5G PUSCH

讲师：捻叶成剑

PUSCH

PUSCH：Physical Uplink Shared Channel / 物理上行共享信道

PUSCH的作用：主要是用来传输上行用户数据，高层信令（比如：RRC）

对于每个PUSCH调度，使用DCI 0_0或DCI 0_1

PUSCH最大支持4层

# of Layers	# of Antenna Ports	Transform Precoding	Precoding Matrix (38.211)
1	2		Table 6.3.1.5-1
	4	enabled	Table 6.3.1.5-2
		disabled	Table 6.3.1.5-3
2	2	disabled	Table 6.3.1.5-4
	4	disabled	Table 6.3.1.5-5
3	4	disabled	Table 6.3.1.5.4-6
4	4	disabled	Table 6.3.1.5-7

PUSCH支持的调制方式：

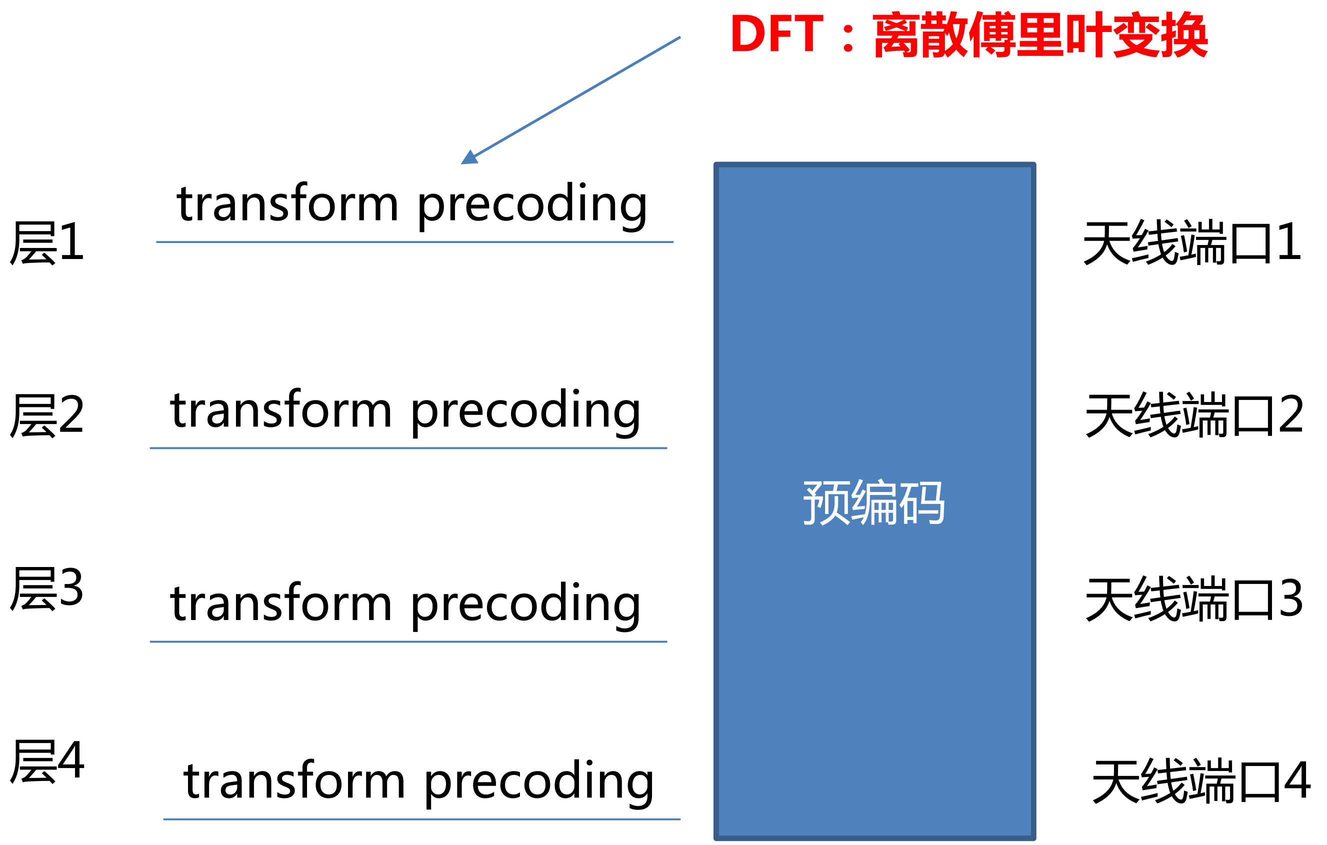
38.211-Table 6.3.1.2-1

Transform precoding disabled		Transform precoding enabled	
Modulation scheme	Modulation order Q_m	Modulation scheme	Modulation order Q_m
		π/2-BPSK	1
QPSK	2	QPSK	2
16QAM	4	16QAM	4
64QAM	6	64QAM	6
256QAM	8	256QAM	8

