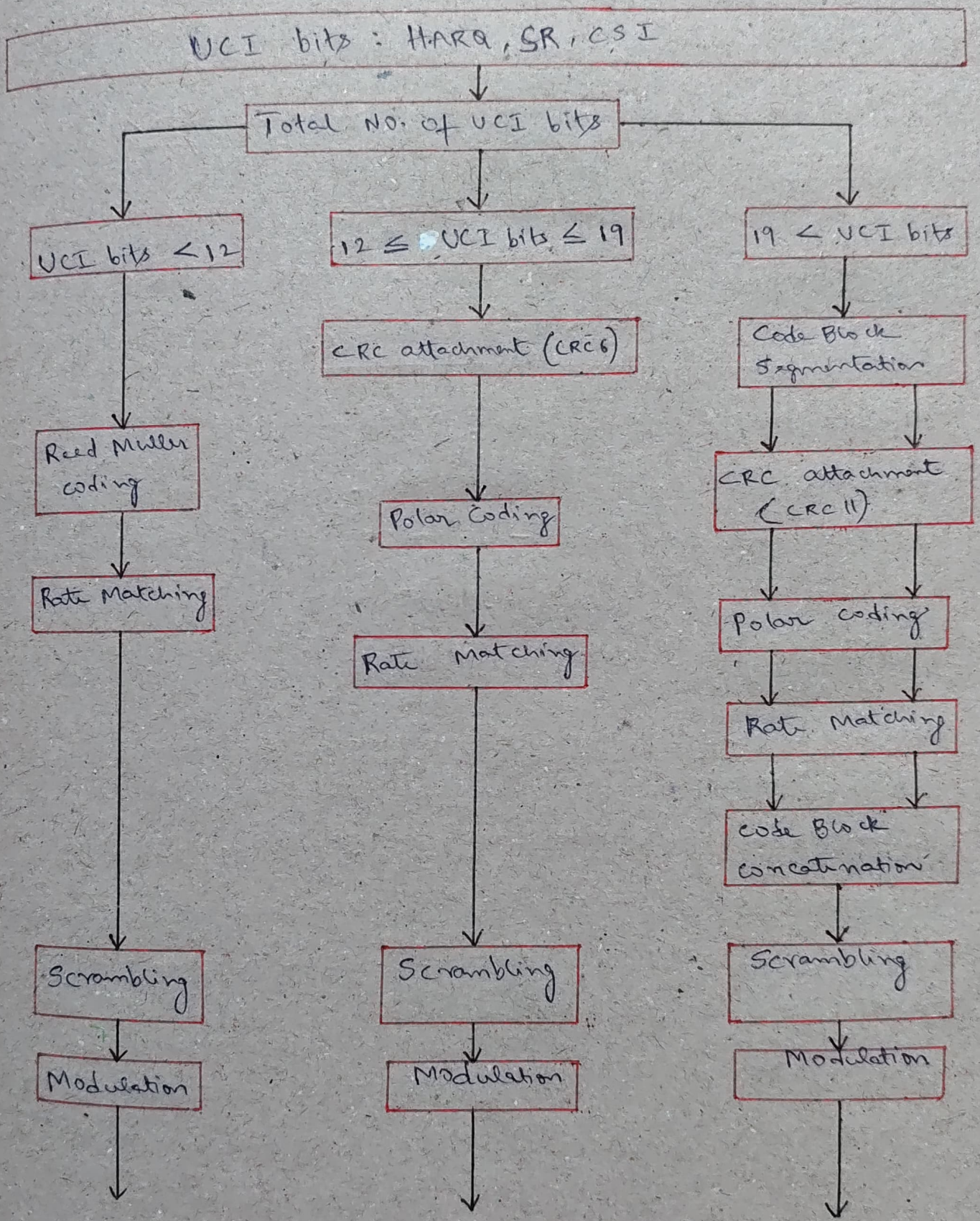
PUCCH Format 2/3/4 : Bit Processing :

HARQ / SR / CSI-Part1 / CSI-Part2



In this section, lets understand the bit processing for PUCCH Formats 2, 3 and 4. Since this part is same for all 3 formats, we'll discuss them combined.

We'll be discussing from the time when the information bits (UCI bits) are generated, to the Modulation.

As we know, PUCCH Formats 2, 3 and 4 carries more than 2 bits, and mainly carries HARQ, SR and CSI bits (Part 1 and Part 2).

Refer flow chart, based on the No. of bits in the UCI, they are handled differently.

- ⊙ No. of UCI bits - 3 to 11 (Small Block length)
- ⊙ No. of UCI bits - 12 to 19
- ⊙ No. of UCI bits - More than 20

Following modules are already discussed in detail in Essential Modules section. (CRC attachment, Polar coding, Rate Matching, Reed Muller coding, Scrambling, Modulation)

When the information bits (UCI bits) are from 3 to 11, then there is no CRC module, and the bits directly goes to Reed-Muller coding. Note: Here Repetition coding and Simplex coding does not come into picture since the PUCCH carries more than 2 bits.

After Reed-Muller coding, the bits goes to the Rate Matching module, and then they go to the Scrambling and Modulation (common for all 3 processing).

When the No. of information bits (UCI bits) are between 12 to 19, then we add CRC 6. Then the (UCI bits + CRC) goes through Polar coding, Rate Matching for Polar codes, Scrambling and Modulation.

When the NO. of UCI bits are more than 20, then they'll go through the Code Block Segmentation, where based on the no. of bits, the block of bits are divided into one or two code Blocks.

If $((A \geq 360$ and $E \geq 1088)$ or $(A \geq 1013))$

divide into 2 code blocks

else

Only one code block

Then we add CRC 11 to each of the code block. Then, each of them (Code Block + CRC11) will go through Polar coding and Rate Matching. Now, both the Rate Matched output will be concatenated. And the concatenated bits goes through Scrambling and Modulation.

We have two different types of CSI information: CSI Part 1 and CSI Part 2. The CSI bits are arranged with Priority. (High Priority to Low Priority)

High Priority
Low Priority

Based on the No. of bits we can transmit at a particular instance with the given resources, we pick the higher priority CSI bits for CSI - Part 1. After that, we pick the bits for CSI - Part 2.

UCI(s) for transmission on a PUCCH	UCI for encoding	Value of E_{UCI}
HARQ-ACK	HARQ-ACK	$E_{UCI} = E_{tot}$
HARQ-ACK, SR	HARQ-ACK, SR	$E_{UCI} = E_{tot}$
CSI (CSI not of two parts)	CSI	$E_{UCI} = E_{tot}$
HARQ-ACK, CSI (CSI not of two parts)	HARQ-ACK, CSI	$E_{UCI} = E_{tot}$
HARQ-ACK, SR, CSI (CSI not of two parts)	HARQ-ACK, SR, CSI	$E_{UCI} = E_{tot}$
CSI (CSI of two parts)	CSI part 1	$E_{UCI} = \min\left(E_{tot}, \left\lceil (O^{CSI\text{-}part1} + L)/R_{UCI}^{max}/Q_m \right\rceil \cdot Q_m\right)$
	CSI part 2	$E_{UCI} = E_{tot} - \min\left(E_{tot}, \left\lceil (O^{CSI\text{-}part1} + L)/R_{UCI}^{max}/Q_m \right\rceil \cdot Q_m\right)$
HARQ-ACK, CSI (CSI of two parts)	HARQ-ACK, CSI part 1	$E_{UCI} = \min\left(E_{tot}, \left\lceil (O^{ACK} + O^{CSI\text{-}part1} + L)/R_{UCI}^{max}/Q_m \right\rceil \cdot Q_m\right)$
	CSI part 2	$E_{UCI} = E_{tot} - \min\left(E_{tot}, \left\lceil (O^{ACK} + O^{CSI\text{-}part1} + L)/R_{UCI}^{max}/Q_m \right\rceil \cdot Q_m\right)$
HARQ-ACK, SR, CSI (CSI of two parts)	HARQ-ACK, SR, CSI part 1	$E_{UCI} = \min\left(E_{tot}, \left\lceil (O^{ACK} + O^{SR} + O^{CSI\text{-}part1} + L)/R_{UCI}^{max}/Q_m \right\rceil \cdot Q_m\right)$
	CSI part 2	$E_{UCI} = E_{tot} - \min\left(E_{tot}, \left\lceil (O^{ACK} + O^{SR} + O^{CSI\text{-}part1} + L)/R_{UCI}^{max}/Q_m \right\rceil \cdot Q_m\right)$

In this table, as we see, CSI - Part 1 also goes with HARQ - ACK and SR.

Table shows different possible combinations of information (UCI s) on PUCCH. (4)

* HARQ - ACK
* HARQ - ACK with SR
* CSI (single part)
* HARQ - ACK, CSI (single part)
* HARQ - ACK, SR, CSI (single part)
* CSI (Two parts)
* HARQ - ACK, CSI (Two parts)
* HARQ - ACK, SR, CSI (Two parts)

PUCCH format	Modulation order	
	QPSK	π/2-BPSK
PUCCH format 2	$16 \cdot N_{symb,UCI}^{PUCCH,2} \cdot N_{PRB}^{PUCCH,2}$	N/A
PUCCH format 3	$24 \cdot N_{symb,UCI}^{PUCCH,3} \cdot N_{PRB}^{PUCCH,3}$	$12 \cdot N_{symb,UCI}^{PUCCH,3} \cdot N_{PRB}^{PUCCH,3}$
PUCCH format 4	$24 \cdot N_{symb,UCI}^{PUCCH,4} / N_{SF}^{PUCCH,4}$	$12 \cdot N_{symb,UCI}^{PUCCH,4} / N_{SF}^{PUCCH,4}$

As we know, the Rate Matching output length is nothing but the no. of bits that can be transmitted on the available resources for Data.

