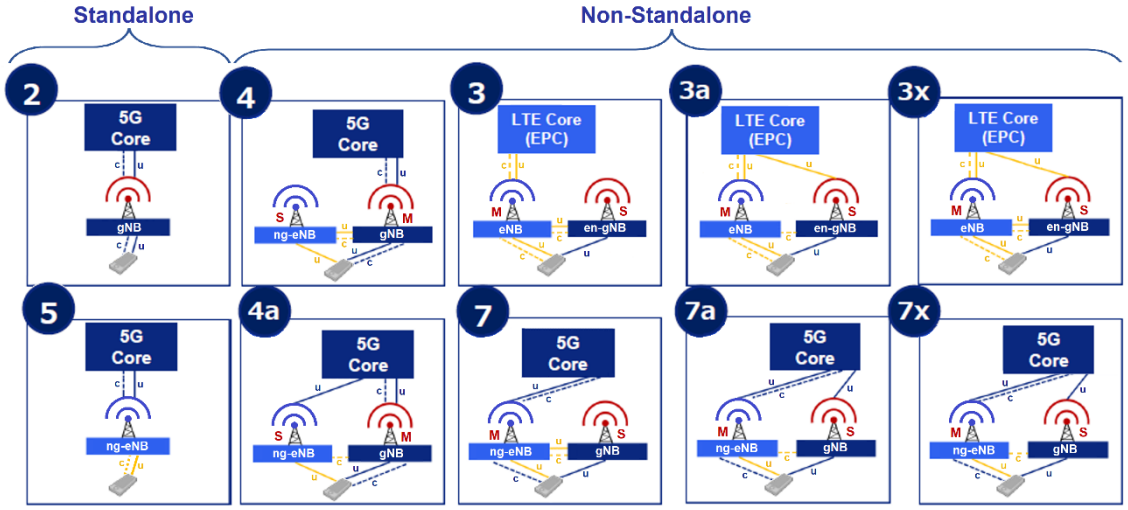
- In **option 3x**, there is a hybrid behavior: **both** the **eNB and/or** the **EPC** may split the **user** plane **traffic**.

The following figure shows **option 3**, **option 3a** and **option 3x** in more detail:




**Similar approaches** for the **splitting** of **user** plane **traffic** involving **5G Core** instead of **EPC** give rise to three **variants**, which are **option 4a**, **option 7a** and **option 7x**.

The following figure summarizes all the **NSA** and **SA** deployment **options** and **variants**:




Some **relevant remarks** on this figure are:

- **Blue** "waves" represent the **LTE radio** (**E-UTRA**) while **red** "waves" represent the **5G** (**NR**) **radio**.
- **Dashed** lines labeled "**c**" represent **control** plane traffic while **solid** lines labeled "**u**" represent **user** plane traffic.
- **Yellow** lines represent **LTE** traffic while **blue** lines represent **5G** traffic.
- An "**M**" identifies the **master node** while an "**S**" identifies the **secondary node**.
- **All** the **LTE Evolved Packet Core** (**EPC**) shown **must** be **5G enabled**.

As an example of a criterion for **user plane traffic splitting**, **VoLTE** traffic is sent to the **eNB** while **eMBB** traffic is sent to the **gNB**.

Several **deployment options** may **coexist** in the **same network**. Actually, **options 2** and **5** may **coexist** for a **considerable time**.

Notice that, **de facto**, **NSA** implies that the **paired LTE** and **5G nodes** must be from the **same manufacturer**, which is a relevant requirement.

Notice that some **UEs** may **only** support **NSA access** and **not SA access** (click 4.9.1).

Source web pages:

- https://devopedia.org/5g-deployment-options
- https://www.telecomhall.net/t/5g-deployment-options-know-all-in-details-options-1-2-3-4-5-6-7/22432/3

## 10.16 RAN sharing

**RAN sharing** consists of a **single deployment** of **RAN physical infrastructure** (i.e., antennas, base stations and backhaul) providing **service** to **customers** from **several operators**, as if the **RAN physical infrastructure** were **owned** by **each** of the **operators**. **Each operator** has and operates **its** own **5G Core**. **RAN sharing** is particularly **suited** to **areas** with **low customer density** (e.g., natural parks, rural areas, etc), but it can **also** be useful in **other scenarios**, such as:

- **Private networks**: shared RAN with dedicated 5G Core.
- **Temporary events** (e.g., Olympic games).
- **Disaster recovery**.

The **shared RAN physical infrastructure** may be **owned** and **managed** by **one** of the **operators** or by a **third party** (e.g., a **tower company**, a railway company, …).

The **NR node** advertises **multiple PLMN IDs** (click 3.5) in `plmn-identityList` from `CellAccessRelatedInfo` (click 3.6.4), within `SIB1` (click 5.3.4), allowing **each UE** to select its provider operator.

**Each operator** can apply its own **policies**, **QoS**, and **security settings**.

**Traffic** is **logically separated** in **each 5G Core**, even though the **RAN** is **shared**.

There are **2 main technical solutions** to provide **RAN sharing**:

- **Multi Operator Core Network** (**MOCN**)
- **Multi Operator RAN** (**MORAN**)

**Multi Operator Core Network** (**MOCN**)

**Each NR node** connects to **multiple** "**access and mobility management functions**" (**AMFs**) and **multiple** "**user-plane functions**" (**UPFs**), where **each AMF** and **each UPF** belongs to the **5G Core** of a **different operator**.

**Operators** can **optionally share** their **spectrum**. If shared, it can be **jointly** used or **partitioned**.

**Characteristics of Multi Operator RAN** (**MORAN**)

**Each operator transmits** in its **own channels**, with its **specific baseband configuration**. To this purpose, the **RAN physical infrastructure** supports **multiple logical configurations**, **one per operator**. Consequently, **MORAN** allows **each operator** to maintain **full control** over its **RAN configuration**.

**Operators cannot share** their **spectrum**. **Each operator** use its **own carriers**, separated from the carriers of the **other operators**.

Source web pages:

- https://www.parallelwireless.com/products/multi-tenant-and-sharing/

## 10.17 Narrowband IoT

**Narrowband IoT** consists of using, **within 5G**, a **GSM 200 kHz carrier** for **low-speed data** for **IoT** devices.

In **5G-R16**, **narrowband IoT** has **one mode** of **operation**, and an **additional** one **not fully standardized**:

- **Standalone**

  Using a specific **GSM 200 kHz channel** (click 10.17) to place the **Narrowband IoT carrier**.

- **Guardband** (**not fully standardized**)

  Using a specific **GSM 200 kHz channel** within the **guardband** of **5G FDD** bands. You can put **one 200 kHz carrier** in **each** of the **two guardbands**. However, this is **difficult** because the **5G guardbands** are **narrower** than the ones of **LTE**.

In **5G-R17**, **narrowband IoT** has **two modes** of **operation**, and an **additional** one **not fully standardized**:

- **Standalone**

  Using a specific **GSM 200 kHz channel** to place the **Narrowband IoT carrier**.

- **5G carrier**

  Using **one group** of **12 15 kHz contiguous subcarriers** to place the **Narrowband IoT carrier**.

- **Guardband** (not **fully standardized**)

  Using a specific **GSM 200 kHz channel** within the **guardband** of **5G FDD** bands. You can put **one 200 kHz carrier** in **each** of the **two guardbands**. However, this is **difficult** because the **5G guardbands** are **narrower** than the ones of **LTE**.

In **LTE**, **narrowband IoT** has **three modes** of **operation**:

- **Standalone**: you use a **specific GSM 200 kHz channel** to place the **Narrowband** IoT carrier.
- **Guardband**: you use it within the **LTE guardband** of **LTE FDD bands**. You can put **one 200 kHz carrier** in **each** of the **two guardbands**.
- **LTE carrier**: you use **one group of 12 15 kHz contiguous subcarriers** to place there the **Narrowband IoT carrier**.

Source web pages:

- https://www.rcrwireless.com/20190718/carriers/three-nb-iot-deployment-models

### 10.17.1 GSM carrier

This section contains an overview of the **GSM carrier**, in relation to its usage in **Narrowband IoT**.

### 10.17.1.1 GSM Frequency Management

Each **GSM** frequency band of **25 MHz** is divided into **125** channels, each being **200 kHz**. Each channel carries one **GSM** radio frequency (**RF**) carrier, except one of them, that is used as **guardband** between **GSM** and other frequency bands. The **25 MHz GSM band** therefore has **124 useful channels**, that transport **124 carriers**.

**GSM** uses **only FDD** (and **no TDD**). The base station operates in **full-duplex**. However, the **UEs** ("**Mobile Station**" or "**MS**" in **GSM** terminology) operate in **half-duplex**, because they use the **same antenna** to **transmit and receive**.

Each **carrier** is modulated (click 11.2.1) using either **GMSK** (1 bit/symbol), **QPSK** (2 bits/symbol), **8PSK** (3 bits/symbol), **16QAM** (4 bits/symbol) or **32QAM** (5 bits/symbol).

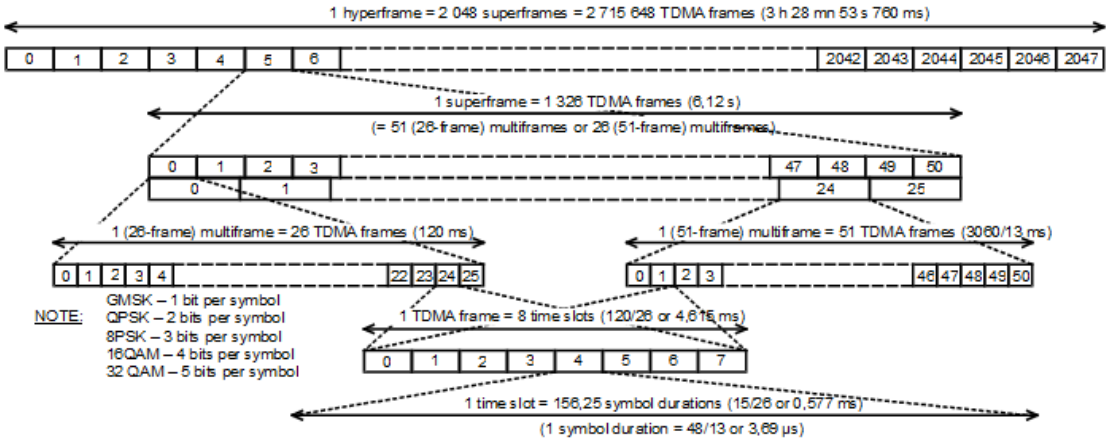
**GMSK**, **QPSK** and **8PSK** have the advantage of **not** requiring amplitude modulation. This allows the usage of low-cost **type C amplifiers** which are very non-linear and do not allow for amplitude modulation.

### 10.17.1.2 GSM Time management

Each GSM carrier is slotted in time. GSM has a hierarchical frame structure composed of **hyperframe**, **superframe**, **multiframe**, **frame** and **slot**, as follows:

- Each **hyperframe** contains **2,048 superframes**.
- Each **superframe** contains either **26** or **51 multiframes**.
- Each **multiframe** contains either **51** or **26 frames** (see further below).
- Each **frame** contains **8** time **slots** (TS).

Hence, each **hyperframe** contains 2,048•51•26 = **2,715,648 frames**. The following figure shows the GSM hierarchical frame structure:




As shown in the figure above, there are two variants of multiframe structure:

- **51-multiframe**: it is sent **Downlink**. It has a duration of 235.4 ms and contains **51** frames. It is divided into **logical channels**, that are time scheduled by the base station. Always occur at **beacon frequency** in **time slot 0** (TS0), and it may also take up other time slots if required by system: for example, TS2, TS4, TS6. For more detail, see the next subsection.

  **51-multiframe** is sometimes called "**control multiframe**".

- **26-multiframe**: it is sent **Uplink**. It has a duration of 120 ms and contains **26** frames. Out of the 26 frames, 24 are used for user data, 1 for the "**Slow Associated Control Channel**" (**SACCH**) and 1 is **idle** (unused).

  **26-multiframe** is sometimes called **traffic multiframe**.

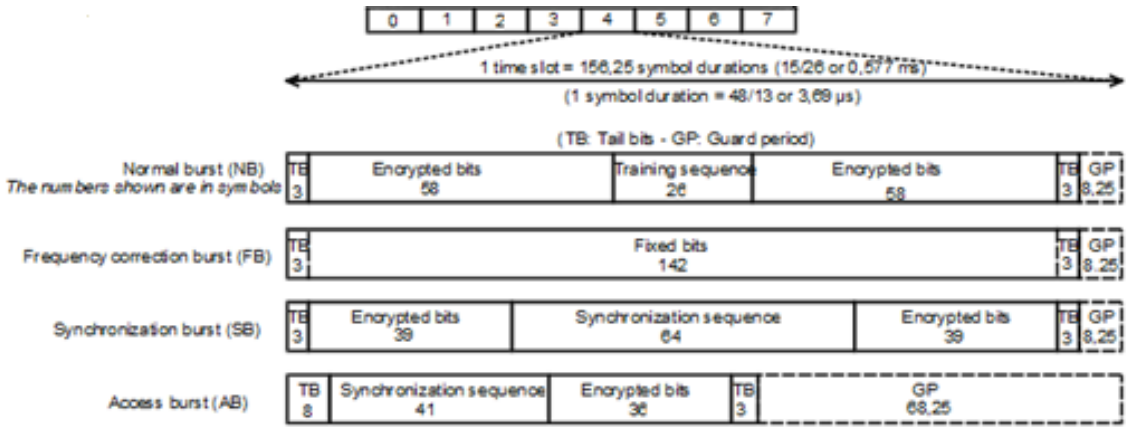
Since the **UE** works in half-duplex, there is a **3-slot** delay between the **Uplink TS0** and the **Downlink TS0**. This **3-slot** time is used by the **UE** to perform various functions (e.g., processing data, measuring signal quality of neighbor cells, etc).

The basic radio resource to be allocated among **UEs** is one **slot**. The **duration** of a **slot** is about **577 µs** (15/26 ms). **Each slot** carries **156.25 symbols**, which results in a symbol rate (or modulation speed) of **270.833 kbauds**. **Each slot** is used to transmit a **burst** of symbols followed by a **guard time**. For most bursts, the **guard time** is **8.25 symbols**, but it can also be **68.25 symbols**. It is used to prevent overlapping between consecutive bursts (a collision), that can happen because of time-synchronization errors. There are four types of bursts:

- Normal Burst (NF).
- Frequency correction Burst (FB).
- Synchronization Burst (SB).
- Access Burst (AB).

It is also defined a Dummy Burst (DB), that is a Normal Burst with dummy data. The Dummy Burst is transmitted when no other burst needs to be transmitted.

The following figure shows the **structure** of these four types of **bursts**. Notice that the Access Burst has a longer guard period (68.25 symbols) than the other ones (8.25 symbols).




As shown in the figure, the different type of **bursts** have the following **fields**:

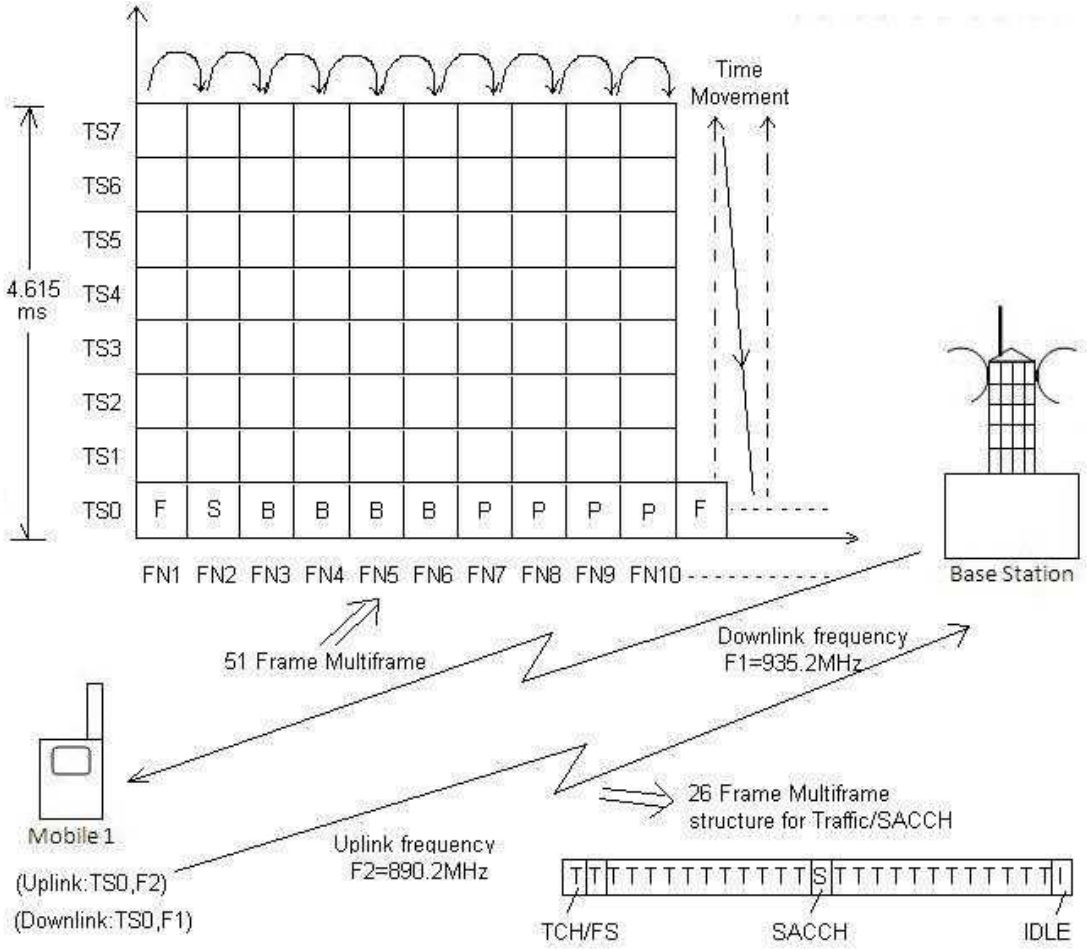
- **Tail**: identifies the **beginning** and **end** of the **burst**. It is **3 symbols** both at the **beginning** and **end** of the **burst**, with the exception of the Access Burst, where the beginning Tail field has **8 symbols**. In the figure above it is labeled as "TB".
- **Coded data**: contains the actual data in the **burst**. It may have **58**, **39** or **36** symbols, depending on the burst type. The last decoded **bit** of each of the two 58-symbol sequences is the "**Stealing flag**". In the figure above it is labeled as "Encrypted bits". The quantity of bits contained in this field depends on the modulation used (click 10.17.1.1)

- **Stealing flag**: indicates whether the **coded data** is **user** data (value "1") or **control** data (value "0"). Most coded data fields carry user data, but under some circumstances, some may be used to send control data. It is the **last decoded bit** within the **coded data** field.
- **Training sequence**: shows the amount of channel fading, helping to decide which equalizer should be used. It helps the adaptive equalizer in both the base station and the **UE** to analyze the radio channel characteristics before coding. It has **26 symbols**. It is sometimes called "**midamble**".
- **Fixed symbols**: provides frequency synchronization. It is **142 fixed** symbols. It is equivalent to an un-modulated carrier, shifted in frequency, with the same guard time as the normal burst. It is broadcast together with the **BCCH**. In the figure above it is labeled as "Fixed bits".
- **Synchronization sequence**: provides time synchronization. It is **64 fixed** symbols. It contains a long training sequence and carries the information of the **TDMA** frame number (**FN**) and base station identity code (**BSIC**) which is 6 bit long and consists of 3 bits of **PLMN** color code with range 0 to 7 and 3 bits of base station color code with range 0 to 7.

#### 10.17.1.3 GSM Physical and Logical channels

During network entry each **UE** is allocated one Downlink slot and one Uplink slot. In the figure below each **UE** is allocated 890.2 MHz in the **Uplink** and 935.2 MHz in the **Downlink**. The **function** of **TS0** is different for **traffic** and **control** multiframes.

The figure below depicts the **frame structure** of the **Downlink control 51-multiframe** (the rectangle at the top left) and the **Uplink traffic 26-multiframe** (the vector at the bottom right).




For the **51-multiframe**, the "**columns**" represent **frames**, numbered sequentially (FN1, FN2, …) from 1 up to 51 (the complete multiframe). The "rows" represent **slots**, **within each frame**, numbered **0** to **7**. The figure depicts **the usage of TS0** in successive **frames**. In frame 2, TS0 is used for the "**Frequency Correction Channel**" (**FCCH**), indicated as "**F**" in the figure. In frame 1, TS0 is used for the "**Synchronization Channel**" (**SCH**), indicated as "**S**" in the figure. In frames 3 to 6, TS0 is used for the "**Broadcast Control Channel**" (**BCCH**), indicated as "**B**" in the figure.

For the **traffic 26-multiframe**, each letter represents **one frame**, from 1 to 26. Frames labeled "**T**" are used for the **Traffic Channel at Full-rate Speech** (TCH/FS). Frame labeled "**S**" is used for the **Slow Associated Control Channel** (SACCH). Frame labeled "**we**" is unused (idle).

Since the **UE** works in half-duplex, there is a **3-slot** delay between the **Uplink TS0** and the **Downlink TS0**. This **3-slot** time is used by the **UE** to perform various functions (e.g., processing data, measuring signal quality of neighbor cells, etc).

Source web pages:

- https://www.rfwireless-world.com/Tutorials/gsm-frame-structure.html

## 10.18 5G Edge Computing

**Edge Computing** consists of: **a**) **deploying computing services physically close** to the **NR nodes** to run user applications and **b**) allowing the **user** to **customize** the **routing** of its **traffic** in relation to the **computing resources** that run **his services**.

**ETSI** standardized "**Multi-access Edge Computing** (**MEC**)", but due to its **complexity** there are other **simplified** approaches under the general term "**Edge Computing**".

### 10.18.1 Fundamental MEC modes

An essential functionality of **MEC** is the **specific routing** of **IP** packets in relation to the **MEC computing resources** that are running the corresponding **MEC applications**. There are **four basic MEC routing modes**, targeted at matching the requirements of a wide range of **MEC applications**:

- **Breakout mode**

  **Session traffic** is **redirected** to the **MEC application** that is running in the **MEC server** located **either locally** or on a **remote server**.

  Some examples of applications that require **breakout mode** include **Content Distribution Networks**, **gaming servers**, and **corporate local area networks**.

- **In-line mode**

  **Session traffic remains** between the **UE** and the **destination server** located on the Internet, but it **traverses** the **MEC application** that is running in the **MEC server**.

  Some examples of applications that require **in-line mode** are **transparent caches** and certain **security applications**.

- **Tap mode**

  **Session traffic** is **duplicated**. **One flow continues** to its destination, while the **other** is forwarded to the **MEC application** that is running in the **MEC server**.

  Some examples of applications that require **tap mode** are **virtual probes** and certain **security applications**.

- **Independent mode**

  **Traffic offloading** is **not** required, but the **MEC application** must be **registered** because it **uses MEC services** such as **DNS** or **Radio Network Information Service** (**RNIS**), among others.

### 10.18.2 Edge Computing in 5G and LTE

The **5G control architecture** is specifically designed to **integrate Edge Computing**. The established operating mode for this is to distribute the **User Plane Function**, locating it between **each base-band unit** (**BBU**, click 4.7.4) and its associated **Edge Computing** server.

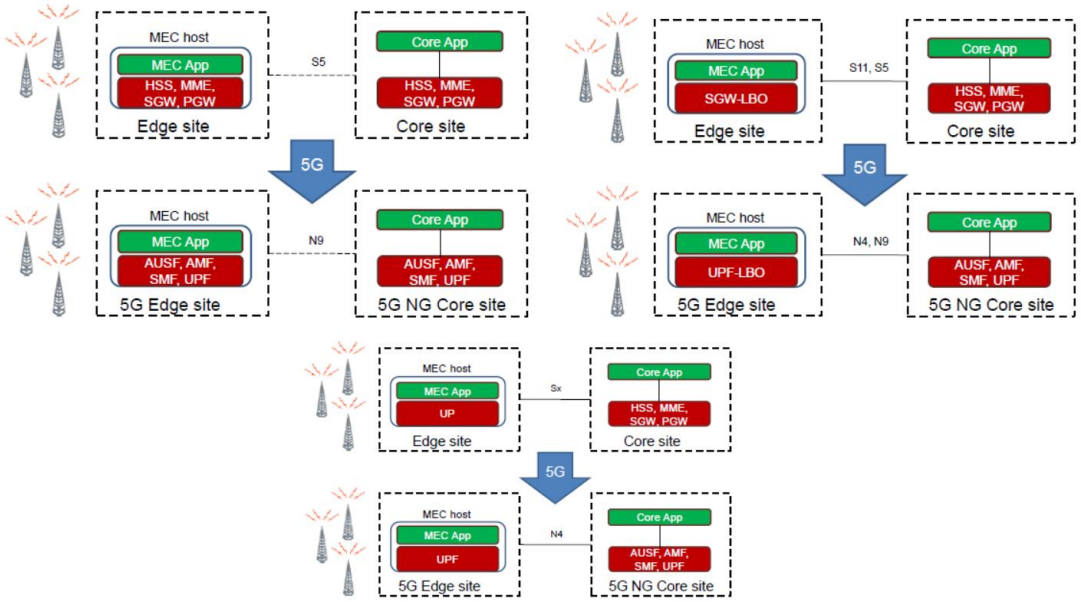
**One** of the essential objectives of **Edge Computing** is the provision of **low-latency service**s, which is why **Edge Computing** servers **must** be sufficiently close to the **BBU**(**s**) through which they serve end users.

**Edge Computing** also be provided on **LTE** networks. However, the **architecture** of **LTE** networks is **not designed** to support **Edge Computing**. For this reason, there is **no**

**single solution** for integrating **Edge Computing** into **LTE** networks, and there are **several** alternative solutions. The main mechanisms for implementing **Edge Computing** in **LTE** networks are:

- **Bump** in the wire
- **Distributed S/PGW**
- **Distributed S/PGW** with **Local Breakout** (**SGW-LBO**)
- **Distributed EPC**

The following figure shows some **Edge Computing** configurations in both **5G** and **LTE** networks, indicating the allocation of **network functions** between the **Edge servers** and the **Core servers**.




As already indicated, the figure shows that for **5G**, **Edge Computing** is **integrated** into the **control architecture**. However, for **LTE** there are **various** functional configuration **alternatives**. **Each** such **alternative** has its specific **advantages** and **disadvantages**, but **none** of them is fully **functionally integrated**.

## 10.18.3 Advantages of Edge Computing

The use of **Edge Computing** is **mandatory** for **applications** with one (or more) of the following **requirements**:

- **Low latency**

  Some examples are robot control, autonomous cars and drones or gaming.

- **Massive data transfer**

  Some examples are image recognition across thousands or millions of cameras or learning models over captured data.

- **Local data processing**

  For example, data must be processed in the same facility where it is received because of high security requirements.

A **complementary** architecture to **Edge computing** is **Fog computing**, where **processing** takes places in a swarm of **terminals**.

For cases where there are **no** specific **technical** or **business constraints**, **processing** can be performed using **cloud computing** in **large data centers**, with data flowing end-to-end across the network. **Cloud computing** is usually **lower cost** and **more reliable** than **edge computing**.

On some cases, the **application requirements** can **only** be satisfied using **either cloud**, **edge** or **fog computing** . However, on **many** other cases **cloud**, **edge** and **fog** are **complementary technologies**, that are advantageously used in a combined way.

A large vertical customer may use **edge computing** for **average load** conditions while **offloading processing peaks** to a **cloud facility**. additionally use the edge computing service during processing peaks or in the event of a failure on their own servers. Conversely, a given **application** may normally operate in **cloud mode** and use instead **edge computing** when it has a low load, in an opportunistic way.

The conclusion is that **cloud**, **edge** and **fog** are **not separate** systems but **rather** constitute a **single network processing architecture** that must be managed in a unified manner. This allows for the **dynamic migration** of **virtual machines**, **containers**, and **processes** between the **three systems** to adapt to the **customer**'s **processing needs**.

In this context, **edge computing** located in the **5G core** takes on **significant importance**. The reason is that it can achieve **very similar capacity**, **cost** and **reliability** to **cloud computing**, while providing a **significantly lower latency** than **cloud computing**.

# 11. NR underlying technologies

## 11.1 Bit, symbol rate, bitrate, throughput, (area) capacity and spectral efficiency

This section contains the **definition** and **explanation** of the following **very relevant terms** that are used throughout this book: **bit** (click 11.1.1) plus **symbol rate**, **bitrate**, **throughput**, **capacity**, **area capacity** and **spectral efficiency** (click 11.1.2).

### 11.1.1 Bit

The **term bit** is used to denote **two different** concepts:

- A **binary digit**

  It is a **digit** from a **binary number system**. Its values are denoted as **0** or **1**. This meaning of "**bit**" applies (among others) to **number representation** and **processing** in mathematics, telecommunications and informatics. It also applies to the **modulation order**, to **bitrate** (click 11.1.2), to **file size**, to **bitfield size** (e.g., in **DCI message fields**, click 6.9.1), to **encoding lengths** of **ASN.1** values and to **identifier size**. The **quantity** of **bits** of any of these elements is expressed as an **integer number** of **bits**. This definition is performed in **number theory**. This definition is where the name "**bit**" comes from: "**bi**nary digi**t**", playing with its plain **English meaning** of "small piece".

- An **information-bit**

  It is the **information** provided when revealing the **outcome** of a **memoryless event** with **two possible results**, where **each result** has a probability of **50%**. This meaning of "**bit**" applies (among others) to **information rate** (i.e., **capacity**, see below), to **file information content** and to **field information content**. The **information content** of a given **element** is expressed as a **real number** of **bits**. This definition of **bit** is performed in **information theory**.

The difference between a **binary digit** and an **information-bit** arises from the **information content** of **binary digit**, which depends on their **probability** (information sources) or **error probability** (mutual information). A few examples to illustrate this are:

- A system may **transmit** "***bd***" **binary digits** per **second**, but the information delivered to the receiver may be **any** quantity "***ib***" of **information-bits** per **second** (where "$ib \leq bd$"). If the **reception** is **extremely reliable** the mutual information between **every received binary digit** and its corresponding transmitted **binary digit** would be close to **one information-bit**. The resulting **information rate** would therefore be close to "***ib***" **information-bits** per **second**. However, if the reception is **extremely unreliable** the mutual information between **every received binary digit** and its corresponding transmitted **binary digit** would be close to **zero information-bits**. The resulting **information rate** would therefore be close to **zero information-bits** per **second**.

- A **lower bitrate** (click 11.1.2) may result in a **higher throughput** (click 11.1.2). If the **modulation order** is **too high** (e.g., **10 bits/symbol**) for the channel characteristics (e.g., a **very noisy channel**), the resulting **error probability** may result in a **very low throughput**. In this case, **reducing** the **modulation order** (e.g., to **2 bits/symbol**) will result in a **higher throughput**.

- A file contains "***bd***" **binary digits**, but its information content may be **any** quantity "***ib***" of **information-bits** (where "$ib \leq bd$"). If the file contains **totally random data**, its **information** content would be "***ib***" **information-bits**. However, if the file contains **highly redundant data**, its information content would be **much lower** than "***ib***" **information-bits**. This is why **files** and **transmitted information** can be **compressed** (if they are redundant) to a **smaller size** in terms of **binary digits**, and yet they can be **exactly recovered** by decompression.

This is why I propose to **change** the name of **information-bits** from "***bits***" to "***ibs***", in order to prevent **confusion** between "**binary digits**" and "**information-bits**". This is what was done when the **binary scaling prefixes** (i.e., "**kibi** (**Ki**)=$2^{10}$", "**mebi** (**Mi**)=$2^{2 \cdot 10}$", "**gibi** (**Gi**)=$2^{3 \cdot 10}$", etc) were introduced, to prevent confusion with the **decimal scaling prefixes** ("**kilo** (**k**)=$10^3$", "**mega** (**M**)=$10^{2 \cdot 3}$", etc).

Do not confuse a **binary digit** with a **binary symbol**. **Binary symbols** are used for example in **binary digital modulations** (**BASK**, **BFSK**, **π/2-BPSK**, **BPSK**, …), in **binary line codes** (**Manchester**, …) and in **information disks**. A **binary symbol** is **sometimes incorrectly** called **bit** because it **represents** a **binary digit**. This **bad practice** frequently causes **confusion** between **symbol rate** and **bitrate** (click 11.1.2).

### 11.1.2 Symbol rate, bitrate, throughput, capacity and spectral efficiency

**Symbol rate**

**Symbol rate** denotes the **symbols** per **second** transmitted at the **physical layer**.

Its units are **bauds**. **One baud** is **one symbol** per **second**. The **symbol rate** is sometimes called the "**baud rate**" or the "**modulation speed**".

**Bitrate**

**Bitrate** denotes a **rate** of **binary digits**.

Its **units** are **binary digits** (click 11.1.1) per **second** (**bits per second**, **bps**).

The **symbol rate** and the **bitrate** are **directly proportional**, through the **modulation order** "$Q_m$" (click 11.2.1):

$$bitrate = Q_m \cdot symbol_rate$$

The **symbol rate** that can be achieved through a channel is usually **limited** by the **channel**'s **bandwidth** while the **modulation order** "$Q_m$" (click 11.2.1) is usually **limited** by the **channel**'s **SNIR** at the **receiver**.

**Throughput**

**Throughput** denotes a **rate** of **information**.

Its units are **information-bits** (click 11.1.1) per **second** (**bits per second**, **bps**).

The **throughput** is the result of the **error correction procedures** (**FEC** and/or **ARQ**) over the **bitrate**.

**Bitrate and throughput**

The **bitrate** and the **throughput** are frequently **confused** because they **seem** to have the **same metric** (**bits per second** (**bps**)). Although they are **both** measured in **bits per**

**second**, these are **different bits per second!** The **bitrate** is measured in **binary digits** per second while the **throughput** is measured in **information-bits** per second.

**Capacity**

**Capacity** denotes the **maximum achievable throughput** of an **element** (e.g., a **bandwidth part**, a **UE**, a **cell**, a **channel**).

**Spectral efficiency**

The **spectral efficiency** is the **capacity** provided by a given **analog bandwidth**. Its **units** are **bits per second**, **per Herz** (**bps/Hz**).

**Term characteristics**

**All** the previous **terms** share the **following characteristics**.

- **Unidirectional**

  They are **unidirectional**, so they should be **qualified** as being **either Downlink** or **Uplink**.

- **Element referred**

  They are referred to a specific **element**. For example, the **capacity** may refer to a **bandwidth part**, a **carrier**, a **channel** or a **UE**. The **area capacity** may refer to a **cell**, a **factory**, a **train**, a **sports stadium**, an **airport terminal** or a **city**.

- **Variability**

  They are **variable** magnitudes. This is, their **value** may depend on **hour**, **weekday**, **season**, **location** within the geographical element, or a combination of them. Therefore, the provided **value** may be an **average** value over a certain **sample**, a **measures** value under certain **conditions** or a value **calculated** with some specific **criteria**.

- **Theoretically capped**

  They are **usually capped** by **physical** and/or **mathematical boundaries** (e.g., **Shannon** limit). The corresponding **theoretical cap** is subject to some **conditions** (e.g., **gaussian additive noise** and **linear channel**). When the **conditions hold**, **no technology** can **improve** the **theoretical cap** value.

- **Technologically capped**

  They are **capped** by **system design** and **capabilities** for **each specific technology**. For example, the **5G cell capacity** is **capped** by the **characteristics** (**maximum channel bandwidth**, **maximum modulation order**, etc) of the **5G technology**. The same applies to **any other technology** (e.g., **LTE**, **DMR** or **TETRA**). The **technological cap** is **always below** the **theoretical cap.**

- **Implementation capped**

  They are **capped** by the characteristics of **each actual implementation** belonging to a **given technology**. For example, **each 5G UE implementation** will have its specific **spectral efficiency**. The **implementation cap** is **always below** the **technological cap.**

- **Shared**

  They are **shared** by **all** the **users** of the **element**. For example, the **capacity** of a **cell** is **shared** by **all** the **UEs camped** in the **cell**. The **capacity** of a **UE** is **shared** by **all** the **applications** running in the **UE**.

## 11.2 Modulation and coding

A **digital modulation** (click 11.2.1) is a **bidirectional mapping** between a **set** of **waveforms** and a **set** of **bit values**, **both** with the **same quantity** of **elements**.

A **channel code** (click 11.2.3) is a technique to **add redundant information** to a **bit-block** in order to **protect** the **bit-block** from **errors arising** along signal propagation.

The **digital modulation** and the **channel code** to be used at any given moment are jointly decided in the so-called **modulation and coding scheme** (**MCS**, click 11.2.4).

In **every physical channel** (except **PRACH**), **each symbol interval** within **each subcarrier** (i.e., **each Resource Element**) contains **one symbol** from a **digital modulation**. At the same time, the **information** contained in **certain groups** of **Resource Elements** is **error-protected** with a **channel code**.

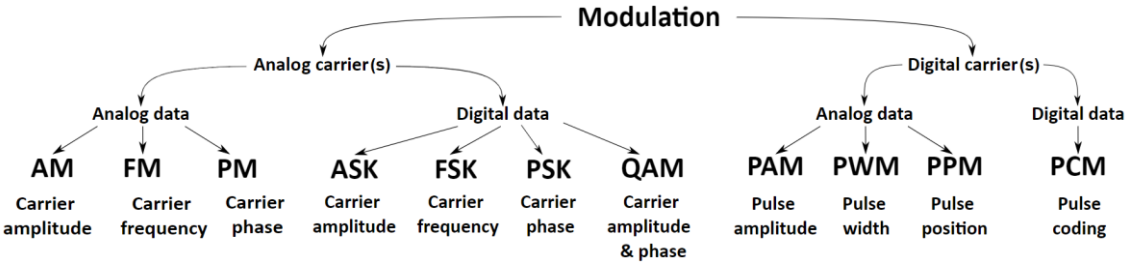
Do **not confuse** a **channel code** (click 11.2.3) with a **line code** (click 11.2.2).

Source web page:

- https://www.techplayon.com/5g-nr-modulation-and-coding-scheme-modulation-and-code-rate/

### 11.2.1 Digital modulations

A **modulation** is a technique to **convey data** by **modifying** the **properties** of a **base signal** called the **carrier**. There are **modulations** for **analog** or **digital data** and with **analog** or **digital carrier(s)**. The following figure shows the **most frequently** used modulations.




A **digital modulation** is a **modulation** of **digital data** over **1 analog carrier** or **1 subcarrier**. A **digital modulation** is a **bidirectional mapping** between a **set** of **analog waveforms** and a **set** of **binary values**, **both sets** with the **same quantity** of **elements**. **Each waveform** is called a **symbol**. The **set** of **symbols** is called the **symbol constellation** of the **digital modulation**. **Each symbol encodes** (i.e., **represents**) a specific **binary value** with a certain **quantity** of **binary digits**.

The **quantity of binary digits encoded** by **each symbol** is called the **modulation order** "$Q_m$" of the **digital modulation**. The **modulation order** "$Q_m$" is "$log_2()$" of the **quantity** of **symbols** in the **constellation**. The **units** of the **modulation order** "$Q_m$" **binary digits** per **symbol**. The **quantity** of **symbols** in the **constellation** is usually a **power** of **2**, such that **each symbol** represents an **integer** quantity of **binary digits**, and **modulation order**

"$Q_m$" is also an **integer**. The **modulation order** that can be effectively used is usually **limited** by the channel's **signal to noise and interference ratio** (**SNIR**).

The **digital modulation techniques** used in **5G** are **Binary Phase Shift Keying** (**BPSK**), **π/2-Binary Phase Shift Keying** (**π/2-BPSK**), **Quadrature Phase Shift Keying** (**QPSK**) and **Quadrature Amplitude Modulation** (**QAM**) with constellations of **16**, **64**, **256** and **1,024** symbols.

The following table shows the **quantity of symbols** in its **constellation** and the **modulation order** "$Q_m$" for **each modulation technique** used in **5G**:

| Modulation | π/2-BPSK | BPSK | QPSK | 16 QAM | 64 QAM | 256 QAM | 1024 QAM |
|---|---|---|---|---|---|---|---|
| Quantity of symbols | 2 | 2 | 4 | 16 | 64 | 256 | 1,024 |
| Modulation order $Q_m$ [binary digits/symbol] | 1 | 1 | 2 | 4 | 6 | 8 | 10 |

The channel's **bitrate** (click 11.1.2) is **directly proportional** to the **modulation order**:

$$bitrate = Q_m \bullet symbol_rate\ modulation_order$$

The **probability** of **error increases** (in a non-linear way) with the **modulation order**, so **increasing** the **modulation order** may result in a **lower throughput** (click 4.9.3).

**Each symbol** from the **constellations** of the **digital modulations** from the table above is represented as a **complex number** where its **phase** (i.e., its "**argument**") represents the symbol **phase**, and its **magnitude** represents the symbol **amplitude**. The **amplitude** of the symbols from **each constellation** is **normalized** to **1** (click 11.2.4). The **actual transmitted symbols** are the **constellation symbols** multiplied by an **amplitude/power** factor usually denoted as "**β**".

Several **tables** presenting **multiple characteristics** (including **modulation techniques**) for each of the **different physical channels** and **signals** are available at 4.11.3.

A brief explanation on a few relevant digital modulations follows:

- **Phase Shift Keying** (**PSK**)

  The **modulated signal** (e.g., each **5G subcarrier**) has **constant frequency and amplitude**. Each **symbol** is represented by a specific **signal phase** that is kept during the **symbol interval**. The **quantity** of **symbols** in the **PSK constellation** is the **quantity** of **different phases** used. The different phases are normally evenly spaced. Frequently used variants are:

  - **Binary PSK** (**BPSK**)

    Constellation of **2 symbols**, represented by **2 phases** spaced **π radians**. It is **very simple** to implement and it is **very robust** in terms of **bit error rate** (**BER**) for a given **signal** to **noise ratio** (**SNR**). In usual implementation (especially with filtering or pulse shaping) it has a high **peak to average power ratio** (**PAPR**), which performs **poorly** with **non-liner power amplifiers**.

  - **π/2 BPSK** (**π/2-BPSK**)

    **Two BPSK** constellations spaced **π/2 radians**, where **constellations alternate** at **each symbol**. Therefore, the signal **always changes phase** at

**every symbol interval**. Its **main advantage** over **BPSK** is that it has a **low PAPR**, which performs **well** with **non-liner power amplifiers**.

- **Quadrature Phase Shift Keying (QPSK)**

  Constellation of **4 symbols**, represented by **4 phases** spaced **π/2 radians**, usually placed at **π/4**, **3π/4**, **5π/4** and **7π/4 radians**. **QPSK** is also known as **Quadriphase-PSK** or as **4PSK**.

  Be aware that **QPSK** is the **same signal** as **4QAM** (see below), but the way to (de)modulate both is **very different**. **QPSK** is usually **preferred** to **4QAM** because **QPSK** processing has lower power consumption, and it has some other implementation advantages.

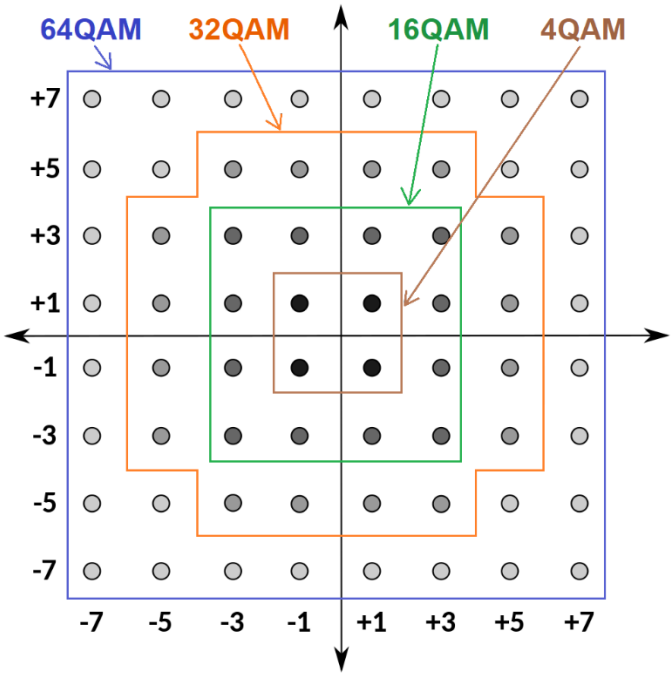
- **8 Phase Shift Keying (8PSK)**

  Constellation of **8 symbols**, represented by **8 phases** spaced **π/4 radians**.

- **Quadrature Amplitude Modulation (QAM)**

  The **modulated signal** (e.g., each **5G subcarrier**) has **constant frequency**. Each **symbol** is represented by a specific **combination** of **signal phase** and **amplitude** that is kept during the **symbol interval**. The **quantity** of **symbols** in the **QAM constellation** is the quantity of combinations of **phase** and **amplitude**. The **quantity of symbols** "**n**" identifies each **QAM variant** and it is used as the prefix of the **variant name** "**nQAM**" or "**n-QAM**".

  The **symbols** of **each constellation** are **evenly spaced** over a **symbol grid**. The following figure shows the **symbol constellations** for **4QAM**, **16QAM**, **32QAM** and **64QAM**:




The modulations **GMSK**, **QPSK** and **8PSK** have the advantage of **not** requiring amplitude modulation. This allows the usage of low-cost **type C amplifiers** which are **very non-linear** and do **not** allow for **amplitude modulation**, which prevents using **QAM**.

**Digital modulations** are sometimes called "**line codes**" or "**line coding**". However, this term is somehow undesirable because it is prone to confusion with "**channel coding**", that is usually shortened to just "**coding**".

Regarding **LTE**, **512QAM** and **1024QAM** were introduced in the latest versions of **LTE Advanced Pro**.

For more detail on **5G modulations**, refer to clause 5.1 of TS 38.211.

Source web pages:

- **PSK**: https://en.wikipedia.org/wiki/Phase-shift_keying
- **QAM**: https://en.wikipedia.org/wiki/Quadrature_amplitude_modulation
- **GMSK**: https://en.wikipedia.org/wiki/Minimum-shift_keying

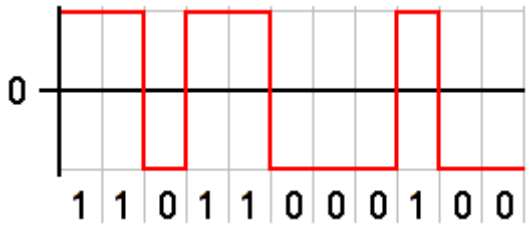
### 11.2.2 Line codes

A **line code** is a technique to **convey digital data** as a **sequence** of **baseband signal shapes**.

Some **frequently used line codes** are:

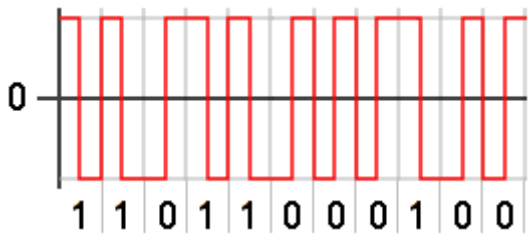
- **Polar non-return-to-zero code**

  It uses a **positive pulse** and a **negative pulse** to represent a **0** or a **1**. A **disadvantage** of this code is that it carries a **DC component** when the **quantity** of **0**'s and **1**'s is **not** the **same**. The following figure illustrates this code where the **positive pulse** represents a **1** and the **negative pulse** represents a **0**:




- **Manchester code**

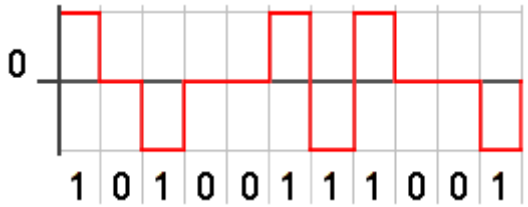
  It uses a **positive-to-negative transition** and a **negative-to-positive transition** to represent a **0** or a **1**. An **advantage** of this code is that it **never** carries a **DC component**. The following figure illustrates this code where the **positive-to-negative transition** represents a **1** and the **negative-to-positive transition** represents a **0**:




- **Bipolar code**

  It uses a **zero pulse** and an **alternating positive** and **negative pulse** to represent a **0** or a **1**. An **advantage** of this code is that it **never** carries a **DC component**.

The following figure illustrates this code where **zero pulse** represents a **0** and the **alternating positive** and **negative pulse** represents a **1**:




Do **not confuse** a **channel code** (click 11.2.3) with a **line code** (click 11.2.2).

Source web page:

- https://en.wikipedia.org/wiki/Line_code

### 11.2.3 Channel codes

A **channel code** is a technique to **add redundant information** to a **bit-block** in order to **protect** the **bit-block** from **errors arising** from the **communication**. **Protecting** from **errors** refers to **detecting** the **errors** and/or **correcting** the **errors**.

**Channel codes** can be divided into **error detecting codes** and **error correcting codes**. The **error correcting codes** may **also** have **error detecting capacity**.

An extremely used **error detecting code** is the **cyclic redundancy check** (**CRC**, click 11.2.3.2).

A few **very relevant error correcting codes** are:

- **Linear-density parity-check** (**LDPC**) codes (click 11.2.3.3).
- **Polar** codes (click 11.2.3.4).
- **Block** codes (click 11.2.3.5).
- **Turbo** codes.

The **CRC**, **LDPC**, **polar** and **block codes** are used in **5G**.

The **turbo codes** were used in **LTE** and **UMTS**. They are **not used** in **5G** for the following reasons. For **long bit-blocks LDPC codes outperform turbo codes** in terms of **throughput** and **error-correction capability**. **Turbo code decoders** have **higher decoding latency** and **power consumption** than **LDPC decoders**, especially for **long bit-blocks**. Notice that **5G** typically uses **longer bit-blocks** than **LTE**. Finally, **LDPC** and **polar codes** offer **better adaptability** to **variable block sizes** and **code rates**, which are characteristic of **5G**.

Do **not confuse** a **channel code** (click 11.2.3) with a **line code** (click 11.2.2).

For more detail on **5G channel codes**, refer to clause 5.3 of TS 38.212.

Source web pages:

- https://www.nrexplained.com/modulation

https://www.rfwireless-world.com/5G/5G-NR-Channel-Coding.html#:~:text=5G%20NR%20Channel%20Coding%20%7C%20Coding%20techniques%20for%205G%20NR%20Channels,-This%20page%20describes&text=Polar%20coding%20is%20used%20for,used%20for%20PDSCH%20and%20PUSCH

#### 11.2.3.1 Code rate

The **code rate** of a **channel code** is a **measure** of how many **redundant bits** are inserted by the **code**, in respect to the **data bits**. The **larger** the **code rate**, the **fewer redundant bits** are **added**. The **code rate** is defined as:

$$code_rate = data_size / transmitted_size = data_size / (data_size + code_size)$$

**Channel code rates** are **always lesser** than **1** (a **code rate** of **1** implies there is **no code**). The ***code_size*** in respect to ***data_size*** can be expressed as:

$$code_size = data_size \bullet (1/code_rate - 1)$$

A **code rate** of **1/2** implies that ***code_size*** is the ***same*** as ***data_size***. A code rate of **1/3** implies that there is **double** ***code_size*** than ***data_size***.

#### 11.2.3.2 Cyclic redundancy check (CRC) codes

**Each CRC code** is an **error detecting** code that consists of a **fixed-size bit sequence** calculated from an **any-sized data block** using a **generating polynomial**.

**Each CRC code** can be characterized by its **bit length** and by its **generating polynomial**.

The simplest **CRC code** is a **1-bit parity check**: if the **quantity of 0**'s is **odd**, you **add** a **1** (the **CRC**), and otherwise you **add** a **0** (the opposite works the same). This **extremely simple code** detects **any erroneous** received **protected block** such that its **quantity** of erroneous bits (irrespective of where they are within the **protected block**) is **odd**. You may think of a **CRC code** as **adding more parity bits**, such that **each parity bit** is computed over a **subset** of the **bit-block**, where the **bits** belonging to the **each subset** are **interleaved** among **themselves** and there is a "**smart**" **subset selection**.

For more detail on **5G CRC codes**, refer to clause 5.1 of TS 38.212.

#### 11.2.3.3 Linear-density parity-check (LDPC) codes

**Low-density parity-check** (**LDPC**) **codes** are **error correcting** codes that are computed in three steps:

1. **Data block segmentation**

   If the **data block** to be protected exceeds a **certain size**, it is **segmented** into so-called **code blocks**, where the **size** of **each code block** is **below** (or equal to) that **certain size**.

2. **Code block encoding**

   **Each code block** is **encoded** using **LDPC** into **1** so-called **encoded block**. The **encoded block** is a **few times larger** than the **code block**, because of the addition of a considerable amount of **LDPC code bits**.

3. **Encoded block rate-matching**

   **Reduce** or **increase** the **encoded block size** to **match** the **available transmission size**. **Reducing** the **size degrades** the **error-correcting capabilities**, while **increasing** it **upgrades** the **error-correcting capabilities**.

   **Reducing** the **size** is done by **removing** ("**puncturing**") some **code bits**, while **increasing** the **size** is done by **repeating** some **code bits**. Increasing the size can be done by **bit repetition** in a **cyclic** way, **or** with **more complex** techniques.

For more detail on **5G LDPC codes**, click 4.12.5 and refer to clause 5.3.2 of TS 38.212.

#### 11.2.3.4 Polar codes

**Polar codes** are **error correcting** codes that consist of **diving** ("polarizing") a **set** of **physical channels** into **reliable** (low noise) **channels** and **unreliable** (high noise) **channels**. **Data bits** are sent over the **reliable channels**, while **control bits** known to **both transmitter** and **receiver** are sent over the **unreliable channels** to **assist** in **decoding**.

**Polar codes** are constructed using a **recursive process** based on a **generator matrix** derived from the **Kronecker power** of a **basic kernel matrix**:

$$G_N = B_N \bullet F^{\otimes n}$$

where:

- "$N = 2^n$" is the **block length**.
- "$F$" is the **kernel matrix**:

$$\begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$$

- "$F^{\otimes n}$" is the **n-th Kronecker power** of "$F$".
- "$B_N$" is a **bit-reversal permutation matrix** to **reorder** the **indices** for **efficient encoding**.

**Polar codes** achieve the **theoretical maximum capacity** of **binary-input symmetric memoryless channels** (**BSCs**) and are the **first codes proven** to do so using a process called **channel polarization**. **Polar codes** are **simple** and **efficient**.

**Polar codes** are used in **5G** for **control channel encoding**.

For more detail on **5G polar codes**, refer to clause 5.3.1 of TS 38.212.

#### 11.2.3.5 Block codes

**Block codes** are **error correcting** codes that encode **data blocks** of **fixed size** and **add** a **fixed quantity** of **redundant bits**.

**Block codes** can be **classified** according to the way they **generate** the **codewords** into:

- **Linear block codes**: they generate the **codewords** as linear combinations of basis vectors.
- **Non-linear block codes**: they generate the **codewords** using **non-linear** algorithms.

The **linear block codes** are **most frequently** used. A few **very common block codes** are:

- **Reed-Muller codes**: can **correct multiple errors**. A (**32**, **k**) **Reed-Muller code** is used to **encode** the **code blocks** of **5G Uplink Control Information** (**UCI**) with a **bitlength** from **3** to **11 bits**. For more detail on this, refer to clause 5.3.3 of TS 38.212.
- **Hamming codes**: can **detect up to 2 errors** and **correct 1 error**.
- **Bose–Chaudhuri–Hocquenghem** (**BCH**) **codes**: can **correct multiple errors**. Commonly used in **telecommunications** and in **storage systems**.

- **Reed-Solomon Codes**: can **correct multiple burst errors**. They are used in **QR codes**, **DVDs** and **CDs**.
- The very simple **repetition codes** are used to **encode 1-bit UCI messages** sent through **PUSCH** (click 7.8.2.3.2).
- The simple **simplex codes** are used to **encode 2-bit UCI messages** sent through **PUSCH** (click 7.8.2.3.3).

For more detail on **5G Reed-Muller codes**, refer to clause 5.3.3.3 of TS 38.212.

### 11.2.4 Modulation and coding scheme (MCS)

**Each modulation and coding scheme** (**MCS**) is a combination of a **digital modulation** (click 11.2.1) and a **channel code** (click 11.2.3).

There is a performance **tradeoff** between **modulation** and **coding**. Using a **higher modulation order** (i.e., more bits per symbol) implies a **higher bitrate** (click 11.1) but it also increases the **error probability**, which requires a **more robust channel coding**, which implies more **coding redundancy**, which reduces the **ratio** of **throughput** (click 11.1) over **bitrate**. The **resulting throughput** may be **lower** (or **higher**) than the one achieved with a **lower modulation order** and **less coding redundancy**.

Consequently, the **digital modulation** and the **channel coding** to be used at any given moment are jointly **selected**, so they are **well coupled** in order to maximize the **throughput**, in respect to the **UE capabilities** and the **current channel conditions**.

**5G** uses a **variety** of **digital modulations** and **channel codings** to better **adapt** to **each communication scenario**, to the **capabilities** of **each UE** and to the **constantly changing channel conditions**.

**Each 5G physical channel** and **signal** (click 4.11.3) uses its **specific set** of **modulation and coding schemes**. The **physical channels PBCH**, **PDCCH**, **PUCCH** and **PRACH** (i.e., the ones other than the shared channels) and the **physical signals** use a **small set** of **modulation and coding schemes**, which are either **fixed** or determined by **configuration**. However, **PDSCH** and **PUSCH** (i.e., the shared channels) use a quite **large set** of **modulation and coding schemes**, and what is more important, **several** are used at the **same time**, and they change in an **extremely dynamic** way. A useful summary of the **modulation** and **coding schemes** used by **each physical channel** and by **each physical signal** is provided in the tables from 4.11.3.

Since **PDSCH** and **PUSCH** can use **several MCS simultaneously**, and they **change** very **rapidly**, it is required to indicate, case by case, **which MCS** is being **used**, in order for the **receiver** to correctly **demodulate** and **decode**. **Each transport block** (click 4.12.1.1) is sent in **1 codeword** (click 4.12.1.2) using **only 1 MCS**. **Each codeword** is sent in **1 assignment** of **Resource Elements** (**REs**, click 1.1.3). The **MCS** used for **each codeword** is conveyed **together** with the information about the **location** of the **REs** that contain the **codeword**. This is information is **conveyed** with **Downlink Control Information** (**DCI**, click 6.9) **messages**.

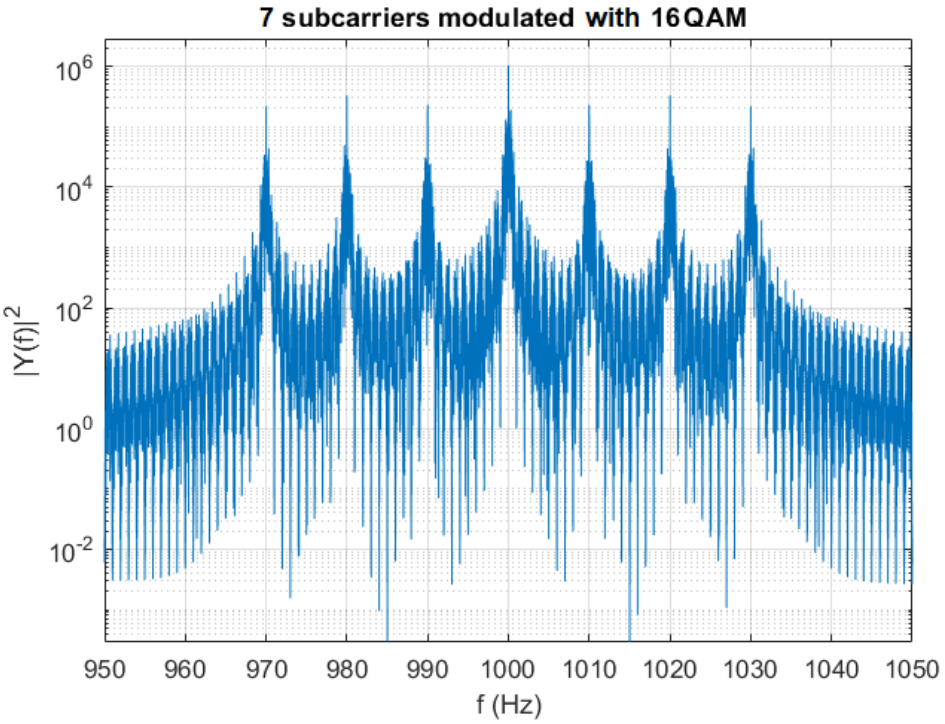
## 11.3 Single-carrier signals and multicarrier signals

Depending on the **quantity** of **carriers** in the **signal**, there are **single-carrier signals** (click 11.3.1) and **multicarrier signals** (click 11.3.2).

### 11.3.1 Single-carrier signals

A **single-carrier signal** (**SC signal**) uses **only one base tone** (i.e., **one carrier**) to convey the **data**. The **entire data stream** is **modulated** (click 11.2.1) onto this **single base tone**.