转换预编码transform precoding

转换预编码就是对层的数据进行数学运算（离散傅里叶变换），在进行了转换预编码之后，再进行端口映射。

使用转换预编码目的是为了降低峰均比。（这里我们不展开讲这个的原理了）



PUSCH的时域资源分配

PUSCH时域资源分配

与PDSCH类似，PUSCH的时域资源分配类型（mapping type），也分为两种，但是细节有区别：

TypeA：在一个时隙当中，PUSCH的符号从0开始，符号长度 4-14 个符号。
TypeB：在一个时隙当中，PUSCH从（0-13）开始，符号长度（1-14）个符号

typeA是时隙调度
typeB是mini slot

上下行Mini slot最重要的区别：
PDSCH是只能2,4,7个符号
而PUSCH是1-14个符号都可以

< 38.214-Table 6.1.2.1-1: Valid S and L combinations >

PUSCH mapping type	Normal cyclic prefix			Extended cyclic prefix		
	S	*L*	*S+L*	*S*	*L*	*S+L*
Type A	0	{4,…,14}	{4,…,14}	0	{4,…,12}	{4,…,12}
Type B	{0,…,13}	{1,…,14}	{1,…,14}	{0,…,12}	{1,…,12}	{1,…,12}

PUSCH时域资源分配

PUSCH的时域调度，主要取决于4个核心参数：

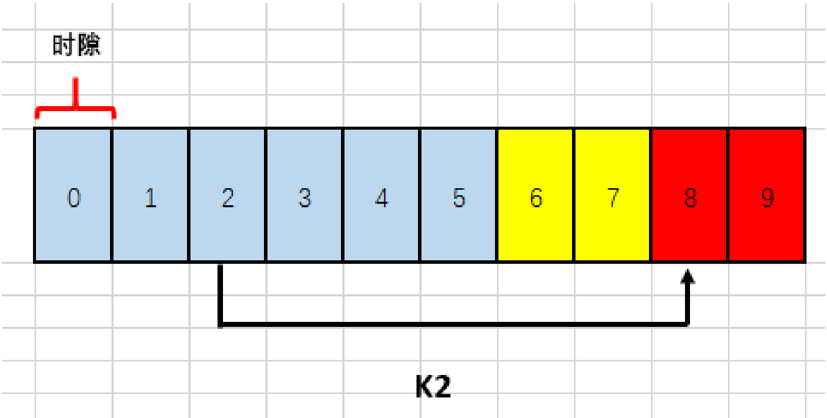
- 时隙偏移值K2
- 时域起始符号S和时域符号数L
- mapping type：时域分配类型，typeA或者typeB