For different PUCCH formats, as the Resource allocation varies, the No. of bits is calculated based on the different Modulation order, as shown in table above. Say for example, PUCCH Format 2 has only QPSK modulation. So, the total no. of bits that are transmitted over the air is given by.

$$16 * N_{symb,UCI}^{PUCCH,2} * N_{PRB}^{PUCCH,2}$$

$$\Rightarrow 16 * (1 \text{ or } 2) * (1 \text{ to } 16)$$

So, this becomes the Rate Matching output length. Similarly, for PUCCH format 3, the Rate Matching output length can be calculated as

$$12 * N_{symb,UCI}^{PUCCH,3} * N_{PRB}^{PUCCH,3}$$

∵ In PUCCH Format 3, the No. of Symbols given to UCI can take upto 14, here we consider 10 symbols are given to UCI and $\frac{\pi}{2}$-BPSK modulation.

$$\Rightarrow 12 * 10 * (1 \text{ to } 16)$$

This is how Rate Matching output length can be calculated, which is nothing but E_{tot}

Note: For information bits (UCI bits) transmission with CSI (Two parts), first we find out the No. of bits that are given to CSI - Part 1, say

$$E_{UCI} = \min\left(E_{tot}, \left\lceil \left(O^{CSI\text{-}part1} + L\right) / R_{UCI}^{max} / Q_m \right\rceil \cdot Q_m\right)$$

And, the remaining bits, after allocating to CSI - Part 1 are given to CSI - Part 2.

$$E_{UCI} = E_{tot} - \min\left(E_{tot}, \left\lceil \left(O^{CSI\text{-}part1} + L\right) / R_{UCI}^{max} / Q_m \right\rceil \cdot Q_m\right)$$

where, O^{ACK} → Bits given to HARQ-ACK

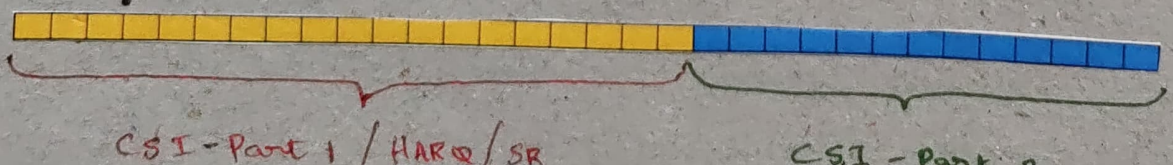
O^{SR} → Bits given to SR

L → CRC length

R_{UCI}^{max} → Maximum Code Rate (0.08 to 0.80)

Q_m → Modulation order (1 or 2 here)

maxCodeRate	Code rate r
0	0.08
1	0.15
2	0.25
3	0.35
4	0.45
5	0.60
6	0.80
7	Reserved



CSI - Part 1 / HARQ / SR

CSI - Part 2

Now, the Scrambling and Modulation modules are same across the 3 different cases (Refer flow chart).

$$c_{init} = n_{RNTI} \cdot 2^{15} + n_{ID}$$

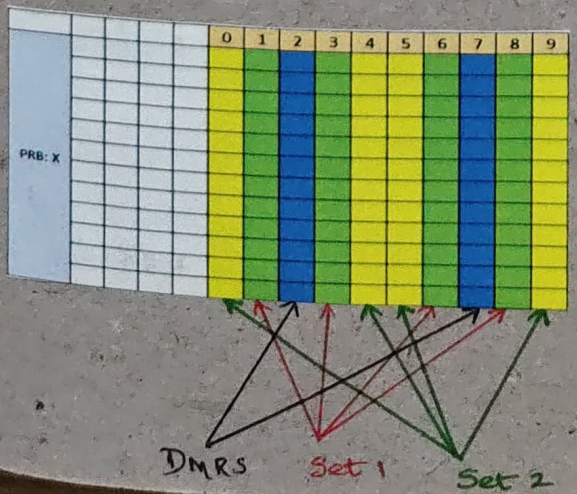
The Scrambling sequence is initialized with n_{RNTI} and n_{ID}. And we can have either QPSK or $\frac{\pi}{2}$ BPSK modulation.

Lets now understand, how CSI-Part 1 is given more priority than CSI-Part 2 during allocation.

Table 6.3.1.6-1: PUCCH DMRS and UCI symbols

PUCCH duration (symbols)	PUCCH DMRS symbol indices	Number of UCI symbol indices sets N_{UCI}^{set}	1st UCI symbol indices set $S_{UCI}^{(1)}$	2nd UCI symbol indices set $S_{UCI}^{(2)}$	3rd UCI symbol indices set $S_{UCI}^{(3)}$
4	{1}	2	{0,2}	{3}	-
4	{0,2}	1	{1,3}	-	-
5	{0, 3}	1	{1, 2, 4}	-	-
6	{1, 4}	1	{0, 2, 3, 5}	-	-
7	{1, 4}	2	{0, 2, 3, 5}	{6}	-
8	{1, 5}	2	{0, 2, 4, 6}	{3, 7}	-
9	{1, 6}	2	{0, 2, 5, 7}	{3, 4, 8}	-
10	{2, 7}	2	{1, 3, 6, 8}	{0, 4, 5, 9}	-
10	{1, 3, 6, 8}	1	{0,2,4,5,7,9}	-	-
11	{2, 7}	3	{1,3,6,8}	{0,4,5,9}	{10}
11	{1,3,6,9}	1	{0,2,4,5,7,8,10}	-	-
12	{2, 8}	3	{1,3,7,9}	{0,4,6,10}	{5, 11}
12	{1,4,7,10}	1	{0,2,3,5,6,8,9,11}	-	-
13	{2, 9}	3	{1,3,8,10}	{0,4,7,11}	{5,6,12}
13	{1,4,7,11}	2	{0,2,3,5,6,8,10,12}	{9}	-
14	{3, 10}	3	{2,4,9,11}	{1,5,8,12}	{0,6,7,13}
14	{1,5,8,12}	2	{0,2,4,6,7,9,11,13}	{3, 10}	-

As we see in this table, based on the No. of PUCCH symbols allocated, the UCI symbols are divided into 3 parts, leaving the DMRS Symbols.

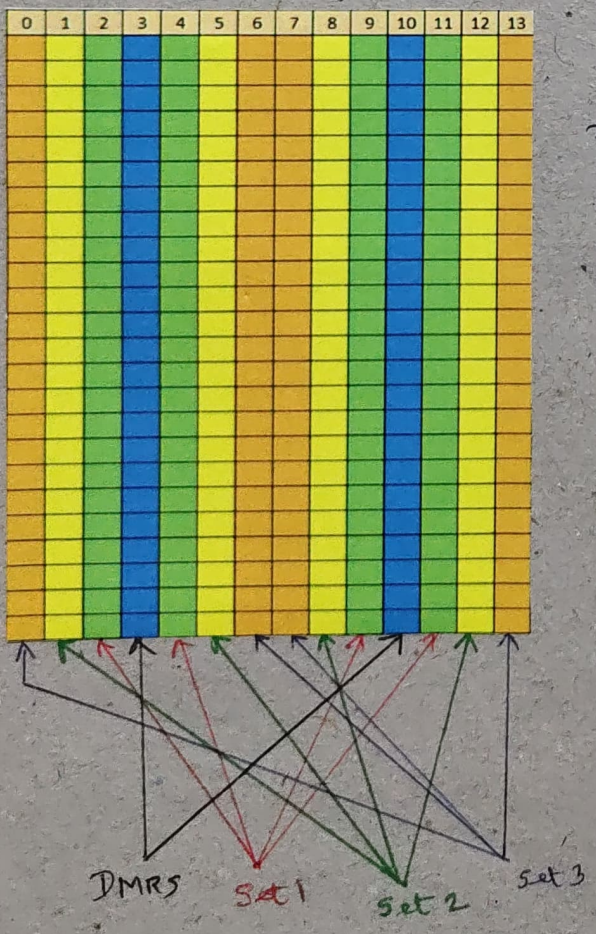


For an example, lets take this case. If the PUCCH duration is 10 symbols, and if the DMRS symbols are at {2,7} indices, then the symbols {1,3,6,8} which are close to DMRS belong to Set 1. And the remaining

symbols {0, 4, 5, 9} close to set 1, they belong to set 2. So, basically, the symbols adjacent to DMRS belong to set 1, the next adjacent symbols belong to set 2, and further adjacent symbols (far from DMRS) belong to set 3. This is how, Table 6.3.1.6-1 is organized.

Note that, Since the channel varies fast, the symbols that are close to DMRS will have better channel estimates compared to set 2 or set 3.

So, utilizing this, CSI-Part 1 and CSI-Part 2 are mapped differently. Even before the data is mapped, they are placed differently after the Rate Matching.



Lets see another example, This is for PUCCH duration of 14 symbols, and there are 3 sets.

Symbols {3, 10} → DMRS

Symbols {2, 4, 9, 11} → Set 1

Symbols {1, 5, 8, 12} → Set 2

Symbols {0, 6, 7, 13} → Set 3

This is how 3 sets are define in Table 6.3.1.6-1.

Find the smallest $j > 0$ such that $\left(\sum_{i=1}^{j} N_{UCI}^{(i)}\right) \cdot N_{UCI}^{symbol} \cdot Q_m \geq G^{(1)}$.

Set $n_1 = 0$;

Set $n_2 = 0$;

Set $\bar{N}_{UCI}^{symbol} = \left\lfloor \left(G^{(1)} - \left(\sum_{i=1}^{j-1} N_{UCI}^{(i)}\right) \cdot N_{UCI}^{symbol} \cdot Q_m \right) \Big/ \left(N_{UCI}^{(j)} \cdot Q_m\right) \right\rfloor$;

Set $M = \mathrm{mod}\left(\left(G^{(1)} - \left(\sum_{i=1}^{j-1} N_{UCI}^{(i)}\right) \cdot N_{UCI}^{symbol} \cdot Q_m \right) \Big/ Q_m, N_{UCI}^{(j)} \right)$;

for $l = 0$ to $N_{symb,UCI}^{PUCCH} - 1$

if $s_l \in \bigcup_{i=1}^{j-1} S_{UCI}^{(i)}$

for $k = 0$ to $N_{UCI}^{symbol} - 1$

for $v = 0$ to $Q_m - 1$

$\bar{g}_{l,k,v} = g_{n_1}^{(1)}$;

$n_1 = n_1 + 1$;

end for

end for

elseif $s_l \in S_{UCI}^{(j)}$

if $M > 0$

$\gamma = 1$;

else

$\gamma = 0$;

end if

$M = M - 1$;

for $k = 0$ to $\bar{N}_{UCI}^{symbol} + \gamma - 1$

for $v = 0$ to $Q_m - 1$

$\bar{g}_{l,k,v} = g_{n_1}^{(1)}$;

$n_1 = n_1 + 1$;

end for

end for

for $k = \bar{N}_{UCI}^{symbol} + \gamma$ to $N_{UCI}^{symbol} - 1$

for $v = 0$ to $Q_m - 1$

$\bar{g}_{l,k,v} = g_{n_2}^{(2)}$;

$n_2 = n_2 + 1$;

end for

end for

else

for $k = 0$ to $N_{UCI}^{symbol} - 1$

for $v = 0$ to $Q_m - 1$

$\bar{g}_{l,k,v} = g_{n_2}^{(2)}$;

In 3GPP specifications, the procedure is defined like this. (complicated math). Lets take an example and understand this process.

Consider PUCCH Format 3, and 14 symbols allocated in total, and 2 PRBs.

Out of 14, two symbols {3, 10} are given for DMRS. So, we left with 12 symbols.

In one symbol, we have 24 REs. So, in 12 symbols, we have $12 * 24 = 288$ REs. where UCI can go.

If we assume $\frac{\pi}{2}$ BPSK, then we have 288 bits after Rate Matching.