In the **frequency-domain**, the **power spectral density** (**PSD**) of the **SC signal** has a **single**, **continuous** band of **frequencies** around the **carrier frequency**. The following figure shows the **PSD** of a **single carrier 16QAM signal**:




Some **relevant remarks** on this figure are:

- The **central power peak** at **1 kHz** corresponds to the **carrier frequency**.
- The **signal bandwidth** is roughly **10 Hz**.

In the **time-domain**, the **SC-signal waveform** is as a **continuous modulation** of the **carrier**. Except in **frequency modulation**, the **base frequency** of the **time signal** is the one of the **carrier**.

Two examples of **analog single-carrier signals** are the **AM** and **FM** commercial radio transmission. An example of **digital single-carrier signal** is the **GSM transmission**.

**Advantages of single-carrier signals**

- **Simpler transmitter** and **receiver**, especially in legacy systems.
- **Lower sensitivity** to **frequency alignment errors** than **multicarrier signals**.
- **Less spectral fragmentation**, which can be advantageous in certain regulatory environments.

**Disadvantages of single-carrier signals**

- **More complex equalization** is required to mitigate **inter-symbol interference** (**ISI**) caused by **multipath fading** (click 12.2.7) in frequency-selective channels.

- **Low peak-to-average power ratio** (**PAPR**, click 11.4). Although **SC signals** generally have **lower PAPR than multicarrier signals**, they can still exhibit some **power peaks** due to channel conditions.

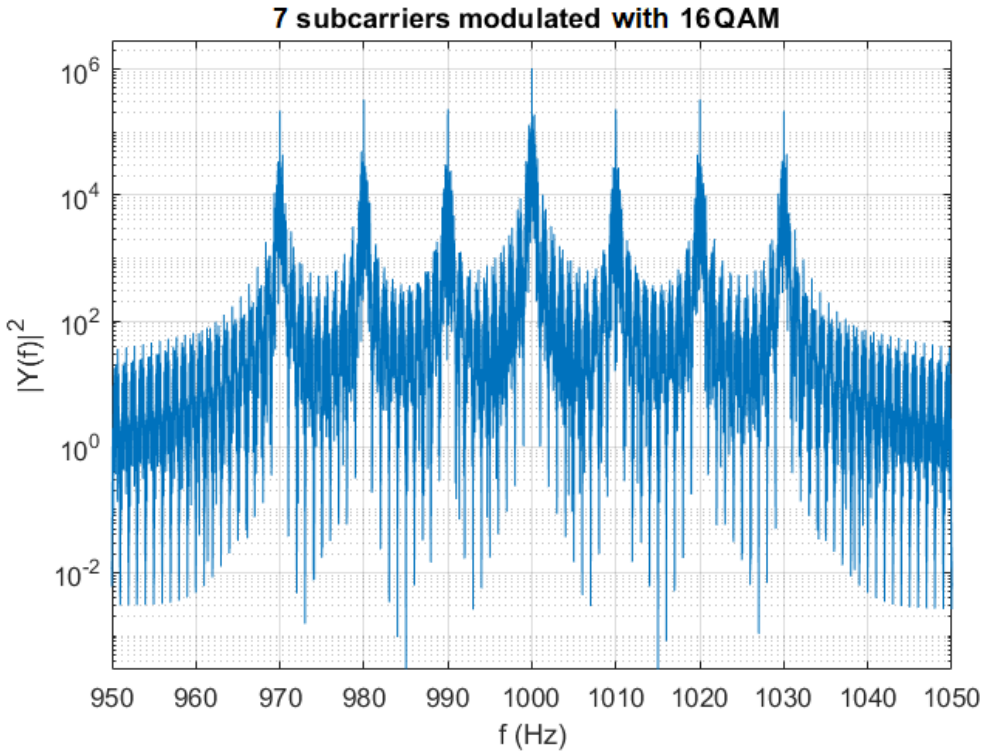
In **summary**, systems using **single-carrier signals** are **simpler** and **more power-efficient** (in most cases) but require **complex equalization** in **frequency-selective channels**.

### 11.3.2 Multicarrier signals

A **multicarrier signal** (**MC signal**) uses **multiple base tones** (i.e., **subcarriers**) to convey the **data**. **Each** among **several substreams** of the **overall data stream** is **modulated** (click 11.2.1) onto **each** of the **subcarriers**. On most systems, the **subcarriers** are **evenly spaced**, with a **spacing considerably lower** than the **overall bandwidth**. **Subcarriers** are **typically orthogonal** among themselves to **avoid interference**.

The **subcarrier spacing** is an important parameter in the **design** of **systems** using **multicarrier signals**. The **quantity** of **subcarriers** in the **system** is the available **bandwidth** divided by the **subcarrier spacing**.

In the **frequency-domain**, the **power spectral density** (**PSD**) of a **multicarrier signal** has **multiple narrow peaks**, **each** corresponding to **each** of the **subcarriers**. The following figure shows the **PSD** of **seven subcarriers**, **each** modulated with **16QAM**:




Some **relevant remarks** on this figure are:

- The **7 power peaks** at **970**, **980**, **990**, **1,000**, **1,010**, **1,020** and **1,030 Hz** correspond to the **central frequencies** of the **7 subcarriers**.
- The signal is centered at **1 kHz**.
- The **subcarrier spacing** is **10 Hz**.

- The **signal bandwidth** is roughly **70 Hz**.

In the **time-domain**, the **waveform** of a **multicarrier signal** is **quite complex**, since it is a combination of **multiple modulated subcarriers**, possibly with **different modulations**.

Some examples of **digital multicarrier signals** are **OFDM** (click 11.14), **OFDMA** (click 11.15) and **SC-FDMA** (click 11.16), which are used in **WiFi**, in **LTE** and in **5G**. Another example is the **DMT signal** used in **broadband DSL**.

**Advantages of multicarrier signals**

- **Highly robust** in respect to **frequency-selective fading** because **equalization** may be **easily performed** per **subcarrier**.
- **Subcarriers** can be **dynamically allocated** in a **very flexible way**, thus allowing to optimize spectrum utilization.
- Work well over a **wide range** of bandwidths, by **just adjusting** the **quantity** of **subcarriers**.

**Disadvantages of multicarrier signals**

- **Higy peak-to-average power ratio** (**PAPR**, click 11.4). This implies **expensive** and **power-inefficient amplifiers**.
- Require **very precise time synchronization** and **frequency alignment** to achieve **orthogonality** between **subcarriers**. This implies **precise oscillators** and **filters** and a **very precise time-source**, which can be **satellite** (e.g., **GPS**) or **terrestrial** (e.g., **PTP**).

In **summary**, systems using **multicarrier signals** are **very flexible**, adapting well to **many channel types**, are **robust** to **frequency-selective fading** (click 12.2.7) and/or **multipath** (click 12.2.3) **channels**, at the cost of **higher complexity**, **higher PAPR** and **higher Doppler sensitivity**.

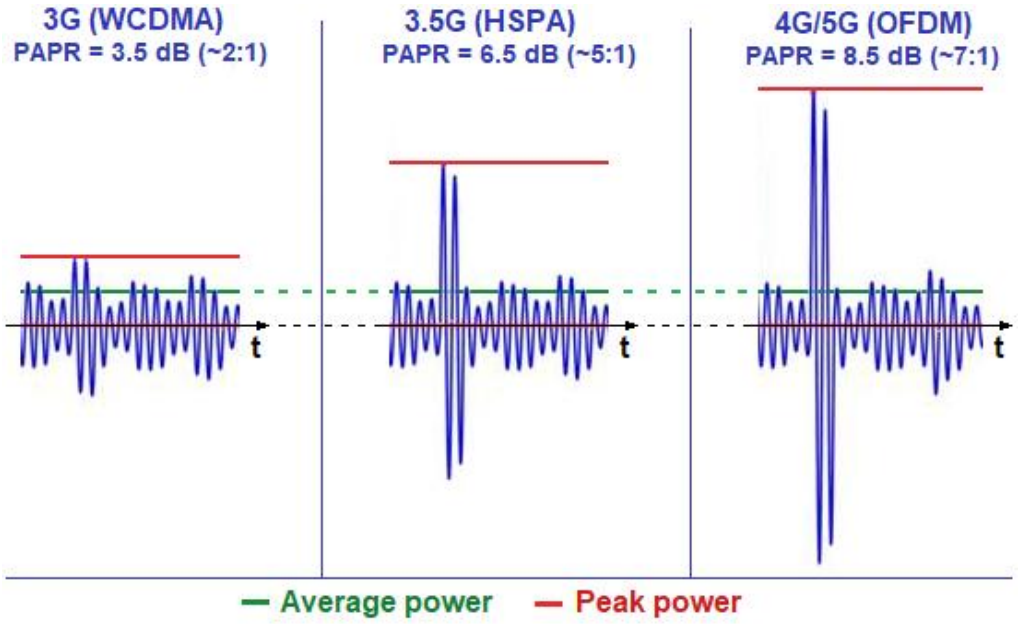
**Multicarrier systems** sometimes use the term "**carrier**" in a somehow **specific way** (click 2.2.5.1).

## 11.4 Peak-to-average power ratio (PAPR)

The **peak-to-average power ratio** (**PAPR**) of a **signal** is its **peak power** divided by its **average power**. It is usually expressed **either** in **dB** or as a **ratio**. The **peak-to-average power ratio** is most frequently computed, on average, for a **family** of **signals**, such as **BPSK**, **16QAM**, **OFDM** or **SC-FDMA**.

The **peak-to-average power ratio** is an important characteristic because a **high PAPR** implies **highly-linear amplifiers** which are usually **more expensive** and **less power efficient**.

The following figure illustrates the **PAPR** for **signal families** used in **3G**, **3.5G** and **4G/5G**. For the case of **4G/5G** it refers to the **Downlink** signal:




The **PAPR** is sometimes referred as "**crest factor**".

Source web page:

- http://ael.chungbuk.ac.kr/lectures/undergraduate/microwave-engineering/2020-1/Lecture-notes/10-multiple-access/L10a-OFDM.htm

## 11.5 Duplexing

### 11.5.1 Duplexing overview

**Duplexing** is the ability to **both transmit** and **receive** over a **given frequency segment** and a **given space**.

**Duplexing** is intrinsically **bidirectional**. **Duplexing** is applied to **point-to-point**, **point-to-multipoint**, **multipoint-to-point** and **multipoint-to-multipoint communications**, as long as **two-way communication** is required. **Duplexing** is **not** required in **broadcast-only** systems (e.g., TV and radio broadcasting).

Some basic **duplexing** techniques are:

- **Frequency-division duplexing** (**FDD**)

  **FDD** consists of **simultaneous transmission** and **reception**, **each** in an exclusive **frequency sub-segment** located within the given **frequency segment**. The **transmission** and the **reception sub-segments must** be separated with a **guardband** in order to **prevent self-interference** (click 12.1.14). **FDD** usually uses **1 transmission sub-segment** and **1 reception sub-segment**. Since **transmission** and **reception** are **simultaneous**, **FDD provides full-duplex communication**.

  For a comparison between **FDD** and **TDD**, click 11.5.3.

  For the specific **frequency-division duplexing** used in **5G**, click 2.2.12.

- **Time-division duplexing** (**TDD**)

  **TDD** consists of **performing transmission** and **reception** in **alternating time slots**, using the **whole** available **frequency segment**. The **transmission** and the **reception slots must** be separated by a **guard period** (click 11.5.2). Since **transmission** and **reception alternate**, **TDD provides half-duplex communication**.

  For a comparison between **FDD** and **TDD**, click 11.5.3.

  For the specific **time-division duplexing** used in **5G**, click 2.3.4.

- **Space-division duplexing**

  **Space-division duplexing** consists of **simultaneous transmission** and **reception** using the **whole frequency segment**, using **two pairs** of **transmit** and **receive** devices in **different spatial locations**, such that **cross-interference** between them is **sufficiently low**.

  In **wired** communications, **space-division duplexing** is done by **laying a cable pair** (copper pair, FO, coax, ..), **one** for **transmission** and the **other** for **reception**. In **wireless** communications, **space duplexing** is done by implementing a **pair** of **independent radio channels**, **one** for **transmission** and the **other** for **reception**. For example, installing **two radio links** between the **same two endpoints** and in the **same frequency** but that **do not cross interfere** because the corresponding **Tx/Rx antennas** are **sufficiently spaced away** in respect to the **wavelength**, to the **antenna radiation patterns** and to the **link distance**.

  **Space-division duplexing** is sometimes called "**spatial duplexing**", "**space-division duplexing**" or "**space duplexing**".

  Since **transmission** and **reception** and **simultaneous**, **space-division duplexing achieves full-duplex communication**.

- **Self-interference cancellation** (**SIC**)

  It is a **signal processing technique** that enables a **radio transceiver** to **simultaneously transmit** and **receive** on a **single frequency segment**, on a **pair** of **partially overlapping frequency segments**, or on **any pair** of **frequency segments** in the same frequency band. When used to allow **simultaneous transmission and reception** on the **same frequency segments**, sometimes referred to as "in-band **full-duplex**" or "simultaneous transmit and receive". **SIC** effectively **doubles spectral efficiency** over **FDD** or **TDD**. SIC is expensive to implement and requires very precise radio frequency components. This is why it is **not** implemented in the **UEs**, and consequently it is **not used** in the communication between the **NR node** and **UEs**. However, **SIC** may be used for **Integrated Access and Backhaul** (**IAB**).

  Since **transmission** and **reception** and **simultaneous**, **self-interference cancellation achieves full-duplex communication**.

  Source web page:

  - https://en.wikipedia.org/wiki/Self-interference_cancellation

- **Electro-impedance duplexing**

  It consists of separating the **transmitted** and **received signals** based on **impedance matching** and **electrical isolation**.

  This **duplexing technique** was used in old **analog telephony** by means of a **hybrid coil** (sometimes called "**hybrid transformer**"). The **hybrid coil** performed the **conversion** to/from **4-wire** transmission (**two simplex communications**) from/to **2-wire** transmission (**full duplex communication**).

  In the **analog telephone handset**, the **microphone** and the **speaker** used **2-wires each**, resulting in a **4-wire** transmission. The **telephone exchanges** and **trunk lines** used **separate pairs** or **channels** for **transmission** and for **reception**, resulting in **4-wire** transmission. However, the **local loop** was a **2-wire twisted copper pair** used for **simultaneous transmission** and **reception** in the **same frequency segment** (300 to 3,400 Hz). The **hybrid coil** at **each end** of the **copper pair** performed the **conversion** to/from **4-wire** from/to **2-wire** transmission.

**Duplexing** (click 11.3), **multiplexing** (click 11.6), **inverse multiplexing** (click 11.7), **diversity** (click 11.8) and **multiple access** (click 11.9) are **different**, but **related**, concepts. For their **relationships**, click 11.10.

### 11.5.2 TDD guard periods

The **TDD guard period** is the **time** that a **TDD system must wait** between the **end** of **reception** and the **beginning** of **transmission**, and **vice versa**. There are **two TDD guard periods**, **one** for the **Tx-to-Rx transition** and the **other** for the **Rx-to-Tx transition**. These **two guard periods must** be observed by **all** the **systems** that are communicating, and their **value** may be **different** at **different systems**.

For the case of **two communicating systems A** and **B**, the **minimum Tx-to-Rx$_A$ guard period** of "**A**" is the **greater** between its **Tx-to-Rx$_A$ transceiver switch time** and both **propagation delays** plus the **Rx-to-Tx$_B$ transceiver switch time** of **B**.

$$Tx\text{-}to\text{-}Rx_A_guard_period_{min} = max(Tx\text{-}to\text{-}Rx_A, Rx\text{-}to\text{-}Tx_B + PD_A + PD_B)$$

The **minimum Rx-to-Tx$_A$ guard period** of "**A**" is the **greater** between the **Tx-to-Rx$_B$ transceiver switch time** of "**B**" and **both propagation delays** plus the **Rx-to-Tx$_A$ transceiver switch time** of "**A**".

$$Rx\text{-}to\text{-}Tx_A_guard_period_{min} = max(Rx\text{-}to\text{-}Tx_A, Tx\text{-}to\text{-}Rx_B - PD_A - PD_B)$$

From these equations it follows that:

$$Tx\text{-}to\text{-}Rx_A_guard_period_{min} + Rx\text{-}to\text{-}Tx_A_guard_period_{min}$$

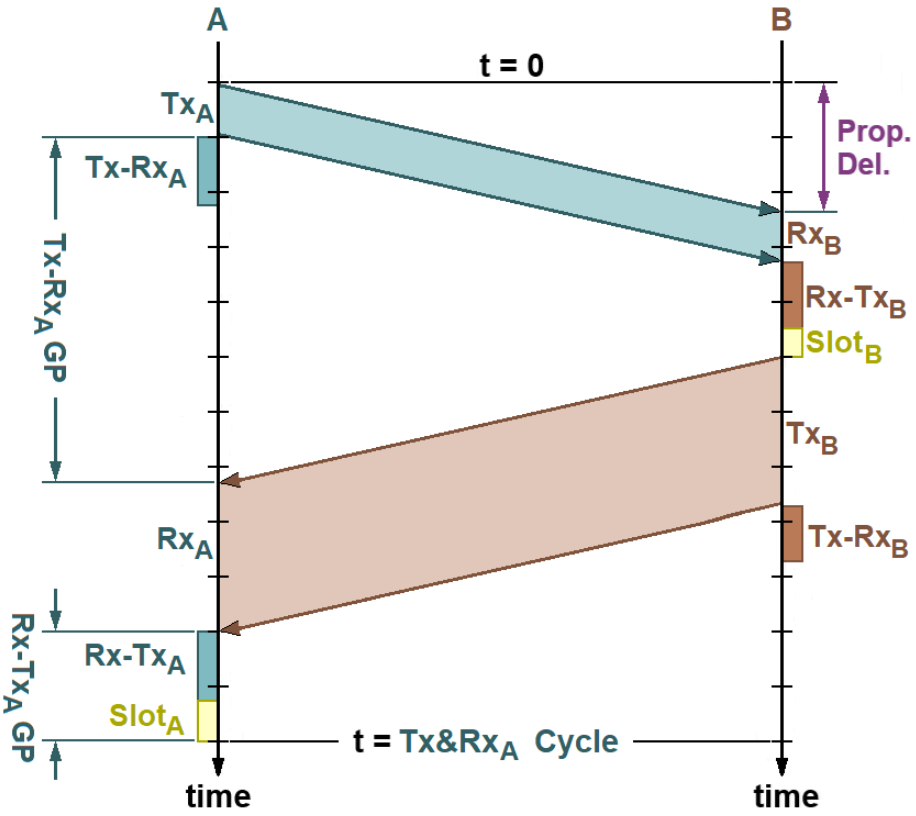
$$= Tx\text{-}to\text{-}Rx_B_guard_period_{min} + Rx\text{-}to\text{-}Tx_B_guard_period_{min}$$

Many communication systems have **slotted transmission**. This implies that **transmission cannot start** at **any moment**, and it can rather **only start** at the **beginning** of a **periodic slot**. The definition of "**slot**" is **system dependent**, and it usually is a **certain quantity** of **symbol intervals**. Do **not confuse** this "**slot**" with the "**slot**" of **5G**, which are **different** (although **related**) **concepts**. For slotted systems, it is **also** required to **wait** until the **beginning** of the **next slot**. In this case, the **slot waiting time** has to be **considered** in the **TDD guard periods**, resulting in:

$$Tx\text{-}to\text{-}Rx_A_guard_period_{min} = max(Tx\text{-}to\text{-}Rx_A, Rx\text{-}to\text{-}Tx_B + PD_A + PD_B + Slot_B)$$

$$Rx\text{-}to\text{-}Tx_A_guard_period_{min} = max(Rx\text{-}to\text{-}Tx_A + Slot_A, Tx\text{-}to\text{-}Rx_B - PD_A - PD_B)$$

The following figure illustrates the **Tx-to-Rx$_A$ guard period** and the **Rx-to-Tx$_A$ guard period** corresponding to system "**A**":




Some relevant remarks on this figure are:

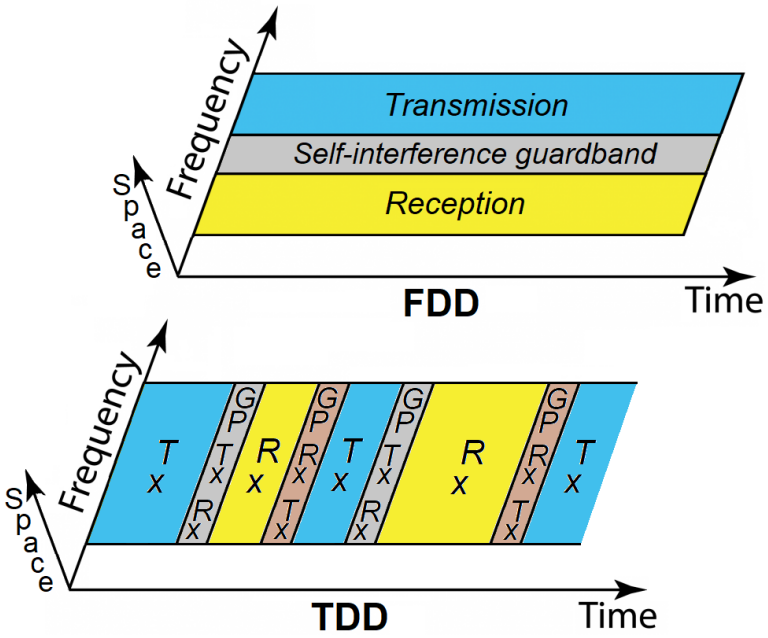
- **Tx-to-Rx** and **Rx-to-Tx** have been shortened to **Tx-Rx** and **Rx-Tx** (respectively).
- **Time** advances **top** to **bottom**: the **top** is the **older events** and the **bottom** the **most recent events**.
- **Propagation delay** is assumed to be **symmetric**, as it is **almost always** the case for terrestrial systems.
- In this example the **transceiver switch time Tx-to-Rx$_A$** is less than the **addition** of **PD$_A$**, **PD$_B$**, **Rx-to-TxB** and **Slot$_B$**. However, this is **not always** the case.
- The figure shows the **start** (shown as "t=0") and **end** (shown as "t= Tx&Rx$_A$ Cycle") instants of a complete **transmission** and **reception cycle** of **A**. **Each Tx&Rx$_A$ cycle starts** right at the **end** of the **previous cycle**.
- We suggest that you check the **guard period formulas** indicated **above** in respect to this figure.

Recall from 11.5.1 that **TDD** may be used for the **four basic communication types**. This implies that the **propagation delay** between the **pairs of systems** involved in the **communication** will most likely be **different**. In this case, it is possible to configure the **TDD minimum guard periods** for the **worst case** of **all** the **pairwise combinations** of systems. For a figure showing **TDD** in a **one-to-many communication**, click 2.3.4.3.

For a comparison between **FDD** and **TDD**, click 11.5.3.

### 11.5.3 Comparison between FDD and TDD

The following figure illustrates the way **FDD** and **TDD** work:




Some **relevant remarks** on this figure are:

- In **FDD**, the **resource overhead** arises from the **self-interference guardband**, while in **TDD** it arises from the **guard periods** (labeled "***GP***"). The **Tx-to-Rx guard period** is colored **gray**, while the **Rx-to-Tx guard period** is colored **brown**. For more detail on the **TDD guard period** and **resource overhead**, click 11.5.2.
- Although in the figure the **Tx-to-Rx guard period** and the **Rx-to-Tx guard period** are **roughly** the **same**, in the **general case** they **may be very different**.
- In **FDD**, **transmission** and **reception** are **simultaneous**, while in **TDD transmission alternates** with **reception**.
- In **FDD**, the **transmission** and **reception** resources (**Tx** and Rx **frequency segments**) are **fixed**, while in **TDD each individual period** of **transmission** or **reception** may have a **variable duration**.
- **Neither FDD nor TDD** use the **space dimension**. However, it is shown in the figure to remind that it is also possible to use **space** for **duplexing**.

**Implementation difficulties and cost**

The **filters** required to **prevent self-interference** (click 12.1.14) in **FDD** are **very difficult** (and **very expensive**) to implement and integrate at **high frequency**.

**Efficiency**

The **FDD overhead** is:

$$Overhead_{FDD} = guardband \,/\, (Tx + Rx)\ \%$$

Although the **FDD guardband** is **always** required to prevent **self-interference**, it can be used by **other systems** that are **not** involved in this communication. If this is the case, the

**overhead** can be considered **zero** because the **self-interference guardband** is actually **used**.

The **TDD overhead** is:

$$Overhead_{TDD} = guard_periods \,/\, (Tx + Rx)\ \%$$

Where ***guard_periods***, are the **two TTD guard periods** (click 11.5.2) of the communications systems, ***Tx*** is the **transmission period** and ***Rx*** is the **reception period**.

Consequently, **TDD** is **increasingly inefficient** as **propagation delay increases**, because the **expression** for the **TDD guard periods** (click 11.5.2) includes **twice** the **propagation delay**. This **inefficiency** can be **mitigated** by using a **Tx+Rx slot much larger** than the **propagation delay**. However, doing this **increases considerably** the **end-to-end delay**, which for many applications (conversational, real time, …) is **unacceptable**.

**Flexible transmission and reception allocation**

**FDD** is quite **rigid** because the transmission and reception **frequency segments** are **fixed** due to the difficulties in building **tunable filters** and **amplifiers** that can cope with the **self-interference** problems.

**FDD** is **very flexible** because it can allocate the **required period** to **transmission** and **reception**, and to **quickly** and **easily change** it in order to satisfy the **changing traffic** demands.

**Massive MIMO usage**

**FDD** is **not** very suitable for **massive MIMO** because it requires **independent channel estimations** for **Downlink** and **Uplink**.

**TDD** is **very suitable** for **massive MIMO** because the **Uplink channel estimation** may also be used for **Downlink**, and this **significantly reduces** the required **MIMO processing**.

**FDD and TDD in 5G**

In **low frequency 5G bands** (**FR1-1**, click 2.2.13.1) **FDD** is **mainly** used because it is **cheaper** to **implement** and it has **better spectral efficiency** than **TDD**. Notice that **low frequency bands** have a **larger reach**, which implies a **higher propagation delay**, which **increases** the **overhead** of **TDD**, thus making it less attractive than **FDD**. Also, in **low frequency 5G bands massive MIMO** is **not** so **frequently used** because of the **larger wavelengths** (roughly between 11.5 m and 50 cm).

In **high frequency 5G bands** (**FR1-2** (click 2.2.13.2), **FR2-1** (click 2.2.13.3) and **FR2-2** (click 2.2.13.4)) **only TDD** is used because the **required FDD filters** are very **complex** (**expensive**), while **TDD** is **much simpler** to implement. Notice that **high frequency bands** have a **smaller reach**, which implies a **smaller propagation delay**, which **increases** the **efficiency** of **TDD**. Also, in **high frequency 5G bands massive MIMO** is **frequently used** because of the **smaller wavelengths** (roughly between 8.6 cm and 5 mm).

## 11.6 Multiplexing and demultiplexing

**Digital multiplexing** (**Mux**) consists of **transmitting** several **individual bitstreams** into **one joint larger bitstream** such that **each** and **every individual bitstream** can be **recovered** in the destination.

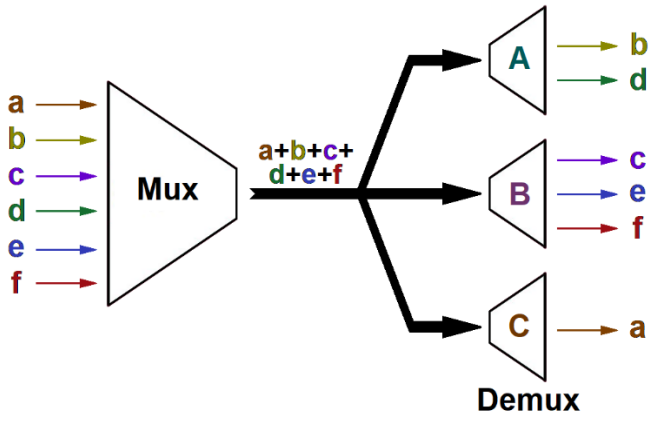
**Digital demultiplexing** (**Demux**) consists of **recovering each** and **every individual bitstream** that was **multiplexed** by the **transmitter**, from the **reception** of the **joint bitstream** (or **frequency segment**).

**Analog multiplexing** and **demultiplexing** is the **same** as their **digital counterparts** but **replacing** "**bitstream**" by a "**signal** within a given **frequency segment**".

**Multiplexing** and **Demultiplexing** are **unidirectional**. **Multiplexing** is performed by the **transmitter**. **Demultiplexing** is performed by the **receiver(s)**. **Multiplexing** and **demultiplexing** are **reciprocal** functions, and they **both must** use the **same technique** for a **correct operation**.

**Mux/Demux** are defined for **only one transmitter**, but there may be **one or more receivers**. Therefore, **Mux/Demux** may be performed in **point-to-point** and **point-to-multipoint** communications. It **cannot** be performed in **multipoint-to-point nor** in **multipoint-to-multipoint** communications. For **several transmitters**, refer to **multiple access techniques** (click 11.9).

The following figure illustrates **multiplexing** and **demultiplexing**:




Some **relevant remarks** on this figure are:

- This example is for **3 receivers** (**Demuxes**), which are labeled "**A**", "**B**" and "**C**". However, in many cases there is only **1 receiver**.
- There are **6 individual bitstreams**, labeled "**a**" to "**f**", and identified by different colors.
- The **6 individual bitstreams** may have **different bitrates**.
- The **addition** of the **throughput** of the **6 individual bitstreams** is below, but **close to**, the **throughput** of the **joint bitstream**.
- **All receivers** receive the **same joint bitstream**.
- **Each receiver demultiplexes** its **corresponding individual bitstreams**. In this example **bitstreams** "**b**" and "**d**" correspond to **receiver "A"**, **bitstreams** "**c**", "**e**" and "**d**" correspond to **receiver "B"** and **bitstream a** corresponds to **receiver** "**C**".

Some basic **digital Mux/Demux** techniques are described in the following subsections. It is also possible to **combine** these **basic multiplexing techniques** in different ways. Such is the case of **5G**, that in **Downlink** combines **frequency-division multiplexing** (**OFDM**, click 11.14) with **time-division multiplexing** (click 11.6.2), and it may also **use space-division multiplexing** for **multi-user MIMO** (click 2.4.3).

**Duplexing** (click 11.3), **multiplexing** (click 11.6), **inverse multiplexing** (click 11.7), **diversity** (click 11.8) and **multiple access** (click 11.9) are **different**, but **related**, concepts. For their **relationships**, click 11.10.

### 11.6.1 Frequency-division multiplexing

The **total transmission bandwidth** of the **joint channel** is divided into **several non-overlapping smaller frequency ranges** and **each individual bitstream** is transmitted using **one specific smaller frequency range**.

**Each** of the **one** or **more receiver**(**s**) can receive (**Demux**) its corresponding **individual bitstream**(**s**) by **selecting** the **corresponding specific smaller frequency range**(**s**).

For example, a **5G carrier** is divided in **non-overlapping subcarriers**, such that **each subcarrier** (or **group** of **subcarriers**) is used to transmit an **individual bitstream**, **each** of which can be destined to a **different UE**.

**Frequency-division multiplexing** is used for **radio channels** in addition to being used for **digital bitstreams**.

### 11.6.2 Time-division multiplexing

The **total transmission time** of the **joint bitstream** is divided into **timeslots** and **each individual bitstream** is transmitted using **one specific stream** of **timeslots**.

**Each** of the **one** or **more receiver**(**s**) can receive (**Demux**) its corresponding **individual bitstream**(**s**) by **selecting** the **corresponding specific stream**(**s**) of **timeslots**.

For example, the **WiFi channel time** acquired by the **access point** contains **frame sequences** (i.e., **individual bitstreams**) such that **each frame sequence** is destined to a **different terminal**. There are also **multicast frames**, but this does **not contradict** this example.

### 11.6.3 Space-division multiplexing

The **space** available for the transmission of the **joint bitstream** is used for **several spatial channels** and **each individual bitstream** is transmitted using **one spatial channel**.

**Each** of the **one** or **more receiver**(**s**) can receive (**Demux**) its corresponding **individual bitstream**(**s**) by **selecting** the **corresponding specific spatial channel**(**s**).

In **wireless** communications, **space multiplexing** is done by implementing **several independent radio channels**. A trivial example consists of installing **two radio links** between the **same two endpoints** and in the **same frequency** but that **do not cross interfere** because the corresponding **Tx/Rx antennas** are **sufficiently spaced away** in respect to the **wavelength**, to the **antenna radiation patterns** and to the **link distance**. **Space multiplexing** is **most frequently** based on **precoding/detection** (click 2.4.2) and **MIMO** (click 2.4.3).

In **space-division multiplexing each** of the **individual bitstreams** is called a **layer**. **Each layer** requires **one** (or more) **exclusive Tx antenna**(**s**) and **one** (or more) **exclusive Rx antenna**(**s**). Therefore, the **quantity** of **layers can** be, **at most**, the **lesser** of the **quantity** of **antennas** at the **transmitter** or at the **receiver**.

**Space-division multiplexing** is sometimes called **space multiplexing**.

### 11.6.4 Code-division multiplexing

The **joint bitstream** is divided into **several streams** of **groups** of **bits**, where **each** such **group** of **bits** encodes **some groups** of **fewer bits**, where **each** such **group** of **fewer bits** corresponds to **each** the **individual bitstreams**. The **bits** from the **joint bitstream** are called "**chips**", because **each** of them is considered to be a "**part**" of a "**proper**" **bit**, where these "**proper**" **bits** are the ones that belong to the **individual bitstreams**. In some systems, a **fixed quantity** of **chips** encodes a **few bits**, where **each bit** belongs to **each** of the **individual bitstreams**.

**Each** of the **one** or **more receiver(s)** can receive (**Demux**) its corresponding **individual bitstream(s)** by applying the **respective code(s)** to the **joint chip stream**.

One frequent technique for **code-division multiplexing** is **Orthogonal Cover Codes** (**OCCs**), which are based on the usage of **orthogonal sequences**.

## 11.7 Inverse multiplexing and demultiplexing

**Digital inverse multiplexing** (**inverse Mux**) consists of **transmitting one main bitstream** as several **individual bitstreams**, such that the **main bitstream** can be **recovered** in the destination(s), and the **capacity** of the **main bitstream** approximates the **addition** of the **capacities** of the **individual bitstreams**.

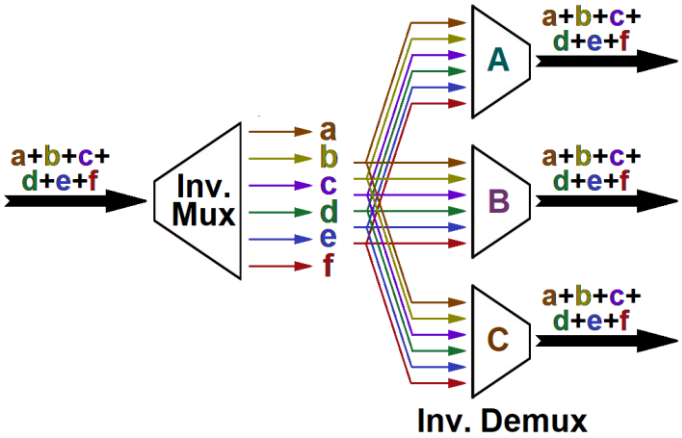
**Digital inverse demultiplexing** (inverse **Demux**) consists of **recovering** the **main bitstream** from **all** the **individual bitstreams**.

**Analog inverse multiplexing** and **demultiplexing** is the **same** as the **digital ones** but **replacing each bitstream** by a **signal** within a given **frequency segment.**

**Inverse Mux** and **inverse Demux** are **unidirectional**. **Inverse Mux** is performed by the **transmitter**. **Inverse Demux** is performed by the **receiver(s)**. **Inverse multiplexing** and **inverse demultiplexing** are **reciprocal** functions, and they **both must** use the **same technique** for a **correct operation**.

**Inverse Mux/Demux** are defined for **only one transmitter**, but there may be **one or more receivers**. Therefore, **Inverse Mux/Demux** may be performed in **point-to-point** (**one transmitter** (**Tx**) and **one receiver** (**Rx**)) and in **point-to-multipoint** (**one Tx** and **several Rx**) communications. It **cannot** be performed in **multipoint-to-point** (**several Tx** and **one Rx**) **nor** in **multipoint-to-multipoint** (**several Tx** and **several Rx**). For **several transmitters**, refer to **multiple access** (click 11.9) **techniques**.

The following figure illustrates **inverse multiplexing** and **inverse demultiplexing**:



Some **relevant remarks** on this figure are:

- This example is for **3 receivers** (**inverse Demuxes**), which are labeled "**A**", "**B**" and "**C**". However, in many cases there is only **1 receiver**.
- The **main bitstream** is split into **6 individual bitstreams**, which are labeled "**a**" to "**f**".
- The **6 individual bitstreams** may have **different bitrates**.
- The **6 individual bitstreams** may be sent over **1 channel**, over **6 individual channels**, or over **any combination** of **shared channels**.
- The **throughput** of the **main bitstream** is below, but **close to**, the **addition** of the **throughput** of the **6 individual bitstreams**.
- **All receivers** receive **all** the **individual bitstream**.
- **Each receiver performs inverse demultiplexing** and recovers the **main bitstream**.

The **basic inverse demultiplexing techniques** are **similar** to the **multiplexing techniques** (click 11.6).

It is also possible to **combine** the **basic inverse multiplexing techniques** in different ways. Such is the case of **5G**, that combines several of them.

In **both Downlink** and **Uplink 5G** uses **frequency inverse multiplexing**, by aggregating the **bitstreams** from **different radio channels** based on **carrier aggregation** (click 10.2). **5G** may also **use space-division multiplexing** based on **single-user MIMO** (click 2.4.3).

**Duplexing** (click 11.3), **multiplexing** (click 11.6), **inverse multiplexing** (click 11.7), **diversity** (click 11.8) and **multiple access** (click 11.9) are **different**, but **related**, concepts. For their **relationships**, click 11.10.

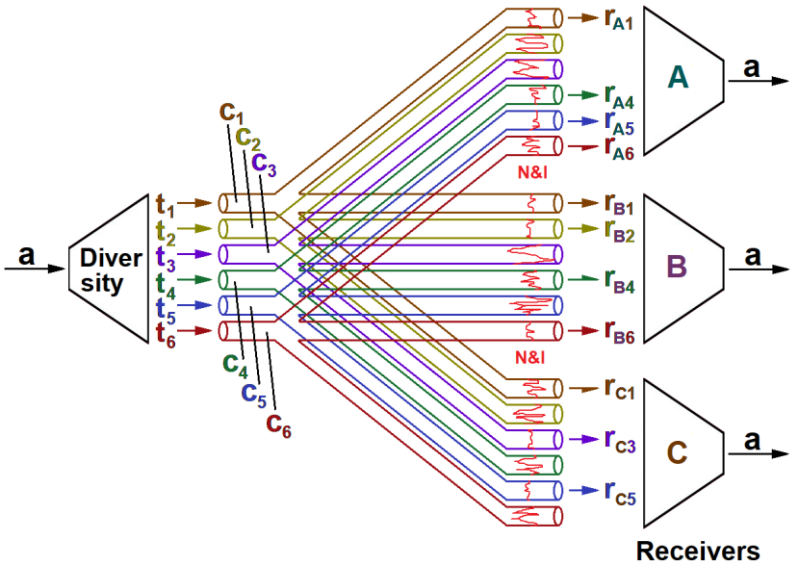
## 11.8 Diversity

**Diversity** consists of **sending one bitstream** "**a**" as **several Tx signals** "$\mathbf{t_i}$", **each** transmitted over **one** (**partly**) **independent channel** "$\mathbf{c_i}$" resulting in an **increased reliability** of the **bitstream**. The "$\mathbf{t_i}$" **signals** may be **redundant** or **not**, **depending** on each specific **diversity technique**.

The **transmitter** applies a **Tx diversity** technique and the **receiver performs** a **Rx diversity combining** technique, suitable for the **Tx diversity** technique.

**Diversity techniques** are based on that (**partly**) **independent channels** experience **fading** and **interference** at **different**, **random times**. **Diversity techniques** are mainly used in **radio communication** and allow to **mitigate fading** (click 12.2.7) and **co-channel interference** (click 12.1.13), resulting in **largely reduced error bursts**. Some **diversity techniques** exploit **multipath propagation** (click 12.2.3), resulting in a **diversity gain**, often measured in **decibels**.

The following figure illustrates **diversity**:




Some **relevant remarks** on this figure are:

- This example is for **3 receivers**, which are labeled **A**, **B** and **C**. However, in many cases there is only **1 receiver**.
- The **6 channels** are labeled "$\mathbf{c_1}$" to "$\mathbf{c_6}$" and are identified by different colors.
- **Each receiver** "**j**" **most likely** receives **different signals** "$\mathbf{r_{j1}}$" to "$\mathbf{r_{j6}}$" over **each** of the **channels** "$\mathbf{c_1}$" to "$\mathbf{c_6}$".
- **Any two receivers** "**j**" and "**k**" **most likely** receive **two different signals**, "$\mathbf{r_{ji}}$" and "$\mathbf{r_{ki}}$" from their **same channel** "$\mathbf{c_i}$".
- **Each receiver** "**j**" **applies** a **diversity combining** technique to its received **signals** "$\mathbf{r_{j1}}$" to "$\mathbf{r_{j6}}$" from channels "$\mathbf{c_1}$" to "$\mathbf{c_6}$" to recover the **transmitted bitstream** "**a**" with a **high reliability**.
- **Noise** and **interference individually experienced** by **each** of the **channels** is shown in the figure as **crooked red lines**.

Some basic **diversity** techniques are:

- **Frequency diversity**

  **Each individual signal** "$\mathbf{t_i}$" is **either** transmitted over a **different frequency segment or spread** over a **wide frequency segment** that is affected by **frequency-dependent fading**. Three examples are:

  - A **multicarrier signal** (click 11.3.2) combined with **subcarrier interleaving**. The **receiver** has to **deinterleave** the **subcarriers**. This is applied in **5G** in **PUCCH-PUSCH interlace** (click 2.2.4.8).
  - A **multicarrier signal** (click 11.3.2) combined with **subcarrier interleaving** and with **forward error correction** (**FEC**, click 11.2.3). The **receiver** has to **deinterleave** the **subcarriers** and **process** the **FEC code**.
  - **Spread spectrum** (e.g., **frequency hopping**). The **receiver** has to perform the **reciprocal operation**.

- **Time diversity**

  Two examples are:

  - **Each individual signal** "$t_i$" is transmitted at a **different time instant**. The **receiver** has to perform **diversity time combining** (see below).
  - A **FEC code** is added to a **bit-block** and the resulting **codeword** is **spread** in **time** by means of **bit-interleaving**. The **bit-interleaving** prevents **error bursts** which improves the **error correction**. The **receiver** has to **deinterleave** the **codeword** and process the **FEC code**.

- **Polarization diversity**

  **Each individual signal** "$t_i$" is **transmitted** and **received** via **one Tx antenna**, where the **different Tx antennas** have **orthogonal polarization**.

- **Space diversity**

  **Each individual signal** "$t_i$" is transmitted over a **different propagation path** using a **Tx antenna array**. In case that the **transmitter** knows the **channel state information** (**CSI**), the corresponding concept is **not space diversity**, and it is **rather precoding** (click 2.4.2). For more detail on **space diversity**, click 11.8.1.

It is also possible to **combine** these **basic combining techniques** in different ways. For example, "**lucky imaging**" uses **selection combining** to choose (typically) the best 10% images, followed by **equal-gain combining** of the selected images.

**Duplexing** (click 11.3), **multiplexing** (click 11.6), **inverse multiplexing** (click 11.7), **diversity** (click 11.8) and **multiple access** (click 11.9) are **different**, but **related**, concepts. For their **relationships**, click 11.10.

**Diversity** (click 11.8), **beamforming** (click 11.12), **precoding** (click 2.4.2) and **MIMO** (click 2.4.3) are **different**, but **related**, concepts. For their **relationship**, click 2.4.4.

Source web pages:

- https://en.wikipedia.org/wiki/Diversity_scheme
- https://en.wikipedia.org/wiki/Diversity_combining

## 11.8.1 Space diversity

**Each individual signal** "$t_i$" is transmitted over a **different propagation path** using a **Tx antenna array**.

If the **Tx antennas** are **spaced** in the order of **one wavelength**, this is called **microdiversity**. If the **Tx antennas** are **very far apart** (e.g., different **5G antenna sites** or distant **WLAN access points**) this is called **macrodiversity** or **site diversity**. One case of **macrodiversity** is **Coordinated Multipoint** (**CoMP**, click 10.11).

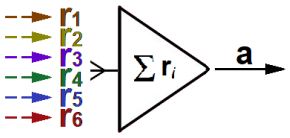
In a **multi-user macrodiversity scheme**, the **terminals** may also be far apart (e.g., in **CoMP**). Therefore, **every constituent link** within **each virtual MIMO link** has **different average link SNR**. This **difference** is mainly due to the **different long-term channel impairments** (e.g., *Free-space path-loss* and **shadow fading**) experienced by the **different constituent links**. **Multi-user macrodiversity** poses unprecedented theoretical and practical challenges. Among many theoretical challenges, perhaps the most fundamental one is to understand how the different average link **SNRs** affect the overall system capacity and individual user performance in fading environments.

One case of a **space diversity technique** is **diversity coding**. In this technique, the **bitstream** is encoded with **several different fully/near orthogonal codes**, and **each such** resulting **individual bitstream** is transmitted over a **different antenna**. **Diversity coding** can be combined with **precoding** (click 2.4.2) when some **channel knowledge** is available at the **transmitter**.

In **space diversity** the **receiver** has to apply a **space diversity combining** technique. Some **space diversity combining** techniques are:

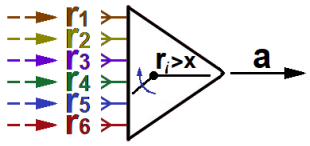
- **Additive Combining**

  The **receiver only** has **1 Rx antenna**. **All** the **individual signals** "$r_i$" arriving to the receiver are **added** by the **single Rx antenna**, possibly followed by some signal grooming.

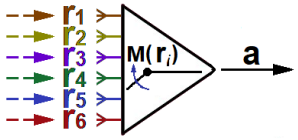



- **Switching Combining**

  The **receiver** has several **Rx antennas**, **each** of which **outputs** a **signal** "$r_i$". **One** of the **individual received signals** is **selected**. If the **power** of the **currently selected individual signal** "$r_j$" falls **below** a certain **threshold** "**x**", the **combiner switches** the selected signal to another **individual signal** "$r_k$" whose **power** is above the **threshold** "$P(r_k) > x$". A **switching combiner** is sometimes called "**scanning combiner**".

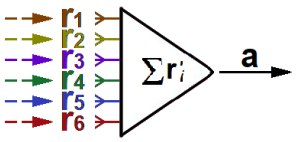



- **Selection Combining**
- The **receiver** has several **Rx antennas**, **each** of which **outputs** a **signal** "$r_i$". The **individual received signal** "$r_j$" with the **highest power**, "$P(r_j) > P(r_i)$" for **all** "$i$" is **selected**.



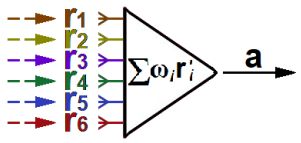

- **Equal-Gain Combining**

  The **receiver** has several **Rx antennas**, **each** of which **outputs** a **signal** "$r_i$". **Each individual received signal** "$r_i$" is **adjusted** in **phase** and **equalized**, resulting in "$r'_i$", and **all** the resulting **signals** "$r'_i$" are **added**.



- **Maximal-Ratio Combining**

  The **receiver** has several **Rx antennas**, **each** of which **outputs** a **signal** "$r_i$". **Each individual received signal** "$r_i$" is **adjusted** in **phase** and **equalized**, resulting in "$r'_i$". **Each** resulting **signal** "$a'_i$" is **weighted** ("$\omega_i$") based on its **S/N** and **all** the **weighted signals** "$\omega_i \bullet r'_i$" are **added**.




## 11.9 Multiple access

**Multiple access** consists of allowing a **set** of **transmitters** to **transmit** through the **same shared media** (radio or cable). Some **multiple access techniques** are:

- **Frequency-division multiple access**
  - **Orthogonal frequency-division multiple access** (click 11.15)
  - **Single-carrier frequency-division multiple access** (click 11.16)
- **Time-division multiple access**
- **Space-division multiple access**
- **Code-division multiple access**

**Multiple access** is based on **multiplexing** (click 11.6), because **multiplexing** is **required** to **share** the **channel** by the **different transmitters**. This is why the **types** of **multiple access** are almost the same as the **types** of **multiplexing**, and they share many characteristics.

**5G** uses **code-division multiple access** for **PUCCH** (click 7.8.2.10).

A **multiple access method must** be part of **every medium access control** (**MAC**, click 11.11) **technique**.

**Multiple access** is sometimes called "**channel access**".

**Duplexing** (click 11.3), **multiplexing** (click 11.6), **inverse multiplexing** (click 11.7), **diversity** (click 11.8) and **multiple access** (click 11.9) are **different**, but **related**, concepts. For their **relationships**, click 11.10.

Source web page:

- https://en.wikipedia.org/wiki/Channel_access_method

## 11.10 Relationships between duplexing, multiplexing, inverse multiplexing, diversity and multiple access

**Duplexing** (click 11.3), **multiplexing** (click 11.6), **inverse multiplexing** (click 11.7), **diversity** (click 11.8) and **multiple access** (click 11.9) are **different**, but **related**, concepts.

**Duplexing vs. Multiplexing**

**Both duplexing** (click 11.3) and **multiplexing** (click 11.6) need to handle **more than one individual bitstream** (or **signal** in a **frequency segment**) over a **larger joint**

**bitstream** (or **signal** in a **frequency segment**). This is why several **duplexing** and **multiplexing techniques** are **very similar**.

However, **duplexing** and **multiplexing** have significant **differences**:

- **Duplexing only** handles **2 individual bitstreams**, while **multiplexing** usually handles **more than 2 individual bitstreams**.
- **Duplexing** communicates in **2 transmission directions**, while **multiplexing** communicates in **only 1 transmission direction**.
- **Frequency-division duplexing** in **radio systems** has significant **self-interference problems** because it implies **simultaneous transmission** and **reception**, and **radio transmitted power** is **much higher** than the **received power** (much higher than in wired systems). However, **frequency-division multiplexing** in **radio systems** does **not** have these **problems** because **all** the **individual bitstreams** (or **signals** in a **frequency segment**) are in the **same transmission direction**.

**Multiplexing vs. Inverse Multiplexing**

**Multiplexing** (click 11.6) and **inverse multiplexing** (click 11.7) perform **opposite** operations.

**Inverse multiplexing vs. diversity**

**Both inverse multiplexing** (click 11.7) and **diversity** (click 11.8) transmit **one main bitstream** as **several individual bitstreams**.

The main difference between them is that **inverse multiplexing** aims to distribute the **throughput** of the **main bitstream** among the **several lower-throughput individual bitstreams** (such that the **main bitstream** can be **recovered** at the receiver), while **diversity** aims to **increase** the **reliability** of the **main bitstream**.

**Multiplexing vs. Multiple Access**

**Both multiplexing** (click 11.6) and **multiple access** (click 11.911.6) transmit several **individual bitstreams** over a **channel**. Actually, **each multiple access technique** is based on some **multiplexing technique**. This is why the **multiplexing techniques** and **multiple access techniques** are **very similar**.

The **main difference** between them is that in **multiplexing** there is **only one transmitter**, while in **multiple access** there are **several transmitters**.

## 11.11 Medium Access Control (MAC)

**Medium Access Control** (**MAC**) is a **protocol sublayer** that enables **several systems** to **effectively communicate** over a **shared transmission media** (**radio** or **cable**).

There are **two basic types** of **MAC communication**:

- **Distributed MAC communication**

  **Several systems communicate directly** among themselves, in the **same way**. An example of **distributed MAC** is **Ethernet** local area networks.

- **Centralized MAC communication**

  A **central node transmits/receives to/from** a **set** of **terminals**. The **terminals** therefore do **not communicate directly** among themselves.

The **MAC sublayer always** has **two components**:

- **Multiple access**

  Consists of allowing a **set** of **transmitters** to **transmit** through the **same shared media** (radio or cable). For more detail on **multiple access**, click 11.9:

- **Transmission coordination**

  Consists of **determining** what **system** has the **right** to **transmit** at **every given moment**.

For the case of **centralized MAC**, a **third component** is **also** required:

- **Multiplexing/Demultiplexing**

  **Digital multiplexing** (**Mux**) consists of **transmitting** several **individual bitstreams** into **one joint larger bitstream** such that **each** and **every individual bitstream** can be **recovered** in the destination

  It is required for the **transmission direction** from the **central node to** the **terminals**. This transmission direction is called **Downlink**.

  For more detail on **multiplexing/demultiplexing**, click 11.6:

Regarding **transmission coordination**, there are **two basic techniques**:

- **Contention techniques**

  **Contention** consists of **each system initially transmitting** an **information block** (usually called a **frame**) whenever the **media** is **free**. The **receiver must acknowledge** the correct reception. If the **transmitter** does **not** receive the **acknowledgement** after a **certain time**, it **waits** a **non-deterministic time** and then it **retransmits** the **frame**. When **two or more frames overlap** in **time** and **frequency** and **code** it is called a "**collision**". **All** the **colliding frames mutually interfere** and consequently **all** of them are **lost**. **Lost frames** due to **collisions** are recovered by **retransmission**, as already indicated.

  Some examples of **contention techniques** are **Aloha**, **slotted Aloha**, **CSMA/CD** (used in **Ethernet**) and **CSMA/CA** (used in **WiFi**).

  Be aware that the **term frame** used in **this context** is **not** the **same** as a **5G frame**.

- **Reservation techniques**

  **Reservation** consists of **each system requesting permission** for **transmission** (i.e., a **reservation request**) to a **given system**, and that **given system** providing **transmission rights**, in a suitable way, to **each requester**.

There are **variants** of these basic techniques that do **not exactly conform** to the explanations above, and there are **also hybrid techniques**.

**Contention** techniques work **well** for **low load**, achieving a **good throughput** and **low latency**. However, for **high loads** the achieved **throughput** is **considerably sub-optimal** and **latency** can **grow large**. **Latency** is in any case **non-deterministic**. **Contention** techniques are **simple**, very **robust** and do **not** have a **single point** of **failure**.

**Reservation** techniques work **well both** for **low** and **high loads**, achieving a **high throughput** and **low latency**. They **also** may provide **deterministic latency bounds**.

They are usually **more complex** than **contention** techniques and have **single point** of **failure**.

**Contention** is more **frequently** used in **distributed MAC** and **reservation** in **centralized MAC**, although there are also **cross cases**.

The **MAC sublayer** is placed within the **link layer** (**layer 2**) of most **protocol architectures**. It is placed **over** the **physical layer** (**layer 1**) and it is **used by** the **logical link sublayer** (which is also part of the **link layer**).

For the **5G MAC sublayer**, click 4.5.4.

## 11.12 Hybrid Automatic Repeat reQuest (HARQ) protocols

A **HARQ protocol** is a **hybrid** between a pure **Automatic Repeat reQuest** (**ARQ**) **error-recovery protocol** and a pure **Forward Error Correction** (**FEC**, click 11.2.3) **channel code**. **HARQ packets** contain a **FEC code** in addition to the **ARQ Error Detection** (**ED**) **code** (which is usually a **CRC**, click 11.2.3.2).

The objective of **HARQ protocols** is to **improve** the **channel reliability** while keeping a **low delay**. **HARQ protocols** do **not** provide **100% reliability**. **HARQ protocols** may **both lose** and **duplicate packets**. Even though **HARQ protocols** do not provide **100% reliability**, they **do preserve** the **order** of **transmitted data**.

**HARQ protocols** almost always have a **maximum quantity** of **retransmissions**, after which the **failed HARQ packet** is **discarded** by the transmitter.

**Depending on** the processing at the **HARQ receiver**, **HARQ protocols** may be classified as:

- **Simple HARQ protocols** (click 11.12.1)
  are the ones where the **HARQ receiver discards** the **erroneous packets**. **Depending on** the processing at the **HARQ transmitter**, **simple HARQ protocols** may be subdivided into:
  - **Simple HARQ type I protocols**
    are the ones where the **HARQ transmitter** sends **data+ED+FEC both** in the **first transmission** and in **successive retransmissions**.
  - **Simple HARQ type II protocols**
    are the ones where the **HARQ transmitter** sends **only data+ED** in the **first transmission** and sends **data+ED+FEC** in **successive retransmissions**.
- **Soft-combining HARQ protocols** (click 11.12.2)
  are the ones where the **HARQ receiver buffers erroneous packets** and uses them to **correctly decode** ulterior **retransmitted packets**. **Depending on** the processing at the **HARQ transmitter**, **soft-combining HARQ protocols** may be subdivided into:
  - **Chase combining HARQ protocols**
    are the ones where the **HARQ transmitter** sends the **same data+ED+FEC** in the **first transmission** and in **every successive retransmission**.
  - **Incremental redundancy HARQ protocols**
    are the ones where the **HARQ transmitter** sends **different FEC** in the **first transmission** and in **every successive retransmission**, while the transmitted **data+ED** remains the **same**.

Several **variants** and **combinations** of these protocols have also been proposed.

**HARQ processes**

**HARQ protocols** usually use **several HARQ processes**, where **each HARQ process** implements a **stop-and-wait HARQ**. When **one** or **more HARQ processes** are **waiting** for an **acknowledgment**, **each successive HARQ process** can **transmit** its **HARQ packet**. For the **stop-and-wait HARQ**, a **1 bit sequence number** is required in every **HARQ packet**, in order to distinguish a **retransmission** of the previously transmitted **HARQ packet** from a **newly transmitted HARQ packet**.

Using **several HARQ processes** implies a **slightly more complex implementation** than the one of **only 1 HARQ process** but **increasing** the quantity of **processes increases throughput** and **reduces delay** (up to the **bandwidth•RTT limit**).

Using **several HARQ processes requires either adding** a **larger sequence number** to **every packet** or devising an **implicit packet indexing**. This is **required** because the **receiver must** be able to **associate each** incoming **HARQ packet** with the correct **HARQ process**, in order to **preserve packet order**.

Using ***n* stop-and-wait HARQ processes** allows to send "***n***" **HARQ packets** while (**N**)**ACKs** are pending to arrive, but it is **not** equivalent to a **sliding window protocol** (e.g., like **HDLC**).

**HARQ protocols in protocol architectures**

**HARQ protocols** do not fit well into the classical **protocol architecture** because: a) their objective is to provide an "**improved**" **physical channel**, this is, a **physical channel** with **better reliability** while **keeping** a **low delay**; b) their **ACK** and **NACK** information is transported over **specific physical layer control channels**. **HARQ** functionality is therefore in between the **physical layer** (**L1**) and the **link layer** (**L2**). To avoid changing the protocol architecture, **HARQ** is quite **frequently** placed within the **Medium Access Control** (**MAC**, click 4.5.4) **sub-layer**, that belongs to the **link layer** (**L2**).

Source web page:

- https://en.wikipedia.org/wiki/Hybrid_automatic_repeat_request

## 11.12.1 Simple HARQ protocols

If a **received packet** is **correct** (either **as such** or correcting it with the **FEC code**), the packet is **acknowledged**. Otherwise, the **packet** is **discarded** and it is **either not acknowledged** or a **negative ack** is sent. The **sender** will therefore **retransmit** the **packet**, and eventually, the **packet** will **correctly arrive**.

**Depending on** the processing at the **HARQ transmitter**, **simple HARQ protocols** may be subdivided into:

- **Simple HARQ type I protocols**
  are the ones where the **HARQ transmitter** sends **data+ED+FEC both** in the **first transmission** and in **successive retransmissions**.
- **Simple HARQ type II protocols**
  are the ones where the **HARQ transmitter** sends **only data+ED** in the **first transmission** and sends **data+ED+FEC** in **successive retransmissions**.

Notice that the **ED** adds **very little overhead** (a few octets), while the **FEC overhead** is **very large**: it is usually the **same**, or a **few times**, the **size** of **data**. On **good channel**

conditions **simple HARQ type I protocols** have **low throughput**, while **simple HARQ type II protocols** perform as well as a **pure ARQ protocol**. On **bad channel** conditions, **simple HARQ type I protocols perform better** than **type II**, because the **first packet transmission** can on some cases **be corrected** thanks to its **FEC code**, thus **avoiding** its **retransmission**.

### 11.12.2 Soft-combining HARQ protocols

**Soft-combining HARQ protocols do not discard** the **erroneous packets**, and they are rather **buffered**. In case the **retransmitted packet** is again **erroneous**, it is **combined** with the **buffered packet(s)** with a **soft-combining** technique that **increases** the **correction probability**. If the **packet** is **still erroneous** in spite of the **combination**, it is also **buffered** to be used in combination with the **previously buffered ones**, and with the **next retransmission** of the **packet**. When the **packet** is finally **received correctly** (either **as such** or by **soft-combining**), it is **acknowledged** and **all** the **buffered packets** are **removed**.