由于现网mapping type都是typeA，因此，重点研究K2，S,和L。

时隙偏移值K2

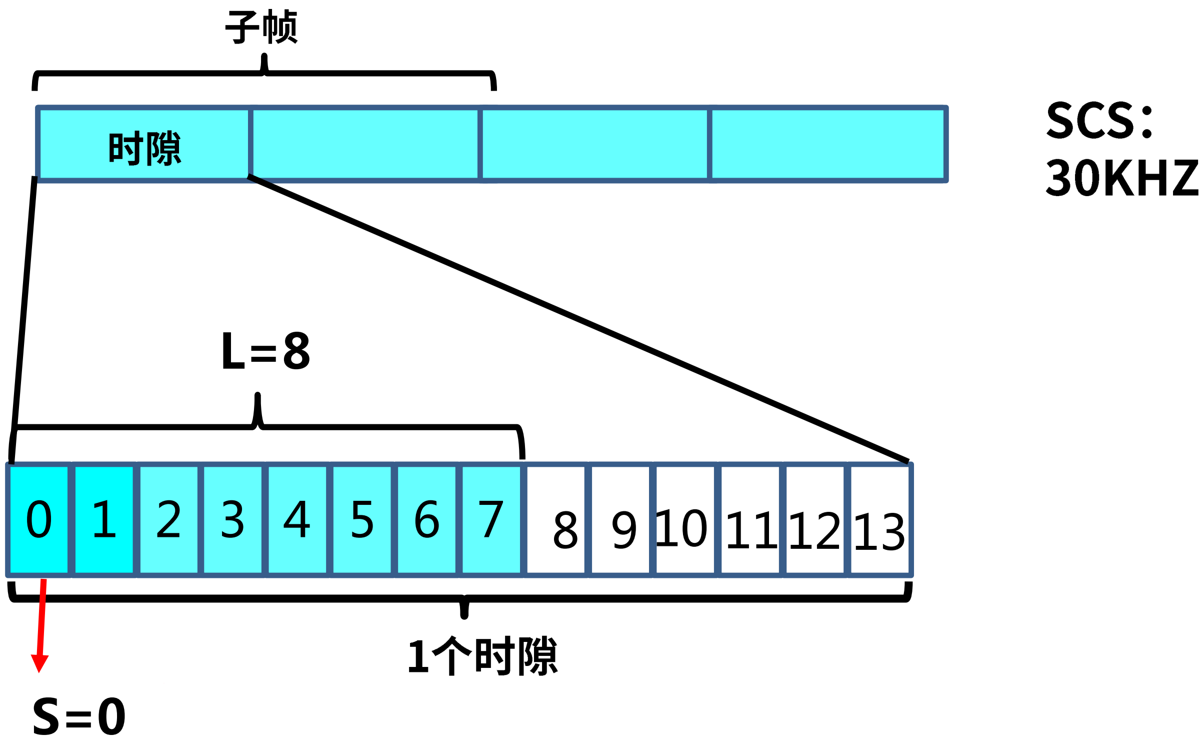
k2指上行调度DCI与其调度的PUSCH之间的时隙间隔。

K2=0表示PUSCH与PDCCH在同一个slot上，K2=1表示PUSCH在PDCCH后面一个slot上，依次类推



时域起始符号S和时域符号数L

举个例子



< 38.214-Table 6.1.2.1-1: Valid S and L combinations >

PUSCH mapping type	Normal cyclic prefix			Extended cyclic prefix		
	S	*L*	*S+L*	*S*	*L*	*S+L*
Type A	0	{4,…,14}	{4,…,14}	0	{4,…,12}	{4,…,12}
Type B	{0,…,13}	{1,…,14}	{1,…,14}	{0,…,12}	{1,…,12}	{1,…,12}

UE如何获取K2，S,L的配置

分为两种情况：

1. 待机状态，可通过DCI确定的查表索引查询默认表格直接获取到K2、mapping type、S和L；

2. RRC连接态时，高层参数PUSCH-Configcommon中的pusch-TimeDomainAllocationList 获取配置列表，然后结合DCI确定的查表索引获取K2、mapping type、 SLIV ，再进一步计算出S和L。

DCI0-0和DCI0-1是调度PUSCH的，在PDCCH里面讲过了。

待机状态查表获取K2，S，L

A表

Table 6.1.2.1.1-2: Default PUSCH time domain resource allocation A for normal CP

Row index	PUSCH mapping type	K_2	S	L
1	Type A	j	0	14
2	Type A	j	0	12
3	Type A	j	0	10
4	Type B	j	2	10
5	Type B	j	4	10
6	Type B	j	4	8
7	Type B	j	4	6
8	Type A	j+1	0	14
9	Type A	j+1	0	12
10	Type A	j+1	0	10
11	Type A	j+2	0	14
12	Type A	j+2	0	12
13	Type A	j+2	0	10
14	Type B	j	8	6
15	Type A	j+3	0	14
16	Type A	j+3	0	10

有2个表格
这里的是A

UE待机状态下，查询默认表格，根据DCI相应字段的数值+1，获得表的行数，查表获取到K2，S，L，mapping type 4个参数

其中，K2的取值取决于子载波间隔

Table 6.1.2.1.1-4: Definition of value j

μ_{PUSCH}	j
0	1
1	1
2	2
3	3

μ	子载波间隔
0	15
1	30
2	60
3	120

DCI 0_0或DCI 0_1中的Time domain resource assignment字段会指示PDSCH的时域位置。该字段共4个bit，所以其值为0-15，假设其值为m，则m+1指示了一个时域资源分配表格的row index（行索引）