Lets consider a case, where

$G^{(1)} \rightarrow$ HARQ + CSI Part 1
$= 64$ bits

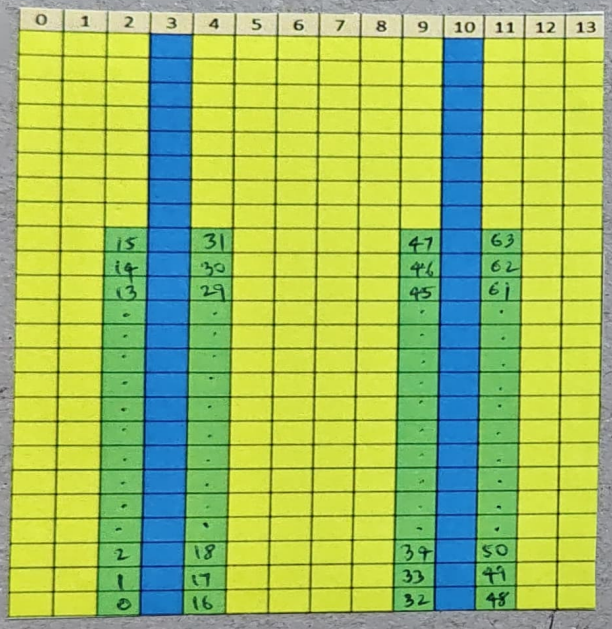
$G^{(2)} \rightarrow$ CSI Part 2
$= 224$ bits

So, in this case, we prioritize $G^{(1)}$ (i.e) HARQ + CSI Part 1 and map them to the symbols that are close to DMRS. (i.e) first we map it to Set 1

Here, we have 64 bits for $G^{(1)}$. And we have

4 (Symbols adj to DMRS Symbols) * 24 (No. of REs per symbol) = 96 bits.

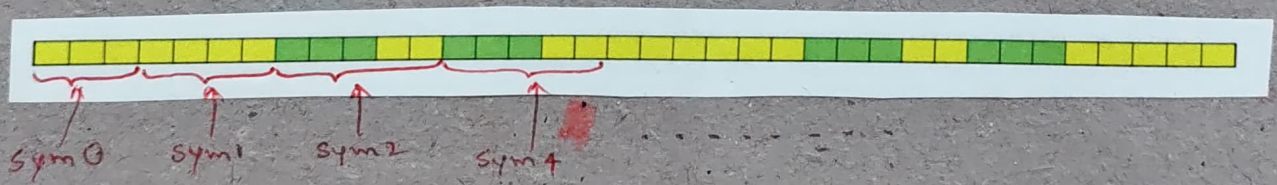
So, we divide 64 bits equally in all 4 symbols adjacent to DMRS symbols. (a) One symbol will take 16 bits. as shown below.



The $G^{(1)}$ bits are arranged as shown in this figure. (a) First we take the Set 1 where we map HARQ / SR / CSI - Part 1. And then, at the remaining locations, we map CSI - Part 2.

So, basically, this is how the bits are arranged.

And, after this, the final sequence looks something like this. (Arranging serially).



This is the way the Rate Matched output bits are arranged in sequence. This sequence goes to scrambling, and then modulation, Post which the Resource Mapping is done.

This is all about PUCCH Format 2/3/4 bit processing.

PUCCH Format-2 : DMRS

DMRS for PUCCH Format 2 is very straight forward. It is generated based on the PN sequence. We've already seen how PN sequence is generated using different C_{init} values. For Format 2 DMRS, C_{init} is defined as follows.

$$c_{init} = (2^{17}(N_{symb}^{slot} n_{s,f}^{\mu} + l + 1)(2N_{ID}^{0} + 1) + 2N_{ID}^{0}) \bmod 2^{31}$$

N_{symb}^{slot} → No. of symbols in a slot
$= 14$.

$n_{s,f}^{\mu}$ → Slot number inside a frame.

l → OFDM Symbol number

So, basically, $\left(N_{symb}^{slot} n_{s,f}^{\mu} + l + 1\right)$ takes care of generating a different index inside a Frame. (i.e.) $(14 * \text{Slot index} + \text{Symbol index})$

N_{ID}^{0} → Scrambling ID derived from PUSCH.
If it is not given, then $N_{ID}^{0} = \text{Cell\_id}$

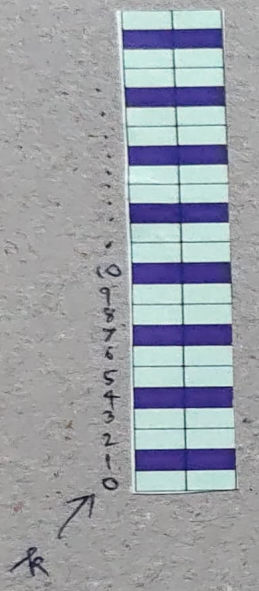
This means, for different cell, different DMRS sequence is generated. This can be a useful thing when, for different neighbouring cells, PUCCH DMRS scheduled at the same time and frequency.

With C_{init} value, we generate PN sequence, and we do QPSK modulation, to generate final sequence.

$$r_l(m) = \frac{1}{\sqrt{2}}(1 - 2c(2m)) + j\frac{1}{\sqrt{2}}(1 - 2c(2m+1))$$
$$m = 0, 1, \ldots$$

$$a_{k,l}^{(p,\mu)} = \beta_{\text{PUCCH.2}} r_l(m)$$
$$k = 3m + 1$$

And, this is how the final DMRS sequence $r_l(m)$ is mapped to PUCCH Format 2 resources. This is pretty straight forward.



As we see, this is how the allocation looks like.

for $m = 0$, $k = 1$

$m = 1$, $k = 4$

$m = 2$, $k = 7$

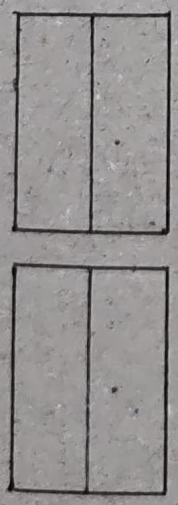
The DMRS and DATA are interleaved in PUCCH Format 2.

So, for each OFDM symbol, we generate unique PN sequence using unique Cinit values. Using which we have unique QPSK modulated DMRS sequence. And then, this unique DMRS sequence is mapped to the REs (DMRS REs)

PUCCH Format 2 : Signal Generation, Receiver and More

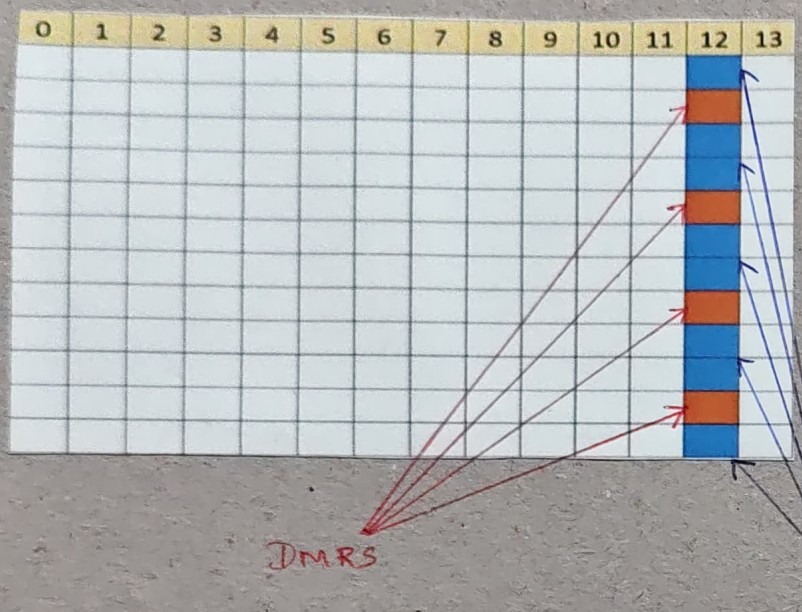
Lets understand PUCCH Format 2 in detail. We've already seen the bit processing of Format 2 and DMRS. In the Bit processing, we've seen different modules between UCI bits to Modulation (Refer flow chart), and understood how the bits are processed. In PUCCH Format 2, Only QPSK modulation is being used. It is a short Format (ii) it takes less number of Symbols. PUCCH Format 2 either takes 1 OFDM symbol or 2 OFDM Symbols.

There is no UE Multiplexing defined in Format 2 like we have in Format 0 and Format 1. This means, if there are 2 UEs that want to send User information they have to send at different locations using F2.



So far, we've seen the very basics of Resource allocation. Lets go in detail and see how differently the PRBs are allocated.

PUCCH Format 2 can take 1 or 2 symbols and 1 to 16 PRBs. We've also seen the Coding Rates ranges from 0.08 to 0.8.

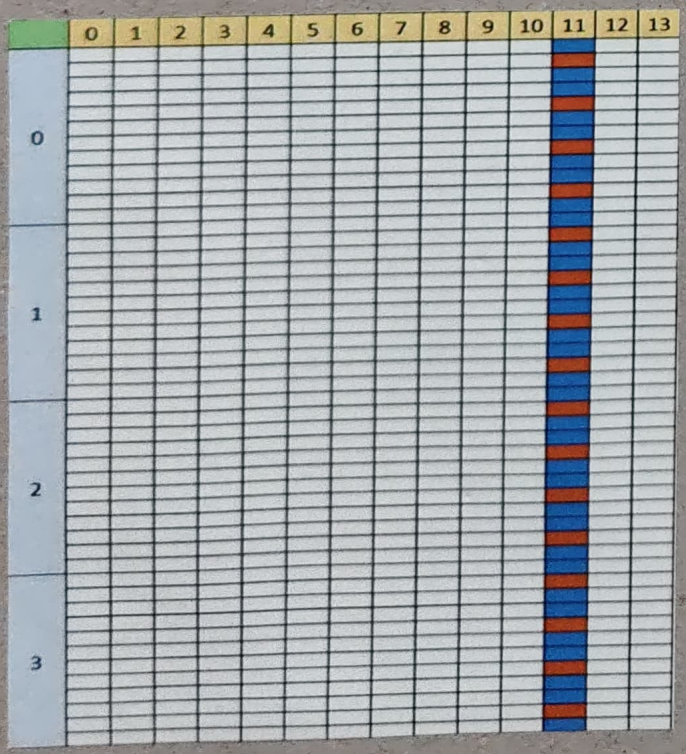


→ This is the smallest allocation using PUCCH Format 2

→ having 1 Symbol and 1 PRB.

→ In 1 Symbol and 1 PRB, we have 8 DATA REs.