**Depending on** the processing at the **HARQ transmitter**, **soft-combining HARQ protocols** may be subdivided into:

- **Chase combining HARQ protocols**
  are the ones where the **HARQ transmitter** sends the **same data+ED+FEC** in the **first transmission** and in **every successive retransmission**.
- **Incremental redundancy HARQ protocols**
  are the ones where the **HARQ transmitter** sends **different FEC** in the **first transmission** and in **every successive retransmission**, while the transmitted **data+ED** remains the **same**.

In **chase combining HARQ protocols** the **receiver** uses **maximum-ratio combining** to combine the **last retransmitted packet** with the **buffered packets**.

In **incremental redundancy HARQ protocols** the **different FEC** information **sent** in **first packet transmission** and in **every successive packet retransmission** is typically generated by **puncturing** the **encoder output**. Thus, at **every retransmission** the **receiver** gains **additional FEC** information.

For **cellular networks**, ample experience has shown that **incremental redundancy HARQ outperforms chase combining**.

### 11.12.3 HARQ protocol in LTE

Source web pages:

- https://www.sharetechnote.com/html/BasicProcedure_LTE_HARQ.html
- https://techlteworld.com/harq-hybrid-automatic-repeat-request-in-lte/
- https://www.techplayon.com/hybrid-automatic-repeat-request-harq-in-lte-fdd/

#### 11.12.3.1 Overview of HARQ protocol in LTE

**Each HARQ packet** that is sent in **1 transport block**, which is attached a **CRC** and converted into **1 code block sequence**. The **code block sequence** may have **1** or **several code blocks**. **Each code block** is **encoded**, adding to **each** of them its specific **CRC** (except when the **sequence** has **only 1 code block**) and its specific **FEC code bits**. The **sequence** of **encoded code blocks** is **1 codeword**, that is sent in **1 resource scheduling**

**block** (i.e., in **1 subframe** within **1 channel**). In case the **resource scheduling block** is larger than the **sequence** of **encoded code blocks**, **padding** is used.

For **Uplink each HARQ packet** is sent to the **eNB** by **single UE**. For **Downlink each HARQ packet** is sent to a **single UE** (or group of **UEs**) by the **eNB**.

The **receiver must acknowledge each received HARQ packet**, **sending either** an **ACK** or a **NACK** (shortened to "(N)ACK") for **each received HARQ packet**. The **(N)ACKs** are sent over the **Physical HARQ Indicator Channel**) (**PHICH**). **Each** (N)ACK occupies **1 PHICH bit**, where value "**1**" is an **ACK** and value "**0**" is a **NACK**.

The **HARQ protocol** has a **maximum quantity** of **retransmissions** after which the **HARQ packet** is discarded. The **HARQ protocol** may therefore **lose packets**. This is **not** a **problem** because the **HARQ protocol** is **not** designed to provide **100% reliability**: it is rather designed to **improve** the **channel reliability** while keeping a **low delay**. Services that require **100% reliability** use the service of the **Radio Link Control** (**RLC**) **protocol**. The **maximum quantity** of **HARQ retransmissions** is configured to **each UE** with parameter `maxHARQ-Tx` (found in `radioResourceConfigCommon` and `radioResourceConfigDedicated`). In order to provide a **low delay**, the **maximum quantity** of **HARQ retransmissions** is **small** (e.g., **2** to **4**).

The **quantity** of **HARQ processes** ranges from **1 to 8**, **depending** on the **duplexing** mode and other characteristics.

The **New Data Indicator** (**NDI**) **bit**, sent by the **eNB** through **PDCCH**, allows the **UE** to determine if an **incoming HARQ packet** is **either** a **retransmitted** one or a **newly transmitted** one. The **NDI bit** works as the **sequence number** in **alternating bit protocols**: the **bit** has the **same value** in **each newly transmitted packet** and in **each and every of its retransmissions**. The **bit changes value** ("toggles") between **two successive newly transmitted packets**.

The **redundancy version two-bits** "***rv***", sent by the **eNB** through **PDCCH**, allow the **UE** to determine what is the **type** of **FEC content** of **each code block** of an **incoming HARQ packet**. One of the **HARQ modes** uses **punctured Turbo codes** with **code rate=1/3**. This means that the **complete FEC code bitlength** is **twice** the **user data bitlength**. The **Turbo code FEC bits** are of **3 types**: a) **systematic bits**; b) **first parity bits** and c) **second parity bits**. The **systematic bits** are inserted **first** into a **circular buffer**, followed by an **alternating bit insertion** from the **first parity bits** and from the **second parity bits**. The **FEC bits attached** to **each code block** are a **consecutive extraction** from the **specific circular buffer** of the **code block**, **starting** from the **position** indicated by the **redundancy version** "***rv***" and **up to** the extent that **fits** into the **resource scheduling block**. The **values** of "***rv***" are {**0**, **1**, **2**, **3**}. The value **0** is **always**, and **only**, used for a **newly transmitted HARQ packet** in order to **maximize** the **quantity** of **systematic bits** sent.

The **Downlink Control Information** (**DCI**) **0 message** sent by the **eNB** through **PDCCH** **may** contain the **following fields**:

- **transport block** ***size***
- ***MCS***
- ***Redundancy version "rv"***
- ***NDI***

- ***HARQ process identifier***
- ***Resource allocation***

The **HARQ processes** have different **characteristics** depending on the **duplexing mode**:

- **Frequency Division Duplexing (FDD)**

  Recall that **Downlink** and **Uplink** use different **frequency segments**, so their **transmissions** are **simultaneous**.

  - **Fixed acknowledgement time**

    The **(N)ACK** for a **HARQ packet** transmitted in **subframe *n*** is **always** received in **subframe *n*+4**, **both** for **Downlink** and for **Uplink**. In terms of time this is **4 ms** (for the **4 subframes**). The **UE** requires **3 ms**, after having received a **HARQ packet**, for its **processing** (**FEC**, …), and therefore, the **UE** sends the corresponding **(N)ACK 4 ms later**. Conversely, the **eNB** requires **3 ms**, after having received a **(N)ACK**, for **processing** (**FEC**, …) the corresponding **new transmission** or **retransmission**.

  - **Fixed quantity** of **HARQ processes**

    It uses **8 HARQ processes**, **both** for **Downlink** and for **Uplink**. Notice that in **FDD Downlink**, the **UE** receives **each (N)ACK 8 subframes after** the corresponding **HARQ packet transmission**. Consequently, **8 HARQ processes** allow for **continuous transmission** until the **(N)ACK** is received.

  - **Periodic retransmissions**

    A **retransmitted HARQ packet** may be sent **every 4 subframes**, **both** for **Downlink** and for **Uplink**.

- **Time Division Duplexing (TDD)**

  Recall that **Downlink** and **Uplink** use the same **frequency segment**, so their **transmissions alternate**, following certain configured patterns.

  - **Variable acknowledgement time**

    The **(N)ACK** for a transmitted **HARQ packet** is received after a **variable** period of **time** that depends on the **TDD duplexing configuration**, **both** for **Downlink** and for **Uplink**. **HARQ feedback timing** varies (e.g., **4 ms**, **6 ms**, **7 ms**, …), and the **UE** uses table 9.1.3-1 from TS 36.213 to determine when to send a **(N)ACK**.

  - **Variable quantity** of **HARQ processes**

    The **quantity** of **HARQ processes** depends on the **TDD duplexing configuration**, **both** for **Downlink** and for **Uplink**. The **maximum quantity** of **HARQ processes** is **7**.

  - **Variable delay retransmissions**

    The **delay** of **Uplink retransmissions** is **inversely proportional** to the **quantity** of **Uplink slots** in the **TDD duplexing configuration**.

  Note that in the **PHICH information** of a **single subframe** it is possible to **transport (N)ACKs** corresponding to **multiple subframes**. Therefore, the

**system works** correctly even when there are more **Uplink subframes** than **Downlink subframes**.

For more detail on **FDD HARQ** in **LTE**, click 11.12.3.2.

For more detail on **HARQ entity structure**, refer to clauses 5.4.2.1 and 5.4.2.2 of TS 36.321.

### 11.12.3.2 FDD HARQ protocol in LTE

The **LTE FDD HARQ processes** have different **characteristics** depending on the **transmission direction**, **Downlink** or **Uplink**.

**LTE FDD Downlink HARQ** is **asynchronous**

In **Downlink**, the **HARQ transmitter** is the **eNB** and the **HARQ receiver** is the **UE**. The **characteristics** of the **FDD Downlink asynchronous HARQ** are:

- **Usage of HARQ processes**

  The **eNB** can freely use **any** of the **8 HARQ processes** for a **newly transmitted HARQ packet**.

- **HARQ processes determination**

  For **each** and **every received HARQ packet**, the **UE knows** its **HARQ process number** because the **eNB** explicitly indicates it in a field within a **DCI 1** or **DCI 2 message** sent through **PDCCH**.

- **Redundancy version determination**

  For **each** and **every received HARQ packet**, the **UE knows** its **redundancy version** "***rv***" because the **eNB** explicitly indicates it in a field within a **DCI 1** or **DCI 2 message** sent through **PDCCH**.

- **Acknowledgement determination**

  This is based on the **HARQ process number**.

- **Retransmission mode**

  It **only** uses **adaptive retransmission** (click 11.12.3.3).

**LTE FDD Uplink HARQ** is **synchronous**

In **Uplink**, the **HARQ transmitter** is the **UE** and the **HARQ receiver** is the **eNB**. The **characteristics** of the **FDD Uplink synchronous HARQ** are:

- **Usage of HARQ processes**

  The **UE** uses the **8 HARQ processes** in a **round-robin** way to **transmit each HARQ packet** (**either newly transmitted** or **retransmitted**).

- **HARQ processes determination**

  For **each** and **every received HARQ packet**, the **eNB** can associate it to one of its **8 HARQ process** because the **eNB** uses its **HARQ processes** in a **round-robin** way. Notice that the **HARQ process id** of the **transmitter** and of the **receiver** may be **different**, but this is **not** a **problem**.

- **Redundancy version determination**

  For **each** and **every received HARQ packet**, the **eNB knows** its **redundancy version** "***rv***" because the **eNB** has **previously** decided the **value** of "***rv***" and has **signaled** it to the **UE** as part of the **Uplink grant** sent in a **DCI 0 message**.

- **Acknowledgement determination**

  The **eNB** sends the **(N)ACK** for **each received HARQ packet 4 subframes after** having **received** the **HARQ packet**. Then, the **UE receives** the **(N)ACK** for **each transmitted HARQ packet 4 subframes after** having **transmitted** the **HARQ packet**. Consequently, the **UE** knows the **HARQ process** to which **each (N)ACK corresponds**.

- **Retransmission mode**

  It can use **both adaptive** and **non-adaptive retransmission** (click 11.12.3.3).

**LTE Downlink and Uplink HARQ performance**

**Downlink asynchronous HARQ** is **more flexible** than **Uplink synchronous HARQ**, because **retransmissions** do not have to be sent at a **fixed subframe**. However, **asynchronous HARQ** has **more** signaling **overhead** than **synchronous HARQ** because the **eNB** has to **explicitly** send the **HARQ process** and "***rv***" for **each** and **every packet**.

### 11.12.3.3 FDD HARQ retransmission modes in LTE

The **FDD HARQ** has **two retransmission modes**: **adaptive** and **non-adaptive**:

**FDD Downlink (asynchronous) HARQ** uses **only adaptive retransmission**.

**FDD Uplink (synchronous) HARQ** may use **both adaptive** and **non-adaptive retransmission**.

- **FDD adaptive retransmissions**

  It consists of the **Modulation and Coding Scheme (MCS)** possibly **changing** in **each successive retransmission** of the **same HARQ packet**. This allows to **adapt MCS** to the **specific channel conditions**, in order to **maximize** the **probability** that **each successive retransmitted HARQ packet** is **correctly received**.

  The **eNB always** uses **adaptive retransmission**. The **eNB** uses (in **each successive retransmission**) the **MCS** and the **redundancy version** "***rv***" that the **eNB** considers **best**.

  The **UE** uses **adaptive retransmission** if it receives a **DCI 0 message** with **NDI not toggle**d. In this case, the **UE retransmits** with the **MCS** and the **redundancy version** "***rv***" contained in the **received DCI 0 message** sent by the **eNB**.

- **FDD non-adaptive retransmissions**

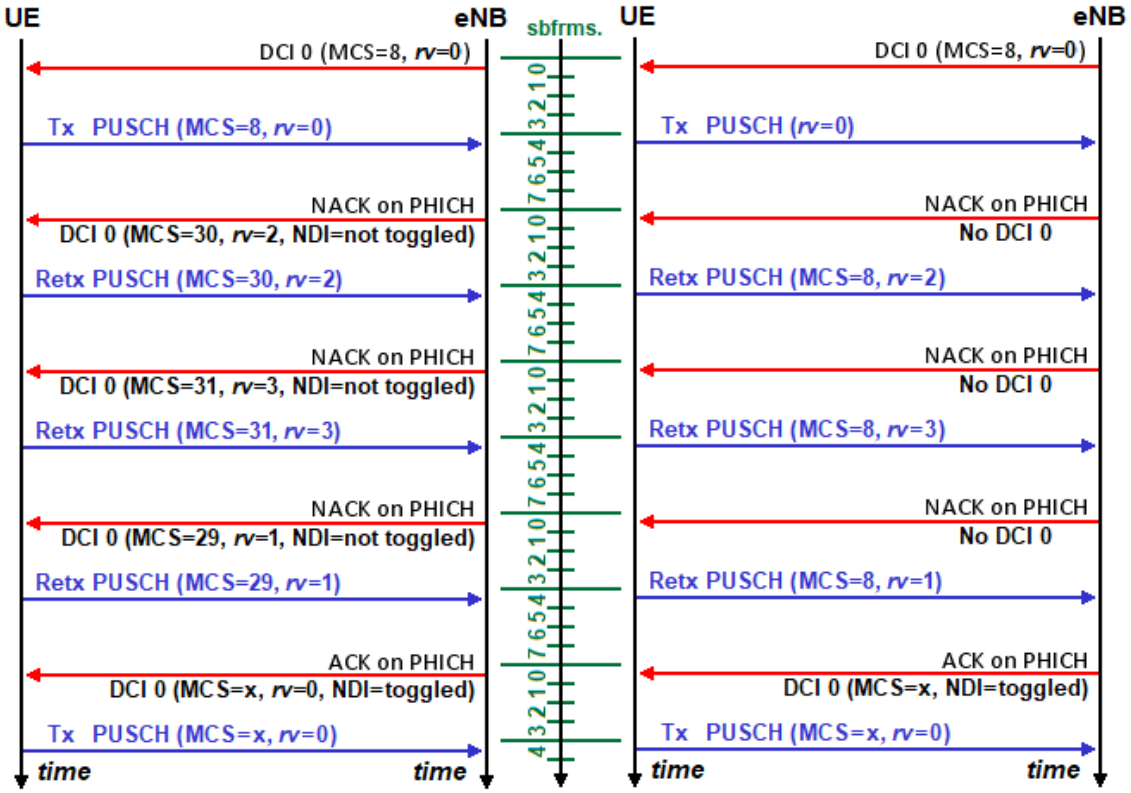
  It consists of the **Modulation and Coding Scheme (MCS)** in **each successive retransmission** remaining the **same** as in the corresponding **newly transmitted HARQ packet**.

  The **eNB never** uses **non-adaptive retransmission**.

  The **UE** uses **non-adaptive retransmission** if it receives "**HARQ feedback (PHICH=NACK)**" and it does **not** receive a **DCI 0 message**. In this case, the **UE**

**retransmits** with the same **MCS** as its **previous HARQ packet**, and with ***rv*** according to the **cyclic sequence** {**0**, **2**, **3**, **1**}.

The following figure is an example of a few packet exchanges of **1 FDD Uplink (synchronous) HARQ process**, showing **adaptive retransmissions** on the **left** and **non-adaptive retransmissions** on the **right**:



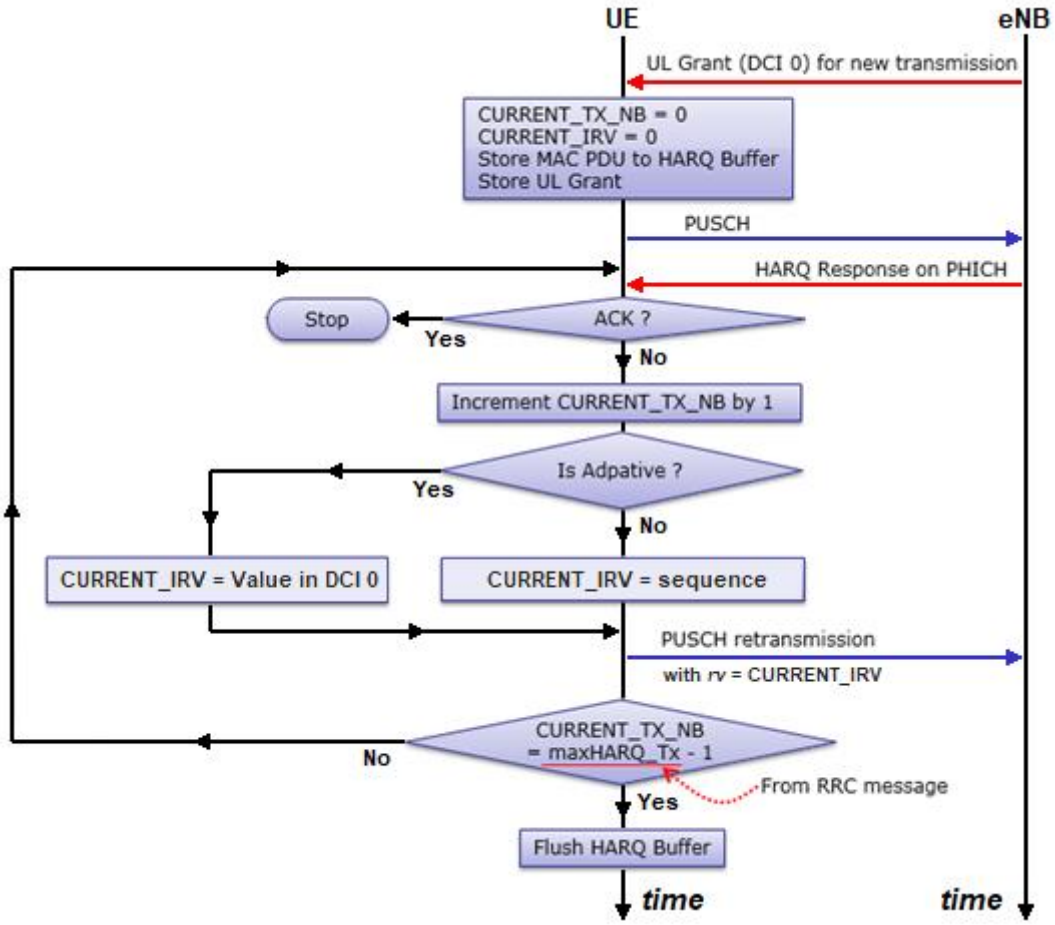

Some **relevant remarks** on this figure are:

- Since the **eNB** uses the **8 HARQ processes round robin**, the **HARQ packet transmissions** of **this process** are spaced away **8 subframes**.
- The label "**sbfrms.**" stands for **subframes**. They are numbered **0** to **7** in the figure to **identify each subframe** with **each** of the **8 HARQ processes**. In this case, subframe 0 is the one used by the HARQ process we are analyzing to transmit its HARQ packets.
- The **eNB** sends the (**N**)**ACK** for **each received HARQ packet** in the **4th subframe after** the one where the **HARQ packet** was **received**.
- The **DCI messages** are sent through **PDCCH**.

### 11.12.3.4 FDD Uplink (synchronous) HARQ protocol in LTE

For **each UE** (i.e., the **HARQ transmitter**), its **HARQ packet** in **each Uplink subframe** is associated in a **round-robin** way to **1** of the **HARQ processes**. Therefore, **each UE** transmits a **HARQ packet** (**either new** or **retransmitted**) over the **same HARQ process number every 8 subframes**. Consequently, the **eNB** knows exactly which **HARQ process** is associated to **each Uplink subframe**. The **eNB** also knows the **redundancy version** "***rv***" because the **eNB** explicitly indicates it to the **UE** (i.e., the **HARQ transmitter**) within a **DCI 0 message**.

The overall processing of a **FDD Uplink** (**synchronous**) **HARQ process** is shown as a **flow diagram** in the following figure (case 1, except **Msg3**):




For more detail on the **FDD Uplink** (**synchronous**) **HARQ process**, refer to clause 5.4.2.2 of TS 36.321.

## 11.12.4 HARQ protocol in UMTS

In **UMTS** the **HARQ protocol** supported **both chase combining** and **incremental redundancy**. In the **incremental redundancy HARQ** configuration, the **first packet transmission** is **coded** with a **punctured 1/3 Turbo code**. In **each successive packet retransmission**, the **code bits** are **usually punctured further** (i.e. only a fraction of the coded bits are chosen) and sent. The **puncturing pattern** used in **each successive retransmission** is **different**, so **different coded bits** are **sent**.

However, in **LTE** and **5G only incremental redundancy HARQ** is supported because ample experience showed that it performs better than **chase combining**.

## 11.13 Cyclic Prefixing

**Cyclic prefixing** consists of **prefixing each symbol** with a **fraction** of its **last part**. The receiver is typically configured to **discard** the **cyclic prefix**, **but** the **cyclic prefix** serves **two relevant purposes**:

- **Reduce inter-symbol interference** (click 12.2.4)

  It provides a **guard interval** with the **previous symbol**, which reduces inter-symbol interference.

- **Reduce inter-carrier interference**

  It causes that the **linear convolution** of a **frequency-selective multipath channel** can be modeled as **circular convolution**, which in turn may transform to the **frequency domain** via a **Discrete Fourier Transform**. This results in **frequency-domain processing**, such as **channel estimation** (click 2.4.5) and **equalization**.

For the **cyclic prefix** to serve its objectives, its duration must be **larger than or equal** to the duration of the **multipath channel** (click 12.2.3). For the actual duration of the **cyclic prefixes** in **5G**, click 2.3.2.

The **cyclic prefix** technique is used **both** when transmitting **Downlink** with **CP-OFDM** (click 11.14) and **Uplink** with **either CP-DFT-s-OFDM** (11.16) or **CP-OFDMA** (11.15)**.**

The concept of a **cyclic prefix** is traditionally associated with **OFDM** systems, although it is also used in single carrier systems to improve the robustness to multipath propagation.

Source web page:

- https://www.techplayon.com/5g-nr-cyclic-prefix-cp-design/
- https://www.electronics-notes.com/articles/radio/multicarrier-modulation/ofdm-cyclic-prefix-cp.php

## 11.14 Orthogonal frequency-division multiplexing (OFDM)

**Orthogonal Frequency Division Multiplexing** (**OFDM**) consists of **one transmitter** that **sends one symbol sequence over each subcarrier** within a **set** of **frequency multiplexed subcarrier**(**s**), and **all** the **subcarriers** are **orthogonal** among themselves. **OFDM** is a **multicarrier** (click 11.3.2) **signal**.

**Each symbol** in the **sequence** may belong to a **modulation constellation** (click 11.2.1) or to a **complex sequence** (click 11.19).

There can be **only one** or **several receivers**. When there are **several receivers**: a) **each receiver selectively receives** the **symbol sequences** in the **subcarriers assigned** to it; b) the **symbol sequence** in **each subcarrier** may be **received** by **only one receiver** (**unicast**) or **by several receivers** (**multicast**).

**Orthogonality requires** a **very precise synchronization** of the **symbol intervals** of **all** the **subcarriers**, a **precise frequency alignment** of **all** the **subcarriers** and the **joint DFT mapping** of **all** the **subcarriers** from the **frequency domain** to the **time domain**.

**Orthogonality** between **subcarriers eliminates crosstalk**, and there is **no need** for **inter-subcarrier guardbands**, **nor** for an individual **filter** for **each subcarrier** (as it is required in conventional **frequency-division multiplexing**).

**OFDM** has **two relevant special subcarriers**: the **null subcarriers** (click 2.2.10.1) and the **DC subcarrier** (click 2.2.10.2).

**OFDM** is usually **combined** with **cyclic prefixing** (click 11.13), and this combination is called **CP-OFDM**. The **cyclic prefix** transmitted in **each symbol interval** within **each subcarrier** is the **final part** of the **symbol transmitted** within the **symbol interval**, and that **symbol** belongs to a **modulation constellation** or to a **complex sequence**.

A **very relevant advantage** of **OFDM** over **single-carrier** schemes is its tolerance to **severe channel conditions** (e.g., narrowband interference and frequency-selective fading in wireless communications or high-frequency attenuation in long copper wires) **without** the need for **complex equalization filters**.

Some disadvantages of **ODFM** over **single-carrier schemes** are:

- It requires **highly linear amplifiers**.
- It has a **high peak-to-average power ratio** (**PAPR**).
- It is sensitive to **Doppler effect** (click 12.1.8), because **Doppler** causes **inter-subcarrier interference**.

**OFDM** is used in the **Downlink** of **5G** and of **LTE**, and it may be used in the **Uplink** of **5G**.

**Two techniques** related to **OFDM** are **orthogonal frequency-division multiple access** (**OFDMA**, click 11.15) and **single-carrier frequency-division multiple access** (**SC-FDMA**, click 11.16).

For a **comparison** between **OFDM**, **OFDMA** and **SC-FDMA**, click 11.18.

For a **comparison** of the **5G processing** between **OFDM**, **OFDMA** and **DFT-s-FDM**, click 10.11.

Source web pages:

- https://en.wikipedia.org/wiki/Orthogonal_frequency-division_multiplexing

## 11.15 Orthogonal frequency-division multiple access (OFDMA)

**Orthogonal frequency division multiple access** (**OFDMA**) consists of **several transmitters**, where **each** of them **sends one symbol sequence over each subcarrier** within a **set** of **frequency multiplexed subcarrier**(**s**) that have been **assigned** to it for a **certain period** of time, and **all** the **subcarriers** from **all transmitters** are **orthogonal** among themselves. **OFDMA** is a **multicarrier** (click 11.3.2) **signal**.

The **subcarriers assigned** to **each transmitter** may be **localized**, **distributed** or **interleaved**, click 11.17.

**Each symbol** in the **sequence** may belong to a **modulation constellation** (click 11.2.1) or to a **complex sequence** (click 11.19). The **cyclic prefix** transmitted in **each symbol interval** within **each subcarrier** is the **final part** of the **symbol transmitted** within the **symbol interval**, and that **symbol** belongs to a **modulation constellation** or to a **complex sequence**.

The **different transmitters** need to **coordinate among themselves what subcarrier** can be used, and at **what time periods**, by **each transmitter**. A given **medium access control** (click 11.11) is required for this **coordination**.

There can be **only one** or **several receivers**. When there are **several receivers**: a) **each receiver selectively receives** the **symbol sequences** in the **subcarriers assigned** to it; b) the **symbol sequence** in **each subcarrier** may be **received** by **only one receiver** (**unicast**) or **by several receivers** (**multicast**).

**Orthogonality requires** a **very precise synchronization** of the **symbol intervals** of **all** the **subcarriers**, a **precise frequency alignment** of **all** the **subcarriers** and a precise **power equalization**, the **three** of them by **all transmitters**, **plus** the **joint DFT mapping** of **all** the **subcarriers** (transmitted by the same system) from the **frequency domain** to the **time domain**.

**Orthogonality** between **subcarriers eliminates crosstalk**, and there is **no need** for **inter-subcarrier guardbands**, **nor** for an individual **filter** for **each subcarrier** (as it is required in conventional **frequency-division multiplexing**).

**OFDMA** has **two relevant special subcarriers**: the **null subcarriers** (click 2.2.10.1) and the **DC subcarrier** (click 2.2.10.2).

**OFDMA** is usually **combined** with **cyclic prefixing** (click 11.13), and this combination is called **CP-OFDMA**.

A **very relevant advantage** of **OFDMA** over **single-carrier** schemes is its tolerance to **severe channel conditions** (e.g., narrowband interference and frequency-selective fading in wireless communications or high-frequency attenuation in long copper wires) **without** the need for **complex equalization filters**.

Some disadvantages of **OFDMA** over **single-carrier schemes** are:

- It requires **highly-linear amplifiers**.
- It has a **high peak-to-average power ratio** (**PAPR**), but this can be solved using the **ODFMA** variant **single-carrier FDMA** (click 11.16).
- It is sensitive to **Doppler effect** (click 12.1.8), because it causes **inter-subcarrier interference**.

**OFDMA** may be **optionally** used in the **Uplink** of **5G** instead of **SC-FDMA**, for **specific** scenarios where **higher flexibility** or **higher spectral efficiency** is required.

**Two techniques** related to **OFDMA** are **orthogonal frequency-domain multiplexing** (**OFDM**, click 11.14) and **single-carrier frequency-division multiple access** (**SC-FDMA**, click 11.16).

For a **comparison** between **OFDM**, **OFDMA** and **SC-FDMA**, click 11.18.

For a **comparison** of the **5G processing** of **OFDM**, **OFDMA** and **DFT-s-FDM**, click 10.11.

## 11.16 Single-carrier frequency-division multiple access (SC-FDMA)

**Single-carrier frequency division multiple access** (**SC-FDMA**) consists of **several transmitters**, where **each** of them **sends one symbol sequence over** a **set** of **frequency multiplexed subcarrier**(**s**) that have been **assigned** to it for a **certain period** of time, such that **each symbol** is **jointly sent** over **all** the **subcarriers assigned** to the **transmitter**, and **all** the **subcarriers** from **all transmitters** are **orthogonal** among themselves. **SC-FDMA** is a **multicarrier** (click 11.3.2) **signal**, but, it exhibits **very**

**similar characteristics** to **single-carrier** (click 11.3.1) **signals**. Since it is a **multicarrier signal**, and it performs a **precoding** before **OFDMA** processing, maybe a **better** term for it would be **linearly precoded OFDMA** (**LP-OFDMA**), as it is sometimes called.

The **subcarriers assigned** to **each transmitter** may be **localized**, **distributed** or **interleaved**, click 11.17.

**Each symbol** in the **sequence** may belong to a **modulation constellation** (click 11.2.1) or to a **complex sequence** (click 11.19). The **cyclic prefix** transmitted in **each symbol interval** within **each subcarrier** is the **final part** of the **signal transmitted** within the **symbol interval**, and that **signal** is **not** a **symbol** that belongs to a **modulation constellation** or to a **complex sequence**. That **signal** is rather a **combination** of a **fraction** of **as many symbols as subcarriers** have been **assigned** to the **transmitter**.

The **different transmitters** need to **coordinate among themselves what subcarrier** can be used, and at **what time periods**, by **each transmitter**. A given **medium access control** (click 11.11) is required for this **coordination**.

There can be **only one** or **several receivers**. When there are **several receivers**: a) **each receiver indiscriminately receives** the **symbol sequence** contained in **all** the **subcarriers sent** by the **transmitter**; b) the **symbol sequence** may be **received** by **only one receiver** (**unicast**) or **by several receivers** (**multicast**).

**Orthogonality requires** a **very precise synchronization** of the **symbol intervals** of **all** the **subcarriers** and a **precise frequency alignment** of **all** the **subcarriers**, a **precise power equalization**, the **three** of them by **all transmitters**, **plus** the **joint DFT mapping** of **all** the **subcarriers** (transmitted by the same system) from the **frequency domain** to the **time domain**.

**Orthogonality** between **subcarriers eliminates crosstalk**, and there is **no need** for **inter-subcarrier guardbands**, **nor** for an individual **filter** for **each subcarrier** (as it is required in conventional **frequency-division multiplexing**).

**SC-FDMA** has **two relevant special subcarriers**: the **null subcarriers** (click 2.2.10.1) and the **DC subcarrier** (click 2.2.10.2).

**SC-FDMA** is usually **combined** with **cyclic prefixing** (click 11.13), and this combination is called **SC-FDMA-CP**.

A **very relevant advantage** of **SC-FDMA** over **actual single-carrier** schemes is its tolerance to **severe channel conditions** (e.g., narrowband interference and frequency-selective fading in wireless communications or high-frequency attenuation in long copper wires) **without** the need for **complex equalization filters**.

Some disadvantages of **SC-FDMA** over **actual single-carrier** schemes are:

- It requires **highly linear amplifiers**.
- It is sensitive to **Doppler effect** (click 12.1.8), because it causes **inter-subcarrier interference**.

**SC-FDMA** is used in the **Uplink** of **LTE**, and it is the **primary** technique used in the **Uplink** of **5G**. The specific **implementation** of **SC-FDMA** used in **Uplink** of **5G** is called **discrete Fourier transform spread orthogonal frequency division multiplexing** (**DFT-s-OFDM**).

**Two techniques** related to **SC-FDMA** are **orthogonal frequency-domain multiplexing** (**OFDM**, click 11.14) and **orthogonal frequency-division multiple access** (**OFDMA**, click 11.15).

For a **comparison** between **OFDM**, **OFDMA** and **SC-FDMA**, click 11.18.

For a **comparison** of the **5G processing** of **OFDM**, **OFDMA** and **DFT-s-FDM**, click 10.11.

**SC-FDMA** is sometimes called "**Linearly-Precoded OFDMA**" (**LP-OFDMA**) or "**Discrete Fourier Transform OFDMA**" (**DFT-OFDMA**). In its initial stages it was called "**subcarrier interferometry**".

Source web page:

- https://www.gta.ufrj.br/ensino/eel879/trabalhos_vf_2014_2/rafaelreis/ofdma_scfdma.html
- https://en.wikipedia.org/wiki/Single-carrier_FDMA

## 11.17 Localized, interleaved and distributed subcarrier mapping

**Subcarrier mapping** consists of a **technique** to **assign** a **specific set** of **subcarriers** to **each transmitter** (i.e., each **UE**) in a **frequency-division multiple access technique**, such as **OFDMA** (click 11.15) or **SC-FDMA** (click 11.16).

Three **relevant subcarrier mapping techniques** are:

- **Localized mapping**

  The **subcarriers assigned** to **each transmitter** are **consecutive**. **Each transmitter** therefore sends its **signal** over the **minimum possible frequency segment** within the **channel**.

- **Interleaved mapping**

  The **subcarriers assigned** to **each transmitter** are **not consecutive** and are rather **equally distributed** over the **entire channel bandwidth**. **Each transmitter** therefore sends its **signal** over the **entire channel bandwidth**. **Each transmitter** sends **no signal** (**zero** amplitude) in the **subcarriers** that are placed **between** the **subcarriers assigned** to it.

- **Distributed mapping.**

  The **subcarriers assigned** to **each transmitter** are **not consecutive** and are **not equally distributed** over the **entire channel bandwidth**. **Each transmitter** therefore sends its **signal** over a **larger bandwidth than required**. **Each transmitter** sends **no signal** (**zero** amplitude) in the **subcarriers** that are placed **between** the **subcarriers assigned** to it.

An **advantage** of **interleaved** and **distributed mappings** is that they provide **frequency diversity** (click 11.8) which improves **reliability**. However, they **distribute** the **subcarriers** over a **wider bandwidth** than required, which has the **disadvantage** of **increasing** the **signal PAPR** (click 11.4).

## 11.18 Comparison between OFDM, OFDMA and SC-FDMA

The following table presents some **relevant characteristics** of **OFDM** (click 11.14), **OFDMA** (click 11.15) and **SC-FDMA** (click 11.16):

| | **OFDM** | **OFDMA** | **SC-FDMA** |
|---|---|---|---|
| **Carrier type** | Multicarrier | | |
| **Purpose** | Multiplexing | Multiple access | |
| **Transmitter(s)** | One | Several | |
| **PAPR** | High | | Low |
| **Spectral efficiency** | Slightly lower in SC-FDMA | | |
| **Subcarrier processing** | Yes | | |
| **Usually with CP** | Yes | | |
| **DFT precoding** | No | | Yes |

Some **relevant remarks** on this table are:

- **SC-FDMA** is **multicarrier** (in spite of its name). It is **multicarrier** because its waveform is **subcarrier based**, which is **particularly clear** when **interleaved** or **distributed subcarrier mapping** (click 11.17) is used. Even though **DFT-s-OFDM** is a **multicarrier signal** is name includes the term "**single-carrier**" because some of the properties of **SC-FDMA** (in particular a **low PAPR**, click 11.4) are similar to the ones of a **single-carrier signal**.
- **SC-FDMA** has a **low PAPR**, **unless interleaved** or **distributed subcarrier mapping** (click 11.17) is used.

**SC-FDMA** has a **lower spectral efficiency** than **OFDMA**, but it also has a **lower peak-to-average power ratio** (**PAPR**). Because of its **lower PAPR**, **SC-FDMA** is **more power efficient**, and also allows for **cheaper amplifiers**. Both advantages are very important for its implementation in the **UE**, and this is why **SC-FDMA** is used **Uplink both** in **5G** and in **LTE**.

Because **OFDM** is a **multiplexing** technique, there is a **single transmitter** that can perform the **DFT mapping** from the **frequency domain** to the **time domain** in a **joint manner** for **all** the **subcarriers** and **symbol intervals**.

Because **OFDMA** and **SC-FDMA** are **multiple access** techniques, they have **several transmitters**: **each** of the transmitters performs the **DFT mapping** from frequency to time in a **joint manner** for **all** its **subcarriers** and **symbol intervals**, but the **overall** combined signal has **not** been produced by **only one transmitter**, as it is done in **OFDM**. This may cause some differences in the way the **combined signal resulting** from **multiple transmitters** is processed at the **receiver**.

## 11.19 Discrete Fourier Transform (DFT) and its inverse (iDFT)

The following sections present different aspects of the **Discrete Fourier Transform** (**DFT**) and of its **inverse** (**iDFT**)

Source web pages:

- https://www.youtube.com/watch?v=QmgJmh2I3Fw
- https://www.youtube.com/watch?v=zqVCqmRQUxo

- https://www.youtube.com/watch?v=nl9TZanwbBk

### 11.19.1 Overview of Fourier transform and its inverse

**Fourier** demonstrated that **any function** "***BS***()" that is represented in the **time-domain** as "***BS***(*t*)" is **exactly** the **same** as a certain **weighted sum** (an **integral**) of **negative** and **positive phasors** (click 11.19.6), **each** of them with a specific **frequency** and **phase**. The value of **each phasor** as a function of the **phasor frequency** is the **frequency-domain representation** "***BS***(*f*)" of the function "***BS***()". **Fourier** also provided a mathematical expression to obtain "***BS***(*f*)" from **any** "***BS***(*t*)". This mathematical expression is the **Fourier transform**.

Consequently, it is possible to obtain the **time-domain representation** "***BS***(*t*)" of **any frequency-domain representation** "***BS***(*f*)" by applying its definition, this is, by computing the **weighted sum** of the **sinusoids** of **different frequencies**. This computation is the **inverse Fourier transform**.

If you apply the **inverse Fourier transform** to a **frequency-domain representation** "***BS***(*f*)" that was obtained by applying the **Fourier transform** to the **time-domain representation** "***BS***(*t*)", the result is the **exact original time-domain representation** "***BS***(*t*)".

If you apply the **Fourier transform** to a **time-domain representation** "***BS***(*t*)" that was obtained by applying the **inverse Fourier transform** to the **frequency-domain representation** "***BS***(*f*)", the result is the **exact original frequency-domain representation** "***BS***(*f*)".

### 11.19.2 Overview of DFT and iDFT

The **Discrete Fourier Transform** (**DFT**) converts "***N***" **consecutive time samples** "***BS***($t_i$)" of the **complex-valued function** "***BS***(*t*)" that represents a **baseband signal** "***BS***()" in the **time domain**, into the **same quantity** "***N***" of **consecutive frequency samples** "***BS***($f_i$)" of the **complex-valued function** "***BS***(*f*)" that represents the same **baseband signal** "***BS***()" in the **frequency domain**.

The **inverse Discrete Fourier Transform** (**iDFT**) converts "***N***" **consecutive frequency samples** "***BS***($f_i$)" of the **complex-valued function** "***BS***(*f*)" that represents a **baseband signal** "***BS***()" in the **frequency domain**, into the **same quantity** "***N***" of **consecutive time samples** "***BS***($t_i$)" of the **complex-valued function** "***BS***(*t*)" that represents the same **baseband signal** "***BS***()" in the **time domain**.

The **time instants** "$t(t_i)$" where the **baseband signal samples** "$BS(t_i)=BS(t(t_i))$" are taken are **evenly spaced**.

The **frequency values** "$f(f_i)$" where the **baseband signal samples** "$BS(f_i)=BS(f(f_i))$" are taken are **evenly spaced**.

There is **only one baseband signal** "***BS***()" but it has **two different representations** "***BS***(*t*)" and "***BS***(*f*)", the former in the **time domain** and the latter in the **frequency domain**. **Both** functions "***BS***(*t*)" and "***BS***(*f*)" represent the **same baseband signal** "***BS***()", but they are **different functions** "$BS(t) \neq BS(f)$".

Applying the **iDFT** to the **frequency-domain samples** "***BS***($f_i$)" that resulted from having applied the **DFT** to the **time-domain samples** "***BS***($t_i$)" produces **exactly** the **same original time-domain samples** "***BS***($t_i$)".

Applying the **iDFT** to the **time-domain samples** "$BS(t_i)$" that resulted from having applied the **DFT** to the **frequency-domain samples** "$BS(f_i)$" produces **exactly** the **same original frequency-domain samples** "$BS(f_i)$".

Some relevant **symbols**, **terms** and **definitions** are:

- "$BS(t)$" is the **complex-valued baseband signal** represented in the **time domain**.
- "$BS(f)$" is the **complex-valued baseband signal** represented in the **frequency domain**.
- "$N$" is the "**DFT size**" or the "**iDFT size**". It is the **quantity** of **time domain samples** and the **quantity** of **frequency domain samples**, which is the **same**. It is **unitless**.
- "$N$-**point DFT**" or "$N$-**point iDFT**" is a **DFT** or **iDFT** with **size** "$N$".
- "$t_i$" is the **index** of the **time instants** where the "$N$" **time-domain samples** of "$BS(f)$" are taken. "$t_i$" ranges from **0** to "$N$-1".
- "$f_i$" is the **index** of the **frequency values** where the **frequency-domain samples** of "$BS(f)$" are taken. "$f_i$" ranges from **0** to "$N$-1".
- "$t(t_i)$" is the **time instant** that corresponds to **index** "$t_i$".
- "$f(f_i)$" is the **frequency value** that corresponds to **index** "$f_i$".
- "$BS(t(t_i))$" is a **time sample value**. It is the value of "$BS(t)$" at **time instant** "$t(t_i)$". By convention, "$BS(t_i)=BS(t(t_i))$".
- "$BS(f(f_i))$" is a **frequency sample value**. It is the value of "$BS(f)$" at **frequency value** "$f(f_i)$". The values of the **frequency-domain samples** are frequently called the "**frequency coefficients**" or the "**coefficients**". By convention, "$BS(f_i)=BS(f(f_i))$".
- "$T_{si}$" is the "**sampling interval**" or "**sampling period**" or "**frequency resolution**". It is the **time difference between** the **time instants** that correspond to **every two consecutive time-domain sample indexes** "$t(t_i)$" and "$t(t_i+1)$":

  $$T_{si}=t(t_i+1)-t(t_i)$$

  Notice that **improving** the **time resolution** (i.e., a **finer-graded time resolution**) implies a **smaller sampling interval** "$T_{si}$" and vice versa.
- "$\Delta f$" is the "**frequency resolution**", the "**frequency bin width**" or the "**bin width**". It is the **frequency difference between** the **frequency values** that correspond to **two consecutive frequency-domain sample indexes** "$f(f_i)$" and "$f(f_i+1)$":

  $$T_{si}=f(f_i+1)-f(f_i)$$

  Notice that **improving** the **frequency resolution** (i.e., a **finer-graded frequency resolution**) implies a **smaller bin width** "$\Delta f$" and vice versa.
- A **sampling interval** is **each time interval** "$(t(t_i), t(t_i+1))$". Notice that it is centered at "$t(t_i)+0.5T_{si}$" and its **duration** is "$T_{si}$". Notice that it is here defined as an **open interval** (by convention). Notice that its definition is **not** the **same** as the one of a **frequency bin**.

- A **frequency bin** is **each frequency interval** "[$f(f_i)$-$\Delta f/2$, $f(f_i)$+$\Delta f/2$)". Notice that it is centered at "$f(f_i)$" and its **width** is "$\Delta f$". Notice that it is a **half-open interval**. Notice that its definition is **not** the **same** as the one of a **sampling interval**.
- "$F_s$" is the **sampling rate**. It is the **rate** with which **time-domain samples** are **taken**.
- "$T_{to}$" is the **total sampled time** or **sampled time** It is the **total time** of **all** the **sampling intervals**. It is the **total time** between "$t(t_{(N-1)})$" and "$t(t_0)$". Its units are **seconds**.
- "$W_{to}$" is the **DFT bandwidth**. It is the **total bandwidth** of **all** the **frequency bins**. It is the **bandwidth** between "$max(f(f_i))$" and "$min(f(f_i))$" plus "$\Delta f$":

  $$W_{to}=(max(f(f_i))+\Delta f/2) - (min(f(f_i))-\Delta f/2)$$

  Notice that "$min(f(f_i))$" is **always negative**. Its **units** are **Hertzs**.

Some relevant **relationships between** five of the **(i)DFT variables** are:

- The **sampling interval** "$T_{si}$" can be expressed as:

  $$T_{si} = 1/F_s = T_{to}/(N\text{-}1) = T_{to}/((W_{to}/\Delta f)\text{-}1)$$

- The **bin width** "$\Delta f$" can be expressed as:

  $$\Delta f = 1/T_{to} = 1/((N\text{-}1)\bullet T_{si}) = F_s/(N\text{-}1)$$
  $$= W_{to}/N = W_{to}/((T_{to}/T_{si})+1) = W_{to}/((T_{to}\bullet F_s)+1)$$

- The **sampling rate** "$F_s$" can be expressed as:

  $$F_s = 1/T_{si} = (N\text{-}1)/T_{to} = (N\text{-}1)/T_{to} = W_{to}\bullet(N\text{-}1)/N$$

- The **total sampled time** "$T_{to}$" can be expressed as:

  $$T_{to} = (N\text{-}1)\bullet T_{si} = (N\text{-}1)/F_s = N/W_{to}$$

- The **DFT bandwidth** "$W_{to}$" can be expressed as:

  $$W_{to} = N\bullet\Delta f = N/T_{to} = ((T_{to}/T_{si})+1)\bullet\Delta f = ((T_{to}\bullet F_s)+1)\bullet\Delta f$$

- The **DFT size** "$N$" can be expressed as:

  $$N = W_{to}/\Delta f = (T_{to}/T_{si})+1 = (T_{to}\bullet F_s)+1$$

In **5G** and in and **most practical applications**, the **time domain complex valued samples** "$BS(t_i)$" are **quantified**, and consequently, they are represented in the **digital domain** (**bit sequences**) and **not** in the **analog domain** (**real numbers**).

## 11.19.3 Intuitive formulation of DFT and iDFT

The **DFT** can be formulated as:

$$BS(f_i) = 1/N\bullet\Sigma_{ti}\ [BS(t_i)\bullet e^{-j2\pi f(f_i)\bullet t(t_i)}]$$

$$BS(f_i) = 1/N\bullet\Sigma_{ti}\ [BS(t_i)\bullet(cos(2\pi f(f_i)\bullet t(t_i)) - j\bullet sin(2\pi f(f_i)\bullet t(t_i))]$$

In which case, the **iDFT** is formulated as:

$$BS(t_i) = \Sigma_{fi}\ [BS(t_i)\bullet e^{+j2\pi t(t_i)\bullet f(f_i)}]$$

$$BS(t_i) = \Sigma_{fi}\ [BS(f_i)\bullet(cos(2\pi t(t_i)\bullet f(f_i)) + j\bullet sin(2\pi t(t_i)\bullet f(f_i)))]$$

where:

- The **range** for "$t_i$" and for "$f_i$" is:

  from "0" to "$N$", in steps of **1**.

- The **indexed values** of the **time instants** "$t(t_i)$" of the **time-domain samples indexed** by "$t_i$" are:

| $t_i$ | 0 | 1 | 2 | … | $N$-2 | $N$-1 |
|---|---|---|---|---|---|---|
| $t(t_i)$ | 0 | $T_{si}$ | $2T_{si}$ | … | $(N-2)T_{si}$ | $(N-1)T_{si}$ |

- The **range** of the **frequency values** "$f(f_i)$" of the **frequency-domain samples** is:

  from "$-n \bullet \Delta f$" to "$m \bullet \Delta f$", in steps of "$\Delta f$".

  where "$n=floor((N-1)/2)$" and "$m=ceiling((N-1)/2)$".

  Notice that **either** "$n=m$" or "$n=m-1$". Consequently, there are **roughly** as many "$n$" **negative frequencies** as "$m$" **positive frequencies**, **either** of which is **roughly** "$N/2$".

  Notice that "$n+m+1=N$", and the "1" corresponds to "$f=0$"

  Notice that "$m \bullet \Delta f$" is **lower** or **equal** than the **Nyquist frequency** "$N/2 \bullet \Delta f=1/(2T_{si})$".

- The **values** of the **frequency values** "$f(f_i)$" of the **frequency-domain samples indexed** by "$f_i$" are:

| $f_i$ | $N-n$ | $N-(n-1)$ | … | $N$-2 | $N$-1 | 0 | 1 | 2 | … | $m$-1 | $m$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $f(f_i)$ | $-n \bullet \Delta f$ | $-(n-1) \bullet \Delta f$ | … | $-2\Delta f$ | $-\Delta f$ | **0** | $\Delta f$ | $2\Delta f$ | … | $(m-1) \bullet \Delta f$ | $m \bullet \Delta f$ |

  Notice that the **indexing** is **split** in **roughly 2 halves**. This has been done to keep the **indexing** the **same** as in the **series formulation** (click 11.19.5) of the **DFT** and **iDFT**.

- Recall that "***cosine***()" is an **even** function: "$cosine(-f)=cosine(f)$"
- Recall that "***sine***()" is an **odd** function: "$sine(-f)=-sine(f)$"

The expression **either** for the **DFT** or for the **iDFT** is **somehow complex**, but **each** of them represents a **simple concept**:

- The **DFT** calculates **each frequency-domain sample** "$BS(f_i)$" as the **correlation** between the **time-domain signal** "$BS(t)$" and a **complex sinusoid** (click 11.19.7) of the corresponding **frequency** "$f(f_i)$". Consequently, **each frequently-domain sample** "$BS(f_i)$" is the "**weight**" that a ***cosine*** and a ***sine*** of the corresponding **frequency** "$f(f_i)$" has in the **time domain signal** "$BS(t)$".

  The **weight** (i.e., the **correlation**) "$BS(f_i)$" of **each frequency** "$f(f_i)$" is obtained by computing the **products** of **each** and **every time-domain sample** "$BS(t_i)$" and its **corresponding value** of the **complex sinusoid**, which is:

  $$cos(2\pi f(f_i) \bullet t(t_i)) - j \bullet sin(2\pi f(f_i) \bullet t(t_i))$$

  and adding **all** of them with "$\Sigma_{ti}$". The **corresponding value** of the **complex sinusoid** is its value at the **same sampling instant** "$t(t_i)$" of the **time-domain sample** "$BS(t_i)$".

Notice that the **variable** of the **complex sinusoid** is "$t(t_i)$" while "$f(f_i)$" is a **fixed parameter** for **each frequency sample**.

- The **iDFT** calculates **each time-domain sample** "$BS(t_i)$" as the **weighted sum** of the **values** of **all** the **phasors** with **frequencies** "$f(f_i)$" at the corresponding **time instant** "$t(t_i)$". This is just the application of the definition (click 11.19) of the **Fourier**'s **frequency-domain representation** of a **signal**.

  **Each element** of the **weighted sum** "$\Sigma_{fi}$" at **time instant** "$t(t_i)$" is the **product** of the **weight** "$BS(f_i)$" (also called **coefficient**) of **1 component phasor** with frequency "$f(f_i)$".

  Notice that the **variable** of the **complex sinusoid** is "$f(f_i)$" while "$t(t_i)$" is a **fixed parameter** for **each time sample**.

## 11.19.4 Remarks on DFT and iDFT

### 11.19.4.1 DFT, iDFT and multicarrier signals

The application of **DFT** and **iDFT** to **multicarrier telecommunications signals** (click 11.3.2) is performed by **aligning each subcarrier** (click 2.2.10) with **each frequency bin** (click 11.19.2).

**Each transmitted symbol** from the **digital modulation constellation** becomes **the coefficient** of **the frequency bin** that corresponds to **the subcarrier** that is hosting the transmitted symbol. Shaping the **signal** (click 2.5) in this way results in **orthogonal subcarriers** with the advantage of **minimum cross interference**.

The **additional (i)DFT frequency bins** over the **quantity** of **subcarriers** hosted by the **channel**'s **transmission bandwidth** are **unused** (i.e., their **coefficient** is **zero**). Notice that the **DFT-size** "$N$" is usually a **power** of **2**, while the **quantity** of **subcarriers** in the **channel**'s **transmission bandwidth** varies with a much **finer granularity**.

### 11.19.4.2 DFT and iDFT size "$N$"

The **minimum** valid value of "$N$" is **jointly** determined by the **sampled time** "$T_{to}$" and by the **highest frequency component** "$F_{max}$" of the **signal**. The **Nyquist sampling requirement** establishes that the **sampling frequency** "$F_s$" has to be **larger** than **twice** the **highest frequency component** "$F_{max}$" of the **baseband signal** "$BS(f)$":

$$F_s > 2F_{max}$$

$$1/T_{si} > 2F_{max}$$

$$(N\text{-}1)/T_{to} > 2F_{max}$$

$$N > 2F_{max} \bullet T_{to} + 1$$

Likewise, the **highest frequency component** of the **signal** "$F_{max}$" has to be **smaller** than **half** the **total DFT bandwidth** "$W_{to}/2$", because otherwise the **iDFT** could **not** reconstruct the **signal**:

$$W_{to} > 2F_{max}$$

$$N \bullet \Delta f > 2F_{max}$$

$$N/T_{to} > 2F_{max}$$

$$N > 2F_{max} \bullet T_{to}$$

This is a **different reasoning** to arrive at a **similar requirement**.

### 11.19.4.3 DFT and iDFT units

**DFT** and **iDFT** are **unitless**, i.e., they do **not** have any **physical unit**. **DFT** and **iDFT** are computed over "$N$", "$t_i$" and "$f_i$", **all** of which are **unitless integer numbers**. It is true that the **signal** "$BS(t_i)$" and "$BS(f_i)$" might associated to some **physical unit**, but this **unit** is **irrelevant** for the computation of their **DFT** and **iDFT**.

It is then possible to conclude that in **DFT** and **iDFT**:

- The **signal units** may be **anything** (V, mV, W, mW, kilo pascals, pH, wind speed, Joules, calories, …).
- The **time** is **measured** and **computed** in **unitless time sample indexes** "$t_i$".
- The **frequency** is **measured** and **computed** in **unitless frequency sample indexes** "$f_i$".

The **dimensions** of **time** and **frequency** are introduced when defining **either** the **total sampled time** "$T_{to}$" or the **DFT bandwidth** "$W_{to}$":

When defining the **total sampled time** "$T_{to}$":

- $T_{si} = T_{to}/(N\text{-}1)$ seconds
- $\Delta f = 1/T_{to}$ Hertz
- $F_s = (N\text{-}1)/T_{to}$ Hertz
- $W_{to} = N/T_{to}$ Hertz

When defining the **DFT bandwidth** "$W_{to}$": $N/W_{to}$

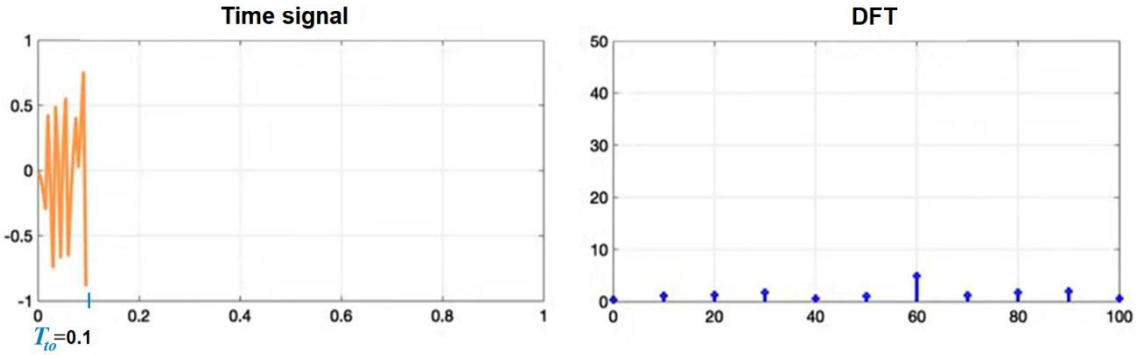
- $T_{si} = N/(W_{to} \bullet (N\text{-}1))$ seconds
- $\Delta f = 1/T_{to}$ Hertz
- $F_s = W_{to} \bullet (N\text{-}1)/N$ Hertz
- $T_{to} = N/W_{to}$ seconds

### 11.19.4.4 Improving the frequency resolution

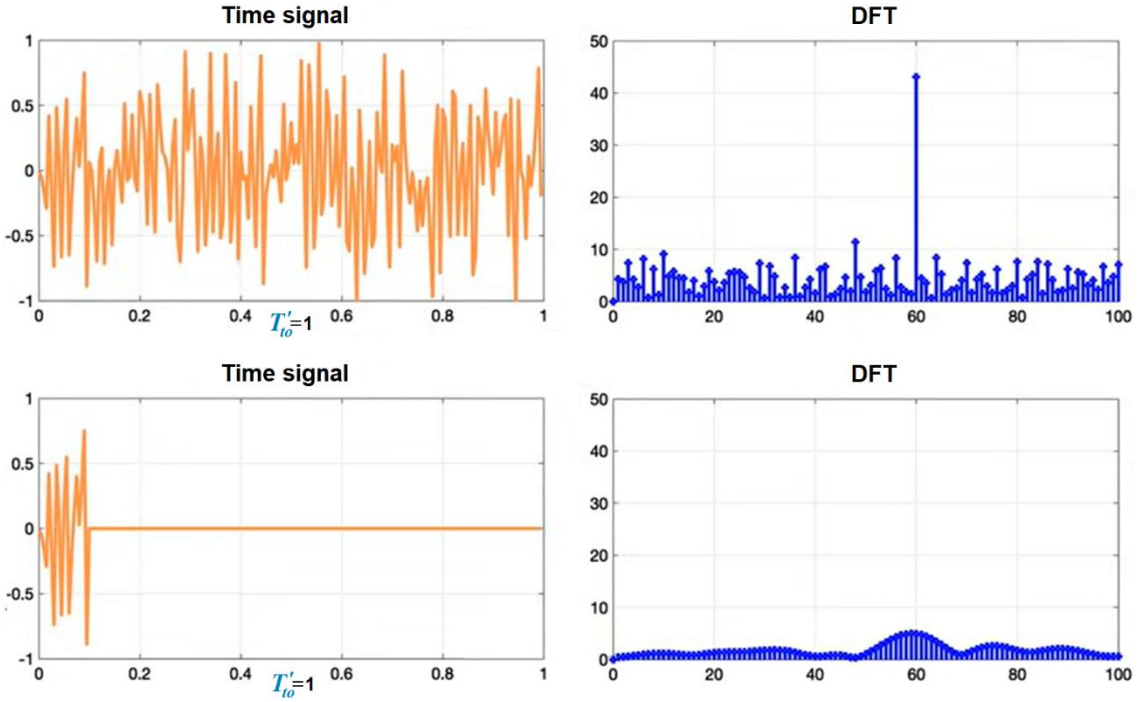
**Improving** the **frequency resolution** consists of reducing the **frequency bin** "$\Delta f$". This is achieved by **increasing** the **sampled time** "$T_{to}$" while **keeping** the **sampling interval** "$T_{si}$" (i.e., **keeping** the **frequency resolution**). This also implies a **larger** (**i**)**DFT size** "$N$". **Increasing** the **sampled time** from "$T_{to}$" to "$T_{to}'=k \bullet T_{to}$" (while **keeping** the **sampling interval** "$T_{si}$") reduces the frequency bin from "$\Delta f$" to "$\Delta f'=\Delta f/k$", and requires (**i**)**DFT size** "$N'=k \bullet N$".

But a **better frequency resolution** is **only** achieved as long as the **increased sampled time** "$T'_{to}$" implies an **increased signal time**. ***If*** the **signal time** only covers "$T_{to}$", ***then***, the **sampled time** has to be **padded** with **zero** from "$T_{to}$" to "$T'_{to}=k \bullet T_{to}$". This **does not really increase** the **frequency resolution**. The reason is that the **quantity** of **valid signal time samples** "$N$" stays the **same** as "$N$", and the remaining "$(k\text{-}1) \bullet N$" **time samples** do **not** add any **information** because they are just **padding**.