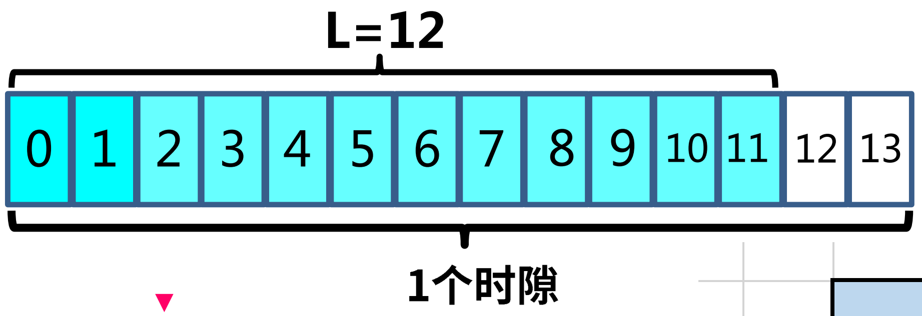
举个例子

假设使用的A表

DCI： Time domain resource assignment=1

子载波间隔30KHZ

Row index=1 +1=2



S=2

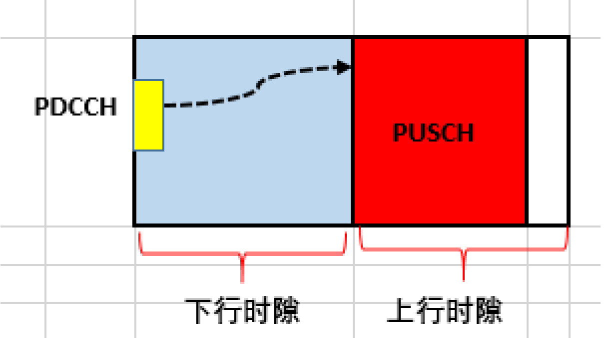


Table 6.1.2.1.1-2: Default PUSCH time domain resource allocation A for normal CP

Row index	PUSCH mapping type	K_2	S	L
1	Type A	j	0	14
2	Type A	j	0	12
3	Type A	j	0	10
4	Type B	j	2	10
5	Type B	j	4	10
6	Type B	j	4	8
7	Type B	j	4	6
8	Type A	j+1	0	14
9	Type A	j+1	0	12
10	Type A	j+1	0	10
11	Type A	j+2	0	14
12	Type A	j+2	0	12
13	Type A	j+2	0	10
14	Type B	j	8	6
15	Type A	j+3	0	14
16	Type A	j+3	0	10

Table 6.1.2.1.1-4: Definition of value j

μ_{PUSCH}	j
0	1
1	1
2	2
3	3

得知J=1
因此K2=1

表A-扩展cp

Table 6.1.2.1.1-3: Default PUSCH time domain resource allocation A for extended CP

Row index	PUSCH mapping type	K_2	S	L
1	Type A	j	0	8
2	Type A	j	0	12
3	Type A	j	0	10
4	Type B	j	2	10
5	Type B	j	4	4
6	Type B	j	4	8
7	Type B	j	4	6
8	Type A	j+1	0	8
9	Type A	j+1	0	12
10	Type A	j+1	0	10
11	Type A	j+2	0	6
12	Type A	j+2	0	12
13	Type A	j+2	0	10
14	Type B	j	8	4
15	Type A	j+3	0	8
16	Type A	j+3	0	10

UE如何知道查哪个表?

UE需要查表的时候就根据目前是普通CP还是扩展CP，去查对应的A表即可。

Table 6.1.2.1.1-1: Applicable PUSCH time domain resource allocation

RNTI	**PDCCH search space**	***pusch-ConfigCommon* includes *pusch-TimeDomainAllocationList***	***pusch-Config* includes *pusch-TimeDomainAllocationList***	**PUSCH time domain resource allocation to apply**
PUSCH scheduled by MAC RAR as described in subclause 8.2 of [6, TS 38.213]		No	-	Default A
		Yes		*pusch-TimeDomainAllocationList* provided in *pusch-ConfigCommon*
C-RNTI, MCS-C-RNTI, TC-RNTI, CS-RNTI	Any common search space associated with CORESET 0	No	-	Default A
		Yes		*pusch-AlloTimeDomaincationList* provided in *pusch-ConfigCommon*
C-RNTI, MCS-C-RNTI, TC-RNTI, CS-RNTI, SP-CSI-RNTI	Any common search space not associated with CORESET 0, UE specific search space	No	No	Default A
		Yes	No	*pusch-TimeDomainAllocationList* provided in *pusch-ConfigCommon*
		No/Yes	Yes	*pusch-TimeDomainAllocationList* provided in *pusch-Config*

RRC连接状态通过信令获取获取K2，S，L

```
PUSCH-ConfigCommon ::=                    SEQUENCE {
    groupHoppingEnabledTransformPrecoding    ENUMERATED {enabled}      OPTIONAL,    -- Need R
    pusch-AllocationList          SEQUENCE (SIZE(1..maxNrofUL-Allocations))
                                        OF  PUSCH-TimeDomainResourceAllocation OPTIONAL,    -- Need R
    msg3-DeltaPreamble            INTEGER (-1..6)            OPTIONAL,    -- Need R
    p0-NominalWithGrant           INTEGER (-202..24)         OPTIONAL,    -- Need R
    ...
}
```

```
  pusch-AllocationList {
     PUSCH-TimeDomainResourceAllocation {
        k2          0
Row 0   mappingType  typeA,
        startSymbolAndLength 00110111'B => 55(Dec)
     },
     PUSCH-TimeDomainResourceAllocation {
        k2          1
Row 1   mappingType typeA,
        startSymbolAndLength '00111000'B => 56(Dec)
     }
```