→ So, the smallest allocation will be 8 DATA REs, with QPSK modulation, we can send 16 bits using PUCCH format 2

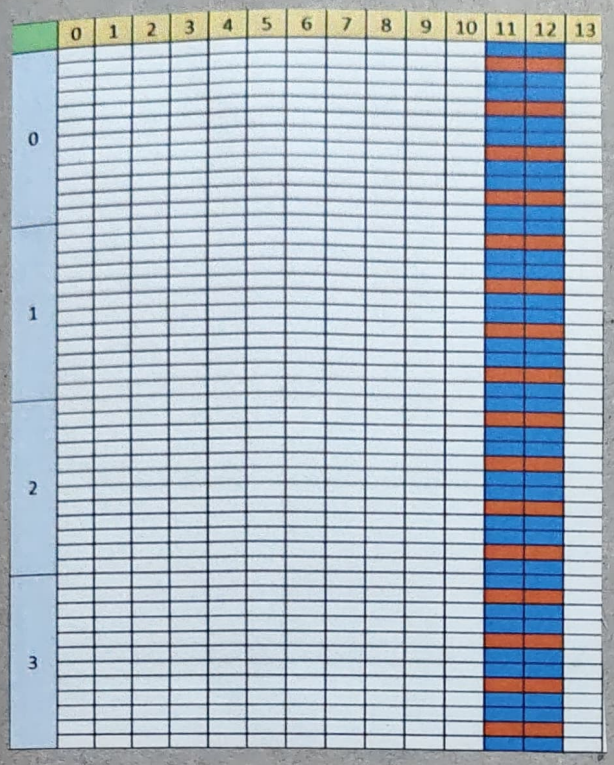


→ It can take upto 16 PRBs. Here, we have 4 PRB allocation.

→ DMRS locations are fixed

→ DMRS and DATA are interleaved

→ Every PRB having 8 DATA REs and 4 DMRS REs.



→ Similarly, here there are 2 Symbols and 4 PRBs.

→ Here, total No. of DATA REs

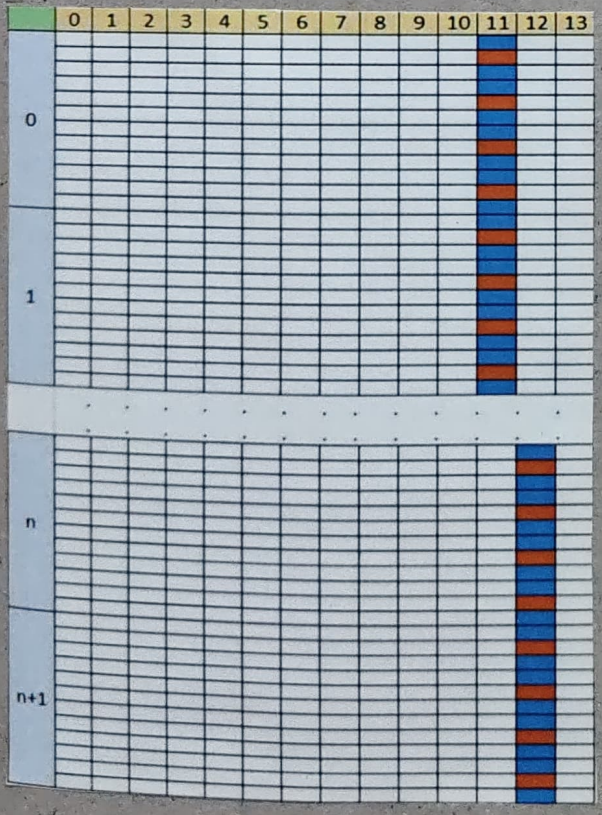
$$\begin{pmatrix}8\ \text{DATA REs in a PRB}\end{pmatrix} * \begin{pmatrix}4\\ \text{PRBs}\end{pmatrix} * \begin{pmatrix}2\\ \text{Symbols}\end{pmatrix}$$

$$= 64 \text{ DATA REs.}$$

→ So, total 128 bits going over the Air.

→ Note that, these 128 bits are not UCI bits. These are after Encoding & Rate Matching.

→ To find out the UCI bits, we can use the code Rate that was applied.

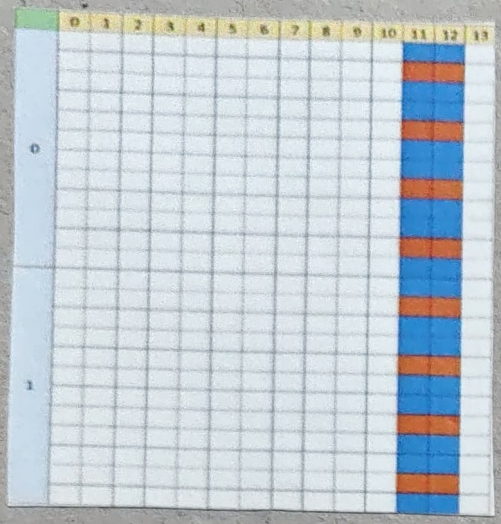


→ PUCCH Format 2 also supports Frequency Hopping.

→ If the Freq. Hopping is enabled, then the second symbol will go at different frequency location.

→ In this example,

- Start PRB = 0
- 2nd hop Start PRB } = n = 100 (say)
- Total No. of PRBs } = 2 (not 4).



→ If the Freq. Hopping is disabled, then it'll look something like this.

→ Where, for both the symbols, Start PRB = 0

As we know, the maximum allocation is 16 PRB, and 2 Symbols. So, the maximum number of DATA RE, for PUCCH Format 2 is 256 RE. (ie)

$$\begin{pmatrix}8 \text{ DATA RE,} \\ \text{in a PRB}\end{pmatrix} * \begin{pmatrix}16 \text{ PRB,}\end{pmatrix} * \begin{pmatrix}2 \text{ Symbol}\end{pmatrix} = 256 \text{ RE,}$$

Using QPSK, maximum of 512 bits can be transmitted. Now, using the different code rates, we can allocate to different UE, based on the Coverage or Channel Conditions. That means,

if there is a UE with Bad Coverage,
then this UE can have Lower Code rate and higher allocation.

(ie), if there is a UE with Good coverage,
then this UE can have Higher code rate and Lower allocation.

So, Code Rate along with the allocation can be used by the UE, to send the UCI information, based on its Channel conditions.

Now, let's understand how the Receiver works. WKT, the DMRS and DATA are interleaved in the resource allocation.

At the receiver, after Resource demapping, first we take the DMRS and estimate the channel. And then, this channel estimates are applied to the data, for equalization.

The DMRS REs can be used together (Averaged) to form Single Channel Estimate, to get better channel estimates per PRB. (implementation specific).

After Equalization, we do Soft Demodulation to get the Log Likelihood Ratios (LLRs). With those LLRs, we do descrambling (WKT, PN sequence is a function of Symbol number and Scrambling ID). Then we take a decision that how many UCI bits were transmitted (3 to 11 / 12 to 19 / More than 20), based on which we pick the respective receiver chain. Say for example, if UCI bits < 12, then we do de-Rate Matching, followed by Reed Muller decoder, and we'll have the UCI bits. If $12 \leq$ UCI bits ≤ 19, then we do de-Rate Matching, followed by Polar decoder, followed by CRC removal, and we'll have the UCI bits. And, if UCI bits > 19, first we'll check whether Code Block Segmentation was done or not at the transmitter side. If this segmentation was done, then at the receiver side, we'll again

do the Segmentation of the de-Scrambling bits, and for each segment, we'll do deRate Matching, Polar decoding, CRC removal, and combine the code block segments to get the find UCI bits.

Note that, PUCCH1 Format 2 does not transmit CSI Part 2 (Something that we've seen in Bit processing). It only carries HARQ/SR/CSI Part 1 (Any combination)

PUCCH Format 3/4 : DMRS

Lets understand DMRS for PUCCH Format 3 and 4. Since the DMRS generation and Mapping part is very similar for Format 3 and 4, we'll understand both of them together.

DMRS for Format 3 and 4 is based on ZC sequence.

$$r_l(m) = r_{u,v}^{(\alpha,\delta)}(m)$$
$$m = 0, 1, \ldots, M_{sc}^{PUCCH,s} - 1$$

Here, as we see, $r_{u,v}^{(\alpha,\delta)}(m)$ is the ZC sequence and it is assigned to $r_l(m)$. And 'm' ranges from 0 to $M_{sc}^{PUCCH,s} - 1$.

$M_{sc}^{PUCCH,s}$ ⟶ No. of Subcarriers in a given symbol in PUCCH, for that Format.

$$M_{sc}^{PUCCH,s} = M_{RB}^{PUCCH,s} N_{sc}^{RB}$$

Here, as we see, the No. of Subcarriers depends on the No. of PRBs allocated ($M_{RB}^{PUCCH,s}$) and the No. of Subcarriers in a PRB (N_{sc}^{RB}).

$N_{sc}^{RB} = 12$ and $M_{RB}^{PUCCH,s}$ is defined as below.

$$M_{RB}^{PUCCH,s} = \begin{cases} 2^{\alpha_2} \cdot 3^{\alpha_3} \cdot 5^{\alpha_5} & \text{for PUCCH format 3} \\ 1 & \text{for PUCCH format 4} \end{cases}$$

where, α_2, α_3 and α_5 are non-negative integers (≥ 0)

For Format 4, the No. of PRBs allocated is Only one. So, the No. of Subcarriers in a given symbol in PUCCH Format 4 is $M_{sc}^{PUCCH,s} = 1 \times 12 = 12$

For Format 3, the No. of PRBs allocated is

$$M_{RB}^{PUCCH,s} = 2^{\alpha_2} \cdot 3^{\alpha_3} \cdot 5^{\alpha_5}$$

$$= \begin{cases} 1 & , \text{if } \alpha_2 = \alpha_3 = \alpha_5 = 0 \\ 2 & , \text{if } \alpha_2 = 1,\ \alpha_3 = \alpha_5 = 0 \\ 3 & , \text{if } \alpha_2 = 0,\ \alpha_3 = 1,\ \alpha_5 = 0 \\ 4, 5, 6, 8, 9, 10, 12, 15, 16 \end{cases}$$

Since, maximum 16 PRBs can be allocated for F3. And, the No. of subcarriers in a given symbol in PUCCH Format 3 is

$$M_{sc}^{PUCCH,s} = 12, 24, 36, 48, 60, 72, 96, 108, 120, 144, 180, 192.$$

And, this will be the length of the ZC sequence that we need to generate.

Now, $r_{u,v}^{(\alpha,\delta)}(m)$ is defined as follows.

$$r_{u,v}^{(\alpha,\delta)}(n) = e^{j\alpha n} \bar{r}_{u,v}(n), \quad 0 \le n < M_{ZC}$$

We've already seen this equation in other channels as well. We'll go over these equations again w.r.t F3 and F4 DMRS as well.

$e^{j\alpha n}$ → Provides cyclic shift to the ZC sequence generated.

$\bar{r}_{u,v}(n)$ → ZC Sequence

Here, α is defined as follows,

$$\alpha_l = \frac{2\pi}{N_{sc}^{RB}} \left(\left(m_0 + n_{cs}(n_{s,f}^{\mu}, l+l') \right) \bmod N_{sc}^{RB} \right)$$

where, $N_{SC}^{RB} = 12$

$n_{cs}(i)$ → cyclic shift, which is a function of slot number and symbol number.