The following figure shows an example of a **signal** with a **total sampled time** "$T_{to}$" of **0.1 sg**, with **DFT-size** "***N***" equal **10**:




Based on the example from the previous figure, the following figure shows the results of **increasing ten times** the **sampled time**, from "$T_{to}$=0.1" to "$T'_{to}$=1". The **two top graphs** show the case where the **frequency resolution** is actually **increased**, while the **two bottom graphs** show the case where the **frequency resolution** is **not increased**:




You may find a **detailed explanation** at:

- https://www.youtube.com/watch?v=QmgJmh2I3Fw

### 11.19.4.5 Improving the time resolution

**Improving** the **time resolution** consists of **reducing** the **sampling interval** "$T_{si}$". This is achieved by **increasing** the **DFT size** "***N***" while **keeping** the **sampled time** "$T_{to}$". **Increasing** the **DFT size** from "***N***" to "$N'=N \bullet T_{to}$" (while **keeping** the **sampled time** "$T_{to}$") reduces the **sampling interval** from "$T_{si}$" to "$T'_{si}=T_{si}/k$" and **increases** the **total bandwidth** from "$W_{to}$" to "$W'_{to}=N \bullet W_{to}$".

Oversampling

**Improving** the **time resolution** is also called "**oversampling**", because it implies handling **more time samples than** it is actually **required**.

### 11.19.4.6 Fast Fourier Transform

The **FFT** is a **computing algorithm** (e.g., like the **bubble algorithm** to **sort lists**). The **FFT** is **not** a **signal transformation**.

Computation the **DFT** or **iDFT** directly from their **mathematical formulation** (click 11.19.5) implies **either** a **N-sized loop** with **another N-sized nested loop** or a **matrix multiplication** between an **N-sized vector** and an **N×N square matrix**. This implies a quantity of operations in the order of "$N^2$".

The **FFT algorithm** makes advantage of **some mathematical properties** and of **symmetries** within the **N×N matrix**. The result is an **accurate computation** with a **substantially lower quantity** of **operations** than the direct computation from the **mathematical formulation**. This has a **double advantage** of **saving both processing time** and **energy consumption**.

In **telecommunications applications**, the **DFT size** "$N$" is usually **large** (e.g., in the **thousands**), which implies a **quantity** of **operations** in the **millions**. Moreover, the **DFT** has to be computed in **real time** for **each symbol interval** (click 2.3.2). A **5G symbol interval** has a duration from ~1.1 µs to 71 µs. This clearly shows that an **efficient computation algorithm** is extremely important.

The **FFT** is therefore a **computing algorithm** and it is **not** a **signal transformation**. The **DFT**, **iDFT**, **Z-tranform** or **Laplace transform** are **actual signal transformations**.

For more detail on **FFT**, refer to:

- https://en.wikipedia.org/wiki/Fast_Fourier_transform

### 11.19.4.7 Correcting factor "1/$N$"

The **correcting factor** "1/$N$" used in the **intuitive formulation** (click 11.19.3) of **DFT** is required because the **correlation** between the **time-domain samples** "$BS(t_i)$" and the **complex sinusoid** "$e^{-j2\pi f(fi)\cdot t(ti)}$" increases the **amplitude** by a factor of "$N$".

Notice that the **only requirement** is that the **DFT value** is "$1/N^{th}$" of the **iDFT** value, and this can be achieved in **infinite ways**. The **DFT** and the **iDFT** are **correctly expressed** in the following way with **any two** "$A$" and "$B$" that satisfy "$A/B = 1/N$":

**DFT** formulated as: $BS(f_i) = A \cdot \Sigma_{ti} \left[ BS(t_i) \cdot e^{-j2\pi f(fi)\cdot t(ti)} \right]$

**iDFT** formulated as: $BS(t_i) = B \cdot \Sigma_{fi} \left[ BS(t_i) \cdot e^{+j2\pi t(ti)\cdot f(fi)} \right]$

The most frequent formulation of **DFT** and **iDFT** is with "$A$=1" and "$B$=$N$".

### 11.19.4.8 Sign of phasor/frequency "$e^{+j}$" or "$e^{-j}$"

The **only requirement** in respect to the **sign** of the **frequency** (i.e., the sign of the **exponent** of the **phasor**) being "+" or "-" is that the **DFT** and the **iDFT** have **different signs**. Consequently, the **DFT** and **iDFT** could also be formulated as:

**DFT** formulated as: $BS(f_i) = A \cdot \Sigma_{ti} \left[ BS(t_i) \cdot e^{+j2\pi/N \cdot f(fi)\cdot t(ti)} \right]$

**iDFT** formulated as: $BS(t_i) = B \cdot \Sigma_{fi} \left[ BS(t_i) \cdot e^{-j2\pi/N \cdot t(ti)\cdot f(fi)} \right]$

### 11.19.4.9 Real-valued baseband signals

The **mathematical formulations** (click 11.19.5) of **(i)DFT** indicate that the **quantity** "$N_t$" of **time samples** is the **same** as the **quantity** "$N_f$" of **frequency samples**, which is denoted "$N$". Therefore, an ***N*-point iDFT implies** an ***N*-point DFT** and **vice versa**.

However, for the **specific** case of a **real-valued baseband signal** in the **time-domain**, the **quantity** of **frequency samples** can be "$N_f = floor(N_t/2)$". This is possible because **every real-valued baseband signal** "$BS(t)$" has a **frequency-domain representation** "$BS(f)$" that is **conjugate** (i.e., Hermitian) **symmetric** in respect to the **frequency axis**. It is then possible to **produce** the **frequency samples** with **negative frequency** from the **frequency samples** with **positive frequency**.

## 11.19.5 Series formulation of DFT and iDFT

The **DFT** and **iDFT** is **usually formulated** as the following series:

**DFT** formulated as: $BS(f_i) = \Sigma_{ti} [BS(t_i) \bullet e^{-j2\pi/N \bullet fi \bullet ti}]$

**iDFT** formulated as: $BS(t_i) = N \bullet \Sigma_{fi} [BS(t_i) \bullet e^{+j2\pi/N \bullet ti \bullet fi}]$

where the **range** for "$t_i$" and for "$f_i$" is from "0" to "$N$", in steps of **1**.

It may **seem** that this **formulation** is **not equivalent** to the **intuitive formulation** (click 11.19.3), but it actually is **completely equivalent**. Let us analyze the **apparent differences**:

**Correcting factor "*N*"**

In this formulation, the **DFT** has a correcting factor "$A$=1", and the **iDFT** has a correcting factor "$B$=$N$". However, in the **intuitive formulation** "$A$=1/$N$" and "$B$=1". This is **completely equivalent**, as it is discussed in 11.19.4.2.

**Correlating phasors** "$e^{\pm j2\pi/N \bullet fi \bullet ti}$"

In this formulation, the **correlating phasors** are "$e^{\pm j2\pi/N \bullet fi \bullet ti}$", while in the **intuitive formulation** they are "$e^{\pm j2\pi f(fi) \bullet t(ti)}$", with the respective **ranges**.

To **prove** that they are the **same**, let us compute the resulting **exponent values**, divided by "$j2\pi$" , of **both formulations**.

Recall from 11.19.2 that "$\Delta f = 1/(N \bullet T_{si})$".

Recall from 11.19.3 that "$n = floor((N-1)/2)$" and "$m = ceiling((N-1)/2)$".

**For non-negative values of "$f(f_i)$"**

The **resulting exponent values** for **non-negative values** of "$f(f_i)$" are:

| | $f_i$ | 0 | | | | | |
|---|---|---|---|---|---|---|---|
| | $t_i$ | 0 | 1 | 2 | … | $N$-2 | $N$-1 |
| **Series** | $1/N \bullet f_i \bullet t_i$ | 0•0 | 0 | 0 | 0 | 0 | 0 |
| **Intuitive** | $f(f_i) \bullet t(t_i)$ | 0•0 | 0 | 0 | 0 | 0 | 0 |
| | $f(f_i)$ | 0 | | | | | |
| | $t(t_i)$ | 0 | $1T_{si}$ | $2T_{si}$ | … | $(N-2)T_{si}$ | $(N-1)T_{si}$ |

| | $f_i$ | 1 | | | | | |
|---|---|---|---|---|---|---|---|
| | $t_i$ | 0 | 1 | 2 | … | $N$-2 | $N$-1 |
| **Series** | $1/N \cdot f_i \cdot t_i$ | 0 | $1/N$ | $2/N$ | … | $(N-2)/N$ | $(N-1)/N$ |
| **Intuitive** | $f(f_i) \cdot t(t_i)$ | 0 | $1/N$ | $2/N$ | … | $(N-2)/N$ | $(N-1)/N$ |
| | $f(f_i)$ | $\Delta f$ | | | | | |
| | $t(t_i)$ | 0 | $1T_{si}$ | $2T_{si}$ | … | $(N-2)T_{si}$ | $(N-1)T_{si}$ |

| | $f_i$ | 2 | | | | | |
|---|---|---|---|---|---|---|---|
| | $t_i$ | 0 | 1 | 2 | … | $N$-2 | $N$-1 |
| **Series** | $1/N \cdot f_i \cdot t_i$ | 0 | $1 \cdot 2/N$ | $2 \cdot 2/N$ | … | $(N-2) \cdot 2/N$ | $(N-1) \cdot 2/N$ |
| **Intuitive** | $f(f_i) \cdot t(t_i)$ | 0 | $1 \cdot 2/N$ | $2 \cdot 2/N$ | … | $(N-2) \cdot 2/N$ | $(N-1) \cdot 2/N$ |
| | $f(f_i)$ | $2\Delta f$ | | | | | |
| | $t(t_i)$ | 0 | $1T_{si}$ | $2T_{si}$ | … | $(N-2)T_{si}$ | $(N-1)T_{si}$ |

and so on, up to:

| | $f_i$ | $m$-1 | | | | | |
|---|---|---|---|---|---|---|---|
| | $t_i$ | 0 | 1 | 2 | … | $N$-2 | $N$-1 |
| **Series** | $1/N \cdot f_i \cdot t_i$ | 0 | $1(m-1)/N$ | $2(m-1)/N$ | … | $(N-2)(m-1)/N$ | $(N-1)(m-1)/N$ |
| **Intuitive** | $f(f_i) \cdot t(t_i)$ | 0 | $1(m-1)/N$ | $2(m-1)/N$ | … | $(N-2)(m-1)/N$ | $(N-1)(m-1)/N$ |
| | $f(f_i)$ | $(m-1) \cdot \Delta f$ | | | | | |
| | $t(t_i)$ | 0 | $1T_{si}$ | $2T_{si}$ | … | $(N-2)T_{si}$ | $(N-1)T_{si}$ |

| | $f_i$ | $m$ | | | | | |
|---|---|---|---|---|---|---|---|
| | $t_i$ | 0 | 1 | 2 | … | $N$-2 | $N$-1 |
| **Series** | $1/N \cdot f_i \cdot t_i$ | 0 | $1 \cdot m/N$ | $2 \cdot m/N$ | … | $(N-2) \cdot m/N$ | $(N-1) \cdot m/N$ |
| **Intuitive** | $f(f_i) \cdot t(t_i)$ | 0 | $1 \cdot m/N$ | $2 \cdot m/N$ | … | $(N-2) \cdot m/N$ | $(N-1) \cdot m/N$ |
| | $f(f_i)$ | $m \cdot \Delta f$ | | | | | |
| | $t(t_i)$ | 0 | $1T_{si}$ | $2T_{si}$ | … | $(N-2)T_{si}$ | $(N-1)T_{si}$ |

As you may see, the **exponent** value of the **series formulation** is identical to the one of the **intuitive formulation**.

**For negative values of "$f(f_i)$"**

The **resulting exponent values** for **negative values** of "$f(f_i)$" are:

| | $f_i$ | $N$-$n$ | | | | | |
|---|---|---|---|---|---|---|---|
| | $t_i$ | 0 | 1 | 2 | … | $N$-2 | $N$-1 |
| **Series** | $1/N \cdot f_i \cdot t_i$ | 0 | $1(N-n)/N$ | $2(N-n)/N$ | … | $(N-2)(N-n)/N$ | $(N-1)(N-n)/N$ |
| **Intuitive** | $f(f_i) \cdot t(t_i)$ | 0 | $-1 \cdot n/N$ | $-2 \cdot n/N$ | … | $-(N-2) \cdot n/N$ | $-(N-1) \cdot n/N$ |
| | $f(f_i)$ | $-n \cdot \Delta f$ | | | | | |
| | $t(t_i)$ | 0 | $1T_{si}$ | $2T_{si}$ | … | $(N-2)T_{si}$ | $(N-1)T_{si}$ |

| | $f_i$ | $N$-($n$-1) | | | | | |
|---|---|---|---|---|---|---|---|
| | $t_i$ | 0 | 1 | 2 | … | $N$-2 | $N$-1 |
| **Series** | $1/N \cdot f_i \cdot t_i$ | 0 | $1(N-(n-1))/N$ | $2(N-(n-1))/N$ | … | $(N-2)(N-(n-1))/N$ | $(N-1)(N-(n-1))/N$ |
| **Intuitive** | $f(f_i) \cdot t(t_i)$ | 0 | $-1 \cdot (n-1)/N$ | $-2 \cdot (n-1)/N$ | … | $-(N-2) \cdot (n-1)/N$ | $-(N-1) \cdot (n-1)/N$ |
| | $f(f_i)$ | $-(n-1) \cdot \Delta f$ | | | | | |
| | $t(t_i)$ | 0 | $1T_{si}$ | $2T_{si}$ | … | $(N-2)T_{si}$ | $(N-1)T_{si}$ |

and so on, up to:

| | $f_i$ | $N$-2 | | | | | |
|---|---|---|---|---|---|---|---|
| | $t_i$ | 0 | 1 | 2 | … | $N$-2 | $N$-1 |
| **Series** | $1/N \cdot f_i \cdot t_i$ | 0 | $1(N-2)/N$ | $2(N-2)/N$ | … | $(N-2)(N-2)/N$ | $(N-1)(N-2)/N$ |
| **Intuitive** | $f(f_i) \cdot t(t_i)$ | 0 | $-1 \cdot 2/N$ | $-2 \cdot 2/N$ | … | $-(N-2) \cdot 2/N$ | $-(N-1) \cdot 2/N$ |
| | $f(f_i)$ | $-2 \cdot \Delta f$ | | | | | |
| | $t(t_i)$ | 0 | $1T_{si}$ | $2T_{si}$ | … | $(N-2)T_{si}$ | $(N-1)T_{si}$ |

| | $f_i$ | $N$-1 | | | | | |
|---|---|---|---|---|---|---|---|
| | $t_i$ | 0 | 1 | 2 | … | $N$-2 | $N$-1 |
| **Series** | $1/N \cdot f_i \cdot t_i$ | 0 | $(N-1)/N$ | $2(N-1)/N$ | … | $(N-2)(N-1)/N$ | $(N-1)(N-1)/N$ |
| **Intuitive** | $f(f_i) \cdot t(t_i)$ | 0 | $-1 \cdot 1/N$ | $-2 \cdot 1/N$ | … | $-(N-2) \cdot 1/N$ | $-(N-1) \cdot 1/N$ |
| | $f(f_i)$ | $-\Delta f$ | | | | | |
| | $t(t_i)$ | 0 | $1T_{si}$ | $2T_{si}$ | … | $(N-2)T_{si}$ | $(N-1)T_{si}$ |

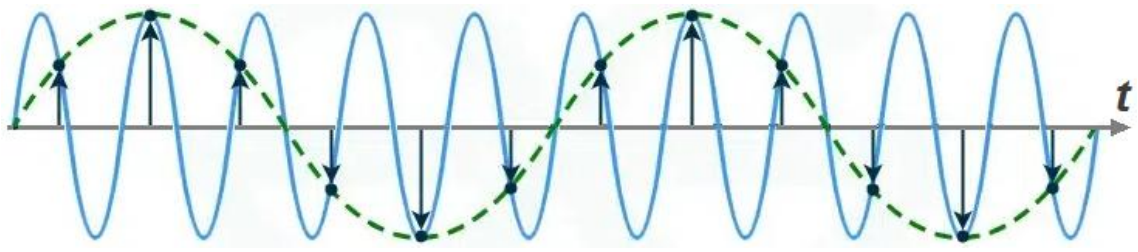
As you may see, the **exponent values** are **different**, but in spite of this, the **result** is the **same**!

Let us just show **one example** from the **second table** above where "$f_i=N-(n-1)$" and for the column where "$t_i=2$". Comparing the results, we get:

| | **Exponent/($j2\pi$)** | **Phasor** | **Time samples of the phasor** |
|---|---|---|---|
| **Series formulation** | $2 \cdot (N-(n-1))/N$ | $e^{j2\pi \cdot 2 \cdot (N-(n-1))/N}$ | Sampled at $F_s=1/T_{si}$ |
| **Intuitive formulation** | $-2 \cdot (n-1)/N$ | $e^{-j2\pi \cdot 2 \cdot (n-1)/N}$ | Sampled at $F_s=1/T_{si}$ |
| **Comparison** | Different, but equivalent | Different, but equivalent | **Identical!** |

The **frequency** of the **phasor** of the **series formulation** is "$2 \cdot (N-(n-1))/N$", that is **higher** than the **Nyquist frequency** (which is "$f_{Ny}=1/2$"). Consequently, the **time samples** of this **phasor** do **not** represent it, and they **rather** represent a **different signal**, whose **frequency** is below the **Nyquist frequency**. This **different signal** is **precisely** the **phasor** of the **intuitive formulation**, whose **frequency** (in **absolute value**) is **below** the **Nyquist frequency**. This is caused by the **aliasing** effect.

The following figure illustrates the **aliasing** effect:




Some **relevant remarks** on this figure are:

- There are **two sinusoids**, with frequencies "$f$" (dashed green sinusoid) and "$5f$" (blue sinusoid).

- The **sampling frequency** "$Fs$" is **higher** than "2$f$", but **lower** than "10$f$", which are the corresponding **Nyquist frequencies**.
- The "$f$" **green sinusoid** can be **correctly represented** by the **time samples**, while the "5$f$" **blue sinusoid** cannot.
- The **time samples** taken from the **blue sinusoid** are **identical** as the ones taken from the **green sinusoid**, as indicated in the **third column** of the table above.
- Consequently, for the **effects** of **(i)DFT**, the "5$f$" **blue sinusoid** is equivalent to the "$f$" **green sinusoid** , as indicated in **columns 1** and **2** of the table above.

This **series formulation** of **DFT** and **iDFT** can also be **formulated** as a **matrix multiplication**.

### 11.19.6 Phasors

A **phasor** is a **complex-valued function** of **time** that represents **both** the **instantaneous amplitude** and the **instantaneous phase** of a **sinusoid** with **maximum amplitude** "$A$", **initial phase** "$\varphi$" and **angular speed** "$\omega$". A **phasor** is usually represented as:

$Ae^{j(\omega t+\varphi)}$ (for a **positive phasor**)

$Ae^{-j(\omega t-\varphi)}$ (for a **negative phasor**)

where "$j$" is the **imaginary** "$\sqrt{-1}$".

The **instantaneous amplitude** of the **sinusoid** represented by the **phasor** is its **real component**:

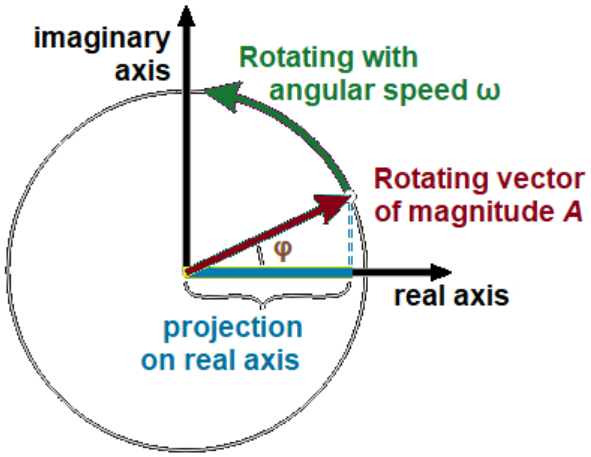
$$sinusoid(t) = real(Ae^{j(\omega t+\varphi)})$$

The **instantaneous phase** of the **sinusoid** represented by the **phasor** is its **phase** (also called "**argument**"):

$$phase(t) = argument(Ae^{j(\omega t+\varphi)})$$

A **phasor** may be interpreted as a **vector** of **magnitude** "$A$", with **initial phase** "$\varphi$" that **rotates** with **angular speed** "$\omega$". The **sinusoid** represented by the **phasor** is its **real component** (i.e., the "shadow" projected by the vector in the **real axis**). The **phase** of the **phasor** is the **phase** of the **vector**.

A **positive phasor** rotates **counterclockwise** while a **negative phasor** rotates **clockwise**. The following figure shows a **positive phasor** rotating **counterclockwise**, and its **projection** on the **real axis**:



A **phasor** may be **represented** using the **sinusoid frequency** "$f$" instead of the **angular speed** "$\omega$" based on their relationship "$\omega=2\pi f$":

$Ae^{j(2\pi f t+\varphi)}$ (for a **positive phasor**)

$Ae^{-j(2\pi f t-\varphi)}$ (for a **negative phasor**)

A **positive phasor** is interpreted as having a "**positive frequency**", while a **negative phasor** is interpreted as having a "**negative frequency**".

A **phasor** may be **represented** as a conventional **imaginary number** that has **one real-valued function** for its **real part** and **another real-valued function** for its imaginary part:

$$Ae^{j(2\pi f t+\varphi)} = A\ [cos(2\pi f t+\varphi)+j\ sin(2\pi f t+\varphi)]$$

This change in representation has been performed by applying **Euler**'s formula:

$$e^{jx} = cos(x)+j\ sin(x)$$

## 11.19.7 Why complex values?

The representations of the **baseband signal** "$BS()$", **both** in the **time-domain** "$BS(t)$" and in the **frequency-domain** "$BS(f)$", are **complex values**. The same happens (obviously) with their **samples** "$BS(t_i)$" and "$BS(f_i)$".

**Complex values in the frequency-domain**

The reason for **complex values** in the **frequency-domain** is that it is required to represent **both** the **maximum amplitude** and the **phase** of the **baseband signal** "$BS(f)$" for **each** value "$f$" of the **frequency** (i.e., the variable of the function "$BS(f)$"). With **only** the **maximum amplitude** for **frequency** "$f$" it would be **impossible** to distinguish a **sine** and a **cosine**. Consequently, it would **not** be **possible** to reconstruct the **original baseband signal** in the **time domain** "$BS(t_i)$".

The **magnitude** (also called **magnitude** or **absolute value**) of the **complex value** returned by "$BS(f)$" is the **amplitude** (i.e., its **maximum value**) of the **sinusoid** of frequency "$f$".

The **phase** (also called **argument**) of the **complex value** returned by "$BS(f)$" is the **phase** of the **sinusoid** of **frequency** "$f$".

For example, "$BS(60\ kHz)$=**2+3j**" indicates that the **60 kHz component** of the **baseband signal** has an **amplitude** of **5** and a **phase** of **0.98 radians**.

**Complex values in the time-domain**

The reason for **complex values** in the **time-domain** is that it is required to represent **both** the **instantaneous value** and the **instantaneous phase** of the **baseband signal** "$BS(t)$" for **each** value "$t$" of **time** (i.e., the **variable** of the function "$BS(t)$"). With **only** the **instantaneous value** for **time** "$t$" it would be **impossible** to determine the **instantaneous phase** of a **time-domain sinusoid**, because a **time-domain sinusoid** takes **each value** "$x$" **twice** in **each cycle** (except for zero and the peaks). Consequently, it would **not** be **possible** to construct the **baseband signal** in the **frequency domain** "$BS(f)$" using **only** the **instantaneous value** of a **time-domain sinusoid**.

The **real part** of the **complex value** returned by "$BS(t)$" is the **instantaneous value** of the **time-domain function**.

The **phase** (also called **argument**) of the **complex value** returned by "***BS***(*t*)" is the instantaneous **phase** of the **time-domain function**.

As it is done in the **frequency domain**, the **phase** is provided in respect to a **cosine function**:

- A **complex value** with **only** a **real** part is **in-phase** with a ***cosine*** function.
- A **complex value** with **only** an **imaginary** part is **in-phase** with a ***sine*** function.

For example, "***BS***(*3.1 sg*)=**2+3j**" indicates that the **signal value** for "*t*=*3.1 sg*" and the phase value for "*t*=*3.1 sg*" is **0.98 radians**.

An **advantage** of **complex values** in the **time-domain** is that it **allows** to represent **baseband signals** in the **time domain** "***BS***(*t*)" whose **frequency-domain representation** "***BS***(*f*)" is **not symmetric** in respect to the "**y**" **axis**. Notice that the **frequency domain representation** "***BS***(*f*)" of **every real-valued baseband signal** in the **time domain** "***BS***(*t*)" is **symmetric** in respect to the "**y**" **axis**. A **non-symmetric frequency domain representation** "***BS***(*f*)" is required to handle a **baseband signal** in the **time domain** "***BS***(*t*)" that make use of its **full spectrum** after its **upconversion** (click 4.5.8.12), and **not** only **half** of its **spectrum**.

A **complex-valued baseband signal** in the **time-domain** is frequently represented with the **2 real valued functions** "***I***(*t*)" and "***Q***(*t*)". The **function** "***I***(*t*)" is called the **in-phase signal component** while the **function** "***Q***(*t*)" is called the **quadrature signal component**. The **in-phase component** "***I***(*t*)" has the **phase** of a ***cosine*** function while the **quadrature component** "***Q***(*t*)" has the **phase** of a ***sine*** function, in **quadrature** with a ***cosine*** function.

**Reference for signal phase**

**Both** in the **time domain** and in the **frequency domain**, the **phase** is provided in respect to a **cosine function**: **phase** "**0**" implies that the **baseband signal** is **in-phase** with "***cosine***()". Notice that:

- A **complex value** with **only** a **positive real part** is **in-phase** with a ***cosine*** function.
- A **complex value** with **only** a **negative real part** is in **antiphase** with a ***cosine*** function.
- A **complex value** with **only** a **positive imaginary part** is **in-phase** with a ***sine*** function.
- A **complex value** with **only** a **negative imaginary part** in **antiphase** with a ***sine*** function.

## 11.20 Pseudo-random sequences

A **pseudo-random sequence** is a **generated sequence** with **mathematical properties** of a **random sequence**. The **mathematical properties** of a **pseudo-random sequence** are (among others) the **same frequency** of **element** values, the expected **frequencies** of **value subsequences** and **low autocorrelation**.

A **pseudo-random sequence** can be characterized by its **generating algorithm**, its **type** of **elements**, its **length**, its **random properties** and its **non-random properties**.

A **family** of **pseudo-random sequences** is a **set** of **pseudo-random sequences** generated with the **same algorithm** where the **sequences have low cross-correlation**. It can be characterized by the **characteristics** of the **sequences** in the set, and by the **quantity** of **sequences** in the set.

This section describes **some binary pseudo-random sequences** and **complex-number pseudo-random sequences** used in **5G**.

For more detail on **pseudo-random sequence generation**, refer to clause 5.2 of TS 38.211.

Source web pages:

- https://www.nrexplained.com/sequence

## 11.20.1 Pseudo-random binary sequences

A **quantity** "$c_{init}$" of **different pseudo random binary sequences** of **length** "$M_{PN}$" **bits**, denoted "$c(n, c_{init}, M_{PN})$", are produced by taking the **first** "$M_{PN}$" **bits** of a **Gold sequence**, as follows:

$$c(n, c_{init}, M_{PN}) = g(n\text{-}1{,}600, 0),$$

with $n \in \{0..M_{PN}\text{-}1\}$, $c_{init} \in \{1..2^{31}\text{-}1\}$ and $M_{PN} < 2^{31}$,

where "$g()$" is a **req-31 Gold sequence** (click 11.20.5), whose **preferred pair** of **M-sequences** (click 11.20.6) is:

$$m_1(n) = m_1(n\text{-}28) \oplus m_1(n\text{-}31)$$
$$m_2(n) = m_2(n\text{-}28) \oplus m_2(n\text{-}29) \oplus m_2(n\text{-}30) \oplus m_2(n\text{-}31)$$

and the **seed** of the **second req-31 M-sequence** "$m_2()$" is "$c_{init}$", while the **seed** of the **first req-31 M-sequence** "$m_1()$" is:

$$m_1(0) = 1 \; ; \; m_1(i) = 0, \text{ for } i=1..30$$

Recall from 11.20.6.3 that changing the **seed** of an **M-sequence** results in a **shift** of the **M-sequence**, that results in a **different Gold sequence**. Recall also that "⊕" is the **exclusive-or** operator.

The function "$c(n, c_{init}, M_{PN})$" therefore produces **up to** $2^{31}$**-1 different pseudo random sequences**, **one** for **each** of the admissible values of $c_{init}$ (all possible values except all zeros), with a **bitlength** "$M_{PN}$" **lesser** than $2^{31}$. Since the **resulting sequences** are **prefixes** of **Gold sequences**, they exhibit **all** their valuable **properties** (click 11.20.5.5).

For more detail on **pseudo-random sequence generation**, refer to clause 5.2.1 of TS 38.211.

## 11.20.2 Low-PAPR type 1 complex sequences

**Each low-PAPR type 1 complex number sequence** is denoted as:

$$r_{u,v}^{(\alpha,\delta,m)}(n) \qquad \text{with } 0 \le n \le (M_{ZC}\text{-}1)$$

**Each low-PAPR type 1 complex number sequence** is obtained by **incrementally** applying a **phase shift** "$\alpha$" to **each successive element** "$n$" of a **low-PAPR type 1 complex number base sequence** "$\bar{r}_{u,v}^{(\delta,m)}(n)$" (click 11.20.2.2), as indicated by the expression:

$$r_{u,v}^{(\alpha,\delta,m)}(n) = e^{jn\alpha}\, \bar{r}_{u,v}^{(\delta,m)}(n), \qquad 0 \le n \le (M_{ZC}\text{-}1)$$

where "$\alpha$" ranges from **0** to **2π**.

#### 11.20.2.1 Low-PAPR type 1 complex sequence overview

**Each low-PAPR type 1 complex number sequence** is denoted as:

$r_{u,v}^{(\alpha,\delta,m)}(n)$ with $0 \leq n \leq (M_{ZC}-1)$

where:

- "$n$" is the **complex number index** (an **integer** number) within the **sequence**.

  "$n$" **ranges** from "0" to "($M_{ZC}$-1)", where "$M_{ZC}$" is the **sequence length**.

- "$M_{ZC}$" (the **sequence length**) is a **function** of "$\delta$" and "$m$", as follows:

  $M_{ZC}(\delta, m) = 12m/2^{\delta}$

- "$\delta$" determines the **spacing** between **2 successive values** of "$M_{ZC}$". The said **spacing** is "$12/2^{\delta}$".

  "$\delta$" can **only** take the values "**0**", "**1**" and "**2**".

- "$m$" is the **index** of the values of "$M_{ZC}$" (the **sequence length**).

  "$m$" is a **natural** number: {**0**, **1** , **2**, …}.

- "$M_{ZC}(\delta, m)$" (the **sequence length**) is always a **multiple** of **3**, as follows:

  - $\delta$=**0** => $M_{ZC}(0, m) = 12m$ = 12, 24, 36, …
  - $\delta$=**1** => $M_{ZC}(1, m) = 6m$ = 6, 12, 18, …
  - $\delta$=**2** => $M_{ZC}(2, m) = 3m$ = 3, 6, 9, …,

  Notice that: $M_{ZC}(0, m) \subset M_{ZC}(1, m) \subset M_{ZC}(2, m)$

- "$\alpha$" is the **phase shift** that is **incrementally** applied as "$e^{jn\alpha}$" to **each successive element** "$n$" of a **low-PAPR type 1 complex number base sequence** "$\bar{r}_{u,v}^{(\delta,m)}(n)$" (click 11.20.2.2) that results in the corresponding **low-PAPR type 1 complex number sequence** "$r_{u,v}^{(\alpha,\delta,m)}(n)$", as indicated by the expression:

  $r_{u,v}^{(\alpha,\delta,m)}(n) = e^{jn\alpha}\, \bar{r}_{u,v}^{(\delta,m)}(n), \quad 0 \leq n \leq (M_{ZC}-1)$

  "$\alpha$" ranges from **0** to **2π**.

  **Each value** of "$\alpha$" produces **1 different low-PAPR type 1 sequence** "$r_{u,v}^{(\alpha,\delta,m)}(n)$" from **each type 1 base sequence** "$\bar{r}_{u,v}^{(\delta,m)}(n)$", where **both sequences** have the **same length** "$M_{ZC}(\delta, m)$".

- "$u$" is the **index** (an **integer** number) that identifies **each** of the **30 groups** of **base sequences type 1** that exist for **each sequence length** "$M_{ZC}(\delta, m)$".

  "$u$" **ranges** from "0" to "29".

- "$v$" is the **index** (an **integer** number) of the **1** or **2 type 1 base sequences** within **each** of the **30 groups** (that correspond to a given **sequence length**).

  "$v$" can **only** take the following values:

  - $v$ = **0** if "$6 \leq M_{ZC}(\delta, m) \leq 60$" (i.e., for **short** "$M_{ZC}$" **sequence length**).

    Consequently, for **each short** "$M_{ZC}$" **sequence length**: a) **each** of the **30 groups** contains **only 1 type 1 base sequence**; b) there are **30 type 1 base sequences**.

- $v \in \{0, 1\}$ if "$M_{ZC}(\delta, m) \geq 72$" (i.e., for **long** "$M_{ZC}$" **sequence length**).

  Consequently, for **each long** "$M_{ZC}$" **sequence length**: a) **each** of the **30 groups** contains **2 type 1 base sequences**; b) there are **60 type 1 base sequences**.

In the TS documents, the variable "$m$" does **not** appear in the definition of the **low-PAPR type 1 sequences**, which are defined "$r_{u,v}^{(\alpha,\delta)}(n)$", instead of "$r_{u,v}^{(\alpha,\delta,m)}(n)$", as it is done here. Adding "$m$" improves the understanding of its **relevance** as a variable that the **determines** the **sequence length**.

### 11.20.2.2 Low-PAPR type 1 base sequences

**Each low-PAPR type 1 complex number base sequence** is denoted as:

$\bar{r}_{u,v}^{(\delta,m)}(n)$ with $0 \leq n \leq (M_{ZC}-1)$

where **all** the variables "$n$", "$M_{ZC}(\delta, m)$", "$\delta$", "$m$", "$u$" and "$v$" have **exactly** the **same semantics** as in the **low-PAPR type 1 sequence** (click 11.20.2). Variable "$\alpha$" from the **low-PAPR type 1 sequence** is **absent** from the **low-PAPR type 1 base sequence** precisely because it is **each** value of "$\alpha$" which converts **each low-PAPR type 1 base sequence** into **1 different low-PAPR type 1 sequence**.

For more detail on **low-PAPR type 1 sequence**, refer to clause 5.2.2 of TS 38.211.

The definition of the **base sequence** "$\bar{r}_{u,v}^{(\delta)}(n)$" depends on the **sequence length** "$M_{ZC}$" as follows.

**Sequence length $M_{ZC} \geq 36$**

The **base sequence** is:

$$\bar{r}_{u,v}^{(\delta,m)}(n) = x(n \bmod N_{ZC})$$

where "$0 \leq n \leq (M_{ZC}-1)$", "$N_{ZC}$" is the **largest prime number lower** than $M_{ZC}$, and:

$$x(r) = e^{-j r(r+1) q \pi / N_{ZC}}$$

$$q = floor(\bar{q} + .5) + v \cdot (-1)^{floor(2\,\bar{q})}$$

$$\bar{q} = N_{ZC}\,(u+1)/31$$

For more detail on **sequence generation** when $M_{ZC} \geq 36$, refer to clause 5.2.2.1 of TS 38.211.

**Sequence length $M_{ZC} < 36$**

The **base sequence** $\bar{r}_{u,v}^{(\delta,m)}(n)$ depends on **several specific tables**.

For more detail on **sequence generation** when $M_{ZC} < 36$, refer to clause 5.2.2.2 of TS 38.211.

In the TS documents, the variable "$m$" does **not** appear in the definition of the **low-PAPR type 1 base sequences**, which are defined "$\bar{r}_{u,v}^{(\delta)}(n)$", instead of "$\bar{r}_{u,v}^{(\delta,m)}(n)$", as its is done here. Adding "$m$" improves the understanding of its **relevance** as a variable that the **determines** the **sequence length**.

## 11.20.3 Low-PAPR type 2 complex sequences

For an explanation on **low-PAPR type 2 complex number sequences**, refer to clause 5.2.3 of TS 38.211.

### 11.20.4 Zadoff-Chu complex sequences

A **Zadoff–Chu (ZC) sequence** (sometimes called "**Chu sequence**" or "**Frank–Zadoff–Chu (FZC) sequence**") is a mathematical sequence of **complex numbers** that when applied to a **signal** gives rise to a **new signal** of **constant amplitude**. When **cyclical shifts** of a **Zadoff–Chu sequence** are imposed upon a **signal**, the resulting set of signals detected at the receiver are **uncorrelated** (i.e., **orthogonal**) with one another.

A generated **Zadoff–Chu sequence** that has **not** been **shifted** is called a "**root sequence**".

The **ZC root sequences**, parametrized by ***u***, are generated with the expression:

$$x_u(n, N_{ZC}) = \exp\left(-j\frac{\pi u n(n + c_f)}{N_{ZC}}\right)$$

The elements in this expression are:

- "***u***" is the **ZC root sequence index**

  The **ZC root sequence index** is a **parameter** of **ZC sequences**, and **each** value of "***u***" produces a **different ZC root sequence**. Notice that "***u***" **must** be an **integer** from **0** to "$N_{ZC}$**-2**" (both included), that is a **co-prime** of "$N_{ZC}$". In case that "$N_{ZC}$" is **prime**, **all** the possible values of "***u***" are **valid**.

- "$N_{ZC}$" is the **sequence length** (i.e., its quantity of **complex numbers**)

  Therefore, "***n***" ranges from **0** to "$N_{ZC}$**-1**". The **sequence length** is **most frequently** a **prime**, because of the interesting resulting properties of the **ZC sequence**.

- $c_f = N_{ZC}$ ***mod*** 2

  It is a **corrective factor** for "$N_{ZC}$" being **even** or **odd**. When "$N_{ZC}$" is **prime** (which implies being **odd**), "$c_f$" is **1**.

The **period** of a **ZC sequence** is the same as its **length** "$N_{ZC}$".

**ZC sequences** are a type of **constant amplitude zero autocorrelation complex sequences**.

**ZC sequences** have the following **properties**:

- **ZC sequence after DFT**

  If "$N_{ZC}$" is **prime**, the **Discrete Fourier Transform** of a **ZC sequence** is another **ZC sequence conjugated**, **scaled** and **time scaled**. The implementation benefit of this property is that a **ZC sequence** can be generated **directly** in the **frequency domain without** having to do the **DFT** of the **sequence**. This is particularly useful for **OFDMA** or **SC-FDMA waveforms** that utilize the **frequency domain** for **signaling**.

- **Zero cyclic autocorrelation**

  The **auto correlation** of a **ZC sequence** with a **cyclically shifted** version of itself is **zero**.

- **Constant cyclic cross-correlation**

  The **cross-correlation** between two **ZC sequences** $x_{u1}(n)$ and $x_{u1}(n)$, with the **same length** "$N_{ZC}$" is constant ($N_{ZC}^{-1/2}$), provided that abs($u_1$-$u_2$) is a co-prime of "$N_{ZC}$".

Web page sources:

- https://arxiv.org/pdf/2211.05702
- https://en.wikipedia.org/wiki/Zadoff%E2%80%93Chu_sequence

## 11.20.5 Gold binary sequence and family

Source web page:

- https://en.wikipedia.org/wiki/Gold_code

### 11.20.5.1 Gold sequence definition

A **Gold sequence** (sometimes called "**Gold code**") is a type of **pseudo random binary sequence** such that the **cross-correlation** (click 11.20.7) between **any two different Gold sequences** (that belong to the **same Gold sequence family**) is $\leq 2^{(k+2)/2}$.

**Each Gold sequence** can be characterized by the quantity "**k**" of **bits** of its **1** or **2 reg-k M-sequences** (click 11.20.6), by the **one** or **two M-sequences** themselves and, when it is composed of **2 M-sequences**, by the **relative shift** "$i$" between its **two M-sequences**.

A **reg-k Gold sequence** is **any M-sequence** "$m(n)$" and any **sequence produced** by the **exclusive-or** of **two reg-k M-sequences** "$m_1(n)$" and "$m_2(n)$" that have a **cross-correlation** $\leq 2^{(k+2)/2}$). **Each** such **pair** of **M-sequences** "$m_1(n)$" and "$m_2(n)$" that **fulfills** the **cross-correlation** condition is called a "**preferred sequence pair**" or just a "**preferred pair**". Expressing the previous definition in a formal way, a **reg-k Gold sequence** "$g(n, i)$" is produced by **both** expressions:

$$g(n, i) = m(n)$$

$$g(n, i) = m_1(n) \oplus m_2(n+i)$$

where the **range** of $n$ and $i$ is $\{0..2^k-2\}$, "+" represents addition ***mod*** $2^k-1$, "$m(n)$" is an **M-sequence**, "$m_1(n)$" and $m_2(n)$ are a **preferred pair**, **all seeds** are **fixed** and "⊕" is an **exclusive-or**.

The **period** of the **sequence** generated by **every reg-k preferred pair** (with unbounded "$n$") is "$2^k-1$" **bits**, which is the **same** as the **period** of **every reg-k M-sequence** (with unbounded "$n$").

**Each** and **every different value** of the **index** "$i$" (with fixed seeds) produces a **different Gold sequence**, and **not** a mere **shift** of a **Gold sequence**.

### 11.20.5.2 Gold sequence family

A **reg-k Gold sequence family** consists of a **preferred pair** (click 11.20.5.1) of **M-sequences** "$m_1(n)$" and $m_2(n)$ **plus all** the **reg-k Gold sequences** produced by the **exclusive-or** of **every possible shift** "$i$" of $m_2(n)$ in respect to "$m_1(n)$". Expressing this in a formal way, a **reg-k Gold sequence family** is produced by the following **generative function** "$g(n)$":

$$g(n) \in \{m_1(n), m_2(n), m_1(n) \oplus m_2(n+i)\}$$

where the **range** of "$n$" is {**0..2$^k$-2**}, $i$ takes **all** the **values** in {**0..2$^k$-2**}, "+" represents addition ***mod*** **2$^k$-1**, **all seeds** are **fixed** and "⊕" is an **exclusive-or**.

**Each Gold sequence** in the **family** that is produced by the **xor** of the **preferred pair** is **identified** by its **index** $i$.

Therefore, a **reg-k Gold sequence family** contains **2$^k$+1 reg-k Gold sequences**: the **two M-sequences** in the **preferred pair plus** the **2$^k$-1 sequences** corresponding to **each value** of the **index** $i$. **All** the **reg-k Gold sequences** in **each reg-k Gold sequence family** are **different**, this is, they are **not** mere **circular shifts** of one another. Be aware that the different **reg-k Gold sequence families** that can be generated by the different **preferred pairs** are **not disjoint**.

A relevant advantage of **Gold sequences** in respect to **M-sequences** is that **each reg-k Gold sequence family** has **many more reg-k Gold sequences** than **all** the **reg-k M-sequences**.

### 11.20.5.3 Gold sequence shifts and seeds

You can produce a **shifted version** "$sg(n)$" of **each** and **every Gold sequence** "$g(n, i)$" by adding a **shift** "$S$" to its argument "$n$":

$$sg(n, i, S) = g(n+S, i)$$

where the **range** of "$n$", "$i$" and "$S$" is {**0..2$^k$–2**} and "+" represents addition "***mod*** **(2$^k$-1)**".

For a **Gold sequence** "$g(n, i)$ ", if the **first M-sequence** is **shifted** by "$S_1$" and the **second** one is **shifted** by "$S_2$", the result is the **Gold sequence** with **index** "$i+S_2-S_1$" **shifted** by "$S_1$":

$$g(n, i, S_1, S_2) = m_1(n+S_1) \oplus m_2(n+i+S_2)\}$$

$$g(n, i, S_1, S_2) = m_1(n+S_1) \oplus m_2(n+S_1+(i+S_2-S_1))\}$$

$$g(n, i, S_1, S_2) = g(n+S_1, i+S_2-S_1) = sg(n, i+S_2-S_1, S_1)$$

where the **range** of "$n$", "$i$", "$S_1$" and "$S_2$" is {**0..2$^k$-2**}, "+" represents addition "***mod*** **(2$^k$-1)**", "-" represents subtraction "***mod*** **(2$^k$-1)**", "$m_1(n)$" and "$m_2(n)$" are a **preferred pair**, **all seeds** are **fixed** and "⊕" is an **exclusive-or**.

Some useful particular cases of "$g(n, i, S_1, S_2)$ ", where "+" and "-" represent addition and subtraction "***mod*** **(2$^k$-1)**" (respectively), are:

- A **shift** "$S_1$" and **no shift** "$S_2$"

  The result is the **Gold sequence** with **index** "$i-S_1$" **shifted** by "$S_1$":

  $$g(n, i, S_1, 0) = g(n+S_1, i-S_1) = sg(n, i-S_1, S_1)$$

- **No shift** "$S_1$" and a **shift** "$S_2$"

  The result is the **Gold sequence** with **index** "$i+S_2$":

  $$g(n, i, 0, S_2) = g(n, i+S_2)$$

- A **shift** "$S_1$" and a **shift** "$S_2=S_1+w$"

  The result is the **Gold sequence** with **index** "$i+w$" **shifted** by "$S_1$":

  $$g(n, i, S_1, S_1+w) = g(n+S_1, i+w) = sg(n, i+w, S_1)$$

- **Index** "$i=0$, a **shift** "$S_1$" and a **shift** "$S_2$"

  The result is the **Gold sequence** with **index** "$-S_1$" **shifted** by "$S_1$":

  $$g(n, 0, S_1, 0) = g(n+S_1, -S_1) = sg(n, -S_1, S_1)$$

- **Index** "$i=0$", **no shift** "$S_1$" and a **shift** "$S_2$"

  The result is the **Gold sequence** with **index** "$S_2$":

  $$g(n, 0, 0, S_2) = g(n, S_2)$$

- **Index** "$i=0$", a **shift** "$S_1$" and a **shift** "$S_2=S_1+w$"

  The result is the **Gold sequence** with **index** "$w$" **shifted** by "$S_1$":

  $$g(n, 0, S_1, S_1+w) = g(n+S_1, w) = sg(n, w, S_1)$$

Recall from (click 11.20.6.3) that changing the **seed** of an **M-sequence** produces a **shifted** version of it. Changing the **seed** of "$m_1(n)$" and/or of "$m_2(n)$" is equivalent to introducing a **non-zero shift** "$S_1$" and/or a **non-zero shift** "$S_2$" (respectively). The resulting **Gold sequences** have already been analyzed slightly above. Therefore, **changing** the **seed** of

**either** or **both M-sequences** results in a **different** (**possibly shifted**) **Gold sequence**. Notice that this behavior is **different** to the one of an **M-sequence** (click 11.20.6).

### 11.20.5.4 Gold sequence quantities

The following table shows some relevant **quantities** in relation to **Gold sequences**, to **Gold sequence families** and to their **preferred pairs**, for **two values** of "k":

| | Shift register length "k" | |
|---|---|---|
| | 7 | 31 |
| **Sequence length = Quantity of shifts = Generative function period ($N_S=2^k-1$) [bits]** | 127 | 2.15E+09 |
| **Quantity of all sequences ($2^{(2^k-1)}$)** | 1.70E+38 | ~1E+646,456,993 |
| **Quantity of M-sequences per generative function ($N_M$)** | 1 | 1 |
| **Quantity of shifted M-sequences per generative function ($N_M \cdot N_S$)** | 127 | 2,147,483,647 |
| **Quantity of generative functions ($N_{GF}=\Phi(2^k-1)/k$)** | 18 | 6.93E+07 |
| **Maximum quantity of M-sequences ($N_{GF} \cdot N_M$)** | 18 | 6.93E+07 |
| **$\Phi(2^k-1)$** | 126 | 2.15E+09 |
| **Quantity of Gold sequences per preferred function pair (1 family) ($N_G=2^k+1$)** | 129 | 2.15E+09 |
| **Quantity of shifted Gold sequences per preferred function pair (1 family) ($N_G \cdot N_S$)** | 16,383 | 4.61E+18 |
| **Quantity of preferred function pairs (families) ($N_{PF}=(2^k-1)/2$)** | 63 | 1.07E+09 |
| **Upper bound for quantity of Gold sequences ($N_{PF} \cdot N_G$)** | 8,127 | 2.31E+18 |

Some **relevant remarks** on this table are:

- The **quantity** of **shifted Gold sequences** per family is **very large**, but the **quantity** of **all sequences** is **several orders** of **magnitude larger**.
- **Both $2^7-1$** and **$2^{31}-1$** are **primes**. This is why their resulting **Φ()** value is **$2^7-2$** and **$2^{31}-2$**, respectively.
- The row "**Upper bound…**" is the **quantity** of **preferred function pairs** multiplied by the **quantity** of **Gold sequences** per **preferred function pair**. This is an **upper bound** that **cannot** be achieved because the families created by the **different preferred function pairs** are **not disjoint**.

#### 11.20.5.5 Gold sequence properties

**Gold sequences** have the following **properties**:

- **Maximum length**

  **Every reg-k M-sequence** has the **maximum possible length** for a **reg-k M-generative function**. This **length** is $2^{k-1}$ bits.

- **Low bound cross-correlation**

  The **highest absolute cross-correlation** between **any two Gold sequences** from the **same family** is $2^{(k+2)/2}+1$ for **even** "**k**", and $2^{(k+1)/2}+1$ for **odd** "**k**".

- **Balance**

  Around **half** of the **Gold sequences** of a **Gold sequence set** are **balanced** (i.e., the quantity of ones and zeros differs by only one).

- **Autocorrelation**

  Every **Gold sequence** has an **autocorrelation** (click 11.20.7) of $2^k-1$ for offset **0**, and **-1** for every other offset. This is relatively high but notice that the **maximum cross-correlation** between **two different Gold sequences** (of the **same set**) is $2^{(k+2)/2}$, which is **much lower** (except the degenerate cases of very low "**k**").

- **Gold sequences** have **better cross-correlation** (click 11.20.7) than **M-sequences**.

#### 11.20.5.6 Gold sequence generation

**Every Gold sequence** in a **Gold sequence family** can be generated with **two k-bit linear feedback shift register** (**LFSR**). **Each LFSR** generates **each** of the **two M-sequence** in the **preferred pair**. **All** the other **Gold sequences** can be generated by introducing a **shift** in the **second M-sequence** from the preferred pair, and doing a **xor between** the **output** of **both LFSR**.

### 11.20.6 M-sequence (binary)

Source web page:

- https://en.wikipedia.org/wiki/Maximum_length_sequence

#### 11.20.6.1 M-sequence definition

A **maximum length sequence** (**M-sequence**, or sometimes **MLS**) is a type of **pseudo random binary sequence** produced by a **linear M-generative function** that corresponds to a **primitive polynomial**. **M-sequences** have **very valuable properties** (click 11.20.6.5).

**Each M-sequence** can be characterized by the **displacement** "**k**" of the "**oldest**" **term** "***m***(***n***-**k**)" of its **M-generative function**. An **M-generative function** with a **displacement** of "**k**" is called a **reg-k M-generative function**. An **M-sequence** with a **reg-k M-generative function** is called a **reg-k M-sequence**.

A **reg-k M-generative function** has the form:

$$m(n) = m(n\text{-}1) \oplus m(n\text{-}2) \oplus \ldots \oplus m(n\text{-}(k\text{-}1)) \oplus m(n\text{-}k) \quad \text{for } n \in \{k..2^k\text{-}2\}$$

$$m(n) = m_n \quad \text{for } n \in \{k\text{-}1..0\}$$

where it **only** includes those **terms** "$m(n\text{-}j) \oplus$" (with $j \in \{1..(k\text{-}1)\}$) that correspond to a **primitive** (i.e., **irreducible**) **polynomial** and "$\oplus$" is the **exclusive-or** operator. The "**oldest**" **term** "$m(n\text{-}k)$" **must always** be present. If it were **not** present, the **M-generative function** would produce its **next value** from **fewer** than its **last** k bits, and consequently, it would **not** be a **reg-k M-generative function**.

The **first** "k" **bits** $m_n$ (with $n \in \{k\text{-}1..0\}$) of a **reg-k M-sequence** are called its "**seed**". The **seed** of an **M-sequence** is a **configuration parameter**. The **seed cannot** be **all zeros** (if it were, the resulting **sequence** would also be **all zeros**). By definition, the **first** "k" **bits** of **every reg-k M-sequence** are the same as its **seed**.

A useful property of an **M-sequence** is that it can be **generated** with a simple **linear feedback shift register** (**LFSR**, click 11.20.6.6).

Notice that the **first value** "$m(0)$" of the **M-sequence** corresponds to the **rightmost value** "$m_0$" of its **seed**, and the **k$^{th}$ value** "$m(k\text{-}1)$" of the **M-sequence** corresponds to the **leftmost value** "$m_{k-1}$" of its **seed**. Likewise, the "**oldest**" **term** "$m(n\text{-}k)$" of the **M-generative function** is the **rightmost** one, and the "**newest**" **term** "$m(n)$" is the **leftmost** one. This implies that the **sequence** (and also the **seed**) has to be **read right to left**, and **not left to right**, which may result somehow counter intuitive.

### 11.20.6.2 M-sequence period and length

When "k" **consecutive bits** "$m(n\text{-}j)$" (with $j \in \{1..k\}$) take the **same value** as **any previous** "k" **consecutive bits**, the **sequence repeats** starting in "$m(n)$". This happens because the **M-generative function** produces its **next** value from the **previous** "k" **bits** of the **sequence**: if these "k" **bits repeat**, the **sequence** also **repeats**.

**M-sequences** are called "**maximum length**" because their **period** (i.e., the **quantity** of **bits** after which they **repeat**) is the **maximum possible**. Since the **sequence repeats** when "k" **consecutive bits** take the **same value** as **any previous** "k" **consecutive bits**, the **maximum possible period** is achieved when the "k" **consecutive bits only repeat** after having already taken **all possible values**. A **non-zero k-bit string** has **$2^k$-1 possible values**. Therefore, the **maximum possible period** is **$2^k$-1 bits**.

Consequently, the **length** of a **reg-k M-sequence** is **$2^k$-1 bits**, which corresponds to the **period** (if unbounded "$n$") of its **M-generative function** (click 11.20.6.1). This is why in the definition of the **M-generative function** it is indicated that $n \in \{0..2^k\text{-}2\}$.

An **M-generative function must** have **at least one term** "$m(n\text{-}j) \oplus$" (with $j \in \{1..(k\text{-}1)\}$). Otherwise, its **period** would **only** be "k" **bits** (a permanent **repetition** of the **seed**).

It is possible to build **sequences** longer than **$2^k$-1 bits** by **cyclically repeating** a **reg-k M-sequence**, up to the desired **bit length**. However, this will **degrade** the **valuable properties** (click 11.20.6.5) of an **M-sequence**.

### 11.20.6.3 M-sequence shifts and seeds

You can produce a **shifted version** "$sm(n, S)$" of an **M-sequence** by **adding** a **shift** "$S$" to the **argument** of its **M-generative function** "$m(n)$":

$$sm(n, S) = m(n + S)$$

where the **range** of "$n$" and "$S$" is $\{0..2^k-2\}$ and "+" represents addition ***mod*** $2^k-1$.

Since a **reg-k M-sequence** includes **all** the **non-zero possible values** (click 11.20.6.1) of "**k**" **consecutive bits**, **any non-zero value** can be used as its **seed**. **Each different seed** produces the **same M-sequence** with a **different shift**, and **not** a **different M-sequence**. Notice that **any seed must** match the **M-sequence** at some position. Therefore, the **seed** determines the **starting position** of the **M-sequence**, which is the same as **performing** a **shift**.

There are $2^k-1$ **different seeds**, **each** of which produces **one shift** among the $2^k-1$ possible ones.

The resulting **shift** of **every** given **seed** is **pseudo-random**. Since an **M-sequence** is **pseudo-random**, the location of **any given pattern** of "**k**" **consecutive bits** (i.e., the **seed**) within the **M-sequence** is also **pseudo-random**. Consequently, increasing the **seed** in **one unit**, results in a **pseudo-random shift quantity**.

### 11.20.6.4 M-sequence quantities

The **quantity** of **different reg-k M-sequences** is the **quantity** of **reg-k M-generative functions**, which is the **quantity** of **primitive polynomials** of degree "**k**":

$$N_{M\text{-}seqs}(k) = \phi(2^k-1)/k$$

where $\phi()$ is the Euler's **totient function** (i.e., the **quantity** of **co-primes**, below $2^k-1$, in respect to $2^k-1$). In case $2^k-1$ is a **prime**, Euler's **totient function** returns the **maximum possible value**, which is $2^k-2$. This is why the selection of "**k**" such that $2^k-1$ is a **prime** is advantageous in **providing** a **large quantity** of different **M-sequences**.

The **quantity** of **all shifts** of **all reg-k M-sequences** is the **quantity** of **reg-k M-sequences** multiplied by the $2^k-1$ **shifts** for **each sequence**.

The following table shows some **relevant quantities** related to **M-sequences**, for **two values** of "**k**":

| | **Shift register length "k"** | |
|---|---|---|
| | **7** | **31** |
| **Sequence length = Quantity of shifts = Generative function period ($N_S=2^k-1$) [bits]** | 127 | 2.15E+09 |
| **Quantity of all sequences ($2^{(2^k-1)}$)** | 1.70E+38 | ~1E+646,456,993 |
| **Quantity of M-sequences per M-generative function ($N_M$)** | 1 | 1 |
| **Quantity of shifted M-sequences per M-generative function ($N_M \cdot N_S$)** | 127 | 2.147.483.647 |
| **Quantity of M-generative functions ($N_{GF}=\Phi(2^k-1)/k$)** | 18 | 6.93E+07 |
| **Maximum quantity of M-sequences ($N_{GF} \cdot N_M$)** | 18 | 6.93E+07 |
| **$\Phi(2^k-1)$** | 126 | 2.15E+09 |

Some **relevant remarks** on this table are:

- The **quantity** of **shifted M-sequences** is **very large**, but the **quantity** of **all sequences** is **several orders** of **magnitude larger**.
- **Both $2^7-1$** and **$2^{31}-1$** are **primes**. This is why their resulting **Φ()** value is **$2^7-2$** and **$2^{31}-2$**, respectively.
- The **quantity** of **seeds** is the same as the **quantity** of **shifts**, this is, the quantity of **shifted M-sequences** per **M-generative function**.
- The **quantity** of **seeds** (equal to **shifts**) is **very** large (**$2^k-1$**), but the **quantity** of **all sequences** is **much**, **much larger**: **$2^{(2^k-1)}$**.

### 11.20.6.5 M-sequence properties

**M-sequences** have the following **properties**:

- **Maximum length**

  **Every reg-k M-sequence** has the **maximum possible length** for a **reg-k M-generative function**. This **length** is **$2^{k-1}$** bits.

- **Balanced**

  **Every reg-k M-sequence** has **$2^{k-1}$ ones** and **$2^{k-1}-1$ zeros**. It is therefore **balanced** in terms of **zeros** and **ones**. This is consistent with the statistics expected in a random sequence.

- **Run distribution**

  **Every M-sequence** has a **run distribution** such that the **fraction** of "**runs**" of **length** "**z**" is **$2^{-z}$**. A "**run**" of **length** "**z**" consists of "**z**" **consecutive** "**1**" or of "**z**"

**consecutive** "**0**", within the **M-sequence**. Therefore, **1/2** of the **runs** have length **1**, **1/4** have length **2**, **1/8** have length **3**, and so on.

- **Low autocorrelation**

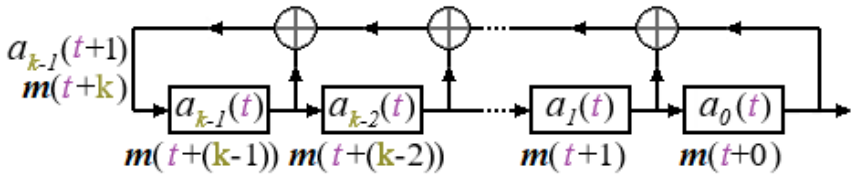
  **Every M-sequence** has an **autocorrelation** (click 11.20.7) consisting of a **Kronecker delta** function (with implementation dependent **DC offset** and **time delay**).