RRC信令中pusch-Configcommon中的PUSCH-Allocationlist获取配置参数列表

具体选择哪一种配置，取决于DCI字段Time domain resource assignment，这里字段的取值，不用+1

在这种情况下，不是直接获取到S和L这两个参数，而是获取了SLIV,通过反向计算来获取S和L

SLIV---起始和长度指示值

PUSCH这里的SLIV计算公式，跟PDSCH的是一样的：

if $(L-1) \le 7$ then

$$SLIV = 14 \cdot (L-1) + S$$

else

$$SLIV = 14 \cdot (14 - L + 1) + (14 - 1 - S)$$

与PDSCH的区别就是S和L的取值有区别。因此，我们还是同样的可以通过SLIV去反向获得S和L的数值

< 38.214-Table 6.1.2.1-1: Valid S and L combinations >

PUSCH mapping type	Normal cyclic prefix			Extended cyclic prefix		
	S	*L*	*S+L*	*S*	*L*	*S+L*
Type A	0	{4,…,14}	{4,…,14}	0	{4,…,12}	{4,…,12}
Type B	{0,…,13}	{1,…,14}	{1,…,14}	{0,…,12}	{1,…,12}	{1,…,12}

SLIV与S和L计算对应表

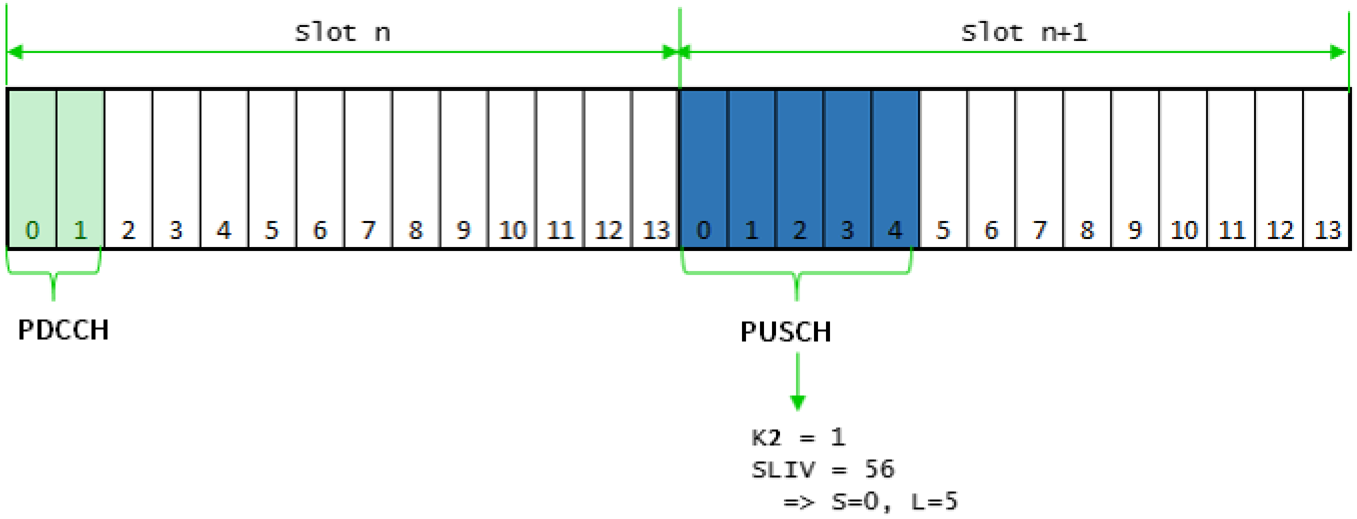
Mapping typeA的SLIV与S和L计算表

S	L	L-1	Last Symbol	SLIV	Valid Mapping Type (Normal CP) PDSCH	Valid Mapping Type (Normal CP) PUSCH
0						
	4	3	3	42	Type A,Type B	Type A,Type B
	5	4	4	56	Type A	Type A,Type B
	6	5	5	70	Type A	Type A,Type B
	7	6	6	84	Type A,Type B	Type A,Type B
	8	7	7	98	Type A	Type A,Type B
	9	8	8	97	Type A	Type A,Type B
	10	9	9	83	Type A	Type A,Type B
	11	10	10	69	Type A	Type A,Type B
	12	11	11	55	Type A	Type A,Type B
	13	12	12	41	Type A	Type A,Type B
	14	13	13	27	Type A	Type A,Type B

举个例子

```
pusch-AllocationList {
    PUSCH-TimeDomainResourceAllocation {
        k2          0
Row 0 - mappingType  typeA,
        startSymbolAndLength 00110111'B => 55(Dec)
    },
    PUSCH-TimeDomainResourceAllocation {
        k2          1
Row 1 - mappingType typeA,
        startSymbolAndLength '00111000'B => 56(Dec)
    }
```

```
DCI :
  Time domain resource assignment = 1
```