$n_{s,f}^{\mu}$ → slot number

l → symbol number

$n_{cs}(n_{s,f}^{\mu}, l)$ is defined as follows.

$$n_{cs}(n_{s,f}^{\mu}, l) = \sum_{m=0}^{7} 2^m \, c(8N_{symb}^{slot} n_{s,f}^{\mu} + 8l + m)$$

which is a function of PN sequence $c(\cdot)$, which takes input as No. of symbols in a slot $\left(N_{symb}^{slot}\right)$, slot number $\left(n_{s,f}^{\mu}\right)$, Symbol number (l) and index value $(m = 0 \text{ to } 7)$.

So, $n_{cs}(n_{s,f}^{\mu}, l)$ computation is basically generate the PN sequence (0's and 1's), multiply them with 2^m (just for converting 8 generated bits to integer). So, n_{cs} takes values between 0 to 255.

This computed n_{cs} value is used in α_l computation.

For F3, $m_0 = 0$.

For F4, m_0 is defined in the below Table.

Table 6.4.1.3.3.1-1: Cyclic shift index for PUCCH format 4.

Orthogonal sequence index l	Cyclic shift index m_0	
	$N_{SF}^{PUCCH,4} = 2$	$N_{SF}^{PUCCH,4} = 4$
0	0	0
1	6	6
2	-	3
3	-	9

So, for F4, we have a parameter called Spreading factor $\left(N_{SF}^{PUCCH,4}\right)$. (We'll see about this in next section)

So, based on the Spreading factor value (2 or 4), and based on the sequence index (0/1/2/3), we pick the m_0 value in F_4. Possible m_0 values in F_4 = 0/6/3/9.

So, based on this m_0 and n_{cs} values, α_l is calculated.

$$\alpha_l = \frac{2\pi}{12} \quad (0 \text{ to } 11)$$

This is how the cyclic shift is provided to the ZC sequence. (ii)

$$r_{u,v}^{(\alpha,\delta)}(n) = e^{j\alpha n} \cdot \bar{r}_{u,v}(n)$$

Note:

Two sequence with different cyclic shift are uncorrelated with each other.

We have two more parameters u and v.

u → provides Group hopping (0 to 29)

v → provides Sequence hopping

$$u = (f_{gh} + f_{ss}) \bmod 30$$

Now, there are 3 different scenarios depending on, which hopping is enabled.

① When both Group hopping and Sequence hopping are not enabled.

$$f_{gh} = 0$$
$$f_{ss} = n_{ID} \bmod 30$$
$$v = 0$$

where, n_{ID} is the Scrambling ID. (can also be cell ID)

② In this case, Group hopping is enabled, and Sequence hopping is disabled.

$$f_{gh} = \left(\sum_{m=0}^{7} 2^m c\left(8\left(2n_{s,f}^{\mu} + n_{hop}\right) + m\right)\right) \bmod 30$$
$$f_{ss} = n_{ID} \bmod 30$$
$$v = 0$$

where, $c(\cdot)$ → PN sequence, taking inputs as slot number $(n_{s,f}^{\mu})$ and Hopping ID (n_{hop})

- n_{hop} = 0, if it is first hop
  1, for second hop.

If intra-slot frequency hopping is enabled, then n_{hop} will take two values {0, 1}.

This PN sequence is initialized with $c_{init} = \lfloor n_{ID}/30 \rfloor$

③ When Sequence hopping is enabled, and Group hopping is disabled.

$$f_{gh} = 0$$
$$f_{ss} = n_{ID} \bmod 30$$
$$v = c\left(2n_{s,f}^{\mu} + n_{hop}\right)$$

where, v → function of PN sequence, taking inputs as slot number $(n_{s,f}^{\mu})$ and Hopping ID (n_{hop}).

This PN sequence is initialized with

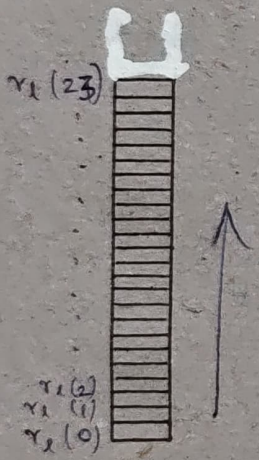
$$c_{init} = 2^5 \lfloor n_{ID}/30 \rfloor + (n_{ID} \bmod 30)$$

where, n_{ID} is the Scrambling ID (can be call ID also). Hence, different sequence hop is generated for different cell / Scrambling ID.

So, based on the different values of u and v, different cyclic shifts are provided to the ZC sequence and we end up having $r_{u,v}^{(\alpha,\delta)}(n)$. ($\delta$ is always 0 for this DMRS).

This way, we've generated sequence $r_l(m)$ of length $M_{sc}^{PUCCH,s}$. And, this sequence is mapped to resources in a very straight forward manner.

$$a_{k,l}^{(p,\mu)} = \beta_{PUCCH,s} \cdot r_l(m).$$
$$m = 0, 1, \ldots, M_{sc}^{PUCCH,s} - 1$$



Now, coming to the DMRS locations in PUCCH F3 and F4, shown in below table.

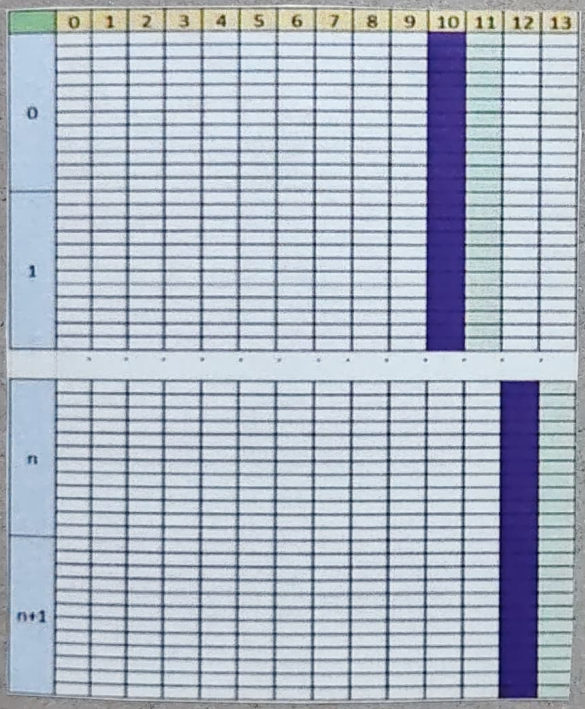
PUCCH length (Symbols)	DM-RS position / within PUCCH span			
	No additional DM-RS		Additional DM-RS	
	No hopping	Hopping	No hopping	Hopping
4	1	0, 2	1	0, 2
5	0, 3		0, 3	
6	1, 4		1, 4	
7	1, 4		1, 4	
8	1, 5		1, 5	
9	1, 6		1, 6	
10	2, 7		1, 3, 6, 8	
11	2, 7		1, 3, 6, 9	
12	2, 8		1, 4, 7, 10	
13	2, 9		1, 4, 7, 11	
14	3, 10		1, 5, 8, 12	

As we know, PUCCH F3 and F4 takes minimum of 4 Symbols and upto maximum of 14 Symbols.
And, PUCCH F3 can take 1 to 16 PRBs,
PUCCH F4 can take only 1 PRB.

→ In case of PUCCH length = 4 symbols where Hopping is disabled, the DMRS position within the PUCCH span is 1

→ This case is applicable for both F3 and F4. In F3, we can have many PRBs, and in F4, we can have only one PRB.

→ In this case, there are 2 PRBs.



→ If the Hopping is enabled, then One DMRS symbol alone wont be enough

→ We need atleast one DMRS symbol on each hop, otherwise we wont be able to do the channel estimation.

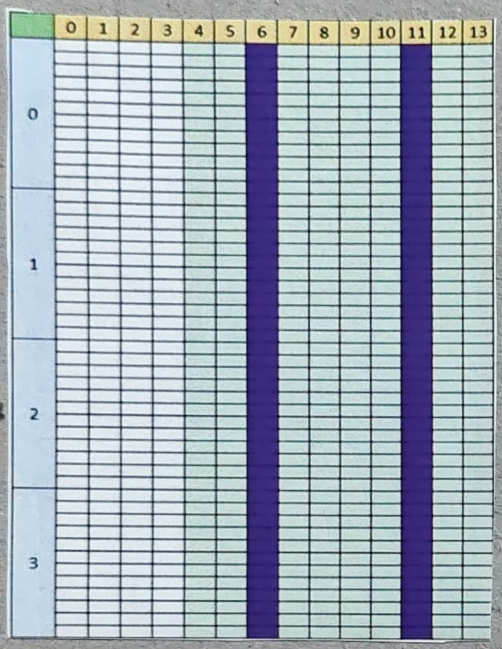
→ This is why, in the Hopping enabled case, we have 2 DMRS symbols.

→ In this case, in the first hop we have DMRS at position 0. In the second hop we have DMRS at position 2.

→ In positions 1 and 3 here, we have PUCCH data.

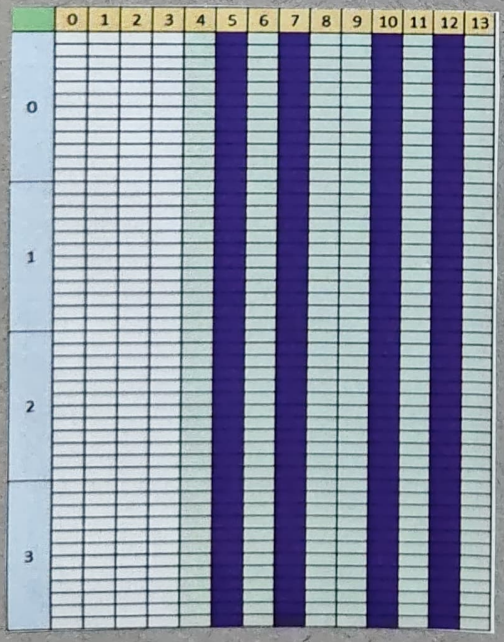
In all the cases (Refer Table), (PUCCH length = 5 to 14 symbols), the DMRS position for Hopping and without Hopping are same. From PUCCH length = 10 to 14 symbols, we can have additional DMRS also.

The Additional DMRS helps to provide better channel estimates and better coverage. When the UE is moving fast and the channel is time varying, then having more number of DMRS symbols helps in better channel estimates.



→ In this case, there are 2 DMRS symbols. There is no additional DMRS.

→ When we have additional DMRS, two more DMRS symbols are added. In this case, the channel Estimates will be better when the channel is varying fast in time.