- **Flat spectrum**

  **Every M-sequence** is **spectrally flat**, except for a **near-zero DC term**.

### 11.20.6.6 M-sequence generation

**Every reg-k M-sequence** can be generated with a **k-bit linear feedback shift register** (**LFSR**), as the one shown in the following figure:




**Any** possible **reg-k M-generative function**:

$$m(n) = m(n\text{-}1) \oplus m(n\text{-}2) \oplus \ldots \oplus m(n\text{-}(k\text{-}1)) \oplus m(n\text{-}k) \quad \text{for } n \in \{k..2^k\text{-}2\}$$

is first **changed variable** to reflect **time instants**, to map it to the **LFSR**:

$$m(t+k) = m(t+(k\text{-}1)) \oplus m(t+(k\text{-}2)) \oplus \ldots \oplus m(t+1) \oplus m(t+0) \quad \text{for } t \in \{0..2^k\text{-}2\text{-}k\}$$

**Each term** "$m(t+j)$" (with $j \in \{1..(k\text{-}1)\}$) of the **reg-k M-generative function** is **mapped** to the **xor feedback loop** (called a "**tap**") of the **register bit** "$a_j(t)$". Recall that the **range** of **j** does **not** include "**k**" because the **rightmost term** "$m(t+0)$" **must always exist** in the **reg-k M-generative function** by definition (click 11.20.6.1). This **term** corresponds to the **feedback loop** of the **rightmost cell** $a_0$, that likewise **must always exist**: **otherwise**, this **LFSR** would have **less** than "**k**" **bits**.

Therefore, the **reg-k M-generative function** indicated above is **mapped** to the **feedback function** of the **LFSR** as follows (see the figure above):

$$m(t+k) = m(t+(k\text{-}1)) \oplus m(t+(k\text{-}2)) \oplus \ldots \oplus m(t+1) \oplus m(t+0) \quad \text{for } t \in \{0..2^k\text{-}2\text{-}k\}$$

$$a_{k-1}(t+1) = a_{k-1}(t) \oplus a_{k-2}(t) \oplus \ldots \oplus a_1(t) \oplus a_0(t) \quad \text{for } t \in \{0..2^k\text{-}2\text{-}k\}$$

$$a_j(t+1) = a_{j+1}(t) \text{ for } j \in \{0..k\text{-}2\} \qquad \text{(this is just the linear shift)}$$

where "⊕" is the **exclusive-or** operator and the **rightmost term must always** be present.

The **seed** of the **M-generative function** is the **same** as the **initial value** of the **LFSR**:

$$a_n(0) = m(t) = m_n \quad \text{for } t \in \{k\text{-}1..0\}$$

Notice that the **first value** "$m(t)$" of the **M-sequence** corresponds to the **rightmost cell** of the **LFSR** "$a_0(t)$", while the **k$^{th}$ value** $m(t+(k\text{-}1))$ corresponds to the **leftmost cell** "$a_{k-1}(t)$". This implies that the **sequence** (including the **seed**) has to be **read right to left**, and **not left to right**, which may result somehow counter intuitive.

Let us illustrate this with an **example** of a **reg-4 M-sequence**, with the **M-generative function**:

$$m(n) = m(n\text{-}3) \oplus m(n\text{-}4) \text{ for } n \in \{4..6\}$$

which is **changed variable:**

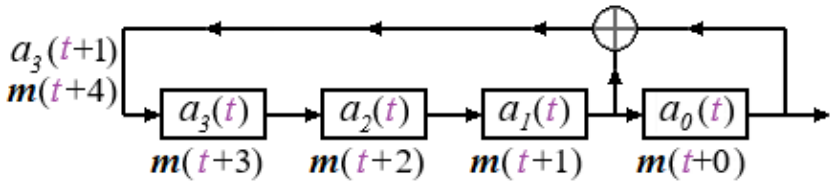
$$m(t+4) = m(t+1) \oplus m(t) \text{ for } t \in \{0..2\}$$

and then **mapped** to the **feedback function** of the **LFSR** as follows (see the figure above):

$$a_3(t+1) = a_1(t) \oplus a_0(t) \quad \text{for } t \in \{4..6\}$$

$$a_j(t+1) = a_{j+1}(t) \text{ for } j \in \{0..2\} \qquad \text{(this is just the linear shift)}$$

This **feedback function** corresponds to the following **4-bit LFSR**:




The following table shows the **15 sequences** that it produces, **each** with an **integer index**, its **seed** and its **shift** in respect to **sequence 1**:

| Seq. | Seed | M-sequence shifts | Shift |
|---|---|---|---|
| 1 | **0 0 0 1** | 1 1 1 0 1 0 1 1 0 0 1 **0 0 0 1** | 0 |
| 2 | **0 0 1 0** | 0 0 1 1 1 1 0 1 0 1 1 **0 0 1 0** | 3 |
| 3 | **0 0 1 1** | 1 1 0 1 0 1 1 0 0 1 0 **0 0 1 1** | 14 |
| 4 | **0 1 0 0** | 0 1 1 1 1 0 1 0 1 1 0 **0 1 0 0** | 2 |
| 5 | **0 1 0 1** | 1 0 0 1 0 0 0 1 1 1 1 **0 1 0 1** | 8 |
| 6 | **0 1 1 0** | 0 1 0 0 0 1 1 1 1 0 1 **0 1 1 0** | 6 |
| 7 | **0 1 1 1** | 1 0 1 0 1 1 0 0 1 0 0 **0 1 1 1** | 13 |
| 8 | **1 0 0 0** | 1 1 1 1 0 1 0 1 1 0 0 **1 0 0 0** | 1 |
| 9 | **1 0 0 1** | 0 0 0 1 1 1 1 0 1 0 1 **1 0 0 1** | 4 |
| 10 | **1 0 1 0** | 1 1 0 0 1 0 0 0 1 1 1 **1 0 1 0** | 9 |
| 11 | **1 0 1 1** | 0 0 1 0 0 0 1 1 1 1 0 **1 0 1 1** | 7 |
| 12 | **1 1 0 0** | 1 0 0 0 1 1 1 1 0 1 0 **1 1 0 0** | 5 |
| 13 | **1 1 0 1** | 0 1 1 0 0 1 0 0 0 1 1 **1 1 0 1** | 10 |
| 14 | **1 1 1 0** | 1 0 1 1 0 0 1 0 0 0 1 **1 1 1 0** | 11 |
| 15 | **1 1 1 1** | 0 1 0 1 1 0 0 1 0 0 0 **1 1 1 1** | 12 |

Some **relevant remarks** on this table are:

- **Both** the **seeds** and the **M-sequences** should be **read right to left**: **first** bit is **rightmost** and **last** bit **leftmost**.
- The **seed** of **each M-sequence** is its **first 4 bits**, that have been highlighted in **bold** font.
- The **first 4 bits** of the **first M-sequence** have been colored in red. This allows to see the **shift** in respect to the **first M-sequence** caused by **each different seed**
- For **each different seed** we get a different **shift** of the **same reg-4 M-sequence**.

- These **15 sequences** are the **15 possible circular shifts** (**0** to **14**) of the same **M-sequence**.
- **All** the **15 possible non-zero values** of the **4-bit LFSR** are present in the **M-sequence** (the **15 seeds**).

#### 11.20.6.7 Example of a reg-2 M-sequence

This example is for a **reg-2 M-sequence**. The **LSFR length** is **2**, which is also the **seed**. The **M-sequence length** is $2^2-1=3$ bits, which is also the **quantity** of **shifts**. The **quantity** of **M-sequences** is $\phi(2^2-1)/2=1$.

The following table shows **all** the possible **3-bit sequences**. For the **3** sequences that are a **shift** of the **M-sequence**, it indicates its **seed**, and the **shift** in respect to **sequence 4**: For the other ones it indicates the type of sequence.

| Seq. | Sequence | Seed | M-sequence shifts | Shift |
|---|---|---|---|---|
| 1 | 0 0 0 | Degenerate sequence | | |
| 2 | 0 0 1 | Not M-sequence | | 0 |
| 3 | 0 1 0 | Not M-sequence | | 2 |
| 4 | 0 1 1 | 1 1 | 0 1 1 | 0 |
| 5 | 1 0 0 | Not M-sequence | | 1 |
| 6 | 1 0 1 | 0 1 | 1 0 1 | 1 |
| 7 | 1 1 0 | 1 0 | 1 1 0 | 2 |
| 8 | 1 1 1 | Period is 1 (not maximum length) | | |

Some **relevant remarks** on this table are:

- **Sequence 1** is degenerate because it is the **all-zero sequence**. It is produced by **every LFSR** when its **seed** is **all zeros**.
- **Sequence 8** is produced by a **LFSR** with **no taps**. It is therefore a repetition of its **initial register**. It does **not** have **maximum length**.
- **Sequences 2**, **3** and **5** have **maximum length**. They are **three shifts** of the same sequence. However, they are **not** generated by a **primitive polynomial**, and therefore, they are **not an M-sequence**.

### 11.20.7 Binary sequence correlation

The **dot product** (denoted "•") of **two binary sequences** of the **same length** "***L***" is the quantity of **positions** with the **same value minus** the quantity of **positions** with a **different value**. The **dot product** ranges from "***-L***" to "***L***". The **dot product** of a **sequence** with itself is "***L***". The **dot product** of a **sequence** with its **inverse sequence** (**bitwise not**) is "***-L***".

The **cross-correlation** of **two binary sequences** of the **same length** "***L***" is the **series** of **results** of the "***L***" **dot products** between **one sequence** and **each** and **every** of the possible **circular shifts** of the **other sequence**. In other words, it is the **series** of **results** of the "***L***" **dot products** between **every** possible **relative circular shift** between **both sequences**. The **cross-correlation** is a **measure** of the **similarity** between **two sequences**.

The **autocorrelation** of a **binary sequence** is the **cross-correlation** of the **sequence** with **itself**.

Source web page:

- https://pages.hmc.edu/harris/class/e11/lect7.pdf

## 11.21 Passive lossless antennas

**All** this **section** refers **solely** to **passive lossless antennas** in **far field** conditions (i.e., when "$d \gg \lambda$").

Section 4.5.8 contains a **5G system description** of an **antenna element** (click 4.5.8.3), an **antenna** (click 4.5.8.5) and an **antenna array** (click 4.5.8.6).

### 11.21.1 Properties of passive lossless antennas

A **passive antenna** is the one that does **not** include any **active component** (nor metamaterials).

A **lossless antenna** is the one that has **no internal losses** due to antenna construction.

The remaining part of this section presents **two relevant properties** of **passive lossless antennas**:

- **Power conservation** (click 11.21.1.1).
- **Power redistribution** (click 11.21.1.2).

#### 11.21.1.1 Power conservation

From the **definitions** above of a **passive antenna** and of a **lossless antenna** it follows that:

- A **passive antenna** does **not amplify** the **power** that is **fed** to it (when used for **transmission**), **nor** the **power** that is **captured** by it (when used for **reception**).
- A **lossless antenna** does **not attenuate** the **power** that is **fed** to it (when used for **transmission**), **nor** the **power** that is **captured** by it (when used for **reception**), and it does **not re-radiate** the **captured power**.

From these **two properties** it follows these **two additional properties**:

- **Power conservation** when used for **transmission**

  A **passive lossless antenna conserves** the **power** "$P_t$" that is **fed** to it: **all** the **power fed** to the **antenna** is **radiated** in **different directions**.

- **Power conservation** when used for **reception**

  A **passive lossless antenna conserves** the **power** "$P_c$" that is **captured** by it from **every direction**: **all** its **captured power** is **output** "$P_r$", through the **antenna terminals**.

  The **output power** at the **antenna terminals** is usually called the **received power**.

These **two additional properties** can also be derived from the principle of **energy conservation**.

#### 11.21.1.2 Power redistribution

Although a **passive lossless antenna does not modify** the **power** "$P_t$" that is **fed** to it, **nor** the **power** "$P_r$" that is **output** by it from **every direction**, a **passive lossless antenna can redistribute** the **power** across the **different directions**.

A **passive lossless antenna** can **transmit** or **receive more power** from **some directions**, at the **cost** of **transmitting** or **receiving less power** (with the **same total amount**) from **other directions**.

**Power redistribution** when used for **transmission**

A **passive lossless antenna** can **radiate** a **different power density flux** "$S_t^v(d)$" in **each direction** "$v$", as long as the **power density flux** that is **radiated** in **every direction adds up** to **exactly** the **power** "$P_t$" that is **fed** to the **antenna**. This **last requirement** arises from **power conservation** (click 11.21.1.1).

For example, the **antenna** can **radiate most** of its **total radiated power** "$P_t$" in **direction** "$v_1$", while it **radiates** a **very low power density flux** in **many** other **directions** that are comprised within a **large solid angle**, as long as the **total radiated power** is "$P_t$" (**power conservation**).

The specific **Tx power redistribution** of an **antenna** can be specified by associating a **weight** "$G_t^v$" to **each** and **every direction** "$v$". The **power density flux** "$S_t^v(d)$" in **direction** "$v$" at distance "$d$" is the **weight** "$G_t^v$" in that same **direction** multiplied by the **evenly distributed** (i.e., **isotropic**) **power density flux** "$P_t/4\pi d^2$":

$$S_t^v(d) = G_t^v \bullet P_t/4\pi d^2$$

The **integral** of the **weight** "$G_t^v$" over **every direction** "$v$" **must** be **1**, so that the **total radiated power** is exactly "$P_t$" (**power conservation**).

**Power redistribution** when used for **reception**

A **passive lossless antenna** can **output** a **different power** "$P_r^v$" from a given **power density flux** "$S_r^v=S_r$" that it **receives** from **every direction** "$v$", as long as its **total output power** "$P_r$" is **exactly** the **same** as the **total output power** "$P_r^{iso}(S_r)$" that would be **output** by an **antenna** that **outputs** the **same power** "$P_r^{iso,v}(S_r)$" from **each power density flux** "$S_r^v=S_r$" that it **receives** from **any direction** "$v$" (i.e., an **isotropic antenna**, click 11.21.2). This **last requirement** arises from **power conservation** (click 11.21.1.1).

The **total output power** is the **same** "$P_r^{iso}(S_r)$", **irrespective** of the **power redistribution** that the **antenna performs**, but the **output signal** (i.e., the **content**) is **different**! The reason is that the **power density flux** "$S_r^v$" that is **arriving** from **each direction** "$v$" has the **same value** "$S_r$", but (in the **general case)** the **signal** from **each direction** "$v$" is **different**.

For example, the **antenna** can **output most** of its **total power** "$P_r^{iso}(S_r)$" from the **power density flux** "$S_r^{v1}$" that is **arriving** from **direction** "$v_1$", while it **outputs** a **very low power** from the **power density fluxes** that are **arriving** from **many** other **directions** that are comprised in a **large solid angle**, as long as the **total output power** is the **same** "$P_r^{iso}(S_r)$" (**power conservation**). In this case, the **signal** that is **output** by the **antenna** would be **very similar** to the **signal** that is **arriving** from **direction** "$v_1$" and would just have a **small component** from the **signals** that are **arriving** from other **directions**.

Notice that the **total output power** is "$P_r^{iso}(S_r)$" **because** the **antenna** is **receiving** an **homogeneous power density flux** "$S_r$" from **every direction** "$v$". In case that the **antenna** receives a **power density flux** "$S_r^v$" that **is lower** than "$S_r$" from **some directions** "$v$", its **total output power** would be **less** than "$P_r^{iso}(S_r)$".

The specific **Rx power redistribution** of an **antenna** can be specified by associating a **weight** "$G_r^v$" to **each** and **every direction** "$v$". The **output power** "$P_r^v$" corresponding to the **power density flux** "$S_r^v$" received from **direction** "$v$" is the **weight** "$G_r^v$" in that **same**

**direction** multiplied by the **power** "$P_r^{iso,v}(S_r^v)$" that an **isotropic antenna** (click 11.21.2) would **output** from that **same incoming power density flux** "$S_r^v$":

$$P_r^v = G_t^v \bullet P_r^{iso,v}(S_r^v)$$

The **integral** of the **weight** "$G_r^v$" over **every direction** "$v$" **must** be **1**, so that the **total received power** is **conserved**.

**Same power redistribution for Tx and for Rx**

Because of the **symmetry** of **Maxwell**'s **equations**, the **power redistribution** (i.e., its **radiation pattern**, click 11.21.5) of a **passive lossless antenna** among the **different directions** is **exactly** the **same** when using the **antenna** for **transmission** and when using the **antenna** for **reception**.

Consequently, the **gain** "$G_t^v$" of an **antenna** used for **transmission** is exactly the same as its **gain** "$G_r^v$" when used for **reception**, in **every direction** "$v$":

$$G^v = G_t^v = G_r^v$$

This **property** is sometimes called the **reciprocity** between using an **antenna** for **transmission** or for **reception**.

### 11.21.1.3 Directional antennas

**Power redistribution** (click 11.21.1.2) allows to build pa**ssive lossless directional antennas**. However, if a **directional antenna increases** its **radiated power density flux** (in **Tx**), or the **output power** (in **Rx**), in a given **direction** it **must reduce** its **radiated power density flux** (in **Tx**), or its **output power** (in **Rx**), in the **same amount** in a **joint** manner from **other directions**.

We can therefore build a **directional passive lossless antenna**, but it always **preserves** the **total radiated power** (in **Tx**) from **every direction** and the **total output power** (in **Rx**) from **every direction**.

**Directional antennas** are therefore **very useful** when it is required to **concentrate** the **radiated/received power** to/from **some directions**. Some such cases are:

- **Radio link**

  This is the case where a **directional antenna** gets its **most**, because it is required to **concentrate all the radiated power** in **only 1 direction**. **Radio links** are frequently used in **mobile communications** for the **backhaul** (click 4.6.1) connectivity.

  You may be tempted to believe that if the **antenna** produces a **perfectly collimated beam** we could reach **unlimited distance** in **vacuum**, but as you may guess, this is **not possible** (click 11.21.7.2).

- **Horizontal omnidirectional communications**

  **Mobile communications** (e.g., **5G**) frequently require that **power** is **radiated** in **every horizontal direction**, but **not** in **every direction**. Consequently, **directional antennas** are **greatly advantageous** by **concentrating** the **power** in the **horizontal directions** while reducing it in the **vertical directions**. For an example of a **vertical radiation pattern** with this approach, click 11.21.5.

- **Sectorization**

  **Mobile communications** (e.g., **5G**) frequently require that **power** is **radiated** in **every horizontal direction**. However, dividing **every horizontal direction** into **sectors**, **each sector** produced by a **directional antenna**, allows to perform **space reutilization** of the **spectrum**.

- **Individual sector(s)**

  **Mobile communications** (e.g., **5G**) sometimes require that **power** is **radiated** in **some horizontal directions** (e.g., covering a town from hill-top). In these cases, a **directional antenna** allows to **concentrate** the **power** in the **required horizontal directions**.

## 11.21.2 Gain of an antenna

For **any** given **antenna surface** "$sf^v$" **orthogonal** to **direction** "$v$", the **antenna gain** "$G^v(sf^v/\lambda^2)$" of an **antenna**, in the **same direction**, is its **effective area** "$A_{eff}{}^v(\lambda, sf^v)$" (click 12.1.6) in that **direction**, **divided** by the **effective area** "$\lambda^2/4\pi$" of an **isotropic antenna** (click 11.21.3).

The **antenna gain** is **unitless**. The **antenna gain** is most frequently expressed in **decibels** over an **isotropic antenna** (**dBi**) or just in decibels (**dB**), but in some formulas it appears as a ratio.

An **isotropic antenna** has an **antenna gain** "$G=G^v=1=0$ **dBi**" in **every direction** "$v$", because an **isotropic antenna** has the **same antenna effective area** in **every direction**.

A **directional antenna** has a **very different antenna gain** "$G^v(sf^v/\lambda^2)$", in **each specific direction** "$v$".

The **antenna gain** "$G^v(sf^v/\lambda^2)$" in **each direction** "$v$" is a **measure** of the **antenna directivity** in that **same direction**.

Because of **power conservation** (click 11.21.1.1), the **integral** of the **antenna gain** "$G^v(sf^v/\lambda^2)$" over **every direction** "$v$" **must** be **1**.

The **antenna gain** "$G^v(sf^v/\lambda^2)$" in **each** and **every direction** "$v$" is the same as its **radiation pattern** (click 11.21.5).

An **antenna gain** "$G^v(sf^v/\lambda^2)$" greater than **1** in a given **direction** "$v$" is **not** a **power amplification**. Recall from 11.21.1.1 that a **passive lossless antenna preserves** the **power**: the **antenna cannot amplify** the **power**.

An **antenna gain** "$G^v(sf^v/\lambda^2)$" greater than **1** in a given **direction** "$v$" indicates that the **antenna** is **increasing** the **radiated power density flux** (in **Tx**) or the **output power** (in **Rx**) in that **same direction**, at the **cost** of **reducing** the **radiated power density flux** or the **output power** in the **same amount** in a **joint** manner from **other directions**.

**Antenna gain** and **antenna surface**

The **antenna gain** "$G^v(sf^v/\lambda^2)$" does **not directly depend** on the **wavelength** "$\lambda$" because the **effective area** "$A_{eff}{}^v(\lambda, sf^v)$" of an **antenna** is **proportional** to "$\lambda^2$", but it is **divided** by the **effective area** "$\lambda^2/4\pi$" of an **isotropic antenna** (click 11.21.3). The "$\lambda^2$" factor in the **numerator** and in the **denominator cancel each other**, and the result is that the **antenna gain** does **not directly depend** on the **wavelength** "$\lambda$".

However, the **antenna gain** "$G^v(sf^v/\lambda^2)$" depends on the **antenna surface** "$sf^v$" divided by the **square** of the **wavelength** "$\lambda^2$". The **antenna gain** "$G^v(sf^v/\lambda^2)$" in **direction** "$v$" is

approximately **linear** with the **antenna surface orthogonal** to that **direction** divided by the **square** of the **wavelength** "$sf^v/\lambda^2$":

$$G^v(sf^v/\lambda^2) \approx K_G \bullet sf^v/\lambda^2$$

This holds as long as the **antenna surface** is **larger** than the **square** of the **wavelength** "$\lambda^2$". For more detail, click 11.21.4.

### 11.21.3 Isotropic antenna

An **isotropic antenna** is the one that has the **same gain** "$G$**=1=0 dBi**" (click 11.21.2) in **every direction**. This is the same as having the same **effective area** "$A_{eff}^{iso}(\lambda)$" (click 12.1.6) in **every direction**.

Consequently, an **isotropic antenna radiates** the **same power density flux** in **every direction** and it **outputs** the **same amount** of **power** from **every direction**.

For **any** given **wavelength** "$\lambda$", the **effective area** "$A_{eff}^{iso}(\lambda)$" of an **ideal point-like isotropic antenna** in **every direction** is:

$$A_{eff}^{iso}(\lambda) = \lambda^2 / 4\pi$$

You can see that its **effective area** is **directly proportional** to the **square** of the **wavelength** "$\lambda^2$": the **larger** the **wavelength** "$\lambda$", the **larger** (to the **square**) the **effective area**.

Consequently, given a certain incoming **power density flux**, for an **isotropic antenna** used for **reception**, the **larger** the **wavelength** "$\lambda$", the **larger** (to the **square**) the **output power** from an **isotropic antenna**. **Double wavelength implies quadruple output power**.

However, this is **not true** for an **isotropic antenna** used for **transmission**, because (recalling from 11.21.1) the **radiated power density flux** in a given **direction** "$v$" is **not proportional** to the **effective area** in that **same direction**, and it is **rather proportional** to the **effective area** in that **same direction divided** by the **effective area** of an **isotropic antenna**.

An important conclusion for an **isotropic antenna**, is that its **radiated power density flux** (when used for **transmission**) is **independent** of the **wavelength** "$\lambda$", but its **output power** (when used for **reception**) is **proportional** to "$\lambda^2$".

It may be somehow surprising that an **ideal point-like antenna** that has **zero surface** can capture **electromagnetic energy** from **any direction** in an **effective area** of "$\lambda^2/4\pi$". This is due to the **Fresnel zone**, where the **interaction** between the **point-like antenna** and the **incoming photons** extends a distance of "$\lambda$" around the **antenna**, and therefore, its **effective area** is proportional to "$\lambda^2$".

Notice that the **ideal point-like antenna** does **not exist** and that **real antennas** have a certain **minimum surface** that is in the order of magnitude of the **wavelength**.

An **isotropic antenna** (click 11.21.3) is the **only antenna** that has the **same effective area** in **every direction**. **Any non-isotropic** (i.e., **directional**, click 11.21.1.2) **antenna** has **different effective areas** in **different directions**.

An **isotropic antenna** is sometimes called an **omnidirectional antenna**.

## 11.21.4 Building a directional antenna

Two ways (among many others) of **building** a **directional antenna** are using **multiple antenna elements** and **incorporating** a **passive capture device** to the **antenna**.

**Multi-element antennas**

A **multi-element antenna** that is built with "***n***" **antenna elements** that are located in a **plane** has an **effective area** (click 12.1.6) in the **direction** that is **orthogonal** to the **plane** which is "***n***" **times larger** than that of **1 antenna element**. This holds as long as: **a)** the **output signals** of **all** the **antenna elements** are **in phase**; and **b)** the **antenna elements** are **sufficiently separated** so that their **effective areas** do **not overlap**.

Notice that the **surface** "$sf^v$" of this **multi-element antenna** is "***n***" **times larger** than **1 antenna element**.

For example, an **antenna** that is built with **20 real isotropic** (click 11.21.3) **antenna elements** placed on a **plane** would have an **effective area** "$A_{eff}^{v}(\lambda, sf^v)$" (in the **direction** "***v***" that is **orthogonal** to the **plane**) that is **20 times larger** than that of **1 real isotropic antenna** "$A_{eff}^{iso}(\lambda, \lambda^2)$":

$$A_{eff}^{v}(\lambda, 20 \bullet \lambda^2) = 20\ A_{eff}^{iso}(\lambda, \lambda^2)$$

It is required (as indicated above) that their **output signals** are **in phase** (**constructive interference**) and that the **distance** between the **center** of **every two antenna elements** is at least **two wavelengths** "$2\lambda$". It has been assumed that the **surface** of the **real isotropic antenna** is "$\lambda^2$".

Consequently, for a given **power density flux**, this **20-element antenna** used for **reception** would **capture** and **output**, from **direction** "***v***", **20 times more power** than **1 real isotropic antenna**, but the **antenna surface** "$sf^v$" is also **20 times larger**.

**Aperture antennas**

An **aperture antenna** that is built with a **passive device** (e.g., a **parabolic reflector** or a **horn**) with a **plane surface** that is "***n***" **times larger** than the **effective area** (click 12.1.6) of an **isotropic antenna** has an **effective area** in the **direction** that is **orthogonal** to the **reflector** or **horn** that is "***n***" **times larger** than that of a **real isotropic antenna**. This holds as long as the **plane surface** of the **reflector** or **horn** is **much larger** than the **wavelength** "$\lambda$".

This **type** of **antennas** are called "**aperture antennas**", because they **capture power** with a **well-defined effective area** (i.e., **aperture**).

Notice that the **surface** "$sf^v$" of this **multi-element antenna** is "***n***" **times larger** than **1 isotropic antenna**.

For example, an **antenna** that is built with a **parabolic reflector** whose **plane surface** is **20 times** the **effective area** of a **real isotropic antenna** would have an **effective area** "$A_{eff}^{v}(\lambda, sf^v)$" (in the **direction** "***v***" that is **orthogonal** to the **reflector**) that is **20 times larger** than that of **1 real isotropic antenna** "$A_{eff}^{iso}(\lambda, \lambda^2)$":

$$A_{eff}^{v}(\lambda, 20 \bullet \lambda^2) = 20\ A_{eff}^{iso}(\lambda, \lambda^2)$$

It is required, as indicated above, that the **plane surface** of the **reflector** is **much larger** than the **wavelength** "$\lambda$". It has been assumed that the **surface** of the **real isotropic antenna** is "$\lambda^2$".

Consequently, for a given **power density flux**, this **parabolic antenna** used for **reception** would **capture** and **output**, from **direction** "***v***", **20 times more power** than **1 real isotropic antenna**, but the **antenna surface** "***sf***" is also **20 times larger**.

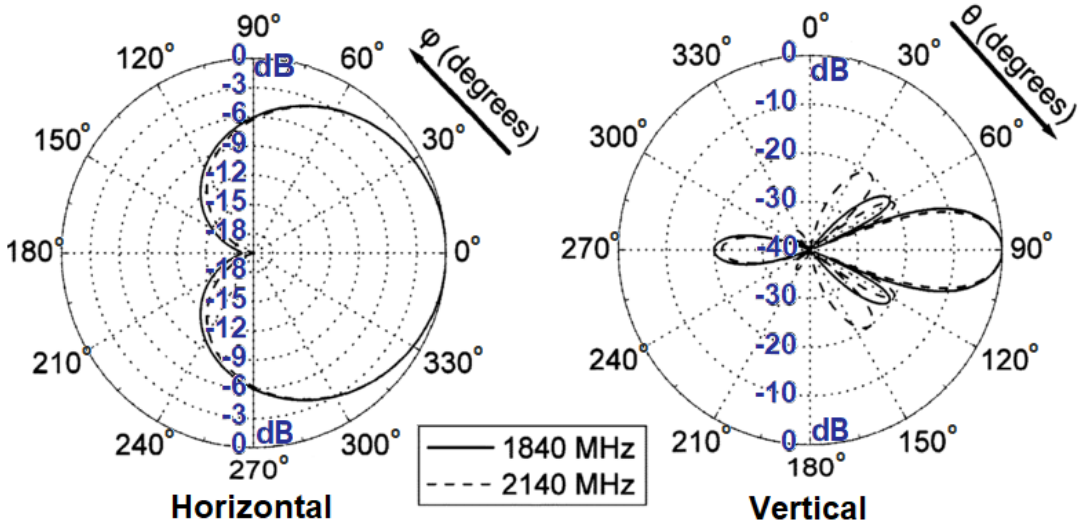
## 11.21.5 Radiation pattern of an antenna

The **radiation pattern** of an **antenna** is its **antenna gain** "$G^v$", in **each** and **every direction** vector "***v***". The **antenna radiation pattern**, same as the **antenna gain**, depends on the **wavelength** "**λ**" (equivalent to **frequency** "***f***" dependency).

The **directivity** of an **antenna** is characterized by its **radiation pattern**.

The **radiation pattern** of an **antenna** is usually represented, for a given **frequency**, by **depicting** its **gain** "$G^v$" in **every horizontal direction** and in **every vertical direction**, divided by its **maximum gain** "$G_{max}{}^v$", and expressed in **decibels** (**dB**). Consequently, the **maximum represented gain value** is "**0 dB**", and the **other gain values** are **negative** in **dB**.

As an example of **antenna directivity**, you may see the following **horizontal** and **vertical radiation patterns** of a **directional antenna**, for **two different frequencies**:




The **horizontal** radiation pattern is also called the **azimuth** "**φ**" radiation pattern and the "**top view**" radiation pattern. The **vertical** radiation pattern is also called the **elevation** "**θ**" radiation pattern and the "**side view**" radiation pattern.

As is the case of this example, the **radiation pattern** of a **directional antenna** usually has a **main lobe** plus **several secondary lobes**.

It is a **fundamental property** of **passive antennas** that the **transmission** and **reception radiation patterns** are the **same**, because of the **reciprocity property** (click 11.21.1.2).

## 11.21.6 Beamwidth of an antenna

The **antenna half-power beamwidth** is the **angular separation** between the **two directions** where the **radiated power drops** to **half** the **maximum value**. It may be seen as the **angular width** of the **main lobe** of an **antenna**. The **half-power beamwidth** is usually just called **beamwidth**.

The **antenna half-power beamwidth** may be determined **both** in the **horizontal** and in the **vertical** planes.

**Half-power** is the same as **-3 dB** over the **maximum gain** of the **antenna** It is therefore easy to determine the antenna **beamwidth** from the **antenna radiation pattern**: just measure the **angle** between the **two directions** that **cross** the **-3 dB circle**. For example, the **horizontal beamwidth** of the **radiation pattern** figure from 11.21.5 is **120º**.

The **half-power beamwidth** is sometimes also applied to **secondary lobes** in addition to the main lobe. In this case, the "**maximum value**" in the definition above is interpreted as the "**maximum value** in the **lobe**".

## 11.21.7 Thought experiments on passive lossless antennas

You may find somehow **illustrative** the following **two thought experiments** on **passive lossless antennas**.

### 11.21.7.1 Power conservation in reception

In **transmission**, the **power conservation** is **completely intuitive**, because it is crystal clear that the **total power** that the **antenna radiates cannot** be **larger** than the **power fed** to the **antenna**, irrespective of the **power redistribution** performed by the **antenna**.

However, in **reception** the **power conservation** is **not intuitive** at all. It "**seems**" that you can **build** an **antenna** in a "**smart**" way, so that its **total receive gain** is **higher** than **1** (i.e., **higher** than **0 dBi**).

Let me show you a **simile** that may be **reasonably intuitive** and **convincing**.

You have a **horizontal flat surface** of **100×100 m**. This represents the **spherical surface around** the **antenna**, that I **turn fla**t to make it more visual.

**Each square** of **10×10 m** represents a **different direction** towards the **antenna**. Then, there are **100 squares** that represent a **total** of **100 directions**.

There is an **homogeneous rain density flux** "$S$" over the **100×100 m entire surface**. The **rain density flux** represents the **power density flux arriving** to the **antenna**.

The **rain falling** in **each square** represents a **plane wave arriving** to the **antenna** from the **direction** that that **square** represents.

The **rainwater** is **colored** with a **different color** for **each direction**, so that **all** the **rain** that falls in **each square** has the **same color**. Then, there are **100 colors**. **Each color** represents a **different signal** that arrives from **each direction**, as a **plane wave**.

Notice that the **rain falling** into **each square** is the **same** in terms of **rain density flux**, but **not** in terms of **color**: **each square** gets the **same rain density flux** but with a **different color**.

**Isotropic antenna**

You have **11,000** pyramidal **funnels**, where **each funnel** has a **surface** of **0.1×0.1 m**. **Each such funnel** is connected to a **flexible pipe**, and all the **11,000 pipes** are connected into **1 single output pipe**. The **joint water flow out** of these **11,000 funnels** represents the **total power output** of an **isotropic antenna**. Because this is an **isotropic antenna**, you place the **same quantity** (**110**) of **funnels** in **each** of the **100 squares**.

The **water flow** (i.e., **output power**) out of the **11,000 funnels** of the **isotropic antenna** is "$P_r$" has an **homogeneous mix** of the **100 colors** (i.e., of the **100 different signals**).

**Directional antenna**

When you **build** a **directional antenna**, you **move** some **funnels** between the **squares**. This represents that the **design** that you chose for your **directional antenna** is **redistributing** the **power** among the **different directions**.

You can **neither add funnels** (**passive antenna**, it **cannot amplify**), **nor remove funnels** (**lossless antenna**): the **quantity** of **funnels** stays at **11,000**.

The **quantity** of **funnels** in **each square**, **divided** by **110**, **represents** the **antenna gain** in that **direction**. If you place **825 funnels** in a given **square**, the **gain** in that **direction** is **7.5**. If you place **1 funnel** in a given **square**, the **gain** in that **direction** is **.009**.

If you place **10,000 funnels** in **1 square**, **20 funnels** in **1 other square** and **10 funnels** in **each** of the **other 98 squares**, you get a **gain** of **90.9** in the **first direction**, a **gain** of **.18** in the **second direction**, and a **gain** of **.09** in **each** of the other **98 directions**.

The **water flow** (i.e., **output power**) out of the **11,000 funnels** of your **directional antenna** is **exactly** the **same** as the one the **isotropic antenna**, but the **color** is completely different! In this case, the **output color** will be **extremely similar** to the **color** of the **square** with **10,000 funnels**, instead of an **homogeneous mix** of the **100 colors**.

I have **intentionally** chosen the **quantity** of **funnels** so it is **not** possible to place **all** of them in the **same square** (see the next subsection).

Notice that if the **rain density fluxes** from **some directions** are **lower** than "$S$", the **water flow** (i.e., **output power**) out of the **11,000 funnels** of your **directional antenna** will be **lower** than "$P_r$". **Two examples** are:

- If there is **no rainfall** (i.e., **no signal**) on the **square** with **10,000 funnels** (i.e., **gain 90.9**), the **output water flow** would be

  $$P_r' = 1{,}000/11{,}000 \bullet P_r = 0.09 \bullet P_r$$

  and its **color** (i.e., its **signal**) would be an **almost homogeneous mix** of **all colors**, **except** the one that corresponds to the **square** with the **10,000 funnels**. This would be basically **interference** with the corresponding **noise**

- If there is **no rainfall** (i.e., **no signal**) on the **99 squares** with **10** (or **20**) **funnels**, the **output water flow** would be

  $$P_r' = 10{,}000/11{,}000 \bullet P_r = .91 \bullet P_r$$

  and its **color** (i.e., its **signal**) would be **exactly** the **same** as the **color** of the **square** with **10,000 funnels**. This would be basically **signal** with the corresponding **noise**.

### 11.21.7.2 Perfectly collimated beam

You know from **math** that a **parabolic reflector** with a **source** of **rays** placed at the **focus**, produces **perfectly parallel rays** out of the **reflector**.

You may therefore be tempted to believe that it is possible to produce a **perfectly collimated radio beam**. If this were possible, you could communicate over any distance in vacuum, as long as you can achieve **sufficiently good alignment** between the **transmitted collimated beam** and the **receiver parabolic reflector**, that could be made **larger** than the **transmitter** to compensate for **certain degree** of **misalignment**.

This sounds really well, but as you can guess, it is **not possible**.

A **perfectly collimated radio beam** would have **zero beamwidth** (click 11.21.6) and **infinite gain** (click 11.21.2). **Zero something** is **sometimes worrisome**, but **infinite something** is a usual sign that **something** is **wrong**.

The reason why it is **not possible** to build a **perfectly collimated beam** with a **parabolic reflector** is **diffraction** (click 12.1.9). **Diffraction** over the **edges** of the **parabolic reflector** causes **side lobes**, which implies that the **gain** of the **main lobe** is **not infinity**, and the **beamwidth** is **not zero**.

The **larger** the **parabolic reflector**, the **lower** the **diffraction** effect, the **smaller** the **beamwidth** and the **larger** the **gain**.

### 11.21.7.3 Sector amplification

We have an **isotropic antenna** fed with **power** "$P_t$", which results in a given **cell size** and **area capacity**.

We build a **sectorized antenna** with **3 antenna elements**, **each** with a **120º spherical wedge** and a **gain** of **4** in every **direction** "***v***" **within** the **spherical wedge**. We replace the **omnidirectional antenna** by the **sectorized antenna**. We **reduce** the **power fed** to the **antenna** to "3/4$P_t$". This results in the **same cell size** and **area capacity**, because **each antenna element** gets "1/4$P_t$", that **multiplied** by the **gain** of **4**, results in the **same value** for the **free-space path-loss** formula (click 12.2.5.1). Consequently, we are **saving power**!

This sounds really well, but as you can guess, it is **not possible**.

The **caveat** is that you **cannot** build an **antenna** with a **120º spherical wedge** and a **gain** of **4** in every **direction** "***v***" **within** the **spherical wedge** because the **integral** of the **antenna gain must** be **1**. If you reduce to **zero** the **gain** in the **remaining 240º**, the **highest gain** you can get for **every direction** "***v***" **within** the **spherical wedge** ºis **3**.

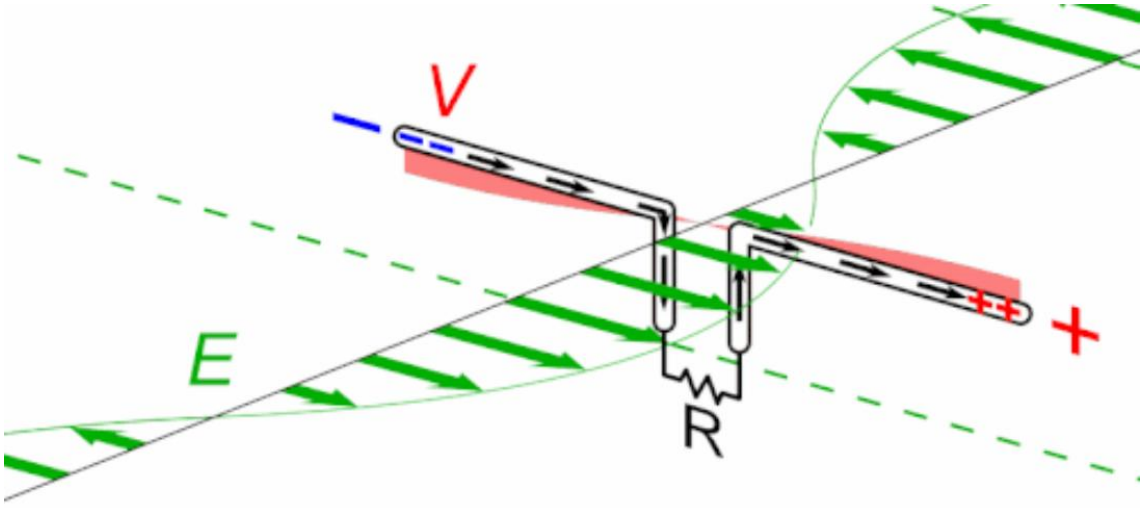
If you see an **antenna horizontal radiation pattern** (click 11.21.5) with an **average gain** that is **higher** than **3** within a **120º horizontal sector** it is because the **power** has been **reduced** from the **vertical directions** (see the corresponding **vertical radiation pattern**). Whatever the **vertical radiation pattern** is, if you define a **120º horizontal sector**, the **most** you can get in **every direction** "***v***" **within** the **sector** is **three times** the **gain** of an **isotropic horizontal radiation pattern**.

## 11.21.8 Dipole antenna

A **dipole antenna** consists of **2 identical metal rods**, where the **antenna terminals** are usually connected to an **extreme** of **each** of the **2 rods**.

A basic design for a **dipole antenna** is **2 identical L-shaped metal rods**, with the **long L-sides aligned** and with the **short L-sides** in **parallel** with a **small separation** between them, where the **2 antenna terminals** are connected to the **2 extremes** of the **2 short**

**sides**. The following figure shows such a **half-dipole reception antenna** interacting with an **incoming electrical wave**:




For the **animated** version of the figure, which is **very illustrative**, refer to the source web page (see below).

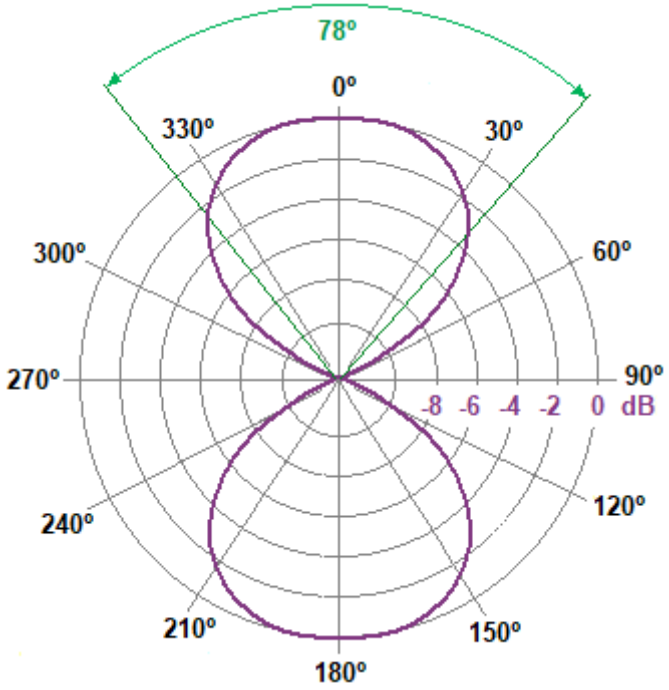
The **axis** of such a **dipole antenna** is the **axis** determined by its **2 long L-sides**.

The **length** of such a **dipole antenna** is the **distance** between the **2 opposite extremes** of its **2 long L-sides**.

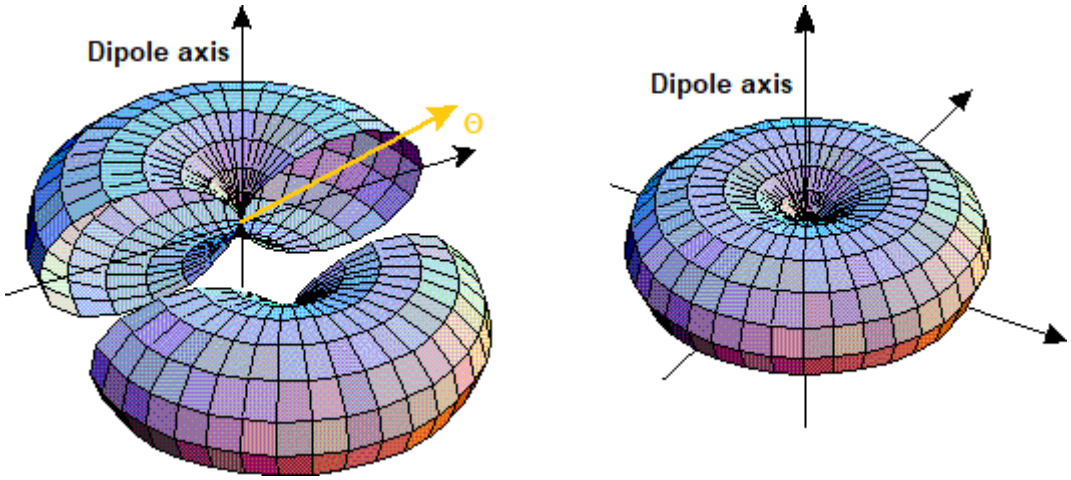
The **length** of a **dipole** is typically determined by the intended operation **wavelength** "λ". The **most commonly used** is the **center-fed half-wavelength dipole**, whose **length** is **half-wavelength** "λ/2".

The **radiation pattern** (click 11.21.5) of a **half-wavelength dipole** is **omnidirectional** in the **orthogonal plane** to its **axis** and it has **2 directional radiation lobes** pointing **outwards** in the **direction** of its **axis**.

The following figure shows the **radiation diagram** of a **half-wavelength dipole** in any **plane** that contains its **axis**:




Consequently, the **3D radiation diagram** of a **half-wavelength dipole** has a similar shape to a **hole-less doughnut**. The following figure shows the **characteristic radiation diagram** of a **half-wavelength dipole**:




**Dipoles** can be used as **self-standing antennas**, as **antenna elements** of a **multi-element** (click 11.21.4) **antenna**, or as **individual antennas** within an **antenna array**.

A **dipole antenna** is quite different from a **monopole antenna**, which consists of a **single metal rod**, where **one antenna terminal** is connected to **1 extreme** of the **rod** and the **other terminal** is connected to the **circuit ground**.

Source web page:

- https://en.wikipedia.org/wiki/Dipole_antenna

## 11.22 Self-interference mitigation

**Self-interference** (**click** 12.1.14) **arises** when a **given system** (**UE** or **NR node**) is **simultaneously transmitting** and **receiving** in the **same channel**. Since **radio** systems **transmit much more power** than they **receive** (e.g., **110** or **130 dB**), the resulting **self-interference** would make **communication impossible**. It is therefore required to **space away transmission** and **reception either** in **frequency** (**guard bands** in **FDD**) or in **time** (**no simultaneous Tx** and **Rx**: **TDD**).

**Self-interference in FDD bands**

**Roughly speaking**, for **FR1 FDD bands** that define a **separation** between **Downlink** and **Uplink channels** of around **40 MHz** at the **NR node**, and **considerably more** at **each UE**, is enough to **avoid self-interference**. The **higher** the **frequency band**, the **greater** the **separation must be**.

The **Downlink** and **Uplink channels** of **5G FDD bands** are **sufficiently spaced away** to **allow** for **simultaneous Tx** and **Rx** (**both** at the **NR node** and at the **UEs**) in **any DL/UL channel pair** and in almost **any combination** of **DL/UL channel pairs** (for **Carrier Aggregation**, click 10.2).

**Self-interference in TDD bands or in TDD+FDD bands**

The **Downlink** and **Uplink symbol intervals** (**TDD slot format**, click 2.3.4.2) for the **same slot** on the **nearby channels** may be different. This implies that the **same system** is **simultaneously transmitting** on a **channel** and **receiving** on a **nearby channel**. If the **subcarrier spacing** of the **nearby channels** is **different** the problem **worsens**, because **each symbol interval** in **one channel** corresponds to **several symbol intervals** on a **nearby channel**. Some solutions for this problem are:

- **Homogeneous TDD channel configuration**

  **Each operator** may configure **all its TDD channels** in the **same band** (in the same area) are configured the **same subcarrier spacing** and with the **same TDD slot format**. This **avoids** (or **largely mitigates**) the **TDD self-interference problems**.

- **Different antennas** and/or **high-quality filters** at the **NR node**

  **Each operator** may install at **each antenna site** a **specific antenna** for **each** of the **nearby channels** , combined with **high-quality** (i.e., **high price**) filters. for **all its TDD channels** in the **same band** (in the same area) are configured the **same subcarrier spacing** and with the **same TDD slot format**. This approach allows an operator to use a **TDD channel adjacent** to an **FDD channel** with **only 5 MHz** of **guardband**.

  An extreme, but **feasible**, example is the **usage** of the **guardband spectrum** that is defined between the **Downlink** and **Uplink semi-bands** (click 2.2.12) of some **FDD bands**.

- **NR node: different antennas** and/or **high-quality filters**

  **Each operator** may install at **each antenna site** a **specific antenna** for **each** of the **nearby channels**, combined with **high-quality** (i.e., **high price**) **filters**. It is also required that the **antennas** are **sufficiently distant** among themselves. This approach allows an **operator** to use a **TDD channel adjacent** to an **FDD channel** with **only 5 MHz** of **guardband**.

- **UEs: Different antennas** and/or **high-quality filters** at the **NR node**

  The **NR node approach** (previous bullet) is **not feasible** at the **UE**: **neither individual distant antennas** nor **high-quality filters**. Consequently, a given **UE cannot simultaneously** use **nearby channels** from a **TDD band** and a from **TDD band**. However, the **NR node** can **simultaneously use** them, and therefore, it can assign **each** of the **nearby channels** to a **different group** of **UEs**.

## 11.23 Same-channel interference mitigation

**Neighbor cells** on the **same channel** have to **mitigate** their **same-channel interference** on **UEs** located in the **border/intersection** between their **coverage areas**. One technique that may largely reduce the **same-channel interference** is by coordinating the **scheduling** of **UEs** between both **cells**. In this way, the **cell** that is **not serving** a given **UE** that is located in the **border** should **avoid scheduling REs** in the same **REs** that are **scheduled** to this **UE**.

**UEs** in the **same channel** may mitigate their **interference** based on **synchronization**, **power control** and **scheduling coordination** by their **serving NR node**.

# 12. Received signal distortion

In a telecommunications system, the **received signal** is usually **substantially different** from the **transmitted signal**. This **distortion** experienced by the **signal** along its transit from the **transmitter** to the **receiver** arises from several **phenomena** that are described in the following subsections.

## 12.1 Individual causes of received signal distortion

**Each telecommunication signal** is transported as one, or more, **electromagnetic beam(s) transmitted** by the transmitter and received by the **receiver(s)**. **Each received signal** is **distorted** in respect to the **transmitted signal** due to **different impairments**. The **main impairments** are due to the following **individual causes**:

- *Absorption*
- *Scattering*
- *Refraction*
- *Reflection*
- *Spreading*
- *Effective area*
- *Time dispersion*
- *Doppler*
- *Diffraction*
- *External Noise*
- *Internal Noise*
- *Thermal Noise and Scattering Noise*
- *External interference*
- *Self-interference*
- *Radio frequency subsystem distortion*

The following subsections contain a brief explanation of **each** of these **individual causes**.

### 12.1.1 Absorption

**Absorption** is the increased **vibration** or **motion** (i.e., **heat**) of a **medium particle** as a consequence of having **absorbed** a **photon** from the **electromagnetic beam**. **Photons** interact with **charged particles**, typically with **electrons**, **polarized molecules** or **nanoparticles**.

**Absorption** is caused by the physical propagation **medium**, and it does **not** happen in **vacuum**.

**Absorption** happens even if you send a **perfectly collimated beam**, because it is a substantially different phenomenon than *Spreading*.

If a **medium particle** that **absorbs** a **photon emits**, as a consequence, **another photon** this is **not absorption** and it is **rather** *Scattering*.

Both **absorption** and *Scattering* cause that the **electromagnetic beam power** is **progressively reduced** as it propagates along the **medium**.

### 12.1.2 Scattering

**Scattering** is the **emission** of a **photon** by a **propagation medium particle** as a consequence of it having **absorbed a photon** from the **beam**. **Photons** interact with **charged particles**, typically with **electrons**, **polarized molecules** or **nanoparticles**. The **emitted photon** is **not** usually sent in the **same direction** as the **absorbed photon**, and it may have a **different energy** (i.e., **different frequency**). The **physics term** for this specific type of **scattering** is "**Rayleigh scattering**".

**Scattering** is caused by the physical propagation **medium**, and it does **not** happen in **vacuum**.

**Scattering** happens even if you send a pe**rfectly collimated beam**, because it is a substantially different phenomenon than *Spreading*.

If the **absorbed photon increases** the **temperature** of the **medium**, this is *Absorption*.

**Both** *Absorption* and **scattering** cause that the **electromagnetic beam power** is **progressively reduced** as it propagates along the **medium**.

### 12.1.3 Refraction

**Refraction** is a **slight change** of **direction** of the electromagnetic beam caused by the propagation **medium** being **non-homogeneous**, and its different components have **different** electromagnetic propagation **speeds**. The **change** of **direction** in the electromagnetic beam takes place at each **boundary** between medium components having different propagation velocities. Depending on the **angle** between the beam propagation direction and the medium boundary, the result will be **refraction** or *Reflection* of the electromagnetic beam. When a beam is not perfectly collimated, part of the beam may be **refracted** and part may be **reflected**. If the physical differences within the medium are **gradual**, this will cause a **curvature** of the electromagnetic beam.

**Refraction** and *Reflection* are therefore caused by the physical propagation medium, and **neither** of **them** happens in vacuum.

Notice that **objects** (ground, sea, buildings, satellites, …) **within** the propagation space (be it **air** or **vacuum**) are part of the "medium" and therefore may cause **refraction** or *Reflection*.

Gradual **refraction** is used in gradual-index optical fibers to make the signal propagate along the fiber. *Reflection* is used in step-index optical fibers to make the signal propagate along the fiber.

Atmospheric **attenuation** is highly dependent on frequency, weather and altitude. For a more detailed analysis on atmospheric **attenuation**, click *Atmospheric Attenuation and Refraction*.

### 12.1.4 Reflection

**Reflection** is a **substantial change of direction** of the electromagnetic beam caused by the propagation **medium** being **non-homogeneous**, and its different components have **different** electromagnetic propagation **speeds**. The **change of direction** in the electromagnetic beam takes place at each **boundary** between medium components having different propagation velocities. Depending on the **angle** between the beam

propagation direction and the medium boundary, the result will be **reflection** or *Refraction* of the electromagnetic beam. When a beam is not perfectly collimated, part of the beam may be **reflected** and part may be **refracted**. If the physical differences within the medium are **gradual**, this will cause a **curvature** of the electromagnetic beam.

**Reflection** and *Refraction* are therefore caused by the physical propagation medium, and **neither** of **them** happens in vacuum.

Notice that **objects** (ground, sea, buildings, satellites, …) **within** the propagation space (be it **air** or **vacuum**) are part of the "medium" and therefore may cause **Reflection** and *Refraction*.

The **Fresnel zone** is a clear case on the formalization of impairments caused by **Reflection** and how to avoid them.

**Reflection** is used in step-index optical fibers to make the signal propagate along the fiber. Gradual *Refraction* is used in gradual-index optical fibers to make the signal propagate along the fiber.

Atmospheric **attenuation** is highly dependent on frequency, weather and altitude. For a more detailed analysis on atmospheric **attenuation**, click *Atmospheric Attenuation and Refraction*.

### 12.1.5 Spreading

Spreading is caused by the **distribution** of the **energy** of the electromagnetic beam throughout the **surface** of the **beam front**. If an electromagnetic beam could be **perfectly collimated** it would **not** suffer **spreading**, but this is **not possible**.

**Spreading** is therefore caused by the **transmitting antenna** and **not** by the physical propagation **medium**. **Spreading** will therefore happen in propagation through either **vacuum** or a physical **medium**.

Regardless of the **directivity** (click 11.21.1.2) of the **antenna**, since the transmitted power is **distributed** along the **surface** of the wavefront, the **power density flux** will be **inversely proportional** to the **square** of the **distance**. This is why the *Free-space path-loss* propagation formula has "$d^2$" in the denominator.

Notice that **spreading** is a substantially different phenomenon than *Time dispersion* (click 12.3)

### 12.1.6 Effective area

The **effective area** "$A_{eff}^{v}$" of a **passive lossless antenna** is defined as:

> "*In a given **direction** "**v**", the ratio of the available **power** at the **terminals** of a **receiving** antenna to the power flux density of a plane wave **incident** on the antenna from that **direction**, the wave being polarization matched to the antenna.*"

This is a **simple concept**, but with **complex implications** (click 11.21).

Notice that the definition refers to the **output power** (i.e., "…***power** at the **terminals***…" and **not** to the **captured power**.

Although the **effective area** "$A_{eff}^{v}$" in a given **direction** is defined by using the **antenna** for **reception**, the **value** of the **effective area** of an **antenna** is exactly the **same** when the **antenna** is used for **transmission**, because of the **reciprocity property** (click 11.21.1.2).

Even though the **value** of the **effective area** of an **antenna** is the **same** when it is used for **transmission** or for **reception**, the **implications** are **very different** in **both** cases:

- When the **antenna** is used for **reception**

  The **antenna output power** from **any given direction** "$v$" is **directly proportional** to its **effective area** "$A_{eff}^{v}$" in that same **direction** "$v$".

- When the **antenna** is used for **transmission**

  The **antenna radiated power density flux** in **any given direction** "$v$" is **directly proportional** to its **effective area** "$A_{eff}^{v}$" in that same **direction** "$v$" **divided** by the **effective area** of the **isotropic antenna** (click 11.21.3).

For reception, a **larger effective area** in **direction** "$v$" implies a **larger output power** from that **same direction**. However, for **transmission** that is **not true**.

The fact that the **output power** is **proportional** to the **effective area** might **look** as if the **receiving antenna amplifies** the **power**, but this is completely **false**: a **passive antenna cannot amplify** (click 11.21.1.1). The **received power** (i.e., the **power output** by the **receiver antenna**) "$P_r$" is **always lower** (actually much lower) than the **power** that is **fed** to the **transmission antenna** (i.e., the **total power radiated** by the **antenna** of the **transmitter**) "$P_t$". A **larger effective area** of the **receiver**'s **antenna only** implies that:

- **Less power** is **uselessly dispersed** throughout space because of *Spreading*.
- **More power** is **usefully output** by the **receiving antenna**, from **all** the power that is **dispersed** through space because of *Spreading*.

The **effective area** of an **antenna** "$A_{eff}^{v}(\lambda, sf^{v})$" on a **given direction** "$v$" depends on the **direction** "$v$", on the **wavelength** "$\lambda$" (click 11.21.3) and on the **physical surface** "$sf^{v}$" (click 11.21.4) of the **antenna** in the **orthogonal plane** to **direction** "$v$".

The **effective area** of an **antenna** is also called "**aperture**" and "**receiving cross section**".

### 12.1.7 Time dispersion

**Time dispersion**, (sometimes called "**Dispersion**"), is caused by **different wavelengths** having **different** propagation **velocities** within a **homogeneous** medium, so that the different **frequencies** contained in the beam disperse **in space along the propagation path**, or what is equivalent, the different frequencies spread **in time**. In other words, the **wave-front** of **some frequencies arrives later** than the **wave-front** of **other frequencies**. **Time dispersion** does not reduce the power of the received electromagnetic beam, but, it affects its modulation (click *Inter-symbol Interference*).

**Time dispersion** is therefore caused by the physical propagation **medium** (when the electromagnetic beam has a **combination of frequencies**) and it does **not** happen in **vacuum**. Notice that **time dispersion** is a substantially different phenomenon than *Spreading*.

The **dispersion relation** relates each wavelength to its frequency. Given the **dispersion relation**, one can calculate the **phase velocity** and **group velocity** of waves in the medium, as a function of frequency.

A combination of **time dispersion** and *Refraction* causes an electromagnetic beam to split in different wavelengths when it crosses a boundary between two different media such that the refraction index is different for different frequencies (i.e., the relationship

between propagation speeds is different for different frequencies. This is precisely what happens when a beam of white light crosses a prism, and the beam is split in a continuum of beams of different colors (i.e., different frequencies).

### 12.1.8 Doppler

The **Doppler** effect (or simply **Doppler**, when in context) is the **change** of the **received frequency** in respect to the **transmitted frequency** when the receiver is **approaching** to or **receding** from the **transmitter**. In sound waves, this can be noticed in the **change of pitch** heard when a vehicle **approaches to** and then **recedes from** an observer. The pitch is higher during the approach and lower during the recession.

When the receiver and the transmitter are **approaching**, **each wave crest** is **transmitted** from a **position closer** to the receiver than the **previous wave crest**. Therefore, each **wave crest** takes **slightly less time** to reach the receiver than the previous wave crest. Hence, the **time between the arrivals** of successive **wave crests** at the receiver **is reduced**, causing an **increase** in the **received frequency**. When the transmitter and the receiver are **receding** the opposite happens, and there is there is a **decrease** in the **received frequency**.