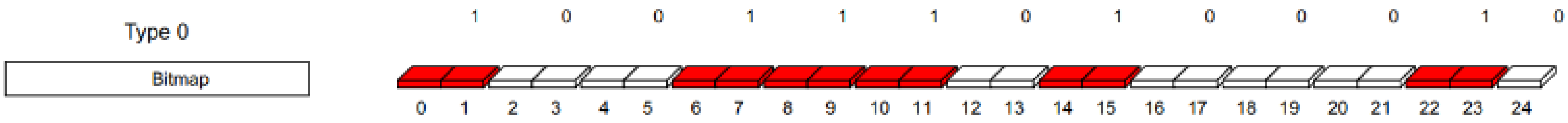
PUSCH的频域资源分配

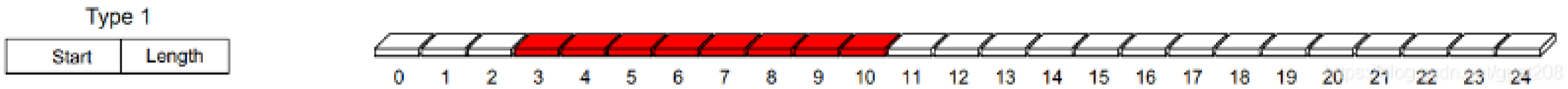
频域资源分配类型

与PDSCH是一样的，也是分为两种类型：type0和type1

Type0：bitmap（位图），指示的资源位置既可以是集中连续的，又可以是根据信道需要灵活分散的

Type1：RIV（开始RB+连续RB长度），在频域RB上集中连续分配的





只有关闭了转换预编码（ transform precoding ）功能后，才支持PUSCH的上行资源分配方案type 0。启用或禁用转换预编码transform precoding时，均支持上行资源分配方案type 1。

DCI与分配类型

DCI0-0调度的PUSCH，仅仅支持type1

DCI0-1调度的PUSCH，类型由高层信令PUSCH-config中的resource Allocation字段来配置，

```
PUSCH-Config ::= SEQUENCE {
    dataScramblingIdentityPUSCH         INTEGER (0..1023) OPTIONAL,
    txConfig                            ENUMERATED {codebook, nonCodebook}
    dmrs-UplinkForPUSCH-MappingTypeA    SetupRelease { DMRS-UplinkConfig }
    dmrs-UplinkForPUSCH-MappingTypeB    SetupRelease { DMRS-UplinkConfig }
    pusch-PowerControl                  PUSCH-PowerControl
    frequencyHopping                    ENUMERATED {intraSlot, interSlot}
    frequencyHoppingOffsetLists         SEQUENCE (SIZE (1..4)) OF
                                            INTEGER (1.. maxNrofPhysicalResourceBlocks-1)
    resourceAllocation                  ENUMERATED { resourceAllocationType0,
                                                     resourceAllocationType1,
                                                     dynamicSwitch},
    pusch-TimeDomainAllocationList      SetupRelease {
                                            PUSCH-TimeDomainResourceAllocationList
```

type0　type1　Dynamicswitch（动态调度）

DCI0-1中的Frequency domain resource assignment字段来进行频域资源分配类型的指示（最高位bit为0代表type0，为1代表type1）

Format 0_1

This is used for the scheduling of PUSCH in one cell.

Field (Item)	Bits	Reference
Identifier for DCI formats	1	
Carrier indicator	0 or 3	
UL/SUL Indicator	0,1	0 - bit for UE not configured with SUL in the cell 1 - bit for UEs configured with SUL in the cell
Bandwidth part indicator	0,1,2	Determined by BandwidthPart-Config in higher layer message and 38.212 - Table 7.3.1.1.2-1
Frequency domain resource assignment	Variable	Variable with Resource Allocation Type
Time domain resource assignment	4	Carries the row index of the items in pusch_allocationList in RRC Number of Bit Length is determined by log(I,2), where I is the number of elements in pusch_allocationList in RRC