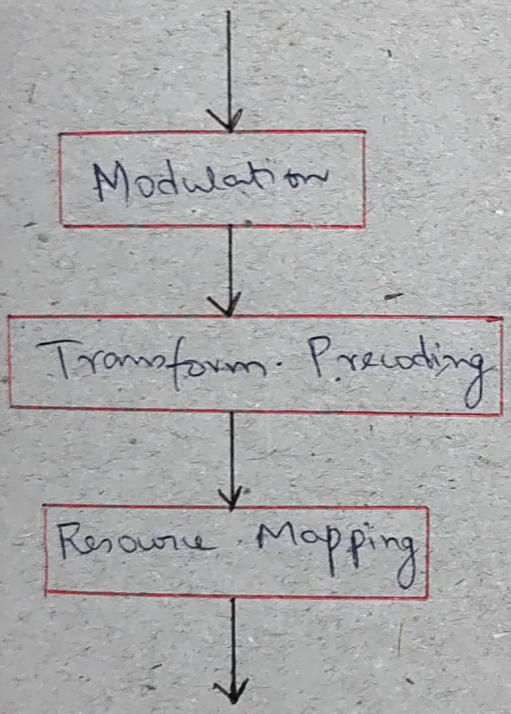


→ In frequency, anyway we have continuous DMRS at every subcarrier. So even if the channel is varying fast in frequency, we take care of it.

→ But, when the channel is varying fast in time, then we take care of it by adding additional DMRS symbols.

PUCCH Format 3 : Signal Generation, Receiver and More

In bit processing part, we've seen the UCI bits going through CRC, Encoding, Rate Matching, Scrambling and Modulation. Now, we'll understand what happens after that and how the bits are processed.



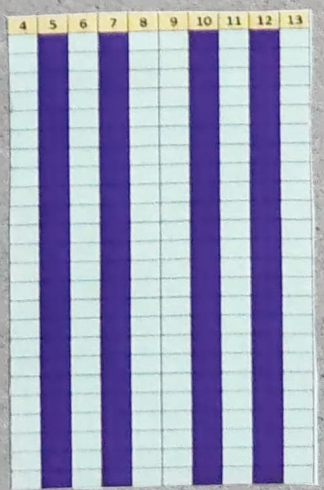
In PUCCH F3, both $\frac{\pi}{2}$ BPSK and QPSK are supported.

After the bits are modulated, they go through Transform Precoding.

Transform Precoding is used to reduce PAPR.

PAPR is the Peak to Average Power Ratio, which makes sure that the Power across the subcarriers does not vary much. So, the Transform Precoding is performed, by taking the DFT of the data mapped to a particular symbol.

As we know, the modulated samples are mapped to the resource, Symbol-by-Symbol. Say first on Symbol 4, then on Symbol 6, then on Symbols 8 and 9, and then on Symbols 11 and 13.



So, in Transform Precoding, we pick Symbol-by-Symbol data and do DFT of it. For example, pick symbol 4 data and find DFT of it. Similarly, pick symbol 6 data and find DFT of it, and so on.

$$z(l \cdot M_{sc}^{PUCCH,s} + k) = \frac{1}{\sqrt{M_{sc}^{PUCCH,s}}} \sum_{m=0}^{M_{sc}^{PUCCH,s}-1} y(l \cdot M_{sc}^{PUCCH,s} + m)\, e^{-j\frac{2\pi mk}{M_{sc}^{PUCCH,s}}}$$

$$k = 0, \ldots, M_{sc}^{PUCCH,s} - 1$$

$$l = 0, \ldots, \left(N_{SF}^{PUCCH,s} M_{symb} / M_{sc}^{PUCCH,s}\right) - 1$$

where, considering our example,

$$N_{SF}^{PUCCH,3} = 1$$

$$M_{sc}^{PUCCH,s} = 24$$

So, for Symbol index 0 (i) $l = 0$, the DFT is

$$z(k) = \frac{1}{\sqrt{24}} \sum_{m=0}^{23} y(m)\, e^{-j\frac{2\pi mk}{24}}$$

(ii) the modulated samples that are mapped on the 24 REs of Symbol index 0, goes through the DFT for Transform Precoding, to reduce the PAPR in PUCCH Format 3.

Similarly, for the next Symbol index (i) $l = 1$, the DFT is

$$z(24 + k) = \frac{1}{\sqrt{24}} \sum_{m=0}^{23} y(24 + m)\, e^{-j\frac{2\pi mk}{24}}$$

So, in total, the No. of Subcarriers $M_{sc}^{PUCCH,s}$ is calculated based on No. of PRBs, allocated in PUCCH Format 3 multiplied by the No. of subcarriers in a PRB.

$$M_{sc}^{PUCCH,s} = M_{RB}^{PUCCH,s} N_{sc}^{RB}$$

So, if the No. of PRBs allocated ($M_{RB}^{PUCCH,3}$) is 10, and the No. of Subcarriers in a PRB (N_{SC}^{RB}) is 12, then the total No. of Subcarriers,

$$M_{SC}^{PUCCH,3} = 10 * 12 = 120.$$

And we take the DFT of 120 modulated samples. This is about the Transform Precoding.

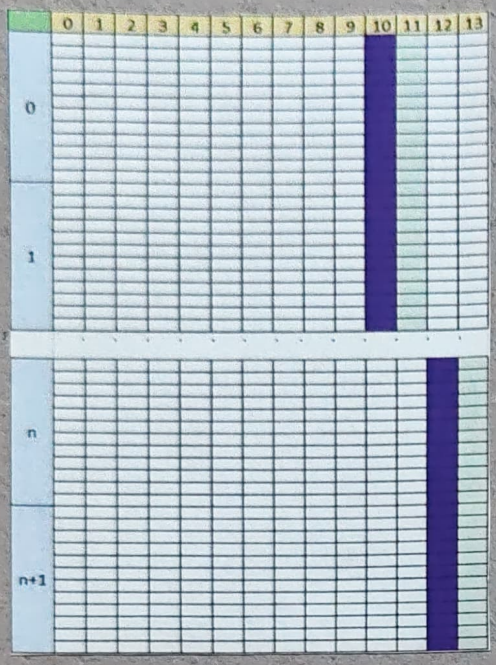
Then, the Transform Precoding data goes through the Resource Mapper module. The Resource Mapper module maps the Transform Precoding data in increasing order of the Symbols. (first it maps on Symbol 4, then on Symbol 6, then on Symbols 8 and 9, and so on.

How do we know the Symbol location in Format 3?

Table 6.4.1.3.3.2-1: DM-RS positions for PUCCH format 3 and 4.

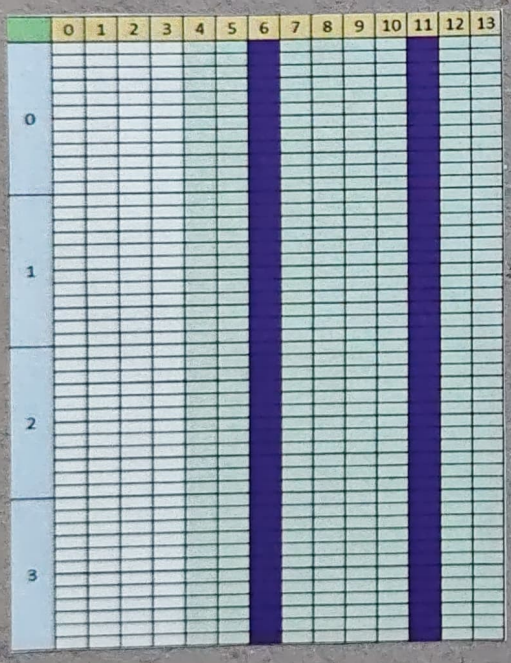
PUCCH length (Symbols)	DM-RS position / within PUCCH span			
	No additional DM-RS		Additional DM-RS	
	No hopping	Hopping	No hopping	Hopping
4	1	0, 2	1	0, 2
5	0, 3		0, 3	
6	1, 4		1, 4	
7	1, 4		1, 4	
8	1, 5		1, 5	
9	1, 6		1, 6	
10	2, 7		1, 3, 6, 8	
11	2, 7		1, 3, 6, 9	
12	2, 8		1, 4, 7, 10	
13	2, 9		1, 4, 7, 11	
14	3, 10		1, 5, 8, 12	

We've seen this DMRS position table. So, wherever the DMRS is not present, there the Transform Precoding data will be mapped. Say for example, PUCCH length = 14 OFDM Symbols are allocated to PUCCH Format 3, there is no Additional DMRS, then Symbol indices 3 and 10 will be used for DMRS, and the remaining 12 Symbols will be used for DATA. If Additional DMRS is configured, then 4 Symbols will be for DMRS, and 10 Symbols for DATA.



→ This is the case when Frequency hopping is enabled.

→ First we'll map the Transform Precoding data on Symbol 11, then we'll map on Symbol 13

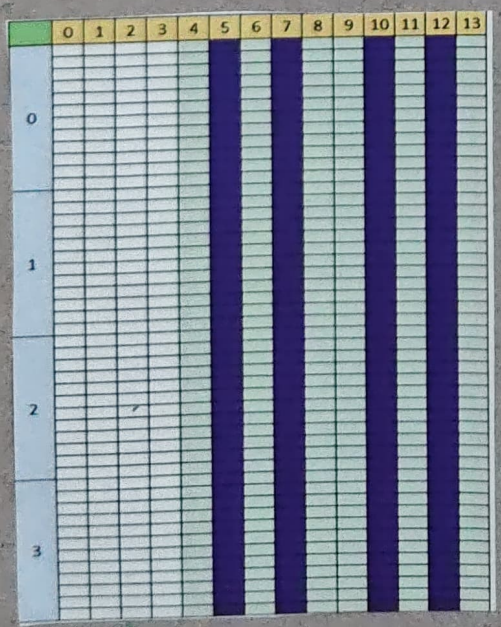


→ Similarly, in this case, there is no Frequency hopping.

→ There are in total 10 Symbols allocated to PUCCH. Out of 10, 2 goes to DMRS.

→ No Additional DMRS positions are configured.

→ The remaining 8 Symbols are used for DATA transfer.



→ And, in this case, Additional DMRS positions are configured.

In PUCCH Format 3, there is no UE Multiplexing. So, if there are two UEs wants to send the data using Format 3, they have to use other resource allocations given to them.

Format 3 is the largest format among all the PUCCH formats that we have, since it takes maximum 14 symbols and 16 PRBs. Let's say, out of 14 Symbols, only 2 are given to DMRS, then

$$\begin{pmatrix}16\\ \text{PRBs}\end{pmatrix} * \begin{pmatrix}12 \text{ Subcarriers}\\ \text{in } 1 \text{ PRB}\end{pmatrix} * \begin{pmatrix}12\\ \text{Symbols}\\ \text{excluding}\\ \text{DMRS}\end{pmatrix} * \begin{pmatrix}2\\ \text{QPSK}\\ \text{modulation}\end{pmatrix}$$

$\Rightarrow$ 4608 encoded bits can be transmitted.