**Doppler shift** refers to the frequency variation of an **individual frequency**.

**Doppler spread** refers to the spread (or shrink) of a **frequency range**.

**Doppler** can be formulated as:

$$f_r = (1 + v/c) \bullet f_t$$

$$\Delta f_r = v/c \bullet f_t$$

The **speed** "$v$" is **positive** when **approaching** and **negative** when **receding**. For the case of electromagnetic waves, "$c$" is the speed of light in the medium. In **Doppler shift** "$f_r$" is the **received frequency** and "$f_t$" the **transmitted frequency**. In **Doppler spread** "$f_r$" is the **received bandwidth** and "$f_t$" the **transmitted bandwidth**.

For a speed $v$ of **230 km/h**, **Doppler** shift is $\mathbf{2.13 \bullet 10^{-7}}$. This may seem really small, but if you apply it to a frequency of **3.5 GHz**, you get a **Doppler** shift of **7.5 kHz**, which is **half a subcarrier**.

### 12.1.9 Diffraction

**Diffraction** is the **bending** of waves around the corners of an **obstacle** or through an **aperture** into the region of geometrical shadow of the **obstacle/aperture**. It is caused by the waves experiencing **constructive** or **destructive wave interference**.

In classical physics, the diffraction phenomenon is described by the Huygens–Fresnel principle that treats each point in a propagating wavefront as a collection of individual spherical wavelets. The characteristic bending pattern is most pronounced when a wave from a coherent source (such as a laser) encounters a slit/aperture that is comparable in size to its wavelength, as shown in the inserted image. This is due to the addition, or interference, of different points on the wavefront (or, equivalently, each wavelet) that travel by paths of different lengths to the registering surface. If there are multiple, closely spaced openings (e.g., a diffraction grating), a complex pattern of varying intensity can result.

The amount of diffraction depends on the size of the gap. Diffraction is greatest when the size of the gap is similar to the wavelength of the wave. In this case, when the waves pass through the gap they become semi-circular.

These effects also occur when an electromagnetic wave travels through a medium with a varying refractive index.

### 12.1.10 External Noise

**External noise** is caused by **random** sources of electromagnetic energy **other than thermal/scattering noise** (caused by the medium itself), that arrive at the receiver. Some sources of external noise are sunlight, lightning, sun storms, human-related (electrical motors, sparks, power lines, …) and other phenomena.

**External noise** is therefore caused by different **random** electromagnetic sources and **not** by the physical propagation medium. **External noise** will therefore happen in propagation through either **vacuum** or a physical **medium**.

**External noise** is a substantially different phenomenon than *Thermal Noise and Scattering Noise* and also substantially different than *Internal Noise*.

### 12.1.11 Internal Noise

**Internal noise** is caused by **random** sources of electromagnetic energy within the **transmitter** and within the **receiver**. Some sources of **internal noise** are the **analog processing** components (transistors, …) and **other elements** within the **transmitter** and within the **receiver**. An example of **internal noise** is the electromagnetic **emissions** of some **switching power supplies**, that can introduce **significant power** within the **receive frequency segments**.

**Internal noise** is therefore caused by different **internal random** electromagnetic sources and **not** by the **physical propagation medium**, nor by other devices. **Internal noise** will therefore happen in propagation through either **vacuum** or a physical **medium**.

### 12.1.12 Thermal Noise and Scattering Noise

**Noise** consists of **random** electromagnetic energy arriving at the receiver.

**Thermal noise** is caused by the propagation **medium**, as well as objects contained in it, that produce electromagnetic energy dependent on temperature.

**Scattering noise** is caused by the propagation **medium** when it is exposed to various electromagnetic beams, such as sunlight, lightnings and man-made electromagnetic beams (electrical motors, sparks, RADAR, various radio beams, ...).

**Thermal/scattering noise** are therefore caused by the propagation **medium** and do **not** happen in **vacuum**. Notice that **thermal/scattering noise** are substantially different phenomena than *External Noise* and also substantially different than *Internal Noise*.

### 12.1.13 External interference

**External interference** is the reception of a **non-random** electromagnetic **beam**, **other** than the **beam** sent by the transmitter (in which case it would be *Self-interference*).

**External interference** is a **particular case** within the more **general concept** of *Beam interference*.

The **non-random** electromagnetic beam is typically man-made, and its characteristics are similar to the transmitted beam. On many cases, the **interfering** beam is produced by a telecommunications system **similar** to the one producing the **interfered** beam.

An example of **external interference** is when two nearby WiFi systems are operating on **overlapping channels**: the electromagnetic beam sent by the access point or by the

terminals belonging to one of the systems will **interfere** at the receiver with the electromagnetic beams sent by the ones belonging to the other system. Another example of **external interference** is when two nearby base stations are not properly configured in frequency and/or time, and their electromagnetic beams **interfere** with each other at the receiver. Yet another example is when a domestic TV amplifier malfunctions, becoming an oscillator, and radiates a substantial amount of electromagnetic energy through the rooftop TV antenna in a frequency range overlapping with mobile communications.

### 12.1.14 Self-interference

**Self-interference** is the distortion caused at the **receiver** by the electromagnetic beams that the **receiver itself** is **transmitting**.

**Self-interference** may be caused (among others) by **poor** (or **defective**) **transmission** and/or **reception filters**, by **coupling** between the transmission and reception **antennas**, by **emissions** from **other system components**, by **insufficient guardband** between **transmission** and **reception** and by **system faults**.

**Self-interference** is a **particular case** within the more **general concept** of *Beam interference*.

### 12.1.15 Radio frequency subsystem distortion

The **radio frequency subsystem distortion** is the signal **distortion** arising from the signal **processing** along the **radio-frequency amplifiers**, **filters**, **frequency shifters** and other **RF** elements.

Notice that this effect does **not** arise in the **path** from transmission to reception, and actually, it takes place **before signal transmission** and **after signal reception**. This makes this effect somehow different to most of the other ones included in this section.

## 12.2 Combined causes of received signal distortion

Some **combinations** of **individual causes** of **received signal distortion** (click 12.1) lead to the following **combined causes** of **received signal distortion**:

- *Noise*
- *Beam interference*
- *Multi-path Interference*
- *Inter-symbol Interference*
- *Free-space path-loss*
- *Attenuation*
- *Fading*

The following subsections contain a brief explanation of **each** of these **combined causes**.

### 12.2.1 Noise

**Noise** is **random** electromagnetic energy arriving to the receiver.

Some of the different types of **noise** are:

- *External Noise*
- *Thermal Noise and Scattering Noise*

- *External Noise*

**Noise** is a substantially different phenomenon than *Beam interference*.

### 12.2.2 Beam interference

**Beam interference** is the reception of **non-random** electromagnetic beam(s) (the **interfering** beams), **other** than the **main beam** received from the transmitter (the **interfered** beam).

The **interfering** beam(s) are typically man-made, and its characteristics are similar to the ones of the **interfered** beam. On many cases, the **interfering** beam is produced by a telecommunications system **similar** to the one producing the **interfered** beam.

We may have the following different types of **beam interference**:

- *External interference*
- *Self-interference*
- *Multi-path Interference*

Notice that *Inter-symbol Interference* is **not** included here because **in spite** of its **name**, it is a substantially **different phenomenon** from **radio interference**.

An example of *External interference* is when two nearby WiFi systems are operating on **overlapping channels**: the electromagnetic beams sent by both access points will **interfere** at any receiver that is within the range of both access points. An example of *Self-interference* is the electromagnetic emissions of some switching power supplies, that can produce significant power within the receive frequency bands. An example of *Multi-path Interference* is transmission between two ships, where the **main** electromagnetic beam may follow a **direct** path between both antennas, and a **secondary** beam (from the antenna radiation pattern) may follow a **reflected** path on the water surface.

**Beam interference** is a substantially different phenomenon than *Noise*.

Depending on the **mutual characteristics** of the **interfering** waves, we may have **destructive** or **constructive wave interference**.

**Destructive wave interference** happens when **two waves** are in **antiphase**, this is, when the peaks of one of them occur at the same time as the valleys of the other one. This implies that both waves (partially) **cancel each other**, resulting in a much weaker wave.

**Constructive wave interference** happens when **two waves** are in **phase**, this is, when the peaks and valleys of both of them occur at the same time. This implies that both waves add their energy, resulting in a stronger beam.

Since telecommunications signals vary substantially in time, interfering beams transmitted by **different** senders will usually experience constructive and destructive **wave interference** at different moments, **greatly distorting the receiving signal**. If the two interfering beams come from the same transmitted beam (same sender and same beam sent), under some specific circumstances, they may experience **sustained constructive wave interference**, and therefore, this **improves** the received signal. However, in the general case, interference, even arising from the same beam sent, will negatively affect the received signal.

### 12.2.3 Multi-path Interference

**Multi-path interference** is the reception of one (or more) **secondary beam(s)** sent by the transmitter that have followed a **different path** to the one followed by the **main** beam received from the transmitter.

**Multi-path interference** is caused by a combination of *Spreading*, *Refraction* and *Reflection*. *Spreading* causes **several beams** while *Refraction* and *Reflection* cause that **each beam** may follow a **different path** towards the receiver.

**Multi-path interference** can be destructive, therefore causing *Fading*. Destructive **multi-path interference** may be largely reduced in **fixed** radio communications (i.e., **radio-links**) by avoiding reflective objects within the **Fresnel zone** between the **transmitting antennas** and the **receiving antennas**.

An example of **multipath-interference** is transmission between two ships, where the **main** electromagnetic beam may follow a **direct** path between both antennas, and a **secondary** beam (from the antenna radiation pattern) may follow a **reflected** path on the water surface.

The **delay spread** is **one type** of measure of the **multipath-interference**. It can be interpreted as the **difference** between the **arrival time** of the **first significant multipath component** (the **line-of-sight** component, in case it exists) and the **arrival time** of the **last significant multipath component**.

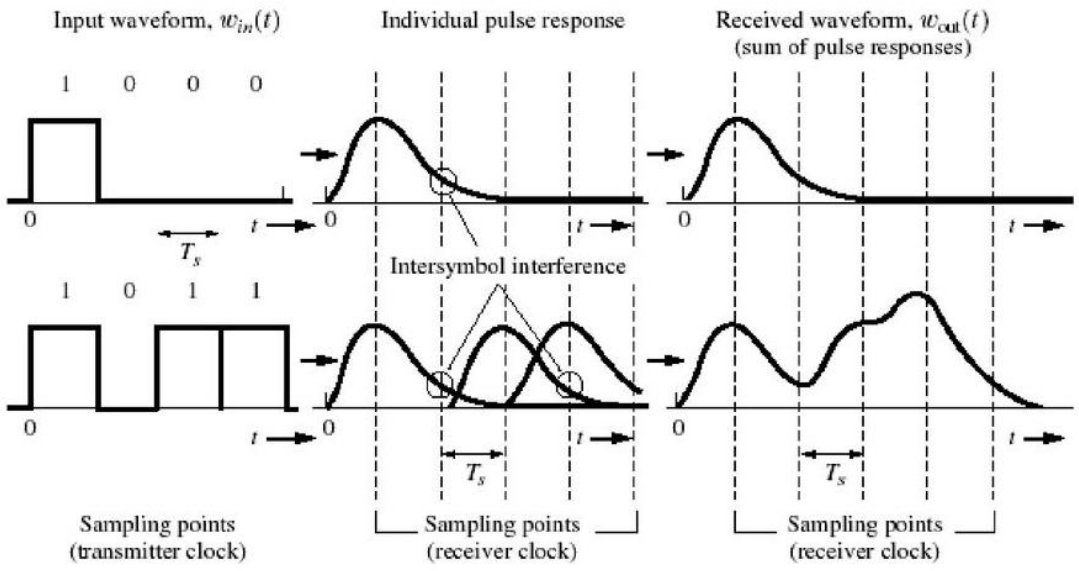
**Multipath-interference** is a **particular case** within the more **general concept** of *Beam interference*.

### 12.2.4 Inter-symbol Interference

**Inter-symbol interference** is a form of distortion of a signal in which **each** symbol interferes with its **previous** and its **subsequent** symbols.

**Inter-symbol interference** may be caused by *Spreading*, variations of *Attenuation* with frequency (in particular in a band-limited channel) and/or *Multi-path Interference*.

The following figure shows a simple example of **inter-symbol interference**, depicting the differences between the transmitted and received signals, and how the symbols interfere with each other:



Notice that **inter-symbol interference** is a concept coming from **telecommunications** and **not** so much from the **physics** of electromagnetic propagation. **Inter-symbol interference** is directly related to the problems we may encounter when trying to identify the correct symbols from the received signal.

Notice that **inter-symbol interference**, **in spite** of its **name**, is a substantially **different phenomenon** from *Beam interference*.

### 12.2.5 Free-space path-loss

**Free-space path-loss** (**FSPL**) is the **power loss** experienced by the **radiated energy** along a **path** in **free-space** and in **vacuum** (i.e., as if it were in **outer space**) in the **direction** "$v$" from a **transmitting antenna** to a **receiving antenna** (i.e., **point-to-point** with **line-of-sight**).

**Free-space path-loss** is **jointly** caused by *Spreading* and by *Effective area*.

**Free-space path-loss** is usually expressed as the **quotient** "$P_r/P_t$" between the **received power** "$P_r$" and the **transmitted power** "$P_t$". Since it is a **ratio**, it is **dimensionless**. It usually expressed in **decibels** (**dBs**), but it is sometimes expressed as a **real number lesser than 1** (i.e., the result of the quotient).

**Free-space path-loss** is the main factor for the **distance limitation** of a radio system in **vacuum** and at **low frequencies** (below 1 GHz), where **atmospheric** *Attenuation* is negligible.

The **free-space path-loss** may be computed with the **free-space path-loss** formula (12.2.5.1). In addition to this formula, there are some relevant variants:

- **Fixed-surface free-space path-loss** formula (click 12.2.5.2).
- **Beamwidth free-space path-loss** formulas (click 12.2.5.3).
- **Path-loss** formulas (click 12.2.5.4).

#### 12.2.5.1 Free-space path-loss formula

The **free-space path-loss** (**FSPL**) formula is derived from the original **Friis** formula. The **FPSL** formula states that the ratio between the **received power** "$P_r$" and the **transmitted power** "$P_t$" is:

$$P_r / P_t = G_t^v \cdot G_r^v \cdot (\lambda/4\pi d)^2$$

$$P_r = [(P_t/4\pi d^2) \cdot G_t^v] \cdot [(\lambda^2/4\pi) \cdot G_r^v]$$

where:

- "$P_t$" and "$P_r$" are (respectively) the **transmitted power** at the **antenna element** of the transmitter and the **received power** in the **antenna element** of the receiver.
- "$G_t^v$" and "$G_r^v$" are (respectively) the **antenna gain** (click 11.21.2) of the **transmitter** and the **antenna gain** of the **receiver**, in respect to a **single isotropic antenna** (click 11.21.3) and in the **direction** "$v$" (i.e., **point-to-point communication**) between **both antennas**.
- "$\lambda$" is the signal **wavelength**.
- "$d$" is the **distance** between **both antennas**.

The **FSPL** formula is **only valid** if **all** the following conditions hold:

- Propagation is in **vacuum**.
- Distance "$d$" is **much larger** than the **wavelength** "$\lambda$" (i.e., **far field** condition).
- "$(4\pi d/\lambda)^2 > G_t^v \cdot G_r^v$", which implies that "$P_r < P_t$", because **higher gain** is **less losses**, and **not power amplification**.
- The **antennas** are correctly **aligned** and have the **same polarization**.
- The **antennas** are in unobstructed **free space**, with **no multipath** propagation.
- The **bandwidth** is **narrow enough** that a **single value** "$\lambda$" for the **wavelength** can be used to represent the whole transmission.
- The **surface** "$sf_t^v$" and "$sf_r^v$" of **each** of the **two antennas** in the **direction** "$v$" between **both antennas** is in the **order** of **magnitude** of the **square** of the **wavelength** "$\lambda^2$". This is why the **antenna gains** appear in the formula as if they were **constant** (click 11.21.2).

Some **relevant remarks** on the **FSPL formula** are:

- The **first part** "$(P_t/4\pi d^2) \cdot G_t^v$" of the **formula** is the **power density flux** that is **radiated** by the **antenna** of the **transmitter** in the **direction** "$v$" to the **receiver**.
  - The **total radiated power** "$P_t$" **divided** by the **surface** of a **sphere** "$4\pi d^2$" is the **power density flux** of an **isotropic antenna** (click 11.21.3). This is the consequence of *Spreading*.
  - The **multiplication** by the **antenna gain** of the **transmitter** "$G_t^v$" is the adjustment because of the **directionality** of the **transmitter antenna** in the **direction** "$v$" to the **receiver antenna**.
- The **second part** "$(\lambda^2/4\pi) \cdot G_r^v$" of the **formula** is the **conversion factor** of the **incoming power density flux** (i.e., the **first part** of the formula) into **output power** that is performed by the **antenna** of the **receiver** in the **direction** "$v$" from the **transmitter**.
  - "$(\lambda^2/4\pi)$" is the **conversion factor** of the **incoming power density flux** (i.e., the **first part** of the formula) into **output power** that is performed by an **isotropic antenna**. Recall that "$\lambda^2/4\pi$" is the **effective area** (click 12.1.6) of an **isotropic antenna**.
  - The **multiplication** by the **antenna gain** of the **receiver** "$G_r^v$" is the adjustment because of the **directionality** of the **receiver antenna** in the **direction** "$v$" to the **transmitter antenna**.
- The **inverse proportion** to "$d^2$" arises from *Spreading*
- The **direct proportion** to "$\lambda^2$" arises **jointly** from the *Effective area* and from the **last condition** above on the **surface** "$sf$" of **each** of the **two antennas**. However, if we **replace** this **condition** (see right below) by **fixed antenna surface**, this **proportion reverses**, and the **received power** "$P_r$" becomes **inversely proportional** to "$\lambda^2$".
- For **any two antennas** "$a$" and "$b$", the **free-space path-loss** is the **same** when "$a$" **transmits** to "$b$" and when "$b$" **transmits** to "$a$". The reason is that the **free-space path-loss** formula is **symmetric** and that **every antenna** (because of

**reciprocity**, click 12.2.5) has the **same transmission** and **reception radiation pattern** (click 11.21.5).This last statement implies that for **every antenna** "$x$" and for **every direction** "$v$", its **transmission gain** "$G_t^{xv}$" and its **reception gain** "$G_r^{xv}$" is the same "$G_t^{xv}=G_r^{xv}=G^{xv}$". Consequently, the **free-space path loss** formula can be expressed in a **symmetric way** as:

$$P^b{}_r / P^a{}_t = G^{av} \bullet G^{bv} \bullet (\lambda/4\pi d)^2 = G^{bv} \bullet G^{av} \bullet (\lambda/4\pi d)^2 = P^a{}_r / P^b{}_t$$

The **PSPL** formula can also be expressed in **logarithmic** scale, where its units are **decibels**, as:

$$P_r / P_t = G_t^v + G_r^v - 20 \bullet log_{10}(\lambda) - 20 \bullet log_{10}(d) - 20 \bullet log_{10}(4\pi)$$

where "$G_t^v$" and "$G_r^v$" are expressed in **decibels**.

### 12.2.5.2 Fixed-surface free-space path-loss formula

When **replacing** in the **FSPL formula** (click 12.2.5.1) the following condition on the **antenna surface** "$sf^v$":

- The **surface** "$sf_t^v$" and "$sf_r^v$" of **each** of the **two antennas** in the **direction** "$v$" between **both antennas** is in the **order** of **magnitude** of the **square** of the **wavelength** "$\lambda^2$".

by this other condition:

- The **area** "$sf_t^v$" and "$sf_r^v$" of **each** of the **two antennas** in the **direction** "$v$" between **both antennas** is **constant** "$Sf_t^v$" and "$Sf_r^v$", and **larger** than the **square** of the **wavelength**: "$Sf_t^v > \lambda^2$" and "$Sf_r^v > \lambda^2$".

the result is that the **received power** "$P_r$" in a **point-to-point communication** in the **direction** "$v$" is **inversely proportional** to the **square** of the **wavelength** "$\lambda^2$" instead of **directly proportional**.

The reason is that for the **antenna gain** "$G^v(sf^v/\lambda^2)$" is approximately **proportional** to "$sf^v/\lambda^2$", because a **larger surface** "$sf^v$" can achieve a **larger gain** in that **same direction**:

$$G^v(sf^v/\lambda^2) \approx K_G \bullet sf^v/\lambda^2$$

Consequently, the **free-space path-loss formula** for the case of **Tx** and **Rx passive lossless antennas** of **fixed surface** "$Sf_t^v$" and "$Sf_r^v$", can be expressed as:

$$P_r / P_t \approx (K_G \bullet Sf_t^v/\lambda^2) \bullet (K_G \bullet Sf_r^v/\lambda^2) \bullet (\lambda/4\pi d)^2$$

$$P_r / P_t \approx K_G^2 \bullet Sf_t^v \bullet Sf_r^v \bullet (4\pi d/\lambda)^2$$

where "$Sf_t^v$" and "$Sf_r^v$" are (respectively) the **fixed surface** of the **Tx antenna** and of the **Rx antenna** in the **direction** "$v$" between **both antennas**.

The formula can also be expressed in terms of **frequency** "$f$" as:

$$P_r / P_t = K_G^2 \bullet Sf_t^v \bullet Sf_r^v \bullet (f / 4\pi d)^2$$

The formula can also be expressed to determine the **distance** "$d$" as:

$$d = f \bullet K_G \bullet (Sf_t^v \bullet Sf_r^v)^{1/2} \bullet (P_r/P_t)^{-1/2} / 4\pi$$

which indicates that the **distance** is **directly proportional** to **frequency** and **inversely proportional** to the **square root** of the **target power ratio** "$P_r/P_t$". If the **frequency** is **doubled**, the **signal reach** is **doubled**. If the **target power ratio** is **divided** by **4,** the **signal reach** is **multiplied** by **2**.

Recall that this formula is for in **point-to-point communication** in **free-space** and in **vacuum**. In **atmospheric transmission** on the **earth surface**, **high frequencies** have **smaller reach** (see **cell range**, click 3.10.1.1) because of their **much higher atmospheric attenuation** and other negative effects.

Since this formula is **derived** from the **FSPL formula**, it is subject to its **same conditions** (click 12.2.5.1), the **except** the **last one**, that we replaced at the beginning of this section.

### 12.2.5.3 Beamwidth free-space path-loss formulas

The **FPSL** formula and the **fixed-surface FPSL** formula are very useful for **point-to-point** communications but are **not directly applicable** to **point-to-multipoint** or **multipoint-to-point** communications.

This case **point-to-multipoint** or **multipoint-to-point** can be modeled by a **beamwidth antenna** with a **fixed** gain "$G^{\omega}$" in **every direction** "$\nu$" within a certain **beamwidth** "$\omega$". This **antenna** is the "**point**" in a **point-to-multipoint** (the **beamwidth antenna transmits**) communication or in a **multipoint-to-point** (the **beamwidth antenna receives**) communication. The **other antennas** are the "**multipoint**".

Regarding the **beamwidth antenna** (i.e., the "**point**") we analyze the following two **beam shapes**:

- A **spherical wedge** (i.e., an "orange segment") of "$\omega=\varphi$" radians.
- A **spherical sector** of "$\omega=\Omega$" steradians.

Regarding the **other antennas** (i.e., the "**multipoint**") we analyze the following two cases:

- The **antenna** can be **oriented**: a **directional antenna** (i.e., **gain > 1**) is used.
- The **antenna cannot** be **oriented**: an **omnidirectional antenna** (i.e., **gain = 1**) is used.

The following table contains the **four beamwidth PSPL formulas** corresponding to the **pairwise combination** of the previous cases:

| | **Beamwidth FPSL** $P^{mp}{}_r/P^{p}{}_t = P^{p}{}_r/P^{mp}{}_t$ | |
|---|---|---|
| "**multipoint**" **antenna** | **Spherical wedge** of "$\varphi$" **radians** | **Spherical sector** of "$\Omega$" **steradians** |
| **Directional** | $G^{mp,\nu} \cdot \lambda^2/8\pi\varphi d^2$ | $G^{mp,\nu} \cdot \lambda^2/4\pi\Omega d^2$ |
| **Omnidirectional** | $\lambda^2/8\pi\varphi d^2$ | $\lambda^2/4\pi\Omega d^2$ |

An example of a **spherical wedge** "**point**" **antenna** with an **omnidirectional** "**multipoint**" **antenna** is **mobile land communications**. The **base station** uses a **sector antenna** with a **radiation pattern** that is an (approximate) **spherical wedge** of "$\varphi$" radians with a constant gain "$G^{BS,\varphi}$". The **base station** a certain **sector** around the **antenna site**. **Each UE** within the **sector** uses an (approximate) **omnidirectional antenna**.

Since **these formulas** are **derived** (see below) from the **FSPL formula**, the are subject to its **same conditions** (click 12.2.5.1).

An example of the **spherical sector** with an **omnidirectional** "**multipoint**" **antenna** is **VSAT communications**. The **satellite** uses an **antenna** with a **radiation pattern** that is an (approximate) **spherical sector** of "$\Omega$" steradians with a constant gain "$G^{Sat,\Omega}$". The

**satellite illuminates** a certain **spot** on the earth's surface. **Each VSAT terminal** within the **spot** uses **a very directional antenna** (a satellite dish) with **very high gain** "$G^{VSAT,v}$" in the **direction** "***v***" towards the **satellite**.

These two examples, and many others, use an **antenna** with an **approximately constant gain** in **every direction** "***v***" within a certain **beamwidth**.

**Spherical wedge FSPL formula**

For a **spherical wedge** of "φ" radians, **power conservation** (click 11.21.1.1) determines that the **highest gain** "$G^{\varphi}$" that can be achieved in **every direction** "***v***" within such a **spherical wedge** is:

$$G_t^{p,\varphi} = 2\pi/\varphi$$

Consequently, the **beamwidth FSPL formula** can be expressed as:

$$\begin{aligned} P^{mp}{}_r / P^{p}{}_t = P^{p}{}_r / P^{mp}{}_t &= G^{p,\varphi} \bullet G^{mp,v} \bullet (\lambda/4\pi d)^2 \\ &= 2\pi/\varphi \bullet G^{mp,v} \bullet (\lambda/4\pi d)^2 \\ &= G^{mp,v} \bullet \lambda^2/8\pi\varphi d^2 \end{aligned}$$

Where "***p***" identifies the **bitwidth antenna** (i.e., the "**point**" **antenna** in the "**point**-to-**multipoint**" or "**multipoint**-to-**point**") and "***mp***" identifies the "**multipoint**" **antenna**.

For the case where the "**multipoint**" **antenna** is **omnidirectional** its **gain** is **1**:

$$P^{mp}{}_r / P^{p}{}_t = P^{p}{}_r / P^{mp}{}_t = \lambda^2/8\pi\varphi d^2$$

**Spherical sector FSPL formula**

For a **spherical sector** of "Ω" steradians, **power conservation** (click 11.21.1.1) determines that the **highest gain** "$G^{\Omega}$" that can be achieved in **every direction** "***v***" within such a **spherical sector** is:

$$G_t^{p,\Omega} = 4\pi/\Omega$$

Consequently, the **beamwidth FSPL formula** can be expressed as:

$$\begin{aligned} P^{mp}{}_r / P^{p}{}_t = P^{p}{}_r / P^{mp}{}_t &= G^{p,\Omega} \bullet G^{mp,v} \bullet (\lambda/4\pi d)^2 \\ &= 4\pi/\Omega \bullet G^{mp,v} \bullet (\lambda/4\pi d)^2 \\ &= G^{mp,v} \bullet \lambda^2/4\pi\Omega d^2 \end{aligned}$$

Where "***p***" identifies the **bitwidth antenna** (i.e., the "**point**" **antenna** in the "**point**-to-**multipoint**" or "**multipoint**-to-**point**") and "***mp***" identifies the "**multipoint**" **antenna**.

For the case where the "**multipoint**" **antenna** is **omnidirectional** its **gain** is **1**:

$$P^{mp}{}_r / P^{p}{}_t = P^{p}{}_r / P^{mp}{}_t = \lambda^2/4\pi\Omega d^2$$

### 12.2.5.4 Path loss formulas

**Path loss** is the **total power loss** experimented by the **radiated energy** along its **path** from the **transmitting antenna** to the **receiving antenna**.

It can be computed using any of the previous **FSPL** formulas (click 12.2.5) and considering the **path** *Attenuation*. Its **units** are **dB/km**.

**Path loss** is caused by *Free-space path-loss* and by *Attenuation*.

### 12.2.6 Attenuation

**Attenuation** is the reduction in power of the electromagnetic beam, between transmitter and receiver, **caused** by the **propagation** through the **physical medium**. **Attenuation** is usually caused by a combination of *Absorption* and *Scattering*.

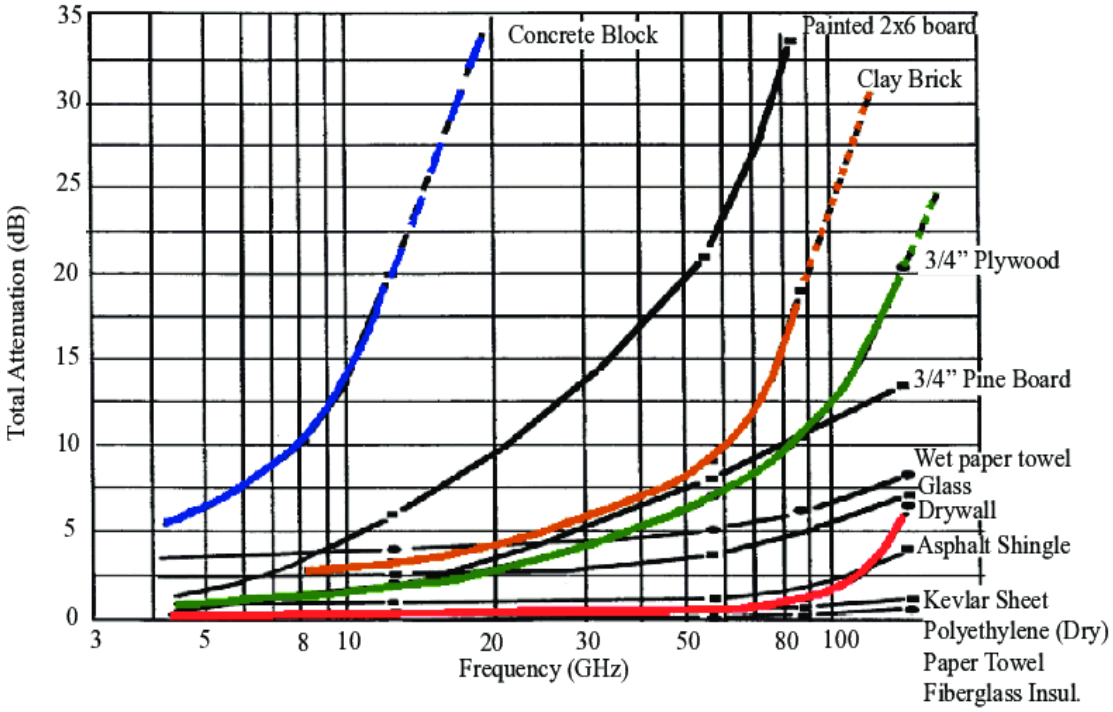
**Attenuation** is therefore caused by the physical propagation **medium**, and it does **not** happen in **vacuum**. Notice that **attenuation** happens even if you send a perfectly collimated beam, because it is a substantially different phenomenon than *Spreading*.

**Attenuation** is usually measured in dB/km. To provide a rough idea of attenuation values in telecommunications, an **optical fiber** has an attenuation around 1 to 2 dB/km.

For radio communications, **atmospheric attenuation** in **clean dry air** below 1 GHz is **so low** (below 0.01 dB/km) that is greatly dominated by *Spreading* and *Effective area*, which are the only factors considered in *Free-space path-loss* formulas for radio communications. However, **clean air atmospheric attenuation** above 1 GHz may be **extremely high** (e.g., up to 1,000 dB/km).

**Atmospheric attenuation** is mainly caused by **water** and **oxygen**, and is highly dependent on **frequency**, on **weather** (temperature, humidity, fog, rain, snow, ...) on **pollution** (particles and other gasses) and on **altitude**. For a more detailed analysis on **atmospheric attenuation**, click *Atmospheric Attenuation and Refraction*.

For mobile communications it is also very important to determine the attenuation of windows, walls, floors and other propagation obstacles for **indoor** communication. The following is an **attenuation** graph (in dB), for **high frequencies** (4 GHz to 150 GHz), presented in a semi-logarithmic scale. The different curves are for different types of indoors obstacles.




### 12.2.7 Fading

**Fading** is a **substantial temporal reduction** of the **received power** of the electromagnetic beam.

**Fading** is usually caused by **substantial temporal changes** in **attenuation** (rain, fog, temperature, moving objects, moving radio system, …), in *Refraction* (rain, fog, temperature, moving radio system, …) and/or in *Reflection* (moving objects, moving radio system, sea/lake conditions, …), affecting the electromagnetic beam.

Increased *Attenuation* obviously causes **fading**.

Changes in *Refraction* cause **fading** when the change in direction of the main electromagnetic beam **prevents** the beam from **illuminating** the receiver.

Changes in *Refraction* and *Reflection* cause **fading** when the different received beams experience **destructive wave interference**.

## 12.3 Differences between phenomena of received signal distortion

This section explains the **differences** between some **pairs** of **phenomena** of received-signal distortion.

**Comparison between *Absorption* and *Scattering***

- In **absorption** the **photons** are **absorbed** by the **physical medium** while in **scattering** they are **re-emitted** in a **different direction**, and possibly with **different characteristics**.
- In **absorption** the **photons** are converted into **heat** (increased medium **temperature**) while in **scattering** the **photons** are roughly **preserved**, but with a **substantial transformation**.
- **Both** are caused by the **physical medium**, **neither** happens in **vacuum**, **both** cause a **reduction** in **received power** and **neither** causes signal **distortion**.

**Comparison between *Absorption* and *Spreading***

- In **absorption** the **photons** are **absorbed** by the **physical medium** while in **spreading** the **photons** are **progressively spaced away** because their trajectories are **not** perfectly **parallel**.
- In **absorption** the **photons** are converted into **heat** (increased medium **temperature**) while in **spreading** the **photons** are **perfectly preserved** in their **direction** and characteristics.
- **Absorption** is caused by the propagation **medium**, while **spreading** is **not**.
- **Spreading happens** in **vacuum**, while **absorption** does **not**.
- **Both** cause a **reduction** in **received power** and **neither** causes signal **distortion**.

**Comparison between** *Spreading* **and** *Time dispersion*

- In **spreading** the photons are **spaced away** in **space**, over the **wave front**, while in **time dispersion** the **photons** are **spaced away** in **time** because of their **different propagation speeds**.
- **Time dispersion** is caused by the propagation **medium**, while **spreading** is **not**.
- **Spreading happens** in **vacuum**, while **time dispersion** does **not**.
- **Spreading** causes a **reduction** of **received power**, while **time dispersion** does **not**.
- **Time dispersion causes a time distortion** of the **signal**, while **spreading** does **not**.

**Comparison between** *Noise* **and beam int**

**Noise** is **random electromagnetic energy** arriving to the receive while **interference** is the reception of a **non-random** electromagnetic beam, other than the main transmitted beam (i.e., the intended received beam).

- The difference between **noise** and **interference** is not always a clear cut. As you may see, the key difference between them is whether the received energy is "**random**" or not. The term "**random**" should be interpreted here as being **reasonably non-deterministic** in respect to the main transmitted beam (i.e., the intended received beam). Thermal noise is clearly random. An electrical engine randomly producing sparks is reasonably random, because the spectral electromagnetic energy distribution of the spark is reasonably random. The signals sent by **two transmitters** of the **same type** (e.g., two WiFi access points) in **overlapping** bands are clearly **not random** to one another. However, if the signals are sent by **many** different same-type transmitters, their combination may be reasonably random, and may be considered more **noise** than **interference**.

**Comparison between** *Thermal Noise and Scattering Noise* **and** *External Noise*

- **Thermal/scattering noise** is caused by the propagation **medium itself**, while **external noise** is caused by **natural** (e.g., lighting, solar storms) or **man-made** (e.g., welding, electrical engines) **phenomena**.
- **Thermal/scattering noise** does **not** happen in **vacuum**, while **external noise does**.
- **Neither** causes a **reduction** in **received power** and **both** cause signal **distortion**.

## 12.4 Causes of distortion phenomena

The following table shows the **causes** of the different **individual** or **combined distortion phenomena**:

| | Causes | | | | |
|---|---|---|---|---|---|
| **Distortion phenomena** | **Phys. med.** | **Tx** | **Rx.** | **Sim. Tx** | **Phen.** |
| *Absorption* | X | | | | |
| *Attenuation* | X | X | | | |
| *Beam interference* | | X | | X | |
| *Diffraction* | X | | | | |
| *Doppler* | | 1 | 1 | | |
| *External interference* | | | | X | |
| *External Noise* | | | | | X |
| *Fading* | X | | | | |
| *Inter-symbol Interference* | X | | | | |
| *Internal Noise* | | X | X | | |
| *Multi-path Interference* | X | X | | | |
| *Noise* | X | X | X | | X |
| *Free-space path-loss* | X | X | X | | |
| *Radio frequency subsystem distortion* | | X | X | | |
| *Effective area* | | X | X | | |
| *Reflection* | X | | | | |
| *Refraction* | X | | | | |
| *Scattering* | X | | | | |
| *Self-interference* | | X | | | |
| *Spreading* | | X | | | |
| *Thermal Noise and Scattering Noise* | X, 2 | | | 2 | |
| *Time dispersion* | X | | | | |

Some **relevant comments** on this table are:

- "**Phys. Med**." stands for: caused by the physical propagation medium.
- "**Tx**" stands for: caused by the transmitter.
- "**Rx**" stands for: caused by the receiver.
- "**Sim. Tx**" stands for: caused by other similar transmitters.
- "**Phen**." stands for: caused by natural or man-made phenomena.
- Note 1: **Other similar transmitters** in combination with a **physical propagation medium** cause **scattering noise**, which is a type of *Thermal Noise and Scattering Noise*.
- *Diffraction* arises mainly from objects within the physical propagation medium, along the propagation path.
- *Effective area* contributes to *Free-space path-loss*, through the receiver antenna.
- *Refraction* and *Reflection* contribute to *Fading*, through *Multi-path Interference*.

## 12.5 Atmospheric Attenuation and Refraction

**Atmospheric** *Attenuation* is mainly caused by **atmospheric** *Absorption* and *Scattering*.

**Atmospheric** *Attenuation* **below 10 GHz** is **so low** (~.01 dB/km) that *Free-space path-loss* is greatly dominated by *Spreading*, which is the only factor considered in *Free-space path-loss* formulas for radio communications.

However, **atmospheric** *Attenuation* above **10 GHz** may be **extremely high** (e.g., up to 1,000 dB/km). It is mainly caused by **water** and **oxygen**, and is highly dependent on **frequency**, on **weather** (temperature, humidity, fog, rain, snow, ...) on **pollution** (particles and other gases) and on **altitude**.

**Water** in any state is an **obstacle** to an **electromagnetic beam**. When the beam passes through water particles, a part of its energy is **absorbed**, and a part is **scattered**. Therefore, the electromagnetic beam is **attenuated**.

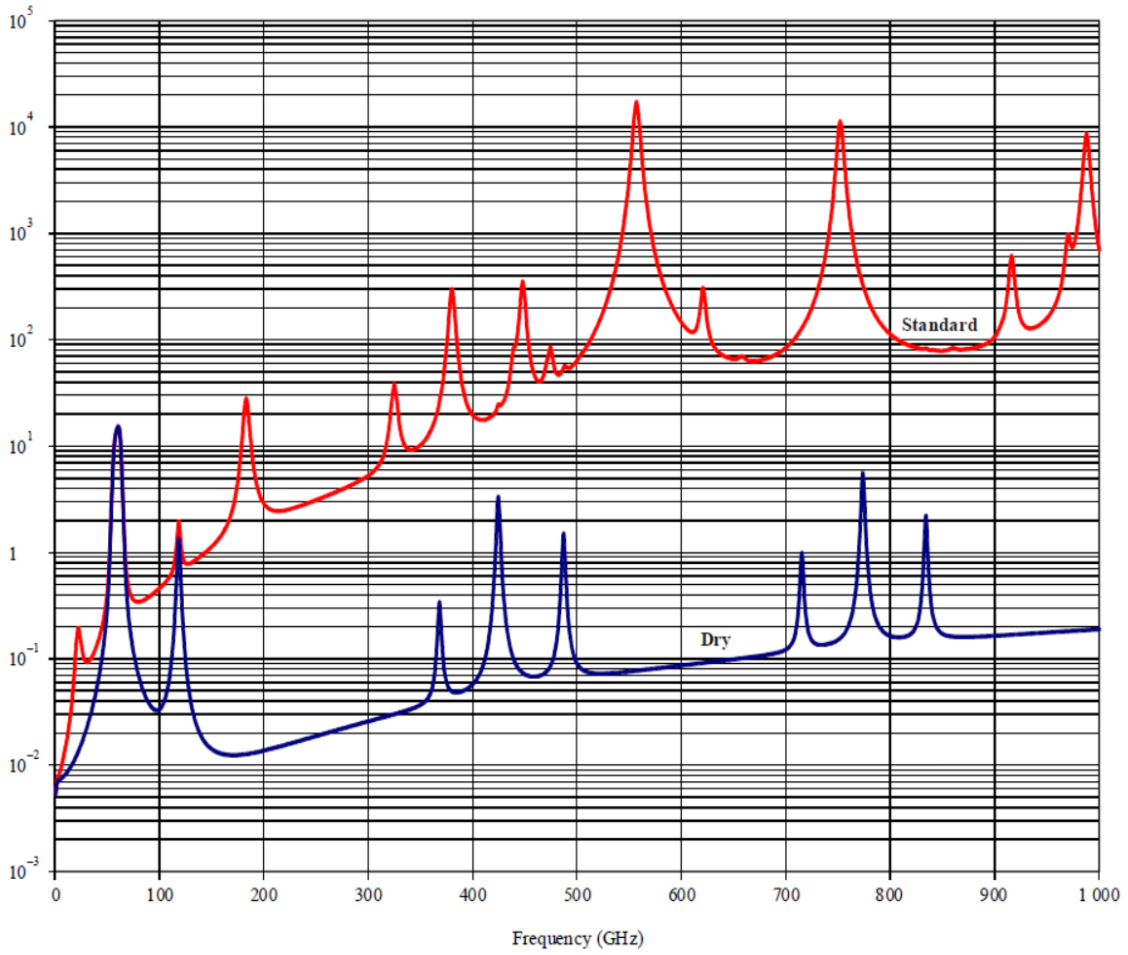
Another problem is the atmospheric *Refraction*, which causes the curvature of the electromagnetic beam leading to the *Fading* phenomenon.

Prediction on atmospheric *Attenuation* and *Refraction* is very important in radio system design. When there are no possibilities to gather data for calculations of the specific attenuation due to rain, clouds and fog, and atmospheric **refractive** index, the values recommended by the International Communication Union's Radiocommunication sector (ITU-R) can be used. But the recommended values are not always exact. In design of the radio links, the most desirable operating frequencies are below 10 GHz, because in such cases atmospheric *Absorption* and rainfall loss may generally be neglected (Freeman, 2007). However, in most countries, the frequency bands below 10 GHz are highly congested. In addition, high frequencies provide larger bandwidth, narrower beam width, good resolution and smaller component size (Bhattacharyya et al., 2000). Therefore, the operating frequencies of 10 GHz and above are increasingly used in design of radio systems. The higher the operating frequency, the greater *Attenuation* due to hydrometeors (rain, cloud, fog, snow, and etc.) is observed (Tamošiūnaitė et al., 2010a), although this is by no means a linear relationship.

In (Ishimaru, 1978), it was mentioned that the electromagnetic beam *Attenuation* due to snow is less than the *Attenuation* due to rain, and that the *Attenuation* due to **dry** snow may be neglected in microwave bands. However, the attenuation due to **wet** snow is higher. Some results of *Attenuation* due to hail are presented in (Ishimaru, 1978).

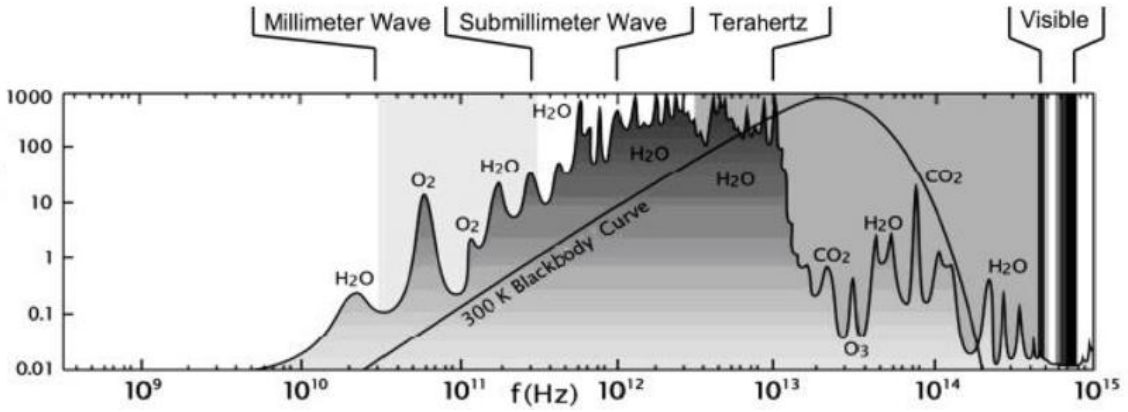
**Atmospheric** *Attenuation* changes **very significantly** as a function of **frequency**, in a **non-linear** way.

The following is an **atmospheric** *Attenuation* graph (in dB/km), for **high frequencies** (0 Hz to 1 THz), presented in a semi-logarithmic scale. The blue curve is for dry air, while the red one is for "standard" humidity (7.5 gr/m3 of water vapor).



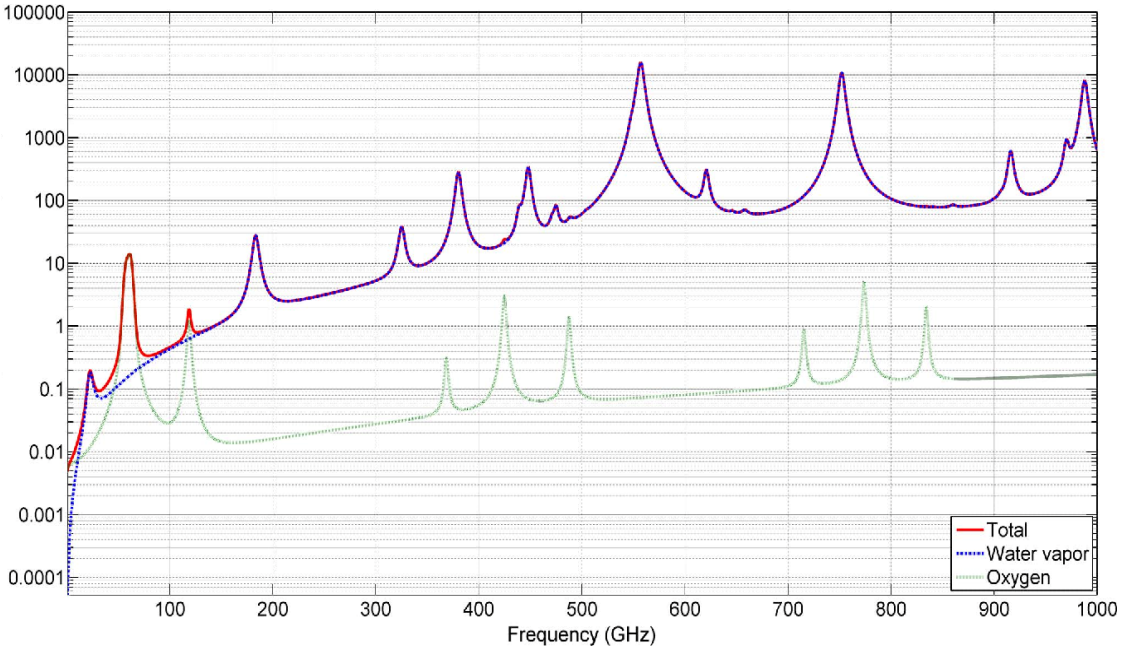

The source of the previous graph is Rec. ITU-R P.676-11.

The following is a clear air **atmospheric** *Attenuation* graph (in dB/km), up to **very high frequencies** (0.5 GHz to 1,000 THz), presented in a logarithmic scale. Recall that adverse weather conditions may significantly increase the *Attenuation*.



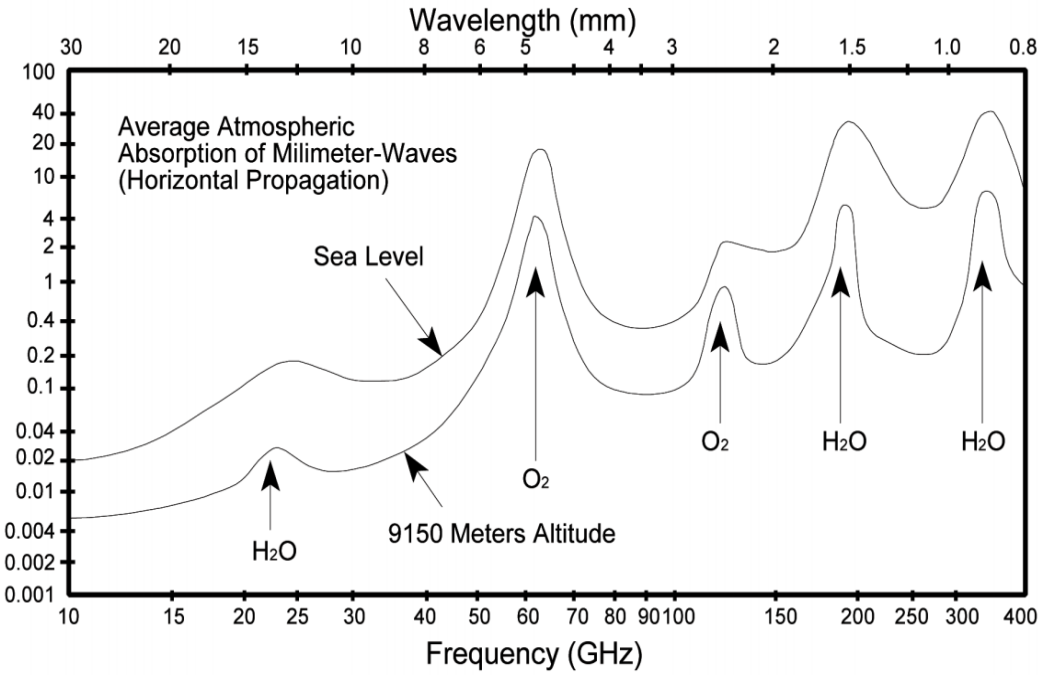

The following is a **water** and **oxygen** *Attenuation* graph (in dB/km), for **high frequencies** (0 Hz to 1 THz), presented in a semi-logarithmic scale. The **green** curve is the attenuation caused by **oxygen**, the **blue dotted** curve is the attenuation caused by **water vapor**, and

the **red** curve is the **addition** of the two previous ones. The **purple curve** is when the **blue dotted** curve and the **red** curve superimpose.




Notice that the **red** curve (addition of **green** and blue) is very similar to the **highest** curve (the **blue dotted** curve for most frequencies) because this is a logarithmic graph.

The following is an **altitude** atmospheric *Attenuation* graph (in dB/km), for **high frequencies** (10 GHz to 400 GHz), presented in a logarithmic scale. The **top** curve is the *Attenuation* at **sea level** while the **bottom** curve is the *Attenuation* at an altitude of 9,150 m (roughly the jetliner flight altitude).



# 13. 3GPP and Mobile Communications Standardization

## 13.1 Introduction to 3GPP and 3GPP standards

The main organization that standardizes Mobile Communications technologies is **3GPP**, that stands for "3rd Generation Partnership. **3GPP** is a global organization arising from **seven** regional and national Standards Development Organizations (**SDO**s): **ETSI** (Europe), **ARIB** and **TTC** (both from Japan), **ATIS** (United States), **CCSA** (China), **TTA** (Korea) and **TSDSI** (India). **3GPP** also includes very substantial industry participation.

Many individual **SDOs** (such as **ETSI**) endorse **3GPP** standards following their own legal procedures (possibly with some minor modifications) and assigning to them their own documents codes. You may therefore find **a given 3GPP** document being published as an official standard document by **several** "de jure" standard bodies, **each** standard body assigning to it its **own different document code**.

Periodically (roughly every year) **3GPP** publishes a "Release". A Release is a complete set of standard documents from **3GPP**, covering all the systems that **3GPP** addresses. A Release **may** include the standard documents for a **new system generation** and **always** includes an **update** of the standard documents for **all** the **preceding generations** (except when deprecated). Each **Release** (except the initial ones) is assigned a **sequential number**, and there is a certain correspondence between the Release numbers and the different "generations" of Mobile Communications technology (click 13.2).

Within each Release there are multiple documents, each of them specifying one element of the overall system or presenting a study on some system element. Each document can either be a "**Technical Specification**" (**TS**) or a "**Technical Report**" (**TR**). A Technical Specification can be considered a "**de facto**" **standard**, while a Technical Report provides some relevant technical information.

Each **3GPP** technical document is identified with a **document code** composed of a **specification number**, and a **version number**, structured as follows:

**TT gg.sss Vrr.vv.bb => TS 51.011 V4.11.1**

where "**TT**" is either "**TS**" (for "Technical Specification") or "**TR**" (for Technical Report"), "**ss.ggg**" is the **specification number**, "**V**" stands for "version" and "**rr.vv.bb**" is the **version number**. Within the **specification** number, "**ss**" is the **series number** (two digits) and "**ggg**" is the **document number** (three digits) within the series. Within the version number, "**rr**" is the **Release** number (one or two digits), "**vv**" is the document **version** number (one or two digits) within the Release, and "**bb**" is the **subversion** number (one or two digits), within the version. Some large documents are divided into **parts**, in which case the **part number** "**pp**" is added after the specification number (with an intermediate hyphen), a follows:

**TT gg.sss-pp Vrr.vv.bb => TS 38.101-1 V15.10.0**

Summarizing, the following document code:

**TS 38.101-1 V15.10.0**

implies that the document is a Technical Specification, series **38**, document number **101**, **part 1**, Release **15**, version **10**, subversion **0**.

The document code is usually preceded by "**3GPP**" and followed by the document date, indicated as "**(yyyy-mm)**", as follows:

**3GPP TS 38.101-1 V15.10.0 (2020-06)**

Technical **Reports** for **publishing** usually have document numbers "**xx.9xx**" while **internal 3GPP** Technical Reports usually have document numbers "**xx.8xx**", "**30.xxx**" or "**50.xxx**.

The **series number** identifies the document **technical area**. The following table lists the technical areas and series numbers as of jun-23:

| Technical area | 3G and beyond / GSM (R99 and later) | GSM only (Rel-4 and later) | GSM only (before Rel-4) |
|---|---|---|---|
| **Requirements** | 21 series | 41 series | 01 series |
| **Service aspects ("stage 1")** | 22 series | 42 series | 02 series |
| **Technical realization ("stage 2")** | 23 series | 43 series | 03 series |
| **Signaling protocols ("stage 3") - user equipment to network** | 24 series | 44 series | 04 series |
| **Radio aspects** | 25 series | 45 series | 05 series |
| **CODECs** | 26 series | 46 series | 06 series |
| **Data** | 27 series | 47 series (no documents) | 07 series |
| **Signaling protocols ("stage 3") - (RSS-CN) and OAM&P and Charging (overflow from 32.- range)** | 28 series | 48 series | 08 series |
| **Signaling protocols ("stage 3") - intra-fixed-network** | 29 series | 49 series | 09 series |
| **Program management** | 30 series | 50 series | 10 series |
| **Subscriber Identity Module (SIM / USIM), IC Cards. Test specs.** | 31 series | 51 series | 11 series |
| **OAM&P and Charging** | 32 series | 52 series | 12 series |
| **Access requirements and test specifications** | N/A | 13 series | 13 series |
| **Security aspects** | 33 series | Several series | Several series |
| **UE and (U)SIM test specifications** | 34 series | Several series | 11 series |
| **Security algorithms (3)** | 35 series | 55 series | (not public) |
| **LTE (Evolved UTRA), LTE-Advanced, LTE-Advanced Pro radio technology** | 36 series | N/A | N/A |
| **Multiple radio access technology aspects** | 37 series | N/A | N/A |
| **Radio technology beyond LTE** | 38 series | N/A | N/A |

When **downloading** a document from the **3GPP** server, its **file name** consists of the **specification number** (**without** the intermediate dot), followed by a **hyphen**, followed by the **version number** expressed as **three digits/letters**, where letters are used to denote numbers above **9** ("**a**" for **10**, "**b**" for **11** and so on). In case the **topic** contains **several parts**, where **each part** is placed on a **separate TS document**, the **part number** is indicated after the specification number with an **intermediate hyphen**. If you want the

complete file naming convention click 3GPP file naming conventions. See the following example of a file name corresponding to a given document code:

**Document code: 3GPP TS 38.101-1 V15.10.0 (2020-06)**

**File name: 38101-1-fa0.zip**

Almost all **3GPP** Technical Specifications are publicly available for free. You may download them, ordered by **series**, from:

www.3gpp.org/ftp/Specs/archive

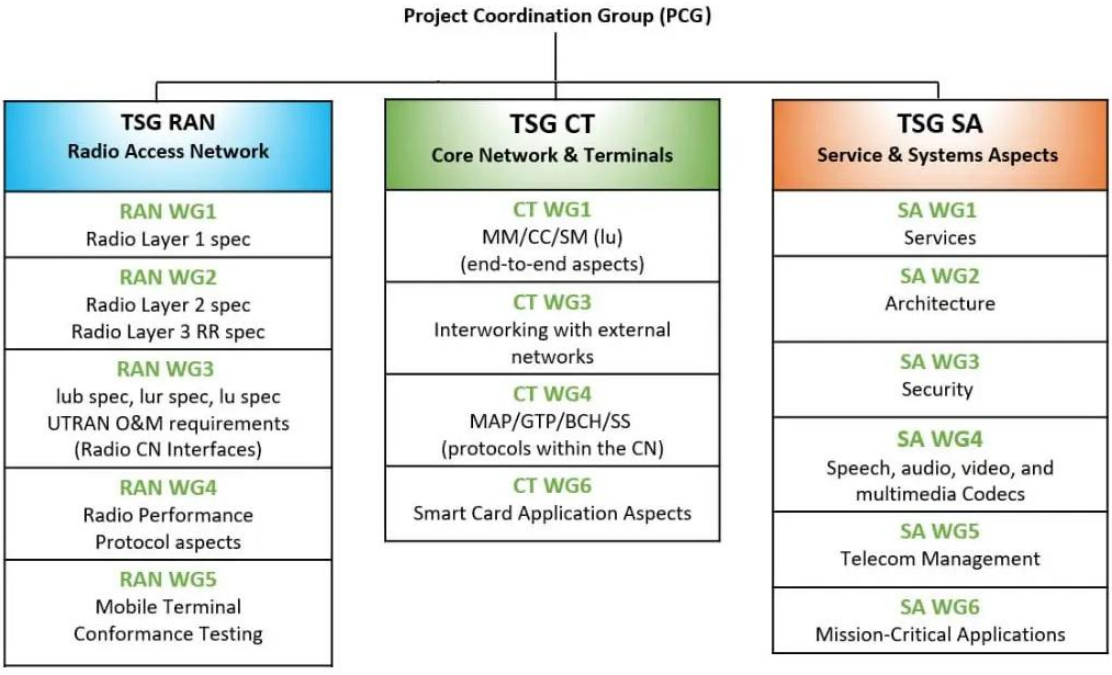
You can also search the **3GPP** document database at:

portal.3gpp.org/Specifications.aspx

The technical work of **3GPP** is structured in **three** broad **Technical Specification Groups** (TSG):

- Services & Systems Aspects (**SA**)
- Radio Access Networks (**RAN**)
- Core Network & Terminals (**CT**)

**Within each** of these **TSGs** there are **several Working Groups**, each addressing a more specific system aspect, as showin in the following figure:




**3GPP** permanently mantains an updated standardization roadmap. The following figure shows an example of a roadmap for **5G** and **6G**:



Source: https://www.nokia.com/about-us/newsroom/articles/5g-advanced-explained/

Source web pages:

- https://www.3gpp.org/about-us/introducing-3gpp
- https://www.3glteinfo.com/how-3gpp-specifications-numbering-works/

## 13.2 Evolution of Mobile Communications Generations

The following table provides an **overview** of some **characteristics** of the different **generations** of **Mobile Communications technology**:

| Tech. | Gen. | 3GPP Releases | Year of First Rel. | Terminal capacity | Maximum terminal BW | Latency (RTT) [ms] | Shown in Phone |
|---|---|---|---|---|---|---|---|
| **5G Advanced** (13.2.2.1) | 5.5G | 18, 19, 20 | 2023 | DL: 40 Gbps<br>UL: 9 Gbps | DL: 6 GHz<br>UL: 3 GHz | sub-ms | 5G+ |
| **5G** (13.2.2.2) | 5G | 15, 16, 17 | 2018 | DL: 15 Gbps<br>UL: 5 Gbps | DL: 3.2 GHz<br>UL: 1.6 GHz | 1 to 2 | 5G |
| **LTE Advanced Pro** (13.2.3.1) | 4.75G | 13, 14 | 2016 | DL: 1.5 Gbps<br>UL: 300 Mbps | DL: 100 MHz<br>UL: 60 MHz | 20 to 30 | 4G+ |
| **LTE Advanced** (13.2.3.2) | 4.5G | 10, 11, 12 | 2011 | DL: 1 Gbps<br>UL: 500 kbps | DL: 100 MHz<br>UL: 40 MHz | 25 to 40 | 4G+ |
| **LTE** (13.2.3.3) | 4G | 8, 9 | 2009 | DL: 296 Mbps<br>UL: 75.4 Mbps | DL: 20 MHz<br>UL: 20 MHz | 30 to 50 | 4G |
| **UMTS-HSPA+** (13.2.4.1) | 3.75G | 7 | 2007 | DL: 42 Mbps | DL: 5 MHz<br>UL: 5 MHz | 40 to 80 | H+ |
| **UMTS-HSPA** (13.2.4.2) | 3.5G | 5, 6 | 2002 | DL: 21 Mbps | DL: 5 MHz<br>UL: 5 MHz | 60 to 120 | H |
| **UMTS-UTRA** (13.2.4.3) | 3G | 99, 4 | 1999 | DL: 2 Mbps | DL: 5 MHz<br>UL: 5 MHz | 100 to 200 | U |
| **EDGE** (13.2.5.1) | 2.75G | 98 | 1999 | DL: 500 kbps | 200 kHz | 300 to 700 | E |
| **GPRS** (13.2.5.2) | 2.5G | 97 | 1998 | DL: 40 kbps | 200 kHz | 500 to 1,000 | G |
| **GSM** (13.2.5.3) | 2G | First ETSI. Later, 3GPP: Ph1-DCS, Ph1-EXT, Ph2 and 96 | 1987 | C: 9.6 kbps | 200 kHz | 200 to 500 | N/A |
| **Analog** (13.2.6) | 1G | N/A | 1982 | C: 2.4 kbps | | N/A | N/A |

### 13.2.1 Main expected characteristics of 6G

Its main **expected** characteristics are:

- Main 3GPP Releases: **21** (**expected 2027-28**), **22** and subsequent ones.
- Mobile phones are expected to show: "**6G**"

## 13.2.2 Main characteristics of 5G (5G and 5.5G)

### 13.2.2.1 Main characteristics of 5G Advanced (5.5G)

Its main characteristics for **Release 18** (the latest release as of the date of this book), are:

- Long name: **5G Advanced**
- Air Interface Technology: **DL CP-OFDM**, **UL SC-FDMA**.
- Main 3GPP Releases: **18** (**mar-24**), **19** (future) and **20** (future).
- Year of first Release: **March 2024**.
- Terminal capacity (CA): **DL 40 Gbps**, **UL 9 Mbps**.
- Latency (RTT): **R18 URLCC**: **1** to **2 ms**. **R19**: **sub-ms**.
- Maximum terminal bandwidth (CA): **DL 32 carriers** (up to **6.4 GHz**), **UL 16 carriers** (up to **1.6 GHz**).
- Channel/Carrier bandwidths:
  - Frequency Range 1: **5**, **10**, **15**, **20**, **25**, **30**, **35**, **40**, **50**, **60**, **70**, **80**, **90** and **100** MHz.
  - Frequency Range 2: **50**, **100**, **200**, **400**, **800**, **1,600** and **2,000** MHz.
  - **Unlicensed spectra** and **Satellite bands**.
- Mobile phones usually show: "**5G+**"
- Global system bandwidth: around **13 GHz**.
- Air Interface is called: "**New Radio**" or "**NR**".
- RAN nodes are called: "**Next generation Node B**", "**gNodeB**" or "**gNB**".
- Terminals are called: "**User Equipment**" or "**UE**".
- Core is called: "**Next Generation Core**", "**NGC**" or "**5G Core**".