什么时候使用类型type0

首先必须是使用DCI0-1格式来调度PUSCH

第一种情况 PUSCH-config中的resource Allocation字段为type0

第二种情况：当PUSCH-config中的resource Allocation字段为dynamicswitch，DCI Frequency domain resource assignment字段最高位bit为0

```
PDSCH-Config ::=                        SEQUENCE {
    dataScramblingIdentityPDSCH         INTEGER (0..1007)    OPTIONAL,
    dmrs-DownlinkForPDSCH-MappingTypeA      SetupRelease { DMRS-DownlinkConfig }  OPTIONAL,
    dmrs-DownlinkForPDSCH-MappingTypeB      SetupRelease { DMRS-DownlinkConfig }  OPTIONAL,
    tci-StatesToAddModList                  SEQUENCE (SIZE(1..maxNrofTCI-States))
                                                OF TCI-State    OPTIONAL,   -- Need N
    tci-StatesToReleaseList                 SEQUENCE (SIZE(1..maxNrofTCI-States))
                                                OF TCI-StateId  OPTIONAL,   -- Need N
    vrb-ToPRB-Interleaver                   ENUMERATED {n2, n4},
    resourceAllocation                      ENUMERATED { resourceAllocationType0,
                                                         resourceAllocationType1,
                                                         dynamicSwitch},
    pdsch-AllocationList                SEQUENCE (SIZE(1..maxNrofDL-Allocations))
                                          OF PDSCH-TimeDomainResourceAllocation     OPTIONAL,
    pdsch-AggregationFactor                 ENUMERATED { n2, n4, n8 } OPTIONAL
```

频域资源分配类型type0

Type 0使用bitmap指示PUSCH所使用的RBG，与PDSCH一致

bitmap存储在DCI0-1的Frequency domain resource assignment字段中

RBG定义没有变化，是跟PDSCH一致的

< 38.214 - Table 5.1.2.2.1-1: Nominal RBG size P, Table 6.1.2.2.1-1: Nominal RBG size P >

Bandwidth Part Size	Configuration 1	Configuration 2
1 – 36	2	4
37 – 72	4	8
73 – 144	8	16
145 – 275	16	16

RRC信令中定义RBG大小

```
PUSCH-Config ::= SEQUENCE {
    dataScramblingIdentityPUSCH             INTEGER (0..1023) OPTIONAL,
    txConfig                                ENUMERATED {codebook, nonCodebook}
    dmrs-UplinkForPUSCH-MappingTypeA        SetupRelease { DMRS-UplinkConfig }
    dmrs-UplinkForPUSCH-MappingTypeB        SetupRelease { DMRS-UplinkConfig }
    pusch-PowerControl                      PUSCH-PowerControl
    frequencyHopping                        ENUMERATED {intraSlot, interSlot}
    frequencyHoppingOffsetLists             SEQUENCE (SIZE (1..4)) OF
                                                INTEGER (1.. maxNrofPhysicalResourceBlocks-1)
    resourceAllocation                      ENUMERATED { resourceAllocationType0,
                                                         resourceAllocationType1,
                                                         dynamicSwitch},
    pusch-TimeDomainAllocationList          SetupRelease {
                                                  PUSCH-TimeDomainResourceAllocationList
                                            }
    pusch-AggregationFactor                 ENUMERATED { n2, n4, n8 }
    mcs-Table                               ENUMERATED {qam256, qam64LowSE}
    mcs-TableTransformPrecoder              ENUMERATED {qam256, qam64LowSE}
    transformPrecoder                       ENUMERATED {enabled, disabled}
    codebookSubset                          ENUMERATED {fullyAndPartialAndNonCoherent,
                                                        partialAndNonCoherent,
                                                        nonCoherent}
    maxRank                                 INTEGER (1..4)
    rbg-Size                                ENUMERATED { config2}
    uci-OnPUSCH                             SetupRelease { UCI-OnPUSCH }
    tp-pi2BPSK                              ENUMERATED {enabled}
    ...
```