(ii) Bits after Rate Matching.

If we take the highest code Rate (0.8), then we can have $4608 * 0.8 \approx 3600$ UCI bits that can be carried by PUCCH F3.

It is a rough number because the blocks are segmented and CRCs are added, so, at max roughly 3600 UCI bits can be carried by PUCCH.

PUCCH Format 3 may take minimum of 4 OFDM symbols and 1 PRB. Out of 4 Symbols, if Frequency hopping is enabled, 2 Symbols are given to DMRS and only two symbols left for DATA. So,

$$\begin{pmatrix}1\\ \text{PRB}\end{pmatrix} * \begin{pmatrix}12 \text{ Subcarriers}\\ \text{in } 1 \text{ PRB}\end{pmatrix} * \begin{pmatrix}2 \text{ Symbols}\\ \text{excluding}\\ \text{DMRS}\end{pmatrix} * \begin{pmatrix}1\\ \frac{\pi}{2} \text{ BPSK}\\ \text{modulation}\end{pmatrix}$$

$\Rightarrow$ 24 encoded bits can be transmitted.

When the SNR is bad (or) channel is bad, in that case $\frac{\pi}{2}$ BPSK can be used.

When the SNR is good (or) channel is good, in that case QPSK can be used.

$\frac{\pi}{2}$ BPSK is also use to have lower PAPR.

So, based on the requirement, different modulations can be used appropriately.

Now, the Receiver part is very similar to how data is processed in PDSCH, PUSCH, and all. We have seen the Transmitter side (Refer flow diagram). At the Receiver side, the reverse of the processing that we've seen in Transmitter side happens.

First, we Demap the data. (i) extract the REs which contains PUCCH DMRS and DATA. We separate DMRS and DATA at the receiver. Using the DMRS, we estimate the channel. This channel estimates are used to Equalize this DATA.

Once we equalize the DATA, we have the modulated samples. Then we do soft demodulation of the modulate samples to get the Log Likelihood Ratios (LLRs). These LLRs goes through Descrambling. Now, based on the No. of UCI bits (preknown at the receiver), the receiver will do De-Rate Matching,

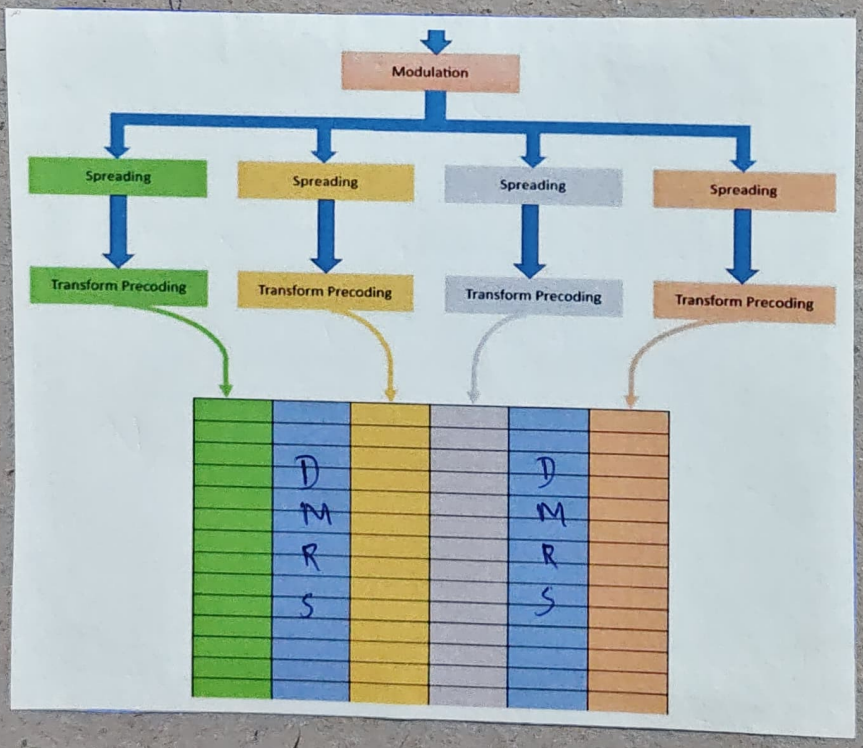
and uses Reed-Muller Decoder / Polar decoder, then CRC removal (need basis) and CodeBlock Concatenation (need basis). So, now we have the Final bits in place.

This is how, whatever we've done at the transmitter, the reverse of that is done at the receiver. Many things here are proprietary. It depends on different organizations, how they have implemented the receiver.

This is all about the PUCCH Format 3.

PUCCH Format 4 : Signal Generation, UE Multiplexing, and Receiver

In the previous sections, we've seen the Bit Processings for Formats 2, 3 and 4. (i.e) how UCI bits goes through multiple blocks (like CRC attach, Reed-Muller / Polar Encoder, Rate Matching, Scrambling and Modulation). In this section, we'll understand the modules after these.



The Modulated data goes through Spreading module, a new module that we haven't seen in other PUCCH Formats. So, basically here the DATA is distributed.

The Spreading function is defined as follows.

$$y(lM_{sc}^{PUCCH,4}+k) = w_n(k)\cdot d\left(l\frac{M_{sc}^{PUCCH,4}}{N_{SF}^{PUCCH,4}} + k \bmod \frac{M_{sc}^{PUCCH,4}}{N_{SF}^{PUCCH,4}}\right)$$

$$k = 0,1,\ldots,M_{sc}^{PUCCH,4}-1$$

$$l = 0,1,\ldots,\left(N_{SF}^{PUCCH,4}M_{symb}/M_{sc}^{PUCCH,4}\right)-1$$

where, $M_{sc}^{PUCCH,4} = 12$

$N_{SF}^{PUCCH,4} = 2$ (assumption) (Spreading Factor)

For $l = 0$,

$$y(k) = w_n(k) \cdot d\left(k \bmod \frac{12}{2}\right)$$

$$= w_n(k)\ d(k \,\%\, 6).$$

where, $w_n(k) \rightarrow$ Multiplication Factor

$y(k) \rightarrow$ Output after Spreading

$d(k \,\%\, 6) \rightarrow$ samples after modulation

$$\Rightarrow \begin{aligned} y(0) &= w_n(0)\ d(0) \\ y(1) &= w_n(1)\ d(1) \\ &\vdots \\ y(5) &= w_n(5)\ d(5) \end{aligned} \quad \Bigg| \quad \begin{aligned} y(6) &= w_n(6) \cdot d(0) \\ y(7) &= w_n(7)\ d(1) \\ &\vdots \\ y(11) &= w_n(11) \cdot d(5) \end{aligned}$$

(ii) the Data after Modulation, is distributed based on different multiplication factors ($w_n(k)$) and different Spreading factors ($N_{SF}^{PUCCH,4} = 2$ or 4).

Lets understand how the Spreading factors are designed.

Table 6.3.2.6.3-1: Orthogonal sequences $w_n(m)$ for PUCCH format 4 when $N_{SF}^{PUCCH,4} = 2$.

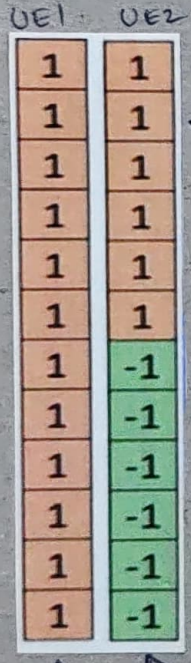
n	w_n
0	[+1 +1 +1 +1 +1 +1 +1 +1 +1 +1 +1 +1]
1	[+1 +1 +1 +1 +1 +1 −1 −1 −1 −1 −1 −1]

Table 6.3.2.6.3-2: Orthogonal sequences $w_n(m)$ for PUCCH format 4 when $N_{SF}^{PUCCH,4} = 4$.

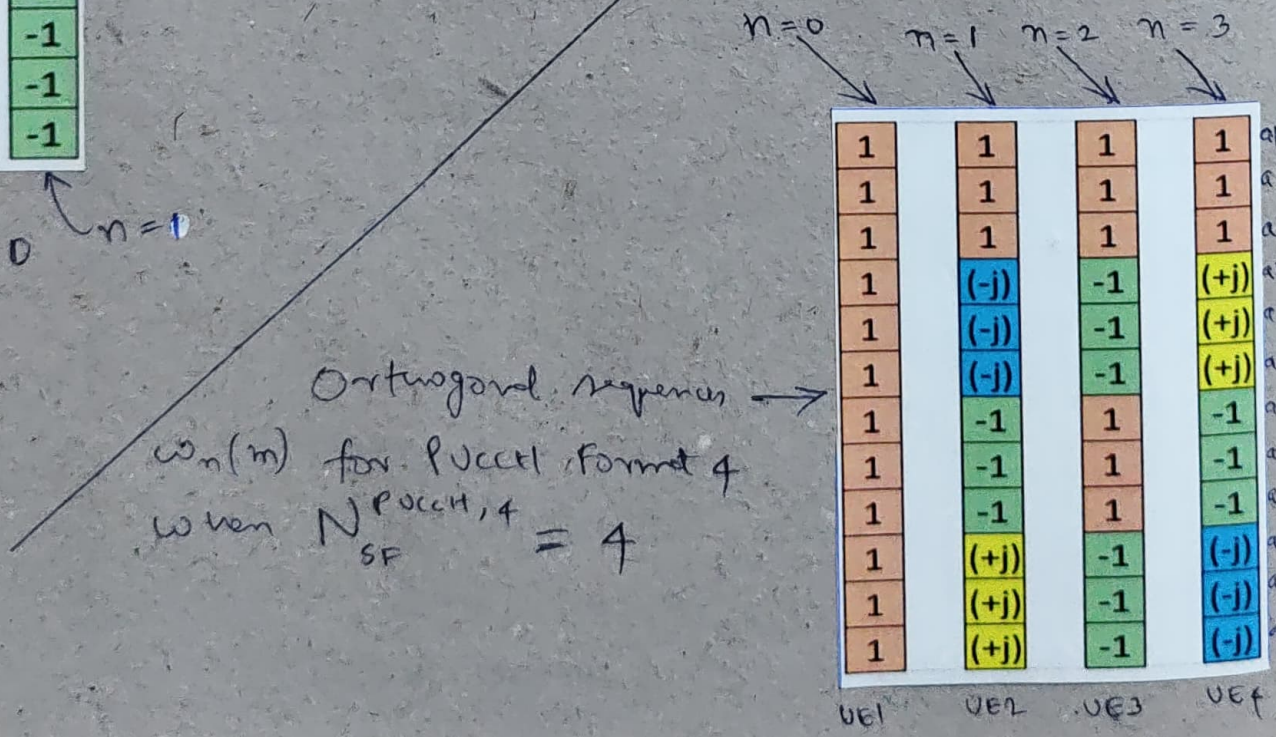
n	w_n
0	[+1 +1 +1 +1 +1 +1 +1 +1 +1 +1 +1 +1]
1	[+1 +1 +1 −j −j −j −1 −1 −1 +j +j +j]
2	[+1 +1 +1 −1 −1 −1 +1 +1 +1 −1 −1 −1]
3	[+1 +1 +1 +j +j +j −1 −1 −1 −j −j −j]

As we see in the Table, for Spreading factor $N_{SF}^{PUCCH,4} = 2$, we have two different arrays. One with all +1's and another array with six +1's and six -1's. So, every data that is being transmitted will be multiplied with either of these 2 arrays on every symbol wherever the data is going. (Not for DMRS).