### 13.2.2.2 Main characteristics of 5G

Its main characteristics are:

- Long name: **5G**
- Air Interface Technology: **DL CP-OFDM**, **UL SC-FDMA** or **CP-OFDM**
- Main 3GPP Releases: **15** (**Dec-18**), **16** (**Dec-19**) and **17** (**2022**).
- Year of first Release: **December 2018**.
- Terminal capacity (CA): **DL 15 Gbps**, **UL 5 Mbps**.
- Latency (RTT): **R15 eMBB**: **5** to **10 ms**; **R16 URLLC**: **1** to **2 ms**.
- Maximum terminal bandwidth (CA): **DL 16 carriers** (up to **3.2 GHz**), **UL 16 carriers** (up to **1.6 GHz**).
- Channel/Carrier bandwidths:
  - Frequency Range 1: **5**, **10**, **15**, **20**, **25**, **30**, **35**, **40**, **50**, **60**, **70**, **80**, **90** and **100 MHz**.

  - Frequency Range 2: **50**, **100**, **200**, **400**, **800**, **1,600** and **2,000 MHz**.
  - **Unlicensed spectra** in **R16**. **Satellite bands** in **R17**.
- Mobile phones usually show: "**5G**".
- Global system bandwidth: around **12 GHz**.
- Air Interface is called: "**New Radio**" or "**NR**".
- RAN nodes are called: "**Next generation Node B**", "**gNodeB**" or "**gNB**".
- Terminals are called: "**User Equipment**" or "**UE**".
- Core is called: "**Next Generation Core Network**", "**NGCN**", "**5G Core Network**" or "**5GCN**".
- Can be deployed in **Non-Standalone** mode, or in **Standalone** mode. Non-Standalone means it requires having **LTE coverage**, using on most cases **LTE signaling** and **LTE core**. **Standalone** is a **complete 5G network**.

For more detail on **maximum UE bandwidth**, refer to TS 38.101 and TS 38.306.

### 13.2.3 Main characteristics of LTE (4G)

#### 13.2.3.1 Main characteristics of LTE Advanced Pro (4.75G)

The **4G/LTE** systems are called **IMT-Advanced** by **ITU-R**.

Its main characteristics are:

- Long name: **Long Term Evolution Advanced Pro**

  Other names: **4.5G**, **4.75G**, **4.9G**, **4.5G Pro**, **Pre-5G**, **Project 5G**.
- Air Interface Technology: **DL CP-OFDM**, **UL SC-FDMA**.
- Main 3GPP Releases: **13** and **14**.
- Year of first Release: **2016**.
- Terminal capacity (CA): **DL 1.5 Gbps**, **UL 300 Mbps**.
- Latency (RTT): **20** to **30 ms**.
- Maximum terminal bandwidth (CA): **DL 5 carriers** (up to **100 MHz**), **UL 3 carriers** (up to **60 MHz**).
- Channel/Carrier bandwidths: **1.4**, **3**, **5**, **10** and **20 MHz**.
- Mobile phones usually show: "**4G+**"
- Global system bandwidth: around **4.7 GHz**
- Air Interface is called: "**Evolved Universal Terrestrial Radio Access**", or "**E-UTRA**".
- RAN nodes are called: "**evolved Node B**" or "**eNodeB**" or "**eNB**".
- Terminals are called: "**User Equipment**" or "**UE**".
- Core is called: "**Evolved Packet Core**" or "**EPC**".

Main advances over **LTE Advanced**:

- Increased battery life about 10 times more than LTE.
- It can use of both licensed (400 MHz to 3.8 GHz) and unlicensed (5GHz) spectrum.
- Increased carrier aggregation capacity, up to 32 carriers.
- Increased spectrum efficiency, by increasing the quantity of antenna paths as well as multi beam approach. It serves single radio cell with 16 to 64 antenna paths.

#### 13.2.3.2 Main characteristics of LTE Advanced (4.5G)

**4G/LTE** systems are called **IMT-Advanced** by **ITU-R**.

Its main characteristics are:

- Long name: **Long Term Evolution Advanced**. Also called **LTE+**.
- Air Interface Technology: **DL CP-OFDM**, **UL SC-FDMA**.
- Main 3GPP Releases: **10**, **11** and **12**.
- Year of first Release: **2011**.
- Terminal capacity (CA): **DL 1 Gbps**, **UL 500 Mbps**.
- Latency (RTT): **25** to **40 ms**.
- Maximum terminal bandwidth (CA): **DL 5 carriers** (**up to 100 MHz**), **UL 2 carriers** (**up to 40 MHz**)
- Channel/Carrier bandwidths: **1.4**, **3**, **5**, **10** and **20 MHz**.
- Mobile phones usually show: "**4G+**"
- Global system bandwidth: around **4.7 GHz**
- Air Interface is called: "**Evolved Universal Terrestrial Radio Access**", or "**E-UTRA**".
- RAN nodes are called: "**evolved Node B**" or "**eNodeB**" or "**eNB**".
- Terminals are called: "**User Equipment**" or "**UE**".
- Core is called: "**Evolved Packet Core**" or "**EPC**".

Main advances over **LTE**:

- Asymmetric bandwidth between **DL** and **UL** (unlike **LTE**, that is **symmetric**).
- Higher order **MIMO** techniques.
- Spectrum efficiency 3 times higher than **LTE**.
- Release 11 added **Coordinated Multipoint** (**CoMP**).
- Release 12 added **256QAM**.
- Release 12 added **Sidelink**, which is direct communication between **UEs**.

#### 13.2.3.3 Main characteristics of LTE (4G)

**4G/LTE** systems are called **IMT-Advanced** by **ITU-R**.

Its main characteristics are:

- Long name: **Long Term Evolution**.
- Air Interface Technology: **DL CP-OFDM**, **UL SC-FDMA**.
- Main 3GPP Releases: **8** and **9**.
- Year of first Release: **2009**.
- Terminal capacity: **DL 296 Mbps**, **UL 75.4 Mbps**.
- Latency (RTT): **30** to **50 m**s.
- Maximum terminal bandwidth: **DL 20 MHz**, **UL 20 MHz**.
- Channel/Carrier bandwidths: **1.4**, **3**, **5**, **10** and **20 MHz**.
- Mobile phones usually show: "**4G**"
- Global system bandwidth: around **4.7 GHz**.
- Air Interface is called: "**Evolved Universal Terrestrial Radio Access**" or "**E-UTRA**".
- RAN nodes are called: "**evolved Node B**" or "**eNodeB**" or "**eNB**".
- Terminals are called: "**User Equipment**" or "**UE**".
- Core is called: "**Evolved Packet Core**" or "**EPC**".

### 13.2.4 Main characteristics of UMTS (3G, 3.5G and 3.75G)

**3G/UMTS** systems are called **IMT 2000** by **ITU-R**.

#### 13.2.4.1 Main characteristics of UMTS-HSPA+ (3.75G)

Its main characteristics are:

- Long name: **Universal Mobile Telecommunication Service**.
- Air Interface Technology: **Code-Division Multiple Access** (**CDMA**).
- Main 3GPP Releases: **7**.
- Year of first Release: **2002**.
- Terminal capacity: **42 Mbps**.
- Latency (RTT): **40** to **80 m**s.
- Maximum terminal bandwidth: **5 MHz**
- Channel/Carrier bandwidth: **5 MHz**.
- Mobile phones usually show: "**H+**"
- Air Interface is called: "**UTRA-High Speed Packet Access +**", "**UTRA-HSPA+**", **UTRA-HSPA(Plus)**, **UTRA-HSPAP** or **UTRA-Evolved HSPA**.
- RAN nodes are called: "**Node B**" or "**NB**".
- Terminals are called: "**User Equipment**" or "**UE**".
- Core is called: "**Core Network**".

#### 13.2.4.2 Main characteristics of UMTS-HSPA (3.5G)

Its main characteristics are:

- Long name: **Universal Mobile Telecommunication Service**.
- Air Interface Technology: **Code-Division Multiple Access** (**CDMA**).
- Main 3GPP Releases: **5** and **6**.
- Year of first Release: **2002**.
- Terminal capacity: **21 Mbps**.
- Maximum terminal bandwidth: 5 MHz.
- Channel/Carrier bandwidth: **5 MHz**.
- Latency (RTT): **60** to **120 ms**.
- Mobile phones usually show: "**H**"
- Air Interface is called: "**UTRA-High Speed Packet Access**" or "**UTRA-HSPA**". It is an amalgamation of two systems: **High Speed Downlink Packet Access** (**HSDPA**) and **High Speed Uplink Packet Access** (**HSUPA**).
- RAN nodes are called: "**Node** B" or "**NB**".
- Terminals are called: "**User Equipment**" or "**UE**".
- Core is called: "**Core Network**".

#### 13.2.4.3 Main characteristics of UMTS-UTRA (3G)

Its main characteristics are:

- Long name: **Universal Mobile Telecommunication Service**.
- Air Interface Technology: **CDMA**.
- Main **3GPP** Releases: **99** and **4**.
- Year of first Release: **1999**.
- Terminal capacity: **2 Mbps**.
- Maximum terminal bandwidth: **5 MHz**.
- Channel/Carrier bandwidth: **5 MHz**.
- Latency (RTT): **100** to **200 ms**.
- Mobile phones usually show: "**U**"
- Air Interface is called: "**UMTS Terrestrial Radio Access**" or "**UTRA**".
- RAN nodes are called: "**Node B**" or "**NB**".
- Terminals are called: "**User Equipment**" or "**UE**".
- Core is called: "**Core Network**".

## 13.2.5 Main characteristics of GSM (2G, 2.5G and 2.75G)

### 13.2.5.1 Main characteristics of GSM-EDGE (2.75G)

Its main characteristics are:

- Long name: **Enhanced Data Rates for GSM Evolution** (**EDGE**)
- Main 3GPP Releases: **98**.
- Year of first Release: **1999**.
- Terminal capacity: **500 kbps**.
- Maximum terminal bandwidth: **200 kHz**.
- Latency (RTT): **300** to **700 ms**.
- Mobile phones usually show: "**E**"
- RAN nodes are called: "**Base Transceiver Stations**" or "**BTS**".
- Terminals are called: "**Mobile Station**" or "**MS**".
- It was an improvement of **GPRS** packet data of **GSM**.

### 13.2.5.2 Main characteristics of GSM-GPRS (2.5G)

Its main characteristics are:

- Long name: **General Packet Radio Service**
- Main 3GPP Releases: **97** and **98**.
- Year of first Release: **1998**.
- Terminal capacity: **40 kbps**.
- Maximum terminal bandwidth: **200 kHz**.
- Latency (RTT): **500 ms** to **1,000 ms**.
- Mobile phones usually show: "**G**"
- RAN nodes are called: "**Base Transceiver Stations**" or "**BTS**".
- Terminals are called: "**Mobile Station**" or "**MS**".
- It was an incorporation of packet data (not circuit switched) to **GSM**.

### 13.2.5.3 Main characteristics of GSM (2G)

Its main characteristics are:

- Long name: **Global System for Mobile Communications** (although its name was initially "**Groupe Spècial Mobile**").
- Initially standardized by: **ETSI**. Its standardization was later up taken by **3GPP**.
- Main 3GPP Releases: **Ph1-DCS**, **Ph1-EXT**, **Ph2**, **96**, **97** and **98**.
- Year of first Release: **1987**.
- Terminal capacity (circuit switched): **9.6 kbps**.
- Maximum terminal bandwidth: **200 kHz**.

- Latency (RTT): **200** to **500 m**s.
- Mobile phones usually show: **no indication**.
- Global system bandwidth: **900 MHz** in Europe in 1986.
- RAN nodes are called: "**Base Transceiver Stations**" or "**BTS**".
- Terminals are called: "**Mobile Stations**" or "**MS**".
- This was the **first digital mobile telephony technology**.
- This was the **first standard for global interoperation**.
- It incorporated the first built-in capacity to send data messages: Short Message Service (**SMS**).

### 13.2.6 Main characteristics of Analog Mobile Telephony (1G)

Its main characteristics are:

- Long name: **Analog Mobile Telephony**.
- Standardized by: **no global standard**. Some specific systems are Nordic Mobile Telephony (**NMT**), Advanced Mobile Phone System (**AMPS**), and Total Access Communication System (**TACS**). **TACS** was a European version of **AMPS**.
- Year of first Release: for **NMT**, **1981** (service provided by Telia in a few cities of Sweden and Norway); for **AMPS**, **1983** (service provided by AT&T in Chicago); for **TACS**, **1985** (service provided by **BT** in a few UK cities).
- **UE** capacity (circuit switched): **2.4 kbps**, with a **modem** over the radio channel.
- Maximum terminal bandwidth: fixed to **30 kHz** for **AMPS**, and to **25 kHz** for **NMT** and **TACS**.
- Latency (RTT): **not applicable**.
- Mobile phones usually show: **no indication**.
- Global system bandwidth: **not globally standardized**. For **AMPS** and **TACS** it was **25+25 MHz** (**Downlink** located in **869** to **894 MHz** and **Uplink** in **824** to **849 MHz**. For **NMT-450** it was **10+10 MHz** in the **450 MHz band**. For **NMT-900** it was **25+25 MHz** in the **900 MHz band**. For **NMT-900** it was **25+25 MHz** in the **900 MHz** band.
- Based on **analog** technology (not digital).
- Very few users of very high-end, with coverage only in a few large cities.

## 13.3 Narrowband standards

They are several standards for **Internet of Things** (**IoT**) or **Machine to Machine** (**M2M**) communication, within the framework of **5G** or of other technologies. Some relevant narrowband standards are (click for the corresponding wikipedia page):

- LPWAN
- 6LoWPAN
- Sigfox

- LoRa and LoRaWAN
- NB-Fi
- Weightless
- DASH7
- MulteFire

### 3GPP Narrowband Cellular Standards

| VTE [11][12] | LTE Cat 1 | LTE-M | | | | NB-IoT | | EC-GSM-IoT |
|---|---|---|---|---|---|---|---|---|
| | | LC-LTE/MTCe | eMTC | | | | | |
| | | LTE Cat 0 | LTE Cat M1 | LTE Cat M2 | non-BL | LTE Cat NB1 | LTE Cat NB2 | |
| **3GPP Release** | Release 8 | Release 12 | Release 13 | Release 14 | Release 14 | Release 13 | Release 14 | Release 13 |
| **Downlink Peak Rate** | 10 Mbit/s | 1 Mbit/s | 1 Mbit/s | | | 250 kbit/s | | 474 kbit/s (EDGE) |
| | | | | | | | | 2 Mbit/s (EGPRS2B) |
| **Uplink Peak Rate** | 5 Mbit/s | 1 Mbit/s | 1 Mbit/s | | | 250 kbit/s (multi-tone) | | 474 kbit/s (EDGE) |
| | | | | | | 20 kbit/s (single-tone) | | 2 Mbit/s (EGPRS2B) |
| **Latency** | 50–100ms | not deployed | 10ms–15ms | | | 1.6s–10s | | 700ms–2s |
| **Quantity of Antennas** | 2 | 1 | 1 | | | 1 | | 1–2 |
| **Duplex Mode** | Full Duplex | Full or Half | Full or Half | | | Half Duplex | | Half Duplex |
| **Device Receive Bandwidth** | 1.4 – 20 MHz | 1.4 – 20 MHz | 1.4 MHz | | | 180 kHz | | 200 kHz |
| **Receiver Chains** | 2 (MIMO) | 1 (SISO) | 1 (SISO) | | | 1 (SISO) | | 1–2 |
| **Device Transmit Power** | 23 dBm | 23 dBm | 20 / 23 dBm | | | 20 / 23 dBm | | 23 / 33 dBm |

### Narrowband IoT

**Narrowband IoT** (**NB-IoT**) is a Low Power Wide Area Network (**LPWAN**) radio technology standard developed by **3GPP** to enable a wide range of cellular devices and services. The specification was frozen in **3GPP** Release 13 (**LTE** Advanced Pro), in June 2016. Other **3GPP IoT** technologies include **eMTC** (**enhanced Machine-Type Communication**) and **EC-GSM-IoT**.

**NB-IoT** focuses specifically **on indoor coverage**, low cost, long battery life, and high connection density. **NB-IoT** uses a **subset** of the **LTE** standard, but limits the bandwidth to a **single narrow-band of 200kHz**. It uses **OFDM** modulation for **Downlink** communication and **SC-FDMA** for **Uplink** communications.

In March 2019, the **Global Mobile Suppliers Association** announced that over **100 operators** have deployed/launched **either NB-IoT** or **LTE-M** networks.

In **5G-R15**, both **Narrowband IoT** and **LTE-M** are supported exactly as in **LTE**. In **5G-R16**, they introduced **Industrial IoT** (**IIoT**), a type of **reserved access protocol** that guarantees transmission resources, eliminating reliability and latency problems.

# 14. Epilogue

## 14.1 Book contents

This book includes a reasonably **complete** and **self-contained** description of the **physical layer** of the **5G new radio** (**5G NR**), up to, and including, **Release 17**. The **5G NR** is the **air interface** of **5G** (i.e., the interface between the **user equipment** (**UE**) and the **NR node**, over the air).

The book addresses the following topics:

- The **5G NR frequency management**. This includes **frequency ranges**, **bands**, **band characteristics**, **duplexing mode**, **channels**, **common resource blocks**, **carriers**, **bandwidth parts**, **subcarriers**, etc.
- The **5G NR time management**. This includes **frames**, **half-frames**, **subframes**, **slots**, **symbol intervals**, **cyclic prefixes**, etc.
- The **5G NR space management**. This includes **beamforming**, **precoding**, **detecting**, **multiple-input multiple-output** (**MIMO**), **channel state information** (**CSI**), etc.
- The structure of **cellular networks**. This includes **cells**, **cell groups**, **sectors**, **antenna sites**, an overview of **deployment planning**, etc.
- The **internal structure** of the **NR node** in detail and an overview of **UE capabilities**. This includes the **functional splits**, the different **functional blocks**, the **HARQ protocol**, the **radio frequency chains**, etc.
- The **UE** configuration. This includes the configuration **parameters** of the **UE**, the **radio resource control** (**RRC**) protocol, an overview of **ASN.1**, etc.
- The **Downlink binary channels** and **signals**. This includes the **synchronization signal** and **PBCH block** (**SSB**), the **physical broadcast channel** (**PBCH**), the **physical Downlink control channel** (**PDCCH**), the **physical Downlink shared channel** (**PDSCH**), etc.
- The **Uplink binary channels** and **signals**. This includes the **physical random access channel** (**PRACH**), the **physical Uplink control channel** (**PUCCH**), the **physical Uplink shared channel** (**PUSCH**), etc.
- The **initial access** procedure. This includes locating the **SSB**, **cell search**, **cell selection**, **CORESET#0**, locating **Point A**, a **step-by-step example** for the complete procedure, etc.
- The **random access** procedure. This includes the random access **preamble formats**, the **4-step contention-based random access** (**CBRA**), the **2-step CBRA**, the **4-step contention-free random access** (**CFRA**), the **2-step CFRA**, the **fallback** variant and the different exchanged **messages**, etc.
- A description of several relevant **NR procedures**. This includes **bandwidth adaptation** (**BA**), **carrier aggregation** (**CA**), **dual connectivity** (**DC**), **supplementary uplink** (**SUL**), **dynamic spectrum sharing** (**DSS**), **multicast-broadcast services** (**MBS**), **handover**, **edge computing**, etc.
- A description of several relevant **NR underlying technologies**: **modulation** and **coding**, **multicarrier signals**, **duplexing**, **multiplexing**, **multiple access**,

**diversity**, **medium access control**, **discrete Fourier transform**, **pseudo-random sequences**, **passive lossless antennas**, and a few others.

- A description of several relevant **received signal impairments**. This includes **absorption**, **scattering**, **refraction**, **reflection**, **spreading**, **effective area**, **time dispersion**, **doppler**, etc.
- An overview of the **standardization process**, **3GPP** and the **different generations** of **mobile communications**.

For more detail on the **book contents**, you may browse through the table of contents.

The following **complementary materials** to this book are also available for download:

- File "**Parameters and data types from TS 38.331 v17_4.txt**"

  Complete listing of **all** the **parameters** and **all** the **data types** from TS 38.331 v17.4.0 in **ASN.1**, in **plain text**. This is **extremely useful** to perform **searches** for **parameters** and **data types much faster** than doing it over the word or pdf versions of TS 38.331.

- File "**5G NR Explained – Azcorra Frequencies.xlsx**"

  Several lists of **5G**, **LTE** and **other frequency bands**, all the **electromagnetic spectrum**, a summary of **mobile communication standards**, **UE capabilities**, **spectrum cost** and other data.

- File "**5G NR Explained – Azcorra Examples.xlsx**"

  Several **sheets** where **each** of them contains the **calculations** used for the different **examples** presented along the book.

- File "**5G NR Explained – Azcorra Calculations.xlsx**"

  Several **sheets** where **each** of them contains the **calculations** used along the book for: **time values**, **TDD**, **PUCCH formats**, **PUCCH OCC**, **PRACH timing**, **PRACH preamble frequency size**, among others.

This book covers **almost nothing** about **Sidelink**, **shared spectrum** operation, and **power control**.

The book and its complementary materials can be downloaded from:

- 5gnrexplained.net

## 14.2 Acknowledgements

**Utmost acknowledgments** to my **beloved family**: my wife **Marifeli**, my daughter **Maite**, my son **Jaime**, my son-in-law **Simone** and my granddaughter **Ainara**. They have suffered my non-stop typing along endless evenings, nights, weekends and vacations. My son **Jaime** deserves particular credit for his very useful technical suggestions (he is, as I am, a telecommunications engineer) and also for his very useful writing suggestions.

**Great recognition** to all my team members from the NETCOM Laboratory at University Carlos III (UC3M), all of them **Professor Doctors**: **Albert Banchs**, **Carlos Jesús Bernardos**, **Rubén Cuevas**, **Antonio de la Oliva**, **Alberto García-Martínez**, **Carmen Guerrero**, **Marco Gramaglia**, **Pablo Serrano**, **Francisco Valera** and **Iván Vidal**. I consulted with them different specific NR aspects and they always provided me with excellent insight on complex technical issues.

**Great recognition** to my IMDEA Networks **colleagues** that research in radio communications. On the one hand, the **Research professor Doctors Joerg Widmer**, **Domenico Giustiniano**, **Vincenzo Mancuso**, **Marco Ajmone-Marsan** and **Marco Fiore**, who provided me with excellent insight on complex technical issues. On the other hand, **Ignacio Berberana** and **Dr**. **Jesús O. Lacruz**. **Ignacio** provided me with an invaluable insight on the insides of a mobile operator (he was top radio expert at Telefonica for 30 years), always within his confidentiality duty. **Jesús** provided me with invaluable insight on the insides of NR processing, as he did implement (under the direction of **Joerg Widmer**) the functional **NR processing** with an **FPGA**.

**Great recognition** to my good friends and colleagues **Professor Doctors Ralf Steinmetz** (TU Darmstadt), **Ioannis Stavrakakis** (NKUA), **Jim Kurose** (U. Massachusetts at Amherst), **Gustavo de Veciana** (U. Texas at Austin), **Edward Knightly** (Rice University), **Fernando Boavida** (U. Coimbra), **Gonzalo Camarillo** (Ericsson) and **Heinrich Stüttgen** (NEC Eurolabs). They greatly inspired me, and I had the privilege of standing on their shoulders of giants.

**High recognition** to several **colleagues** in the 5TONIC 5G co-creation laboratory. From **Telefónica**, **Dr. Luis Contreras**, **Jesús Folgueira**, **Javier Gavilán** (currently at **Mavenir**) and **Dr. Diego López**. From **INTERDIGITAL**, **Dr. Alain Mourad**. From **Ericsson**, **Manuel Lorenzo**, **Marc Molla** and **Iván Rejón**. From **Andrew**, an **Amphenol** company (formerly **CommScope**), **Pedro Torres-Martos**. They provided me with invaluable insight on deep technical issues of **5G NR**, as well as practical issues, always within their confidentiality duty.

**High recognition** to several **colleagues** from the Ministry of Economy and Digitization. My boss, the Vice minister **Roberto Sánchez**. My chief of cabinet, **Juan Santaella**. My deputy directors, **Pedro Alonso** (operators), **Javier Amorós** (supervision), **Lorenzo Avello** (regulation), **Antonio Fernández-Paniagua** (spectrum decisions), **Cristina Morales** (mmedia and internet services), **Marta Serrano** (spectrum control). My division directors **María Lara** (budget) and **José María Moragón** (evaluation). They provided me with invaluable insight on regulatory and business aspects of 5G, always within their confidentiality duty.

**Patronage recognition** to the **European Commission**: their support to European research and development of top excellence and relevance, through the coordination of industry and academia, brings a spark of hope that Europe will be able to remain competitive in respect to the USA and China. **Mario Campolargo**, **Peter Stuckmann**, **Bernard Barani**, **Remy Bayou**, **Philippe Lefebvre**, **Pavlos Fournogerakis**, **Thibaut Kleiner** and **Francisco Guirao** (plus many others) all deserve **huge recognition**.

**Patronage recognition** to my employers, **University Carlos III of Madrid** (**UC3M**) and the **Regional Government of Madrid**: they directly funded my work for many years, which has allowed me to offer this book for free to the academic, scientific and industry communities. Within the Regional Government, particular acknowledgement to **Alfonso González Hermoso de Mendoza** (father of the IMDEA initiative), **Lucía Figar**, **Rocío Albert**, **Ana Isabel Cremades** and **María Luisa Castaño** for their **extraordinary** support to the **IMDEA initiative**.

**Special recognition**, plus my **thanks** and **acknowledgement**, to every public source of information and non-copyrighted figures that it was extremely interesting to read and inspire on. Some figures and text fragments of this book have been obtained, after

correcting and/or adapting them, from the TS documents, different referenced non-copyrighted web sites, from Wikipedia and from other public sources.

## 14.3 About the author

Prof. Dr. Arturo Azcorra graduated from Loy Norrix High School (Michigan, USA) in 1980. In 1986, he received his M.Sc. degree in Telecommunications Engineering from the Universidad Politécnica de Madrid (Madrid, Spain), with "Sobresaliente" (Outstanding) grade, and was subsequently awarded the Price Waterhouse Prize for Best Student in 1986. He then obtained his Ph.D. in Telecommunications Engineering from the same university in 1989, receiving the National Award COIT-AEIT-ANIEL (AMETIC) to the Best PhD Thesis (Premio Nacional a la Mejor Tesis Doctoral). In 1993, he obtained an MBA from the Instituto de Empresa, graduating first in his class.

He began his career in the Madrid Subway Company (Compañía del Metropolitano de Madrid), while simultaneously performing research work in telecommunications at UPM. He was an associate professor at UPM from 1989 to 1998, when he moved to University Carlos III of Madrid (UC3M) where he became Full Professor[1] and founder of the NETCOM Research Group on Networks and Communication Technologies, which he has coordinated since, at the Department of Telematics Engineering. He was later appointed Deputy Vice-Provost for Academic Infrastructures at the UC3M and held this post from 2000 to 2007. He was a Visiting Researcher at ICSI University of California at Berkeley (USA) in 1999 and at the Massachusetts Institute of Technology (MIT) (USA) in 2002. In 1998 he was appointed Director of the UC3M Telefónica Chair and worked in this role until 2009. In June 2002 he earned the distinction of being named IEEE (Institute of Electrical and Electronics Engineers) Senior Member. From 2003 to 2008 he was Director of REDIMadrid,[2] the Telematics Research Network of the Community of Madrid. He is the founder and director since 2006 (except for occasional periods in which he has held other public positions) up to 2024 of the international research institute IMDEA Networks.

In November 2009 he was appointed Director General for Technology Transfer and Corporate Development at the Spanish Ministry of Science and Innovation (MICINN)[3][4][5], a position he held until his appointment as Director General at the Centre for the Development of Industrial Technology (Centro para el Desarrollo Tecnológico Industrial - CDTI), the Spanish agency to fund research in private companies, on May 30, 2010.[6][7][8] In February 2012 he returned to his post as full professor at the Department of Telematics Engineering at University Carlos III of Madrid and Director of IMDEA Networks Institute.[9][10] In April 2021, he took leave from these posts, and incorporated to the Spanish Ministry of Economic Affairs and Digital Transformation as Director General of Telecommunications and Organization of Audiovisual Media Services[11] On October 6, 2022, he was promoted to Undersecretary of Telecommunications and Organization of Audiovisual Media Services. In July 2023, he regained his position as full professor in the Department of Telematics Engineering at Universidad Carlos III de Madrid and director of IMDEA Networks. On December 2024 he voluntarily stepped down from the direction of IMDEA Networks.

In October 2018 he was appointed member of the European Academy of Sciences "Academia Europaea". He received, in 2020, the prestigious **Reginald Fessenden Award**[12] for his contributions to the development of 5G technology. Granted by the ACM-MSWIM International Conference, the award ("for his pioneering

contribution to the midhaul and core of 5G networks") recognizes scientific contributions in the fields of wireless communications, networking and mobile systems.

In November 2021 he was elevated to the rank of **IEEE Fellow** in the "Technical Leader" category, for the scientific and industrial impact of his research contributions to the development of 5G technology. In this way, he becomes the first scientist in Spain to be designated IEEE Fellow in the "Technical Leader" category.

He has participated in or directed 63 advanced research projects, mainly in the Framework Programmes for Research and Technological Development of the European Union. He was the coordinator of the European Networks of Excellence E-NET, E-NEXT and CONTENT, and of the European projects CARMEN, TRILOGY, 5G-CROSSHAUL and 5G-TRANSFORMER. He has licensed technological developments in telecommunications for Ericsson-Telebit Denmark, NEC Europe and Huawei China. He has also performed advisory and technological consulting work for organizations such as Servicom, the European Space Agency, MFS-Worldcom, the Community of Madrid, RENFE, REPSOL, the Spanish Ministry of Education, Culture and Sports (Ministerio de Educación, Cultura y Deporte), Ericsson-Telebit and the Spanish Ministry of Science and Technology. He was a Member of the Advisory Board of the Future Internet PPP, European Commission, since February 2012 until December 2017. He was a member of General Assembly of the 5G Infrastructure Association from 2013 to 2021.[13] From 2013 to 2017 he was also a member of the Steering Board of the NetWorld2020 European Technology Platform (ETP) (formerly Net!Works & ISI ETPs), during which he was also the Chairman of its Expert Advisory Group.[14] Since 2014 he is partnership Board member of the European Union 5G-PPP,[15] and from 2015 to 2021 he was Vice-president of the 5TONIC Laboratory.[16] Arturo Azcorra is the founder of the ACM CoNEXT conference series, of which he was the first General Chair. He was a member of the Standing Committee of the IEEE INFOCOM Conference from 2005 to 2014, and has chaired prestigious international conferences such as IEEE INFOCOM, ACM CoNEXT and PROMS-IDMS. He has authored over one hundred and sixty technical publications in journals and international and national conferences.

## 14.4 Book principles

These are a few principles on writing technical books that I consider relevant and that have guided me while writing this book:

- **Electronic is in, paper is out!**

  Reading technical books in electronic format is full of advantages. Aside from the obvious environmental advantage of using less paper, in electronic format you can:

  - Follow the cross-links that populate the book.
  - Navigate with clickable, foldable table of contents (pdf bookmarks)
  - Do an electronic search for a term or sentence.
  - Write comments and notes, even substantially long, in a clean way and directly over the document. You can also later edit or remove these comments.

- **Science please**, **not literature!**

  Write with short clear sentences: subject, verb, predicate. Avoid long sentences with subordinate clauses, which usually cause confusion.

  Avoid pronouns as much as possible. A few repetitions of the same object name in a paragraph might be bad in a novel, but in a technical book it is very likely good for the reader to better recall the object name.

  Never mix pronouns referring to different nouns. In case you are referring many times to the same object in the same context: use only one pronoun when referring to the same name several times in the same context.

- **Right to the point!**

  Begin every section with a short explanation/definition of the section's topic, even if not completely accurate, and if possible, in one sentence.

  Next, indicate the main characteristics/attributes of the section's topic.

  Never start a section with preambles, justifications or contextualization: these longer and more detailed explanations should come later in the section.

- **Say something once**, **why say it again!**

  Avoid duplicate explanations: this is a source of incoherences. When explaining a concept, if you need a **subordinate explanation** of another concept, **do not copy** the explanation and rather insert a link to the place where the corresponding concept is properly explained. If absolutely necessary, you may insert a **short summary** of the explanation, in addition to the link.

- **Holistic is beautiful!**

  In spite of the previous principle, sections summarizing cross-concept issues are absolutely essential. When you explain concepts one by one, there is a high risk of missing the (sub)system view. It is therefore essential to have one section for each (sub)system, where the whole is explained and it is also essential to use links (see the next principle) to the places where the corresponding individual elements of the (sub)system are explained.

- **Links! links! links!**

  The explanation of most concepts requires the usage of different terms referring to other concepts. Each such term should have a link to the place where it is explained. Links are extremely important in a technical book! This is one of the reasons why it is much better to have a technical book in electronic form than on paper.

- **One and only one terminology!**

  Define properly, and coherently use, a single terminology throughout the book. When the field itself does not have a single globally accepted terminology (which is quite frequent), in the definition of each term that you do in the book, include a mapping to the different related terms usually accepted in the field.

- **An image is worth a thousand words!**

  Images are great to clarify concepts and help the reader recall complex questions. Use images as much as possible. Design them very carefully because an image is equally powerful to clarify a concept as to misguide about it.

- **An image alone is not enough!**

  Precisely because an **image** is worth a **thousand words**, on most cases, images are **not** self-explanatory. Images must therefore be followed by text that explains their meaning and implications. This allows the reader to get the full understanding of the image, and, to avoid possible misunderstandings over its meaning.

- **The matrix!**

  Describe **technical systems** with a **bidimensional matrix**, that has **both** a **structural dimension** and a **conceptual dimension**. If a system is only described in **1** of these **dimensions**, the reader has an incomplete view. Structure the book contents on sections following 1 dimension, and add sections written on the other dimension, where these later sections have links to each of the corresponding former sections.

- **Logic is logical!** (but **not obvious**)

  **Natural language** does **not** adapt well to express statements from **mathematical logic**. It is therefore essential to make sure that the usage of "**each**", "**every**", "**any**", "**each** and **every**", "**all**", "**both**", "**each of both**", "**and**", "**or**", "**and/or**", "**if**", "**if** and **only if**", "**may**" (which is the **equivalent** to "**may not**"), "**might**", "**can**", "**cannot**" and similar statements is the **correct** one. Use **parentheses** to **group** logical items and subexpression in every **sentence** that expresses a **complex logical statement**.

- **It's a colorful world!**

  Using **colors** to quickly identify an **object** both in **text** and **figures** is **extremely useful** to facilitate that the reader follows an explanation. Color is always useful, but it is particularly useful to identify **terms** in **equations**, **object types** in programming code (e.g., **ASN.1** code) and **object types** in figures.

- **What the hell is "XUR-TT"?**

  Whenever a given **acronym** is used in a **section**, write it **expanded** the **first time** that it is used, to save the reader the need to recall what it is. In **successive usages** within the **section**, you may use the **acronym** as such. Notice that in our technical field there are **LOTS** of acronyms.

- **Clickable**, **foldable**, **table of contents!**

  Technical books are (usually) consulted going to a specific chapter, section or subsection. Even when they are read linearly (like a novel), they are read linearly **once** and afterwards **consulted many times**. Having a clickable, foldable, table of contents, such as the one allowed for by pdf readers is a **must**. This is one of the reasons why it is much better to have a technical book in electronic form than on paper.

- **Conventional table of contents!**

  A clickable, foldable, table of contents (see the previous point) is a **must**, but it is also a **must** to have an actual table of contents in the book because the clickable/foldable one cannot be searched. I consider it better to insert it at the end of the book, to avoid beginning the book with a long table of contents, but other authors prefer to insert it at the beginning.

## 14.5 Copyright

This book is copyrighted and registered with **ISBN 978-84-09-78065-5**.

This book is free of charge for training, educational purposes, and research.

## 14.6 Bibliography


- **3GPP** Technical Specifications (standards). Downloadable from:

  www.3gpp.org/ftp/Specs/archive/

- ETSI standards of **3GPP** documents:

  https://www.etsi.org/deliver/etsi_ts/125200_125299/125212/

- Individual **UE** model information:

  www.globalcertificationforum.org/

- Overview of 5G characteristics, by release:

  5G Release 15, Qualcomm, December 2022

  https://www.qualcomm.com/content/dam/qcomm-martech/dm-assets/documents/5G-NR-Release-15.pdf

  5G Release 16, Qualcomm, July 2020

  https://www.qualcomm.com/content/dam/qcomm-martech/dm-assets/documents/propelling_5g_forward_-_a_closer_look_at_r16.pdf

  5G Release 17, Qualcomm, March 2022

  https://www.qualcomm.com/content/dam/qcomm-martech/dm-assets/documents/powerpoint_messaging_-_3gpp_release_17_completing_the_first_phase_of_5g_evolution.pdf

- Mobile communications technical explanations

  http://howltestuffworks.blogspot.com/

  https://devopedia.org/5g-new-radio

  https://sharetechnote.com/

  https://sharetechnote.com/html/5G/Handbook_5G_Index.html

  https://sharetechnote.com/html/6G/6G_WhatIsIt.html

  https://www.5gfundamental.com/

  https://www.nrexplained.com

  https://www.rfwireless-world.com/5G/

https://5g.smarttelecomedu.com/

https://www.electronics-notes.com/articles/connectivity/

- Excellent document on antennas:

  "*Recommendations for base station antennas*", NGMN, v13.0, July 2025. Downloadable from:

  https://www.ngmn.org/wp-content/uploads/NGMN_BASTA_Recommendations-for-Base-Station-Antennas_V13.0.pdf

- Mobile communications tools and technical explanations

  https://5g-tools.com/

- **3GPP** Glossary

  **3GPP** TR 21.905: "Vocabulary for **3GPP** Specifications".

- KPIs: Euro-5G - Supporting the European 5G Initiative (D2.6 Final report on programme progress and KPIs), Version: v0.2.

- 5G New Radio Commercial Deployments

  en.wikipedia.org/wiki/List_of_5G_NR_networks

- Detailed information on actually deployed base stations

  https://www.cellmapper.net/map

- "*5G NR: The Next Generation Wireless Access Technology*", second edition, Erik Dahlman, Stefan Parkvall, Johan Sköld, Elsevier Academic Press, 2021.

- "*Understanding the 5G NR Physical Layer*", Javier Campos, Keysight Technologies.

- "*Physical Cell ID Allocation in Cellular Networks*", Sofia Nyberg, Master thesis, Department of Computer Science, Linköping University, 2016.

- "*C-RAN Employing xRAN Functional Split: Complexity Analysis for 5G NR Remote Radio Unit*", Jay Kant Chaudhary, Atul Kumar, Jens Bartelt and Gerhard Fettweis, TU Dresden, Germany. May 2019.

- "*Configured Grant for Ultra-Reliable and Low-Latency Communications: Standardization and Beyond*", by M. Gerami and B. Singh, IEEE Communications Standards, Volume 6 , Issue, 4, December 2022.

- Glossary of Technical terms:

  www.mpirical.com/glossary

# 15. Acronyms, symbols and terms

When searching this list, notice that **symbols** (e.g., "(", "**μ**", "**π**", etc) may be ordered **first** or **last** in the list, without a clear criterion (MS-Word ordering).

(**N**)**AC**K: represents either a Negative Acknowledgement (NACK) and/or an Acknowledgement (ACK).

(**NG**)**EN-DC**: NG-RAN E-UTRA and New Radio Dual Connectivity. It is dual connectivity to one master ng-eNB and to one secondary gNB. It is specific of Non-Standalone deployment option 4. NG-RAN is Next-Generation Radio Access Network. E-UTRA is Evolved-Universal Terrestrial Radio Access, which is the LTE radio interface.

**μ**: Numerology. It is the combination of subcarrier spacing and cyclic prefix used. It is a natural number (starting on cero). The subcarrier spacing is $2^{\mu} \bullet 15$ kHz.

$10 \bullet log_{10}(P/1\text{mW}) = 30 + 10 \bullet log_{10}(P)$

**3GPP**: 3rd Generation Partnership Project. An industry association to standardize mobile technology.

**5GC**: 5G Core Network. The "core" or "core network" is the overall control system of a mobile communications network.

**5GCN**: 5G Core Network. The "core" or "core network" is the overall control system of a mobile communications network.

**5GS**: 5G System. It is the combination of the 5G RAN and the 5G Core.

**ACK**: Acknowledgement. It is a control frame sent to confirm the correct reception of a data frame.

**A-CSI**: Aperiodic channel state information.

**AI-RNTI**: Availability indication Radio Network Temporary Identifier.

**AMF**: Access and Mobility Management Function. The function (part of the 5G Core) that handles the UE access to the network as well as UE mobility across the network.

**ARFCN**: Absolute Radio-Frequency Channel Number. It is an integer number, that identifies, in GSM, the center frequency of both the Downlink and the Uplink channels, in each GSM channel pair.

**AS**: Access Stratum. access stratum. Protocol stack between the radio access network and the UE to transport data over the air interface.

**ASN.1**: Abstract Syntax Notation One. It is a standard interface description language for defining data structures that can be serialized and deserialized in a cross-platform way. Currently specified in the ITU X.680 series.

**BBU**: Base-band Unit. A processing unit (subsystem of a base station), that (de)modulates the received/transmitted carrier signals in base band. Each base band carrier signal is previously/afterwards (de)shifted in frequency (by the Radio Frequency subsystem) from/to its proper frequency location.

**BCCH**: Broadcast Control Channel. It is a Downlink logical binary channel. It is used for sending broadcast style information to the user equipment within that cell.

**BCH**: Broadcast Channel. It is a Downlink transport binary channel.

**BCS**: Bandwidth combination set. Asymmetric channel bandwidth combination sets are used in asymmetric channel configurations.

**BFR**: Beam Failure Recovery. A procedure to repair the air interface beam failure.

**BPSK**: Binary Phase-Shift Keying. A modulation in phase with a constellation having 2 symbols, implying 1 bit/symbol. Do not confuse with π/2-BPSK, that uses 2 constellations, shifted 90º, having 2 symbols each.

**BS**: Base Station.

**BSR**: Buffer Status Report. A type of Media Access Control (MAC) Control Element.

**BW**: Bandwidth. It is a range of frequencies. Normally used to define the range of frequencies of a band, channel or signal. It is sometimes used (not very correctly) to identify the throughput or bit rate of a digital channel.

**BWP**: Carrier bandwidth parts. Each of the (very) different configurations that a 5G carrier may have over time.

**CAG**: Closed Access Group.

**CAPC**: Channel Access Priority Class.

**CBG**: code block group.

**CBGTI**: code block group transmission information.

**CC**: Component Carrier. One of the carriers that is aggregated using the Carrier Aggregation technique.

**CCCH**: Common Control Channel. It is a Downlink and Uplink logical binary channel. It is used for control information between the UE and the network, while the UE is doing the initial access (and therefore does not have an RRC).

**CDM**: Code Division Multiplexing. It is used in different 5G Reference Signals.

**CDMG**: Code Division Multiplexing Group. It is used in different 5G Reference Signals.

**CD-SSB**: cell-defining synchronization signal and PBCH block (SSB). It is the SSB that provides CORESET#0, that allows the UE to receive the system information block 1, which contains the NCGI of the cell. It is always located in the initial bandwidth part.

**CE**: a Control Element (message or PDU) in the media access control sublayer.

**CFR**: Common Frequency Resource. The frequency resources used in multicast broadcast service.

**CG**: Configured grant.

**CG-DFI**: Configured grant Downlink Feedback Indication. It conveys the HARQ-ACKs for PUSCH HARQ.

**CG-SDT-CS-RNTI**: Configured grant Small Data Transmission Configured Scheduling Radio Network Temporary Identifier.

**CG-Type**: PUSCH configured grant type. There are CG-Type 1 and CG-Type 2 (click 7.11).

**CG-UCI**: Configured grant Uplink Control Information

**CI**: Cell Identity. A static identifier for each cell within a gNode B. It is from 4 to 14 bits long. Together with the gNode B Identitiy (gNB ID) constitutes the New Radio Cell Identity (NCI).

**CI-RNTI**: Cancellation Indication Radio Network Temporary Identifier.

**CLI**: Crosslink Interference. Interference between two, possibly distant, base stations transmitting in the same TDD channel.

**CMAS**: Commercial Mobile Alert Service. It is a type of public warning system widely used in United States and Canada.

**C-MTC**: Critical Machine-Type communication. It is sometimes considered as a 5G use case, with stringent requirements for latency, reliability, and high availability, very related to Ultra Reliable Low-Latency Communications (URLLC).

**Control/shared subcarrier spacing values**: the subcarrier spacing values that can be used in control physical channels (Physical Downlink Control Channel and Physical Uplink Control Channel) and in shared physical channels (Physical Downlink Shared Channel and Physical Uplink Shared Channel). These are 15, 30, 60, 120, 480 and 960 kHz.

**CORESET#0**: it is a shorthand for "CORESET for CSS-Type 0", this is, for "Control Resource Set for Common Search Space Type 0".

**CORESET**: Control Resource Set. They are a set of Resource Elements where the Physical Downlink Control Channel (PDCCH) Downlink Control Information (DCI) messages may be found.

**COT**: Channel Occupancy Time.

COW: Cell on Wheels. A truck that contains all the equipment

**CP**: Cyclic Prefix. A copy of the last part of a symbol that is sent before the symbol itself.

**CPAC**: conditional PSCell addition and change. It is a type of handover procedure.

**CP-OFDM**: Cyclic-Prefix Orthogonal Frequency Division Multiplexing.

**CQI**: Channel Quality Indicator.

**CRB**: Common Resource Block. It denotes the Resource Blocks of each Common Resource Block range.

**CRC**: Cyclic Redundancy Check. It is a type of error detecting channel code.

**CRI**: CSI-RS Resource Indicator, where "**CSI-RS**" stands for "Channel State Information - Reference Signal".

**C-RNTI**: Cell Radio Network Temporary Identifier.

**CRS**: Cell-Specific Reference Signal.

**CSC**: Common subcarrier. It denotes the subcarriers of each Common Resource Block range.

**CSI**: Channel-State Information.

**CSI-RS**: Channel-State Information Reference Signals.

**CSI-RSRP**: Channel State Information Reference Signal Received Power. It is the linear average over the power contributions (in Watt) of the Resource Elements that carry Channel State Information Reference Signal symbols for power measurement.

**CSI-RSRQ**: Channel State Information Reference Signal Received Quality.

**CS-RNTI**: Configured Scheduling Radio Network Temporary Identifier. It is a unique **UE** identification used for Semi-Persistent Scheduling (SPS) in Downlink and for configured grant in Uplink.

**CSS**: Common Search Space. It is a class of search space for CORESETs.

**CSS-Type**: common search space type. There are CSS-Type 0, 0A, 0B, 1, 1A, 2, 2A and 3 (click 6.8.5.5).

**CT**: Core Network and Terminals. One of the Technical Specification Groups of 3GPP.

**CWB**: Channel Bandwidth.

**DAPS**: Dual Active Protocol Stack. It is a handover technique where the link with the new base station is established before releasing the one with the former base station, in order to reduce the interruption time caused by handover.

**dB**: decibel. It is a logarithmic unit to express the value of unitless ratios that span a large range of values. It is typically used to measure ratios between 2 power values (e.g., the signal to noise ratio). The value in decibels of the ratio "$X_a/X_b$" is defined as "$10 \bullet log_{10}(X_a/X_b)$".

**dBm**: decibels over 1 milliwatt. It is a logarithmic unit of electrical power. It is typically used to measure radio power. The value in dBm of "$P$" watts "$X_a/X_b$" is defined as:

**DC subcarrier**: Direct Current subcarrier. It is the subcarrier placed in frequency 0 in the baseband representation of the **OFDM** signal.

**DC**: it has two completely different meanings that can be determined depending on context. Its two meanings are: a) Direct Current; b)) Dual Connectivity. The capacity of a UE to be simultaneously connected to two different base stations, that may have the same technology (e.g., two gNBs) or different technologies (e.g., one gNB and one eNB).

**DCCH**: Dedicated Control Channel. It is a Downlink and Uplink logical binary channel. It is used for control information between the UE and the network, once the UE has an RRC connection.

**DCI**: Downlink Control Information.

**DDM-Type**: PDSCH DM-RS configuration type. There are DDM-Type 1 and DDM-Type 2 (click 6.14.5.2).

**DFA-Type**: type of PDSCH frequency-domain resource allocation. There are 2 DFA types: DFA-Type 0 (click 6.14.3.2) and DFA-Type 1 (click 6.14.3.3).

**DFI**: Downlink Feedback Indication.

**DFT**: Discrete Fourier Transform.

**DFT-s-OFDMA**: Discrete Fourier transform spread orthogonal frequency division multiple access. It is the specific 5G implementation of a SC-FDMA technique.

**DL**: Downlink.

**DL-SCH**: Downlink Shared Channel. It is a Downlink transport binary channel.

**DM-RS**: Demodulation Reference Signal.

**DRB**: Data Radio Bearer. It is a radio bearers used for user data. DRBs provide unicast communication.

**DRX**: Discontinuous Reception.

**DSS**: Dynamic Spectrum Sharing. A technique that allows a channel to be used with a 5G carrier and an LTE carrier alternating over time.

**DTCH**: Dedicated Traffic Channel. It is a Downlink and Uplink logical binary channel.

**EARFCN**: Extended Absolute Radio-Frequency Channel Number. It is an integer number that identifies, in LTE, the center frequency of the next-above-middle subcarrier of a carrier.

**ECP**: Extended duration cyclic prefix. A cyclic prefix that implies 12 symbols in each slot.

**EIRP**: Equivalent Isotropic Radiated Power. It is the total radiated power from a transmitter antenna times the numerical directivity of the antenna in the direction of the receiver. It is also the power delivered to the antenna times the antenna numerical gain.

**eLTE**: evolved LTE. It is an LTE RAN whose nodes can connect to the 5G Core (5GC).

**eMBB**: enhanced Mobile Broadband. It is a 5G "use case" (i.e., a service class) targeted to mobile communications (smart phones, laptops, …).

**eMTC**: enhanced Machine Type Communications.

**eNB**: enhanced Node B. It is the term to denote the specific base stations of LTE. It is the same as an eNodeB.

**EN-DC**: E-UTRA and New Radio Dual Connectivity. It is dual connectivity to one master eNB and to one secondary en-gNB. It is specific of Non-Standalone deployment option 3. E-UTRA is Evolved-Universal Terrestrial Radio Access, which is the LTE radio interface.

**en-gNB**: EPC-connected gNB. It is a gNB connected to the LTE Evolved Packet Core (EPC) via the S1-U interface.

**eNodeB**: enhanced Node B. It is the term to denote the specific base stations of LTE. It is the same as an eNB.

**EPRE**: Energy Per Resource Element.

**ETWS**: Earthquake and Tsunami Warning System. It is a type of public warning system.

**FDD**: Frequency Division Duplexing.

**FDM**: Frequency Division Multiplexing.

**FDMA**: Code-Division Multiple Access.

**FDRA**: Frequency-Domain Resource Assignment. It is a field of DCI messages.

**FEC**: Forward Error Correction.

**FFT**: 1) Fast Fourier Transform: an algorithm that computes the Discrete Fourier Transform (DFT) of a sequence, or its inverse (iDFT). 2) Fast Fourier Transformer: a device that performs the Fast Fourier Transform.

**FR**: Frequency Range. A way to classify operating frequency bands according to their frequency location. FR1 is from 410 MHz to 7.125 GHz. FR2 is from 24.25 GHz to 71 GHz. Within FR1, FR1-1 is below 3 GHz and FR1-2 is above 3 GHz. Within FR2, FR2-1 is below 52.6 GHz and FR2-2 is above 52.6 GHz.

**FR1**, **FR1-1**, **FR1-2**, **FR2**, **FR2-1**, **FR2-2**: see "**FR**".

**G-CS-RNTI**: Group Configured Scheduling Radio Network Temporary Identifier.

**GF**: Galois Field. The formal name for a finite arithmetic algebraic structure. Most coding operations are done in a Galois Field with two elements denoted "GF(2)".

**GHz**: Gigahertz. It is $10^9$ Hertz.

**GINs**: Group IDs for Network selection.

**GMSK**: Gaussian Minimum Shift Keying. A type of digital modulation used in GSM.

**gNB ID**: gNB Identity. A static identifier for each and every gNode B within a Public Land Mobile Network. It is from 22 to 32 bits long. Together with the Cell Identity (CI) constitutes the New Radio Cell Identity (NCI).

**gNB**: next generation NodeB. It is the term to denote the specific base stations of 5G. It is the same as a gNodeB.

**gNodeB**: next generation NodeB. It is the term to denote the specific base stations of 5G. It is the same as a gNB.

**GNSS**: Global Navigation Satellite Systems (e.g., Global Positioning System or Galileo).

**Gold sequence**: it is a type of binary sequence. They can be produced by the exclusive-or of two specific M-sequences of the same length.

**GPS**: Global Positioning System. A satellite-based positioning system with global coverage.

**G-RNTI**: Group Radio Network Temporary Identifier.

**GSM**: Global System for Mobile communications.

**HAPS**: High-altitude platform stations.

**HARQ**: Hybrid Automatic Repeat reQuest protocol.

**HC-Type**: HARQ-ACK codebook type. There are HC-Type 0, HC-Type 1 and HC-Type 3 (click 3.7.5, 7.6.1).

**HIBS**: High-altitude IMS base stations.

**HO**: Handover.

**HRNN**: Human-Readable Network Names. The list of human-readable network names contained in SIB10 corresponding to the network identifiers listed in SIB1.

**IAB**: Integrated Access and Backhaul. It allows for multi-hop backhauling using the same frequencies employed for User Equipment (UE) access or a distinct, dedicated, frequency.

**IAB-MT**: Integrated Access and Backhaul Mobile Termination. It is a subsystem of an IAB node. In many aspects regarding the 5G Downlink and Uplink (but not Sidelink) specifications, the IAB-MT behaves as a UE.

**iDFT**: Inverse Discrete Fourier Transform.

**IE**: Information Element. A message field or configuration parameter.

**IMT**: International Mobile Telecommunications. The ITU term for mobile communications systems (GSM, UMTS, LTE, 5G, …).

**IMT-2000**: International Mobile Telecommunications 2000. The ITU term for 3G/UMTS systems.

**IMT-2020**: International Mobile Telecommunications 2020. The ITU term for 5G systems.

**IMT-Advanced**: International Mobile Telecommunications Advanced. The ITU term for 4G/LTE.

**INT-RNTI**: Interruption Radio Network Temporary Identifier.

**IoT**: Internet of Things. It is a reference to the communication among ordinary objects (sensors, light switches, washing machines, …).

**ITS**: Intelligent Transportation Systems. A vertical 5G application area consisting of a range of advanced transport- and traffic-management services.

**ITU**: International Telecommunications Union. A Standards Development Organization.

**ITU-R**: The Radiocommunication Sector within International Telecommunications Union. ITU is a Standards Development Organization.

**ITU-T**: The Telecommunications Standardization Sector within International Telecommunications Union. ITU is a Standards Development Organization.

**kHz**: kilohertz. It is $10^3$ Hertz.

**Ki**: Kilo binary unit prefix. It is a unit multiplier of $\mathbf{2^{10}}$ (=1,024). For example, 1 Kib is $\mathbf{2^{10}}$ bits and 1 KiB is $\mathbf{2^{10}}$ bytes.

**KPI**: Key Performance Indicator.

$\boldsymbol{k_{SSB}}$: k-stationary Synchronization Signal and PBCH Block (SSB).

**LAA**: License-Assisted Access. A technique used by LTE and 5G where the UE is attached to the network using a licensed carrier, and one or more additional unlicensed carriers are used to increase the UE data rate.

**LBRM**: link-based rate-matching.

**LBT**: Listen Before Talk. A channel access procedure used in unlicensed spectrum.

**LCID**: logical channel identifier.

**LCP**: Long normal cyclic prefix. One of the two types of Normal cyclic prefix, that has a longer duration than the Short normal cyclic prefix.

**LDPC**: Low Density Parity Check. A type of error-detection channel codes.

**LFSR**: Linear Feedback Shift Register. A simple circuit that can produce M-sequences and Gold sequences (using two LFSRs).

**LI**: Layer Indicator.

**LRR**: Link Recovery Request. One of the information elements that can be sent in Uplink Control Information (UCI).

**LSB**: Least Significant Bit(s). The least significant bit(s) of a bitfield, usually represented rightmost.

**LTE**: Long Term Evolution. It is the same as 4G and IMT-Advanced.

**MAC CE**: MAC Control Element.

**MBMS**: Multimedia Broadcast Multicast Service. A multicast/broadcast service in LTE.

**MBS**: Multicast-Broadcast Services. A family of multicast/broadcast services in 5G, introduced in R17.

**MCC**: Mobile country code. Three decimal digits that identify each country having a mobile network. Together with the Mobile Network Code uniquely identifies a mobile network operator.

**MCCH**: Multicast Control Channel. LTE and UMTS have MCCH, but 5G does not.

**MCCH-RNTI**: Multicast Control Channel (MCCH) Radio Network Temporary Identifier. LTE and UMTS have MCCH, but 5G does not.