So, if two UE's are multiplexed, then one UE will be given $n=0$ and second UE will be given $n=1$. So, this is the Orthogonal Cover Code (OCC) that helps us multiplex two UEs.



← Orthogonal sequences $W_n(m)$ for PUCCH Format 4 when $N_{SF}^{PUCCH,4} = 2$



Orthogonal sequences → $W_n(m)$ for PUCCH Format 4 when $N_{SF}^{PUCCH,4} = 4$

0	1	2	3	4	5	6	7	8	9	10	11	12	13	
								1		1	1		1	a1
								1		1	1		1	a2
								1		1	1		1	a3
								1		1	1		1	a4
								1		1	1		1	a5
								1		1	1		1	a6
								1		1	1		1	a1
								1		1	1		1	a2
								1		1	1		1	a3
								1		1	1		1	a4
								1		1	1		1	a5
								1		1	1		1	a6

0	1	2	3	4	5	6	7	8	9	10	11	12	13	
								1		1	1		1	b1
								1		1	1		1	b2
								1		1	1		1	b3
								1		1	1		1	b4
								1		1	1		1	b5
								1		1	1		1	b6
								-1		-1	-1		-1	-b1
								-1		-1	-1		-1	-b2
								-1		-1	-1		-1	-b3
								-1		-1	-1		-1	-b4
								-1		-1	-1		-1	-b5
								-1		-1	-1		-1	-b6

⊙ The data going here is multiplied with +1s ($n=0$)

⊙ The first 6 data on any symbol is repeated in the next 6 subcarriers.

⊙ The data going here is multiplied with +1s and -1s. ($n=1$)

⊙ Negative of first 6 data goes in the next 6 subcarriers, in all symbols.

These are the orthogonal cover codes, that helps us multiplexing multiple UEs.

With spreading factor $N_{SF}^{PUCCH,4} = 4$, we can multiplex 4 UEs. Each UE is using one cover code ($n=0$, $n=1$, $n=2$, $n=3$).

This is how the OCCs are designed and this is how spreading is taking place.

After spreading, the data goes to Transform Precoding. Transform Precoding is nothing but taking DFT of the samples $y(l \cdot M_{sc}^{PUCCH,s} + k)$ after spreading.

$$z(l \cdot M_{sc}^{PUCCH,s} + k) = \frac{1}{\sqrt{M_{sc}^{PUCCH,s}}} \sum_{m=0}^{M_{sc}^{PUCCH,s}-1} y(l \cdot M_{sc}^{PUCCH,s} + m)\, e^{-j\frac{2\pi mk}{M_{sc}^{PUCCH,s}}}$$

$$k = 0, \ldots, M_{sc}^{PUCCH,s} - 1$$

$$l = 0, \ldots, \left(N_{SF}^{PUCCH,s} M_{symb} / M_{sc}^{PUCCH,s}\right) - 1$$

This DFT helps reducing the PAPR.

Here $M_{sc}^{PUCCH,0} = 12$.

For $l=0$,

$$z(k) = \frac{1}{\sqrt{12}} \sum_{m=0}^{11} y(m)\, e^{-j\frac{2\pi mk}{12}}$$

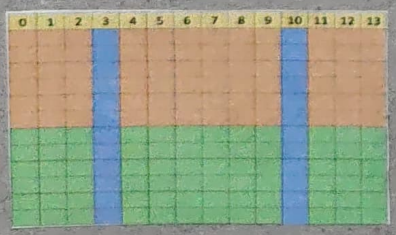
Now, using the factors l and $M_{sc}^{PUCCH,0}$, we find the DFT of the next symbol, and so on (Refer Flow diagram)

Now, How the DMRS and DATA are mapped? We've already seen this table in previous section. (Refer there)

Table 6.4.1.3.3.2-1: DM-RS positions for PUCCH format 3 and 4.

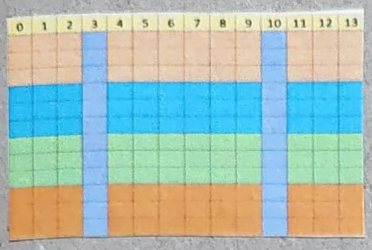
PUCCH length (Symbols)	DM-RS position / within PUCCH span			
	No additional DM-RS		Additional DM-RS	
	No hopping	Hopping	No hopping	Hopping
4	1	0, 2	1	0, 2
5	0, 3		0, 3	
6	1, 4		1, 4	
7	1, 4		1, 4	
8	1, 5		1, 5	
9	1, 6		1, 6	
10	2, 7		1, 3, 6, 8	
11	2, 7		1, 3, 6, 9	
12	2, 8		1, 4, 7, 10	
13	2, 9		1, 4, 7, 11	
14	3, 10		1, 5, 8, 12	

The symbol numbers for DMRS is mentioned in this table. And the remaining symbols are for DATA part.



→ This is how the DATA is mapped for Spreading Factor $N_{SF}^{PUCCH,4} = 2$

→ (i) the data is multiplied with an array of six +1s and six -1s. (ii) OCC.



⇒ This is how the DATA is mapped for Spreading Factor $N_{SF}^{PUCCH,4} = 4$.

These couple of examples are for PUCCH length of 14 symbols (No additional DMRS) configuration. Here, out of 14 symbols, two symbols are for DMRS and the remaining 12 are for DATA.

And, as we see in the first example, the DATA is only going to the first 6 REs, and the remaining 6 REs the data is repeated (with Negation). So, in this case,

$$(6\ \text{REs}) * (12\ \text{Symbols}) * \left(\begin{matrix}2\\ \text{QPSK}\end{matrix}\right)$$

⇒ 144 bits are encoded after Rate Matching.

If we take the maximum Code Rate as 0.8, then $(114\ \text{bits} * 0.8) \approx 93$ UCI bits can be transmitted using PUCCH Format 4.

RECEIVER DESIGN

In the DMRS generation, we've seen the cyclic shift for DMRS.

Table 6.4.1.3.3.1-1: Cyclic shift index for PUCCH format 4.

Orthogonal sequence index l	Cyclic shift index m_0	
	$N_{SF}^{PUCCH,4} = 2$	$N_{SF}^{PUCCH,4} = 4$
0	0	0
1	6	6
2	-	3
3	-	9

If Spreading factor $N_{SF}^{PUCCH,4} = 2$, the cyclic shift exists at indices 0 and 6. If the Spreading factor $N_{SF}^{PUCCH,4} = 4$, then the cyclic shift exists at indices 0, 6, 3 and 9.

Because we are not using the OCC for DMRS (OCC is used only for DATA), so we should have some way, to have uncorrelated / orthogonal DMRS to estimate the channel for each UE.

Lets assume there are 2 UEs that are sending UCI through PUCCH Format 4. And, we need to have a receiver that can separate the data and decode the UCI at the receiver end. So, we pick the spreading factor $N_{SF}^{PUCCH,4} = 2$, where there are two cyclic shift indices (m_0). For UE 1, $m_0 = 0$. And for UE 2, $m_0 = 6$.

For UE1, the OCC for DATA is $n = 0$.
For UE2, the OCC for DATA is $n = 1$.
} Refer Table 6.3.2.6.3-1

(ii) $n = 0 \Rightarrow [+1 \ +1 \ +1 \ +1 \ +1 \ +1 \ +1 \ +1 \ +1 \ +1 \ +1 \ +1]$

$n = 1 \Rightarrow [+1 \ +1 \ +1 \ +1 \ +1 \ +1 \ -1 \ -1 \ -1 \ -1 \ -1 \ -1]$

So, first we'll estimate the channel. As we know, for different cyclic shifts, two different generated signals will be uncorrelated.

So, based on DMRS, using the cyclic shift property of the ZC sequence, we can estimate the channel on each subcarrier.

Now, once we estimated the channel, we need to apply the effect of the channel on DATA.

For DATA, we've used OCC, and the DATA is also spread.

Lets take an example.

WKT, RE 0 and RE 6 have the same DATA. (ii)

RE 0 $\longrightarrow$ x_1 $\longrightarrow$ z_1

RE 1 $\longrightarrow$ x_2 $\longrightarrow$ z_2

RE 2 $\longrightarrow$ x_3 $\longrightarrow$ z_3

RE 3 $\longrightarrow$ x_4 $\longrightarrow$ z_4

RE 4 $\longrightarrow$ x_5 $\longrightarrow$ z_5

RE 5 $\longrightarrow$ x_6 $\longrightarrow$ z_6

RE 6 $\longrightarrow$ x_1 $\longrightarrow$ $-z_1$

RE 7 $\longrightarrow$ x_2 $\longrightarrow$ $-z_2$

RE 8 $\longrightarrow$ x_3 $\longrightarrow$ $-z_3$

RE 9 $\longrightarrow$ x_4 $\longrightarrow$ $-z_4$

RE 10 $\longrightarrow$ x_5 $\longrightarrow$ $-z_5$

RE 11 $\longrightarrow$ x_6 $\longrightarrow$ $-z_6$

Using this property, we can write

$$y_1 = h_1 x_1 + h_2 z_1 + w_1$$

$$y_2 = h_3 x_1 + h_4 (-z_1) + w_2$$

Here, h_1, h_2, h_3 and h_4 are known to us using the channel estimates using DMRS. And we've received y_1 and y_2 at the receiver. We need to find x_1 and z_1.

Since we have 2 equations and 2 unknowns, we can build an equalizer to find x_1 and z_1. Similarly, we can find out x_2 to x_6 and z_2 to z_6.

And, similarly for other symbols also (whichever symbol is allocated to PUCCH F4), we can

equalize the data.

Once we equalize the data, we pass it through the Receiver Chain. (ii) Recall, at the Transmitter side, we done Transform Precoding (where we took 12-point DFT). So, at the receiver, we're supposed to do 12-point IDFT. Then the data will go through Demodulation and Descrambling. Then based on the UCI size, the Rate-dematching, Reed-Muller Decoding / Polar Decoding, CRC removal (need basis) and Code block Concatenation (need basis) is done. to receive the UCI bits at the receiver.

In Summary, PUCCH F4 can multiplex 2 or 4 UEs, takes only one PRB, 4 to 14 symbols, maximum 115 UCI bits (approx.) can be transmitted.

This is all about PUCCH Format 4.