**MCG**: Master Cell Group, used in Dual Connectivity. It is a group of serving cells associated with the Master RAN node. It comprises the Special Cell, which is known as the Primary Cell, and optionally one or more Secondary Cells.

**MCS**: Modulation and Coding Scheme. It represents both the modulation and the channel code used in the subcarrier.

**MCS-C-RNTI**: Modulation and Coding Scheme Cell Radio Network Temporary Identifier. It is a unique UE identification used for indicating an alternative MCS table for PDSCH and PUSCH.

**MHz**: Megahertz. It is $10^6$ Hertz.

**Mi**: Mega binary unit prefix. It is a unit multiplier of $\mathbf{2^{20}}$ (~ $1{,}049 \cdot 10^3$). For example, 1 Mib is $\mathbf{2^{20}}$ bits and 1 MiB is $\mathbf{2^{20}}$ bytes.

**MIMO**: Multiple-Input Multiple-Output. A radio technology consisting of using several transmitting and/or several receiving antennas. This improves radio transmission and allows in some cases multiple space reutilization of the spectrum.

**mMTC**: massive Machine-Type Communications. It is a 5G "use case" (i.e., a service class) targeted to massive sensing and actuator applications.

**MNC**: Mobile Network Code. Two or three decimal digits (notice MNC "01" is not the same as MNC of "001") that identify a mobile network operator, within a Mobile Country Code.

**MOCN**: Multi Operator Core Network. A technique for Radio Access Network sharing.

**MORAN**: Multi Operator Radio Access Network. A technique for Radio Access Network sharing

**MRB**: Multicast Radio Bearers. Provide IP multicast communication.

**MRC**: Maximum Ratio-Combining. It is a beamforming technique for Multiple-Input Multiple-Output (MIMO).

**MR-DC**: Multi Radio Dual Connectivity. Dual connectivity using two different radio technologies. For example, dual connectivity to one master gNB and to one secondary ng-eNB.

**MRT**: Maximum Ratio Transmission. It is a beamforming technique for Multiple-Input Multiple-Output (MIMO).

**MS**: Mobile Station. A UE in GSM terminology.

**MSB**: Most Significant Bit(s). The most significant bit(s) of a bitfield, usually represented leftmost.

**M-sequence**: Maximum length sequence. It is a type of pseudorandom binary sequence. They are sometimes called MLS or n-sequence. M-sequences are spectrally flat, with the exception of a near-zero DC term.

**Msg0**, **Msg1**, **Msg2**, **Msg3** and **Msg4**: Messages of the random access procedure. They are not acronyms. Other messages are MsgA and MsgB.

**MsgA**: Message of the random access procedure. It is not an acronym. Other messages are Msg0, Msg1, Msg2, Msg3, Msg4 and MsgB.

**MsgB**: Message of the random access procedure. It is not an acronym. Other messages are Msg0, Msg1, Msg2, Msg3, Msg4 and MsgA.

**MSGB-RNTI**: Message B Radio Network Temporary Identifier. It is used to send the MsgB of the random access procedure.

**MSI**: Minimum System Information. It comprises the PBCH System Information, the MIB and SIB1.

**Multi-TRP**, **Multi-TRxP**: Multi Transmission/Reception Point. Having several Transmission/Reception points simultaneously interacting with each UE.

**MU-MIMO**: Multi-User Multiple-Input Multiple-Output. A radio technology consisting of simultaneously performing MIMO with more than one UE.

**MUSIM**: Multiple Universal Subscriber Identity Modules.

**N/A**: Not Applicable.

**NACK**: Negative Acknowledgement. It is a control frame sent to confirm the **incorrect** reception of a data frame. Notice that NACK is different from (N)ACK, which represents either a Negative Acknowledgement (NACK) and/or an Acknowledgement (ACK).

**NAS**: Non-Access Stratum. Protocol between the core and the UE used to manage the establishment of communication sessions and for maintaining continuous communication with the UE as it moves.

**NB-IoT**: Narrowband Internet of Things.

**NCD-SSB**: non cell defining synchronization signal and PBCH block (SSB). It is a SSB that does not containsCORESET#0.

**NCGI**: New Radio Cell Global Identity. It identifies every NR cell globally. It is similar to the EUTRA Cell Global Identifier (ECGI) used in LTE. It is constructed from the Public Land Mobile Network Identifier (PLMN ID) to which the cell belongs to and the New Radio Cell Identity (NCI) of the cell.

**NCI**: New Radio Cell Identity. A 36-bit static identifier assigned to each and every cell within a Public Land Mobile Network. It is composed of the gNode B Identity (gNB ID), which is from 22 to 32 bits, and the Cell Identity (CI) which is from 4 to 14 bits.

**NCP**: Normal duration cyclic prefix. A cyclic prefix that implies 14 symbols in each slot.

**NDI**: new data indicator. It is the "alternating bit" of each HARQ process.

**NDL**: Normal Downlink. Term applied to a channel or carrier to indicate that it is contained in the Downlink part of an FDD band (i.e., not contained in a SDL band).

**NE-DC**: New Radio and E-UTRA Dual Connectivity. It is dual connectivity to one master gNB and to one secondary ng-eNB. It is specific of Non-Standalone deployment option 4. E-UTRA is Evolved-Universal Terrestrial Radio Access, which is the LTE radio interface.

**NGCN**: Next Generation Core Network. It is the term to denote the 5G Core Network. The "core" or "core network" is the overall control system of a mobile communications network.

**NG-DC**: Next Generation Dual Connectivity. This term is sometimes used to refer to New Radio Dual Connectivity.

**ng-eNB**: next-generation enabled eNB. It is an enhanced Node-B (an eNB, this is, an LTE base station) with added functionality that enables it to interoperate with the 5G core.

**NPN**: Non-Public Network. A private mobile network. They can be a Standalone Non-Public Network (SNPN) or a Public Network Integrated Non-Public Network (PNI-NPN).

**NR**: New Radio. The air interface of 5G.

**NR-ARFCN**: New Radio Absolute Radio-Frequency Channel Number. It is an integer number that identifies, in 5G, different carrier elements, one of them being the center frequency of the next-above-middle subcarrier of a carrier.

**NR-DC**: New Radio Dual Connectivity. It is dual connectivity to one master gNB and one secondary gNB. It is specific of standalone deployments.

**NSA**: Non-Standalone mode. A way to deploy the 5G RAN, while keeping the control in LTE. This requires having LTE coverage in addition to the 5G RAN coverage.

**NSIB-Cell**: A cell where SSB is broadcast, but where neither `SIB1` nor CORESET#0 configuration are broadcast.

**NSSB-Cell**: A cell where SSB is not broadcast. Notice that every NSSB-Cell is also an NSIB-Cell.

**NTN**: Non-Terrestrial Networks. It typically refers to satellite networks.

**NUL**: Normal Uplink. Term applied to a channel or carrier to indicate that it is contained in the Uplink part of an FDD band (i.e., not contained in a SUL band).

**NW**: Network. Sometimes used to refer in a generic way to the functions of the Radio Access Network, including the base stations, and the core.

**OAM**: Operations And Maintenance. The set of systems and protocols for network management and operation.

**OCC**: Orthogonal Cover Code. A line code used in Physical Downlink Shared Channel Demodulation Reference Signals (within each Code Division Multiplexing Group) and in Physical Uplink Control Channel.

**OFDM**: Orthogonal Frequency Division Multiplexing.

**OFDMA**: Orthogonal Frequency Division Multiple Access.

**OOB**: out of band. It can be used as “out of band emissions” to refer to the power that is emitted outside of the assigned frequency segment.

**OSI**: Other System Information. It comprises all the System Information Blocks (other than SIB1) and all the positioning System Information Blocks (posSIBs). In other contexts it means “Open Systems Interconnection”.

**OTA**: Over-the-air. It usually refers to measurements and tests for the air interface.

**PaDU**: Packet Data Unit. It is a specific subsystem in 5G radio access network. It can be, among others, an IP packet or an Ethernet frame.

**Paired spectrum bands**: bands where separated frequency ranges are assigned for Downlink and Uplink transmission. The paired spectrum are Frequency Division Duplexing (FDD) bands, Supplementary Downlink (SDL) ands and Supplementary Uplink (SUL) bands.

**PAPR**: Peak-to-average power ratio. A characteristic of modulation techniques. High PAPR usually means high power consumption.

**PBCH**: Physical Broadcast Channel.

**PCCH**: Paging Control Channel. It is a Downlink logical binary channel. It is used to page the UEs whose location at cell level is not known to the network.

**PCell**: Primary Cell. It is used to start the initial access and is considered as main cell in the Master Cell Group (MCG). It will come under the MCG. There might be many cells under MCG.

**PCH**: Paging Channel. It is a Downlink transport binary channel.

**PCI**: Physical Cell Identity.

**PCP**: PRACH-specific cyclic prefix.

**PDCCH**: Physical Downlink Control Channel. Carries DCI messages.

**PDSCH**: Physical Downlink Shared Channel.

**PDU**: it may be a Protocol Data Unit or Packet Data Unit, depending on context. Protocol Data Unit is a generic term in the protocol architectures of many systems. Packet Data Unit is a specific subsystem in the 5G radio access network.

**PDU**: it may be either a Protocol Data Unit or Packet Data Unit. Protocol Data Unit is a generic term in protocol architectures of many systems. Packet Data Unit is a specific subsystem in 5G radio access network.

**PEI-RNTI**: Paging Early Indication Radio Network Temporary Identifier.

**PLMN ID**: Public Land Mobile Network Identifier. An identifier that uniquely identifies each mobile communications network operating in the world. It is composed of 3 decimal digits for the Mobile Country Code (MCC) plus 2 or 3 decimal digits for the Mobile Network Code (MNC).

**PLMN**: Public Land Mobile Network. A mobile communications network providing public service to users.

**PMI**: Precoding Matrix Indicator.

**PNI-NPN**: Public Network Integrated Non-Public Network. A type of non-public network (NPN).

**PO**: PUSCH transmission occasion in PUSCH configured grants.

**Point A**: is the center frequency of subcarrier 0, from Common Resource Block 0, of all the CRB ranges (refer to TS 38.211).

**posSIB**: positioning System Information Block.

**PRACH**: Physical Random Access Channel.

**PRB**: Physical Resource Block. In 5G, it is one of the Resource Blocks (i.e., 12 contiguous subcarriers), numbered 0 (lowest frequency) to n of a carrier bandwidth part.

**PRG**: Physical Resource Block Group. It is a range of consecutive Physical Resource Blocks (PRBs), used for frequency resource allocation of PDSCH. The number of consecutive PRBs is from 2 to 16, in powers of 2, and it depends on the number of PRBs of the corresponding bandwidth part or CORESET#0.

**PRG**: Precoding Resource Block Group.

**P-RNTI**: Paging Radio Network Temporary Identifier.

**PRS**: Positioning Reference Signals. They are sent Downlink for positioning purposes.

**PSCell**: Primary Secondary Cell. It will come under the Secondary Cell Group (SCG). It performs the initial access under SCG.

**PSK**: Phase-Shift Keying. A type of digital modulation.

**PS-RNTI**: Power Saving Radio Network Temporary Identifier.

**PSS**: Primary Synchronization Signal. It is by the base station sent (together with the Secondary Synchronization Signal) for the UEs to acquire Downlink synchronization.

**PSym**, **PSymbol**: PRACH-specific symbol. The specific symbols used in the RA preamble.

**pTAG**: primary Timing Advance Group. A Timing Advance Group that contains a Primary Cell.

**PTI**: Precoding Type Indicator.

**PTM**: point to multipoint.

**PTP**: point to point.

**PT-RS**: Phase-Tracking Reference Signal.

**PUCCH**: Physical Uplink Control Channel. It carries UCI messages.

**PUSCH**: Physical Uplink Shared Channel.

**QAM**: Quadrature-Amplitude Modulation.

**QCL**: Quasi Co Location. Two antenna ports are said to be quasi co-located if properties of the channel over which a symbol on one antenna port is conveyed can be inferred from the channel over which a symbol on the other antenna port is conveyed.

**QoS**: Quality of Service. The characteristics of a data connection/flow in terms of throughput, delay, delay jitter, packet-loss probability, and other relevant metrics.

**QPSK**: Quadrature Phase-Shit Keying.

**RA**: Random Access.

**RACH**: Random Access Channel. It is a Transport binary channel.

**RAPID**: Random Access Preamble Identifier. It is a 6-bit identifier, sent to the UE, whose value matches the specific RA preamble sequence received by the base station.

**RAR**: Random Access Response. Sometimes called "Msg2". It is one the messages in the random access procedure.

**RA-RNTI**: Random Access Radio Network Temporary Identifier.

**RAT**: Radio Access Technology. A generic term to denote different technologies of air interfaces (e.g., GSM, UMTS, LTE, 5G, WiFi, …).

**RB**: It may be a Resource Block or a Radio Bearer, depending on context. A 5G resource block is 12 contiguous subcarriers. A LTE resource block is all the symbols in one subframe over 12 contiguous subcarriers. A radio bearer is one of the channels over the PDCP protocol.

**RE**: Resource Element. It is 1 symbol over 1 subcarrier.

**RedCap**: Reduced Capability UE. It is a standard for 5G UEs well-suited for a wide range of IoT applications.

**REG**: Resource Element Group. It is 1 symbol over 1 Resource Block (i.e., over 12 contiguous subcarriers).

**RF**: Radio Frequency. Usually refers to the radio frequency elements of the system (amplifiers, filters, frequency shifters, …).

**RI**: Rank Indicator.

**RIM**: Remote Interference Management. Mitigation of Crosslink Interference of an Uplink TDD signal caused by a distant base station Downlink transmission in the same channel.

**RIM-RS**: Remote Interference Management Reference Signal.

**RIT**: Radio Interface Technology. Term used in ITU-R to denote the different radio telecommunication technologies.

**RIV**: Resource Indicator Value. It is used for frequency-domain resource allocation. This is the term used in TS 38.331, but in other TS documents it is called "Resource Indication Value".

**RLC**: Radio Link Control. It is a logical link control (layer 2) protocol.

**RLF**: Radio Link Failure. A type of error condition in the air interface.

**RMSI**: Remaining Minimum System Information. It is another name for System Information Block 1 (SIB1).

**Rn**: Release n, where n is an integer number. It is one of the 3GPP periodical releases of a complete set of Technical Specifications.

**RNTI**: Radio Network Temporary Identifier.

**RO**: random access preamble transmission occasion.

**RRC**: Radio Resource Control. It is a RAN control protocol to manage the configuration of the radio link.

**RRM**: Radio Resource Management. It is a control function to manage the radio configuration of multiple cells.

**RSRP**: Reference Signal Received Power. It is the linear average over the power contributions (in Watt) of a given set of Resource Elements. It is used to convey the received power level of a specific reference signal (CSI-RS or SSB).

**RSRP**: Reference Signal Received Power.

**RSRQ**: Reference Signal Received Quality.

**RTT**: Round Trip Time. It is the channel propagation delay from the transmitter to the receiver plus the one from the receiver to the transmitter. It is also used for transit delay in networks, instead of for propagation delay.

**SA**: Service & Systems Aspects. One of the Technical Specification Groups of **3GPP**.

**SA**: Standalone mode. Deploy a complete 5G network, including 5G RAN, 5G signaling and 5G core.

**SAR**: Specific Absorption Rate. It is a measure (in units of Watts/kilogram) of the amount of RF power absorbed by the human body whenever a radio device transmits.

**SBAS**: Satellite-based Augmentation Systems. They improve the accuracy and reliability of Global Navigation Satellite Systems information by correcting signal measurement errors and by providing additional information.

**SC**: Subcarrier. One of the individual carrier signals that compose the overall carrier.

**SCell**: Secondary Cell. It will come under Master/Secondary Cell Groups. The UE might be configured with one or more SCells in connected mode. SCells can be activated or deactivated according to traffic.

**SCG**: Secondary Cell Group, used in Dual Connectivity. It is a group of serving cells associated with the Secondary RAN Node. It comprises the Special Cell, which is known as the Primary Cell, and optionally one or more Secondary Cells.

**SC-OFDMA**: Single-Carrier Orthogonal Frequency Division Multiple Access.

**SCP**: short normal cyclic prefix. One of the two types of normal cyclic prefix, that has a shorter duration than the long normal cyclic prefix.

**SDL**: Supplementary Downlink.

**SDMA**: Spatial Division Multiple Access. A technique to perform multiple access over the same medium using multiple receive antennas. It is equivalent to Multi User Multiple-Input Multiple Output (MU-MIMO) in the Uplink direction.

**SDO**: Standards Development Organization.

**SDT**: Small Data Transmission.

**SDU**: Service Data Unit.

**Semi-Persistent Scheduling V-RNTI**: Semi-Persistent Scheduling for Vehicle to X (V2X) Radio Network Temporary Identifier.

**SFI**: Slot Format Indication. Informs the UE if a symbol is Downlink, Uplink or Flexible. SFI can be Static, semi-static (both through Radio Resource Control) or Dynamic (through DCI).

**SFI-RNTI**: Slot Format Indication Radio Network Temporary Identifier.

**SFN**: System Frame Number. A cyclic serial number carried by each frame. In 5G, the SFN is 10 bits, allowing for a 1,024-frame cycle.

**SIB**: System Information Block.

**SIB1-Cell**: A cell where `SIB1` is periodically broadcast through PDSCH.

**SIM**: Subscriber Identity Module. It is the usual (SIM) card introduced into the phone that allows the network to identify a subscriber.

**SINR**: Signal to Interference and Noise Ratio. The division between the signal plus interference power and the noise power, typically measured at the receiver.

**SI-RNTI**: System Information Radio Network Temporary Identifier. It is the RNTI used to mask the **CRC** of the Downlink Control Information message (Format 1_0) that locates the System Information Blocks (SIBs) within PDSCH. The value of SI-RNTI is 16 1s.

**SISO**: Single-Input Single-Output. The conventional radio technology using 1 transmitting and 1 receiving antenna. This term was introduced to clarify the system characteristics in respect to MIMO (Multiple-Input Multiple-Output).

**SL**: Sidelink, this is, transmission direction between UEs.

**SL-CS-RNTI**: Sidelink Transmission (Configured Scheduled) Radio Network Temporary Identifier.

**SLIV**: Start Symbol and Length Indicator Value. A single value that conveys both a start symbol interval and a quantity of symbol intervals. It is used for time-domain resource allocation.

**SL-RNTI**: Sidelink Transmission Radio Network Temporary Identifier.

**SMF**: Session Management Function. The function (part of the 5G Core) that handles the sessions between UEs and the network.

**SNPN**: Standalone Non-Public Network. A type of non-public network (NPN).

**SNR**: Signal to noise ratio. The division between the signal power and the noise power, typically measured at the receiver.

**SpCell**: Special Cell. It is a Primary Cell and/or a Primary Secondary Cell.

**SP-CSI-RNTI**: Semi-Persistent Channel State Information Radio Network Temporary Identifier.

**SPS**: Semi-persistent Scheduling.

**SR**: Scheduling Request. A request sent from a UE to the base station when it wants to transmit.

**SRB**: Signaling Radio Bearer. It is a radio bearer used for Radio Resource Control and Non-Access Stratum signaling.

**SRS**: Sounding Reference Signals, sent Uplink for positioning and other purposes.

**SS/PBCH** block: Synchronization Signal and PBCH Block (SSB). It is the combination of the Primary Synchronization Signal (PSS), Primary Synchronization Signal (SSS) and Physical Broadcast Channel (PBCH). It is the same as SSB. It is sent in 4 symbols over 240 subcarriers.

**SSB**: Synchronization Signal and PBCH Block. It is the combination of the Primary Synchronization Signal (PSS), Primary Synchronization Signal (SSS) and Physical Broadcast Channel (PBCH). It is the same as SS/PBCH. It is sent in 4 symbols over 240 subcarriers.

**SSBRI**: SSB Resource Indicator. It indicates which SSB instance has been received by the UE with the best signal quality.

**SS-RSRP**: Synchronization Signal Reference Signal Received Power. It is the linear average over the power contributions (in Watt) of the Resource Elements that carry a Synchronization Signal.

**SS-RSRQ**: Synchronization Signal Reference Signal Received Quality. It is the ratio N•SS-RSRP/NR carrier RSSI, where N is the number of resource blocks in the NR carrier RSSI measurement bandwidth.

**SSS**: Secondary Synchronization Signal. It is sent by the base station (together with the Primary Synchronization Signal) for the UEs to acquire Downlink synchronization.

**sTAG**: secondary Timing Advance Group. A Timing Advance Group that only contains secondary Cells (and no Primary Cell).

**SUL**: Supplementary Uplink.

**TAG**: Timing Advance Group. One or more serving cells with the same Uplink Timing Advance and the same Downlink timing reference cell.

**TBoMS**: transport block processing over multi-slot.

**TBS**: transport block size. The size (in bits) of the transport block that is contained in one PDSCH grant.

$\boldsymbol{T_c}$: 5G Physical Layer Timing unit. It is the sampling time for a subcarrier spacing of 480 kHz. $T_c$ = 1 / (SCS • Nf) = 1 / (480•103•4,096). $T_c$ ~ 0.509 ns. $T_c = 64 \bullet \boldsymbol{T_s}$.

**TCI**: Transmission Configuration Indication. It is a field of some Downlink Control Information messages.

**TC-RNTI**: Temporary Cell Radio Network Temporary Identifier. A C-RNTI provisionally allocated during random access procedure. If the procedure is successful the TC-RNTI become a C-RNTI.

**TDD**: Time Division Duplexing.

**TDRA**: Time-Domain Resource Assignment. It is a field of DCI messages. It is also a resource to be assigned for PDSCH and PUSCH transmissions.

**THz**: Megahertz. It is $10^{12}$ Hertz.

**TPC**: Transmit Power Control.

**TPC-PUCCH-RNTI**: Transmit Power Control-PUCCH – Radio Network Temporary Identifier.

**TPC-PUSCH-RNTI**: Transmit Power Control-PUSCH – Radio Network Temporary Identifier.

**TPC-SRS-RNTI**: Transmit Power Control-Sounding Reference Symbols – Radio Network Temporary Identifier.

**TR**: Technical Report. A type of **3GPP** document.

**TRP**: Transmission/Reception Point. More frequently denoted as “TRxP” (see below).

**TRS**: Tracking Reference Signals.

**TRxP**: Transmission/Reception Point. It is an individual radio unit in the 5G RAN.

$T_s$: LTE Physical Layer Timing unit. It is the sampling time for a subcarrier spacing of 15 kHz. TS = 1 / (SCS • Nf) = 1 / ($15 \cdot 10^3 \cdot 2{,}048$). $T_s \sim 32.552$ ns. $T_c = 64 \cdot T_s$.

**TS**: Technical Specification. A relevant type of **3GPP** document.

**TSG**: Technical Specification Group. Each of the three broad groups in which **3GPP** organizes its work.

**TSN**: Time-Sensitive Networking. Network service where latency variations (jitter) and accurate clock distribution are as important as low latency in itself.

**TTI**: Transmission Time Interval. Normally refers to a frame, subframe or slot.

**UCI**: Uplink Control Information.

**UDM-Type**: PUSCH DM-RS configuration type. There are UDM-Type 1 and UDM-Type 2 (click 7.11.5.2).

**UE**: User Equipment. A UE in 5G, LTE and UMTS terminology.

**UFA-Type**: type of PUSCH frequency-domain resource allocation. There are 3 UFA types (click 7.11.3): Type 0 (click 7.11.3.2), UFA-Type 1 (click 7.11.3.3) and UFA-Type 2 (click 7.11.3.4).

**UL**: Uplink.

**UL-SCH**: Uplink Shared Chanel. It is a Transport binary channel.

**Unpaired spectrum bands**: bands where the same frequency range is assigned for Downlink and Uplink transmission. The unpaired spectrum bands are Time Division Duplexing (TDD) bands.

**UPF**: User Plane Function. The function (part of the 5G Core) that handles user data.

**URLLC**: Ultra-Reliable Low-Latency Communications. It is a 5G "use case" (i.e., a service class) targeted to hard real-time applications (e.g., robotic control).

**USS**: UE-Specific Search Space. It is a class of search space for CORESETs.

**V2V**: Vehicle-to-Vehicle communication.

**V2X**: Vehicle to Everything communication.

**VRB**: Virtual Resource Block. It is a Resource Block in a configurable **virtual** space. Each VRB is mapped to one Physical Resource Block.

**VRG**: Virtual Resource Block Group. It is a range of consecutive Virtual Resource Blocks (VRBs), used for frequency resource allocation of PDSCH. The number of consecutive VRBs is from 2 to 16, in powers of 2, and it depends on the number of Physical Resource Blocks of the corresponding bandwidth part or CORESET#0.

**WDCMA**: Wideband Code Division Multiple Access. A media access control technique, widely known for its usage in 3G systems.

**WP5D**: Working Party 5 D. It is the working party (workgroup) within ITU-R in charge of the coordinated definition of International Mobile Telecommunication (IMT) standards.

**WRC**: World Radiocommunication Conference. It is a conference organized by ITU to globally agree on Radio Regulations, the use of the radio-frequency spectrum as well as geostationary and non-geostationary satellite orbits. It is held every three to four years.

**ZC sequence**: Zadoff-Chou sequence. It is a complex-valued sequence which, when applied to a signal, gives rise to a new signal of constant amplitude, before and after the DFT. The signals arising from cyclic shifted versions of a Zadoff–Chu sequence are uncorrelated with one another.

**Zero forcing**: a beamforming technique for Multiple-Input Multiple-Output (MIMO).

**π/2-BPSK**: π/2 Binary Phase-Shift Keying. A binary phase modulation using 2 constellations, shifted 90º, having 2 symbols each. It has 1 bit/symbol. It has a very low peak-to-average power ratio (PAPR) and high RF amplifier power efficiency at lower data-rates, which makes it very suitable for Uplink transmission.

# 16. Contents

1. Introduction ..... 1
1.1 Overview of a 5G network ..... 1
1.1.1 5G major network subsystems ..... 1
1.1.2 NR overview: NR nodes, antenna sites and cells ..... 2
1.1.3 NR overview: channels, carriers and bandwidth parts ..... 4
1.1.4 NR overview: bandwidth part frequency-time-space multiplexing ..... 5
1.2 Main 3GPP TS documents referred in this book ..... 9
1.3 Fonts and colors used for different 5G elements ..... 12
2. NR frequency, time and space ..... 15
2.1 NR numerology ..... 15
2.1.1 Numerology, subcarrier spacing, slot duration and symbol interval duration 16
2.1.2 Symbol rate per MHz of bandwidth ..... 17
2.1.3 Justification and implications of 7 numerologies ..... 17
2.1.4 Types of numerologies ..... 19
2.1.5 Multiple simultaneous numerologies ..... 20
2.2 NR frequency management ..... 23
2.2.1 Band organization of the spectrum, and frequency ranges ..... 26
2.2.2 Frequency block organization of a band ..... 31
2.2.3 Channel configuration of a frequency block ..... 34
2.2.4 CRB range configuration of a cell's channel ..... 57
2.2.5 Carrier configuration of each cell's channel ..... 71
2.2.6 Carrier parameters ..... 78
2.2.7 Bandwidth part configuration of each carrier ..... 92
2.2.8 Bandwidth part parameters ..... 105
2.2.9 Resource block group (RBG) configuration of each bandwidth part ... 129
2.2.10 Subcarriers ..... 131
2.2.11 Frequency alignment of different elements ..... 140
2.2.12 Frequency division duplexing (FDD) in 5G ..... 142
2.2.13 Summary of 5G operating bands ..... 144
2.2.14 Summary of LTE and other frequency bands ..... 150
2.2.15 Global frequency raster and absolute radio frequency channel number (ARFCN) ..... 154
2.3 NR time management ..... 156
2.3.1 Transmission time structure ..... 156
2.3.2 Duration of cyclic prefix, symbol, symbol interval and slot ..... 159

2.3.3 Configuration of cyclic prefix for subcarriers, bandwidth parts and carriers 164
2.3.4 Time division duplexing (TDD) in 5G ..... 164
2.3.5 Time division duplexing (TDD) parameters ..... 176
2.3.6 Time synchronization ..... 187
2.4 NR space management ..... 191
2.4.1 Beamforming ..... 191
2.4.2 Precoding and detecting ..... 195
2.4.3 Multiple-Input Multiple-Output (MIMO) ..... 197
2.4.4 Relationships between diversity, beamforming, precoding and MIMO 207
2.4.5 Channel State Information ..... 208
2.5 NR signals ..... 209
2.5.1 Overview of NR signal processing ..... 210
2.5.2 5G NR Tx signal processing ..... 210
2.5.3 Baseband signal representation ..... 212
3. Cellular networks and cells ..... 213
3.1 Cell, antenna site and sector ..... 213
3.1.1 Cell ..... 213
3.1.2 Neighbor cells ..... 214
3.1.3 Antenna site ..... 215
3.1.4 Sector ..... 215
3.2 Cell types and cell groups ..... 216
3.2.1 Cell types under the network point of view ..... 216
3.2.2 Cell types and cell groups under the UE point of view ..... 218
3.2.3 Cell types based on their size ..... 221
3.3 Relationship between cells, NR nodes, channels and UEs ..... 222
3.4 Network types ..... 223
3.5 Network, node and cell identity ..... 224
3.5.1 Network identity ..... 224
3.5.2 RAN node identity ..... 225
3.5.3 Cell identity ..... 225
3.5.4 System information area identity ..... 227
3.6 Network, node and cell identity parameters ..... 228
3.6.1 Network, node and cell identity parameters within `SIB1` ..... 228
3.6.2 Network, node and cell identity parameters within the Non-SI parameter group 228

3.6.3 Network, node and cell identity parameters sent in RRC inter-node messages 231
3.6.4 `CellAccessRelatedInfo` data type ..... 232
3.6.5 `PLMN-IdentityInfoList` and `PLMN-IdentityInfo` data types 235
3.6.6 `PLMN-Identity` data type ..... 237
3.6.7 `NPN-Identity-r16` data type ..... 237
3.7 Cell group parameters ..... 238
3.7.1 Cell group parameters within the Non-SI parameter group ..... 239
3.7.2 Cell group parameters sent in RRC inter-node messages ..... 240
3.7.3 `CellGroupConfig` data type ..... 240
3.7.4 `MAC-CellGroupConfig` data type ..... 242
3.7.5 `PhysicalCellGroupConfig` data type ..... 243
3.8 Cell identification, types, UE configuration and quantity ..... 244
3.9 Cell parameters ..... 246
3.9.1 Serving cell parameters within `SIB1` ..... 246
3.9.2 Serving cell parameters within the Non-SI parameter group ..... 246
3.9.3 `ServingCellConfigCommonSIB` data type ..... 247
3.9.4 `SCellConfig` data type ..... 249
3.9.5 `SpCellConfig` data type ..... 251
3.9.6 `ServingCellConfigCommon` data type ..... 252
3.9.7 `ServingCellConfig` data type ..... 255
3.9.8 `ReconfigurationWithSync` data type ..... 257
3.9.9 `ServCellIndex` and `SCellIndex` data types ..... 258
3.10 Deployment planning ..... 258
3.10.1 Cell reach ..... 259
3.10.2 Cell range ..... 265
3.10.3 Cell area ..... 268
3.10.4 Cell area capacity ..... 269
3.10.5 Coverage ..... 271
3.10.6 Densification techniques ..... 273
4. NR nodes and UEs ..... 285
4.1 NR node definition ..... 285
4.2 NR node types ..... 285
4.3 NR node functional blocks, physical functions, payload processing, protocol architecture and data flow ..... 286
4.4 NR node functional splits ..... 286

4.5 NR node functional blocks ..... 289
4.5.1 SDAP and RRC functional block ..... 289
4.5.2 Packet Data Convergence Protocol (PDCP) functional block ..... 289
4.5.3 Radio link control sublayer ..... 291
4.5.4 Medium access control sublayer ..... 293
4.5.5 MAC data units: PDUs, subPDUs, subheaders, SDUs, CEs and payloads 295
4.5.6 HARQ protocol in 5G ..... 307
4.5.7 High and Low physical functional blocks ..... 322
4.5.8 Radio Frequency (RF) functional block ..... 330
4.6 Backhaul, midhaul and fronthaul ..... 348
4.6.1 Backhaul ..... 350
4.6.2 Midhaul ..... 351
4.6.3 Fronthaul ..... 352
4.7 NR node subsystems ..... 353
4.7.1 Central Unit (CU) ..... 353
4.7.2 Distributed Unit (DU) ..... 354
4.7.3 Radio Unit (RU) ..... 356
4.7.4 Baseband Unit (BBU) ..... 357
4.7.5 Remote Radio Unit (RRU) ..... 357
4.7.6 Multiband antenna ..... 358
4.7.7 Active Antenna Unit (AAU) ..... 359
4.7.8 NR node cost estimates ..... 360
4.8 Labels in NR node figures ..... 364
4.9 User Equipment (UE) ..... 365
4.9.1 UE capability and categories ..... 366
4.9.2 UE capability delivery ..... 368
4.9.3 UE throughput and capacity ..... 370
4.9.4 Reduced capability (RedCap) UEs ..... 375
4.10 NR node and UE protocol architecture ..... 376
4.10.1 NR radio access protocol stacks ..... 376
4.10.2 User-plane protocol stacks ..... 377
4.10.3 Control-plane protocol stacks ..... 380
4.10.4 Data flow across the user-plane protocol stack ..... 382
4.10.5 Non-access stratum (NAS) ..... 384
4.11 NR binary channels and signals ..... 384
4.11.1 Logical binary channels ..... 386

4.11.2 Transport binary channels ..... 388
4.11.3 Physical binary channels and signals..... 389
4.11.4 Scheduling, transmission/reception occasions and transmissions/receptions..... 398
4.11.5 Antenna ports..... 406
4.12 PDSCH and PUSCH payload processing..... 408
4.12.1 PDSCDH and PUSCH Transport blocks and codewords..... 409
4.12.2 PDSCH and PUSCH Tx payload processing overview..... 411
4.12.3 PDSCH and PUSCH transport block CRC appending..... 416
4.12.4 PDSCH and PUSCH code block segmentation and CRC appending .. 417
4.12.5 PDSCH and PUSCH code block LDPC encoding ..... 422
4.12.6 PDSCH and PUSCH LDPC rate-matching ..... 425
4.12.7 PDSCH and PUSCH rate-matched block concatenation..... 430
4.12.8 PUSCH codeword and PUCCH codeword (UCI) multiplexing..... 431
4.12.9 PDSCH and PUSCH codeword scrambling ..... 431
4.12.10 PDSCH and PUSCH modulation of each scrambled codeword ..... 434
4.12.11 PDSCH and PUSCH symbol to layer mapping..... 435
4.12.12 PUSCH transform precoding (optional)..... 437
4.12.13 PDSCH and PUSCH precoding (optional)..... 437
4.12.14 PDSCH and PUSCH symbol sequence to VREs mapping ..... 438
4.12.15 PDSCH and PUSCH payload VRE to payload PRE mapping..... 441
4.12.16 PDSCH and PUSCH Rx payload processing overview ..... 445
4.12.17 Mapping to PDSCH/PUSCH terms and symbols in TS documents. 449
5. UE configuration parameters and the RRC protocol..... 452
5.1 Classifications of UE parameters ..... 455
5.1.1 Criterion of production protocol layer..... 455
5.1.2 Criterion of belonging to System Information or not..... 455
5.1.3 Criterion of configuration type (common or dedicated)..... 456
5.1.4 Criterion of belonging to Minimum System Information (MSI) or to other System Information ..... 457
5.2 UE parameter groups..... 457
5.3 Stages in UE physical layer configuration ..... 458
5.3.1 PCI parameter group..... 459
5.3.2 PBCH system information parameter group ..... 460
5.3.3 `MIB` parameter group..... 462
5.3.4 `SIB1` parameter group ..... 463
5.3.5 Other SIBs and posSIBs parameter group..... 466

5.3.6 Non-SI parameter group ..... 469
5.3.7 Relationship between configuration type and delivery mode ..... 471
5.4 UE parameters for some physical layer elements ..... 472
5.5 Other SIBs and posSIBs mapping and scheduling parameters ..... 473
5.5.1 Overview of other SIBs and posSIBs mapping parameters ..... 473
5.5.2 Overview of other SIBs and posSIBs scheduling parameters ..... 474
5.5.3 Other SIBs and posSIBs mapping and scheduling parameters within `SIB1` 475
5.6 Other RRC parameters and variables ..... 484
5.6.1 RRC parameters sent by the UE to the NR node ..... 484
5.6.2 RRC UE variables ..... 485
5.7 RRC parameter specification ..... 485
5.7.1 RRC UE data types and constants ..... 485
5.7.2 RRC UE parameters: identification and instances ..... 486
5.7.3 RRC UE parameters: semantics ..... 490
5.7.4 `SIB1` data type ..... 512
5.7.5 `SystemInformation` data type ..... 515
5.7.6 `SystemInformation-IEs` data type ..... 516
5.7.7 `PosSystemInformation-r16-IEs` data type ..... 518
5.7.8 Structure of parameters within the Non-SI parameter group ..... 520
5.8 The RRC protocol ..... 527
5.8.1 RRC services and functions ..... 528
5.8.2 RRC protocol states and transitions ..... 529
5.8.3 RRC procedures and processing delay requirements ..... 532
5.8.4 RRC access control ..... 535
5.8.5 RRC NR messages ..... 536
5.8.6 RRC inter-node messages ..... 543
5.8.7 RRC UE capability retrieval framework ..... 544
5.8.8 RRC transport of NAS messages ..... 544
5.8.9 RRC UE assistance information ..... 545
5.8.10 RRC message segmentation ..... 546
5.9 Brief overview of ASN.1 ..... 546
5.9.1 ASN.1 simple and built-in data types and constants ..... 547
5.9.2 ASN.1 named data types and constants ..... 548
5.9.3 ASN.1 structured data types and constants, and parameters ..... 548
5.9.4 ASN.1 multilevel structured specification of data types and constants 551
5.9.5 ASN.1 identifier, names, uniqueness and duplicates ..... 552

5.9.6 ASN.1 data types: specification, parent/son, ancestor/descendant and top/bottom ..... 553
5.9.7 ASN.1 data type and parameter instances and their quantity ..... 557
5.9.8 ASN.1 data type and parameter ancestor trees ..... 559
5.9.9 ASN.1 other aspects ..... 562
6. NR Downlink physical channels and signals ..... 563
6.1 SSB (PSS, SSS and PBCH) ..... 563
6.1.1 SSB overview ..... 563
6.1.2 SSB scheduling ..... 563
6.1.3 SSB contents ..... 577
6.1.4 SSB processing ..... 578
6.1.5 Tables of per-band SSB subcarrier spacing, SSB pattern and synchronization raster ..... 579
6.2 PBCH overview ..... 581
6.3 PBCH scheduling ..... 582
6.4 PBCH payload and DM-RS contents ..... 582
6.4.1 PBCH contents: PBCH system information parameters ..... 582
6.4.2 PBCH contents: `MIB` representation ..... 584
6.5 PBCH payload and DM-RS processing ..... 584
6.6 PBCH parameters ..... 585
6.7 PDCCH overview ..... 585
6.8 PDCCH scheduling ..... 587
6.8.1 Structure of frequency & duration resources ..... 588
6.8.2 PDCCH frequency-domain resources ..... 589
6.8.3 PDCCH time-domain resources ..... 598
6.8.4 Blind decoding ..... 598
6.8.5 PDCCH search spaces ..... 599
6.8.6 PDCCH common search space Type 0 ..... 607
6.8.7 PDCCH COntrol REsource SETs (CORESETs) ..... 612
6.8.8 PDCCH CORESET#0 ..... 615
6.8.9 PDCCH scheduling of payload PREs and DM-RS PREs ..... 624
6.9 PDCCH payload contents: DCI messages ..... 624
6.9.1 DCI message fields ..... 626
6.9.2 DCI Format 0_0 messages ..... 630
6.9.3 DCI Format 0_1 messages ..... 631
6.9.4 DCI Format 0_2 messages ..... 635
6.9.5 DCI Format 1_0 messages ..... 635

6.9.6 DCI Format 1_1 messages ..... 638
6.9.7 DCI Format 1_2 message ..... 641
6.9.8 DCI Format 2_x messages ..... 641
6.9.9 DCI Format 4_0, 4_1 and 4_2 messages ..... 646
6.9.10 Radio Network Temporary identifier (RNTI) ..... 647
6.10 PDCCH payload processing ..... 652
6.10.1 PDCCH Tx payload processing ..... 653
6.10.2 PDCCH Rx payload processing ..... 654
6.11 PDCCH DM-RS contents ..... 655
6.12 PDCCH parameters ..... 655
6.12.1 PDCCH parameters within `MIB` ..... 656
6.12.2 PDCCH parameters within `SIB1` ..... 656
6.12.3 PDCCH parameters within the Non-SI parameter group ..... 657
6.12.4 `PDCCH-ConfigCommon` data type ..... 659
6.12.5 `PDCCH-Config` data type ..... 662
6.12.6 `ControlResourceSet` data type ..... 665
6.12.7 `ControlResourceSetId`, `ControlResourceSetId-r16` and `ControlResourceSetId-v1610` data types ..... 668
6.12.8 `SearchSpace` data type ..... 669
6.12.9 `SearchSpaceId` data type ..... 674
6.12.10 `PDCCH-ServingCellConfig` data type ..... 674
6.13 PDSCH overview ..... 675
6.14 PDSCH scheduling ..... 676
6.14.1 PDSCH Rx semi-persistent scheduling ..... 677
6.14.2 PDSCH Rx dynamic scheduling ..... 677
6.14.3 PDSCH frequency-domain resources ..... 678
6.14.4 PDSCH time-domain resources ..... 685
6.14.5 PDSCH scheduling of payload PREs, DM-RS PREs, PT-RS PREs and empty PREs ..... 693
6.14.6 PDSCH cell-wide scheduling ..... 713
6.15 PDSCH payload contents ..... 715
6.16 PDSCH payload processing ..... 715
6.16.1 UE determination of PDSCH modulation order and target code rate ..... 715
6.16.2 UE determination of PDSCH redundancy version and quantity of received sequence(s) ..... 716
6.16.3 UE determination of PDSCH quantity and size of transport block(s) ..... 717
6.16.4 UE determination of PDSCH LDPC base graph ..... 719

6.17 PDSCH DM-RS and PT-RS contents ... 719
6.17.1 PDSCH DM-RS contents ... 719
6.17.2 PDSCH PT-RS contents ... 721
6.18 PDSCH parameters ... 722
6.18.1 PDSCH parameters within `MIB` ... 722
6.18.2 PDSCH parameters within `SIB1` ... 723
6.18.3 PDSCH parameters within other SIBs and posSIBs ... 723
6.18.4 PDSCH parameters within the Non-SI parameter group ... 724
6.18.5 `PDSCH-ConfigCommon` data type ... 726
6.18.6 `PDSCH-Config` data type ... 727
6.18.7 `PDSCH-ConfigBroadcast-r17` data type ... 731
6.18.8 `PDSCH-TimeDomainResourceAllocation` data type ... 733
6.18.9 `PDSCH-TimeDomainResourceAllocation-r16` data type ... 735
6.18.10 `PDSCH-ServingCellConfig` data type ... 738
6.18.11 `DMRS-DownlinkConfig` data type ... 739
6.19 PSS and SSS ... 740
6.19.1 PSS and SSS scheduling ... 741
6.19.2 PSS contents and processing ... 741
6.19.3 SSS contents and processing ... 742
6.20 CSI-RS ... 744
6.20.1 CSI-RS scheduling ... 745
6.21 PRS ... 746
6.21.1 PRS scheduling ... 747
6.22 RIM-RS ... 748
7. NR Uplink physical channels and signals ... 750
7.1 PRACH overview ... 750
7.2 PRACH scheduling ... 751
7.2.1 PRACH frequency-domain resources ... 751
7.2.2 PRACH time-domain resources ... 757
7.3 PRACH contents and processing ... 763
7.3.1 RA preamble ... 764
7.3.2 RA preamble Formats ... 765
7.3.3 RA preamble sequences per cell ... 772
7.3.4 RA preamble formal definition ... 773
7.4 PRACH parameters ... 775
7.5 PUCCH overview ... 775

7.6 PUCCH scheduling ..... 777
7.6.1 PUCCH scheduling of HARQ-ACKs ..... 778
7.6.2 PUCCH semi-persistent scheduling of scheduling requests (SRs) ..... 780
7.6.3 PUCCH scheduling of multiple UCI types ..... 781
7.6.4 UE determination of which PUCCH resource to use ..... 782
7.6.5 PUCCH resources ..... 784
7.6.6 PUCCH resource `format` variants and types ..... 795
7.6.7 PUCCH resource sets ..... 807
7.6.8 Several PUCCH transmissions in the same slot ..... 807
7.6.9 PUCCH scheduling of payload PREs and DM-RS PREs ..... 809
7.7 PUCCH payload contents: UCI messages (also sent through PUSCH) ..... 812
7.7.1 Content types of UCI messages ..... 813
7.7.2 Contents of UCI messages ..... 815
7.8 PUCCH payload processing ..... 816
7.8.1 PUCCH resource `format` payload processing summary ..... 817
7.8.2 PUCCH payload Tx processing ..... 819
7.8.3 PUCCH payload Rx processing ..... 841
7.8.4 Mapping to PUCCH processing terms used in TS documents ..... 841
7.9 PUCCH parameters ..... 843
7.9.1 PUCCH parameters within `SIB1` ..... 844
7.9.2 PUCCH parameters within the Non-SI parameter group ..... 844
7.9.3 `PUCCH-ConfigCommon` data type ..... 848
7.9.4 `PUCCH-Config` data type ..... 849
7.9.5 `PUCCH-ConfigurationList-r16` data type ..... 853
7.9.6 `PUCCH-ResourceSet` data type ..... 853
7.9.7 `PUCCH-Resource` data type ..... 855
7.9.8 `PUCCH-format0` to `PUCCH-format4` data types ..... 857
7.9.9 `PUCCH-ResourceExt-v1610` data type ..... 859
7.9.10 `PUCCH-FormatConfig` data type ..... 862
7.9.11 `PUCCH-FormatConfigExt-r17` data type ..... 864
7.9.12 `SchedulingRequestResourceConfig` data type ..... 865
7.9.13 `SchedulingRequestResourceConfigExt-v1610` and `-v1700` data types ..... 867
7.9.14 `SchedulingRequestConfig` data type ..... 868
7.9.15 `SchedulingRequestToAddMod` data type ..... 869
7.9.16 `DMRS-BundlingPUCCH-Config-r17` data type ..... 870

7.10 PUSCH overview ..... 871
7.11 PUSCH scheduling ..... 872
7.11.1 PUSCH Tx semi-persistent scheduling ..... 872
7.11.2 PUSCH Tx dynamic scheduling ..... 874
7.11.3 PUSCH frequency-domain resources ..... 874
7.11.4 PUSCH time-domain resources ..... 881
7.11.5 PUSCH scheduling of payload PREs, DM-RS PREs, PT-RS PREs, SRS PREs and empty PREs ..... 887
7.12 PUSCH payload contents ..... 892
7.13 PUSCH payload processing ..... 893
7.14 PUSCH DM-RS and PT-RS contents ..... 893
7.15 PUSCH parameters ..... 894
7.15.1 PUSCH parameters within `MIB` ..... 894
7.15.2 PUSCH parameters within `SIB1` ..... 894
7.15.3 PUSCH parameters within the Non-SI parameter group ..... 895
7.15.4 `PUSCH-ConfigCommon` data type ..... 897
7.15.5 `PUSCH-Config` data type ..... 899
7.15.6 `PUSCH-TimeDomainResourceAllocation` data type ..... 902
7.15.7 `PUSCH-Allocation-r16` data type ..... 904
7.15.8 `PUSCH-ServingCellConfig` data type ..... 910
7.15.9 `DMRS-UplinkConfig` data type ..... 911
7.16 Sounding reference signals (SRS) ..... 913
8. NR initial access procedure ..... 915
8.1 Cell selection procedure ..... 916
8.2 Cell search: getting the primary and secondary synchronization signals (PSS and SSS) ..... 921
8.2.1 Example of getting PSS/SSS ..... 922
8.3 Getting PBCH System Information, beam selection and `MIB` ..... 923
8.3.1 Example of getting PBCH System Information and `MIB` ..... 924
8.4 Getting CSS-Type 0 and CORESET#0 ..... 927
8.4.1 Example of getting CORESET#0 (PDCCH) frequency location and size 927
8.4.2 Example of getting CSS-Type 0 ..... 931
8.4.3 `MIB` parameter `pdcch-ConfigSIB1` ..... 932
8.5 Getting the PDCCH DCI messages ..... 934
8.6 Getting PDSCH and `SIB1` ..... 935
8.6.1 Location of PDSCH and its MCS ..... 935

8.6.2 Location of PDSCH payload and DM-RS REs ..... 937
8.6.3 Example of getting PDSCH and SIB1 ..... 938
8.7 Getting the CRB ranges ..... 940
8.7.1 `MIB` parameter `ssb-SubcarrierOffset` ..... 941
8.7.2 Variable "$k_{SSB}$" ..... 942
8.7.3 `SIB1` parameter `offsetToPointA` ..... 945
8.7.4 Example of getting the CRB ranges ..... 946
8.8 Getting the initial bandwidth parts and carriers ..... 949
8.9 PUCCH transmission before RRC connection establishment ..... 950
8.10 Initial random access procedure ..... 951
8.11 UE configuration and RRC connection establishment ..... 951
8.12 Initial access procedure parameters ..... 951
8.12.1 Initial access procedure parameters within `MIB` ..... 952
8.12.2 Initial access procedure parameters within `SIB1` ..... 952
8.12.3 `MIB` data type ..... 953
9. NR random access procedure ..... 956
9.1 The 4 RA procedures (and 1 variant) ..... 956
9.1.1 The 4-step and 2-step RA procedures ..... 958
9.1.2 The contention-based (CBRA) and the contention-free (CFRA) RA procedures ..... 960
9.1.3 The 4-step CBRA procedure ..... 961
9.1.4 The 4-step CFRA procedure ..... 968
9.1.5 The 2-step CBRA procedure ..... 969
9.1.6 The 2-step CFRA procedure ..... 971
9.1.7 Computation of RA-RNTI ..... 973
9.2 RA resource selection ..... 973
9.2.1 RA preamble sequence, PRACH occasion and beam sweeping ..... 974
9.2.2 RA resource selection for CBRA/CFRA procedures ..... 976
9.2.3 RA resource selection for beam failure recovery (BFR) ..... 976
9.2.4 RA resource selection for SI request (On-demand SI) ..... 976
9.2.5 RA resource selection for reconfiguration with sync ..... 977
9.2.6 RA resource selection for PDCCH Order ..... 977
9.3 Messages for RA procedures ..... 978
9.3.1 Msg0 (Downlink, RA preamble assignment with(out) PUSCH grant activation) ..... 978
9.3.2 Msg1 (Uplink, Random Access preamble) ..... 979
9.3.3 Msg2 (Downlink, Random Access Response) ..... 980

9.3.4 Msg3 (Uplink, Scheduled Transmission) ..... 981
9.3.5 Msg4 (Downlink, Contention Resolution) ..... 983
9.3.6 MsgA (Uplink, RA preamble and PUSCH payload) ..... 983
9.3.7 MsgB (Downlink, 3 variants) ..... 985
9.4 Triggering and characteristics of a RA procedure ..... 987
9.4.1 Triggering entities of RA procedure ..... 987
9.4.2 Triggering events and characteristics of RA procedure ..... 988
9.5 Random access procedure parameters ..... 990
9.5.1 Random access parameters within `SIB1` ..... 990
9.5.2 Random access parameters within the non-SI parameter group ..... 991
9.5.3 `RACH-ConfigCommon` data type ..... 995
9.5.4 `BeamFailureRecoveryConfig` data type ..... 998
9.5.5 `RACH-ConfigGeneric` data type ..... 1000
9.5.6 `MsgA-ConfigCommon-r16` data type ..... 1003
9.5.7 `MsgA-PUSCH-Config-r16` data type ..... 1004
9.5.8 `RACH-ConfigCommonTwoStepRA-r16` data type ..... 1005
9.5.9 `RACH-ConfigGenericTwoStepRA-r16` data type ..... 1007
9.5.10 `RACH-ConfigDedicated` data type ..... 1011
9.5.11 `CFRA` data type ..... 1012
9.5.12 `CFRA-TwoStep-r16` data type ..... 1013
9.5.13 `RA-InformationCommon-r16` data type ..... 1015
10. NR procedures and techniques ..... 1018
10.1 Bandwidth adaptation (BA) ..... 1018
10.2 Carrier aggregation (CA) ..... 1018
10.2.1 Carrier aggregation control and configuration ..... 1019
10.2.2 Signaling in carrier aggregation ..... 1022
10.2.3 Cross-carrier scheduling ..... 1023
10.2.4 Carrier aggregation architecture ..... 1023
10.2.5 Carrier aggregation objectives ..... 1025
10.2.6 Carrier aggregation and time synchronization ..... 1026
10.3 Dual connectivity (DC) ..... 1026
10.3.1 Types of RAN nodes in dual connectivity ..... 1028
10.3.2 Types of dual connectivity ..... 1028
10.3.3 Types of cell groups in dual connectivity ..... 1029
10.3.4 Dual connectivity and carrier aggregation ..... 1030
10.4 Supplementary Uplink (SUL) ..... 1030

10.5 Supplementary Downlink (SDL) ..... 1032
10.6 5G/LTE static frequency-domain sharing ..... 1033
10.7 5G/LTE dynamic spectrum sharing (DSS) ..... 1033
10.8 Multicast-broadcast services (MBS) ..... 1035
10.9 Small data transmission (SDT) ..... 1036
10.10 Use cases ..... 1036
10.11 Handover ..... 1037
10.12 Processing of OFDM, OFDMA and DFT-s-OFDM in 5G ..... 1038
10.12.1 OFDM processing ..... 1038
10.12.2 OFDMA processing ..... 1039
10.12.3 DFT-s-OFDM processing ..... 1039
10.12.4 Summary of (SC)OFDM(A) processing differences ..... 1041
10.12.5 Labels on the figures for (SC)OFDM(A) processing ..... 1041
10.13 Coordinated Multipoint (CoMP) ..... 1042
10.14 Integrated Access and Backhaul (IAB) ..... 1042
10.15 5G deployment options: Standalone (SA) and Non-Standalone (NSA). 1043
10.16 RAN sharing ..... 1046
10.17 Narrowband IoT ..... 1047
10.17.1 GSM carrier ..... 1047
10.18 5G Edge Computing ..... 1052
10.18.1 Fundamental MEC modes ..... 1052
10.18.2 Edge Computing in 5G and LTE ..... 1052
10.18.3 Advantages of Edge Computing ..... 1053
11. NR underlying technologies ..... 1055
11.1 Bit, symbol rate, bitrate, throughput, (area) capacity and spectral efficiency 1055
11.1.1 Bit ..... 1055
11.1.2 Symbol rate, bitrate, throughput, capacity and spectral efficiency ..... 1056
11.2 Modulation and coding ..... 1058
11.2.1 Digital modulations ..... 1058
11.2.2 Line codes ..... 1061
11.2.3 Channel codes ..... 1062
11.2.4 Modulation and coding scheme (MCS) ..... 1065
11.3 Single-carrier signals and multicarrier signals ..... 1065
11.3.1 Single-carrier signals ..... 1066
11.3.2 Multicarrier signals ..... 1067
11.4 Peak-to-average power ratio (PAPR) ..... 1068

11.5 Duplexing ..... 1069
11.5.1 Duplexing overview ..... 1069
11.5.2 TDD guard periods ..... 1071
11.5.3 Comparison between FDD and TDD ..... 1073
11.6 Multiplexing and demultiplexing ..... 1074
11.6.1 Frequency-division multiplexing ..... 1076
11.6.2 Time-division multiplexing ..... 1076
11.6.3 Space-division multiplexing ..... 1076
11.6.4 Code-division multiplexing ..... 1077
11.7 Inverse multiplexing and demultiplexing ..... 1077
11.8 Diversity ..... 1078
11.8.1 Space diversity ..... 1080
11.9 Multiple access ..... 1082
11.10 Relationships between duplexing, multiplexing, inverse multiplexing, diversity and multiple access ..... 1082
11.11 Medium Access Control (MAC) ..... 1083
11.12 Hybrid Automatic Repeat reQuest (HARQ) protocols ..... 1085
11.12.1 Simple HARQ protocols ..... 1086
11.12.2 Soft-combining HARQ protocols ..... 1087
11.12.3 HARQ protocol in LTE ..... 1087
11.12.4 HARQ protocol in UMTS ..... 1093
11.13 Cyclic Prefixing ..... 1094
11.14 Orthogonal frequency-division multiplexing (OFDM) ..... 1094
11.15 Orthogonal frequency-division multiple access (OFDMA) ..... 1095
11.16 Single-carrier frequency-division multiple access (SC-FDMA) ..... 1096
11.17 Localized, interleaved and distributed subcarrier mapping ..... 1098
11.18 Comparison between OFDM, OFDMA and SC-FDMA ..... 1099
11.19 Discrete Fourier Transform (DFT) and its inverse (iDFT) ..... 1099
11.19.1 Overview of Fourier transform and its inverse ..... 1100
11.19.2 Overview of DFT and iDFT ..... 1100
11.19.3 Intuitive formulation of DFT and iDFT ..... 1102
11.19.4 Remarks on DFT and iDFT ..... 1104
11.19.5 Series formulation of DFT and iDFT ..... 1108
11.19.6 Phasors ..... 1111
11.19.7 Why complex values? ..... 1112
11.20 Pseudo-random sequences ..... 1113
11.20.1 Pseudo-random binary sequences ..... 1114

11.20.2 Low-PAPR type 1 complex sequences ..... 1114
11.20.3 Low-PAPR type 2 complex sequences ..... 1116
11.20.4 Zadoff-Chu complex sequences ..... 1117
11.20.5 Gold binary sequence and family ..... 1118
11.20.6 M-sequence (binary) ..... 1122
11.20.7 Binary sequence correlation ..... 1128
11.21 Passive lossless antennas ..... 1129
11.21.1 Properties of passive lossless antennas ..... 1129
11.21.2 Gain of an antenna ..... 1132
11.21.3 Isotropic antenna ..... 1133
11.21.4 Building a directional antenna ..... 1134
11.21.5 Radiation pattern of an antenna ..... 1135
11.21.6 Beamwidth of an antenna ..... 1135
11.21.7 Thought experiments on passive lossless antennas ..... 1136
11.21.8 Dipole antenna ..... 1138
11.22 Self-interference mitigation ..... 1141
11.23 Same-channel interference mitigation ..... 1142
12. Received signal distortion ..... 1143
12.1 Individual causes of received signal distortion ..... 1143
12.1.1 Absorption ..... 1143
12.1.2 Scattering ..... 1144
12.1.3 Refraction ..... 1144
12.1.4 Reflection ..... 1144
12.1.5 Spreading ..... 1145
12.1.6 Effective area ..... 1145
12.1.7 Time dispersion ..... 1146
12.1.8 Doppler ..... 1147
12.1.9 Diffraction ..... 1147
12.1.10 External Noise ..... 1148
12.1.11 Internal Noise ..... 1148
12.1.12 Thermal Noise and Scattering Noise ..... 1148
12.1.13 External interference ..... 1148
12.1.14 Self-interference ..... 1149
12.1.15 Radio frequency subsystem distortion ..... 1149
12.2 Combined causes of received signal distortion ..... 1149
12.2.1 Noise ..... 1149
12.2.2 Beam interference ..... 1150

12.2.3 Multi-path Interference ..... 1151
12.2.4 Inter-symbol Interference ..... 1151
12.2.5 Free-space path-loss ..... 1152
12.2.6 Attenuation ..... 1157
12.2.7 Fading ..... 1157
12.3 Differences between phenomena of received signal distortion ..... 1158
12.4 Causes of distortion phenomena ..... 1160
12.5 Atmospheric Attenuation and Refraction ..... 1161
13. 3GPP and Mobile Communications Standardization ..... 1164
13.1 Introduction to 3GPP and 3GPP standards ..... 1164
13.2 Evolution of Mobile Communications Generations ..... 1167
13.2.1 Main expected characteristics of 6G ..... 1167
13.2.2 Main characteristics of 5G (5G and 5.5G) ..... 1168
13.2.3 Main characteristics of LTE (4G) ..... 1169
13.2.4 Main characteristics of UMTS (3G, 3.5G and 3.75G) ..... 1171
13.2.5 Main characteristics of GSM (2G, 2.5G and 2.75G) ..... 1173
13.2.6 Main characteristics of Analog Mobile Telephony (1G) ..... 1174
13.3 Narrowband standards ..... 1174
14. Epilogue ..... 1177
14.1 Book contents ..... 1177
14.2 Acknowledgements ..... 1178
14.3 About the author ..... 1180
14.4 Book principles ..... 1181
14.5 Copyright ..... 1184
14.6 Bibliography ..... 1184
15. Acronyms, symbols and terms ..... 1186
16. Contents ..... 1204