资源分配Type0举例1

假设：

PUSCH-config中

resource Allocation字段为type0

rbg-Size：configuration 1

RRC信令获取BWP带宽：273RB

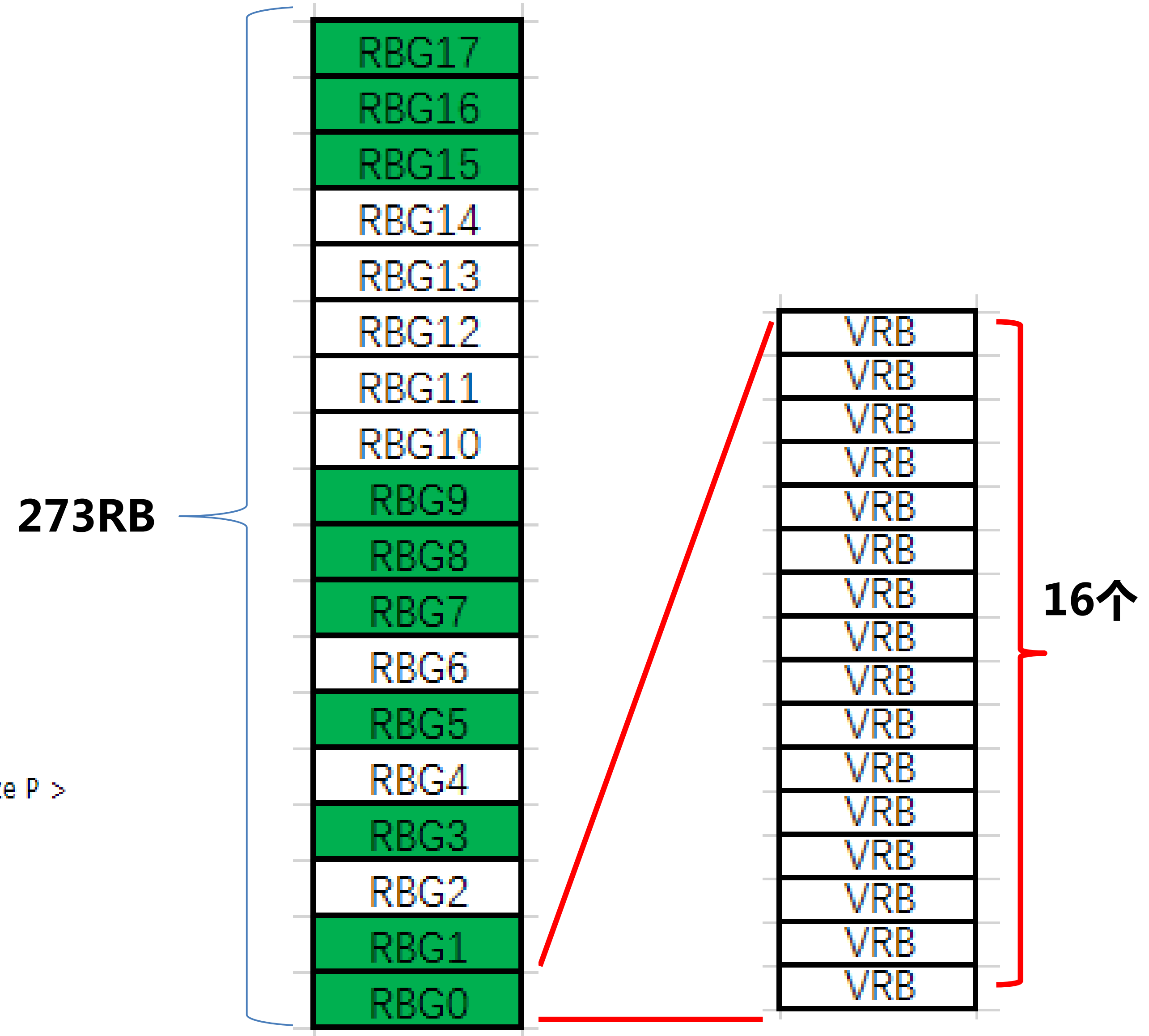
DCI0-1中：

Frequency domain resource assignment
"1101010111000000111"

可知RBG=16个VRB

< 38.214 - Table 5.1.2.2.1-1: Nominal RBG size P, Table 6.1.2.2.1-1: Nominal RBG size P >

Bandwidth Part Size	Configuration 1	Configuration 2
1 – 36	2	4
37 – 72	4	8
73 – 144	8	16
145 – 275	16	16



什么时候使用类型type1

DCI0-0调度的PUSCH

DCI0-1调度的PDSCH

- PUSCH-config中的resource Allocation字段为type1

- 当PUSCH-config中的resource Allocation字段为dynamicswitch，DCI0-1 Frequency domain resource assignment字段最高位bit为1

Type1的频域资源分配

Type1类型下，通过RIV（Resource Indication Value）反推 RB起始位置S和映射的连续RB个数L，即可完成映射。与PDSCH是一致的。

RIV的公式与PDSCH的也是一模一样的

$$\text{if } (L_{RBs}-1) \le \lfloor N_{BWP}^{size}/2 \rfloor \text{ then}$$

$$RIV = N_{BWP}^{size}(L_{RBs}-1)+RB_{start}$$

$$\text{else}$$

$$RIV = N_{BWP}^{size}(N_{BWP}^{size}-L_{RBs}+1)+(N_{BWP}^{size}-1-RB_{start})$$

RIV 在DCI的 Frequency domain resource assignment字段中

RIV计算举例子

$$N_{BWP}^{size}=50 \qquad (L_{RBs}-1) \le \lfloor N_{BWP}^{size}/2 \rfloor$$

	L_{RBs}=1			L_{RBs}=2			…	L_{RBs}=26		
RB_{Start}	0	1	49	0	1	48	…	0	1	24
RIV	0	1	49	50	51	98	…	1250	1251	1274

$$(L_{RBs}-1) > \lfloor N_{BWP}^{size}/2 \rfloor$$

	L_{RBs}=27			…	L_{RBs}=48			L_{RBs}=49		L_{RBs}=50
RB_{Start}	0	1	23	…	0	1	2	0	1	0
RIV	1249	1248	1226	…	199	198	197	149	148	99

Type1的频域资源分配举例子

假设

PUSCH-config中
resource Allocation字段为type1

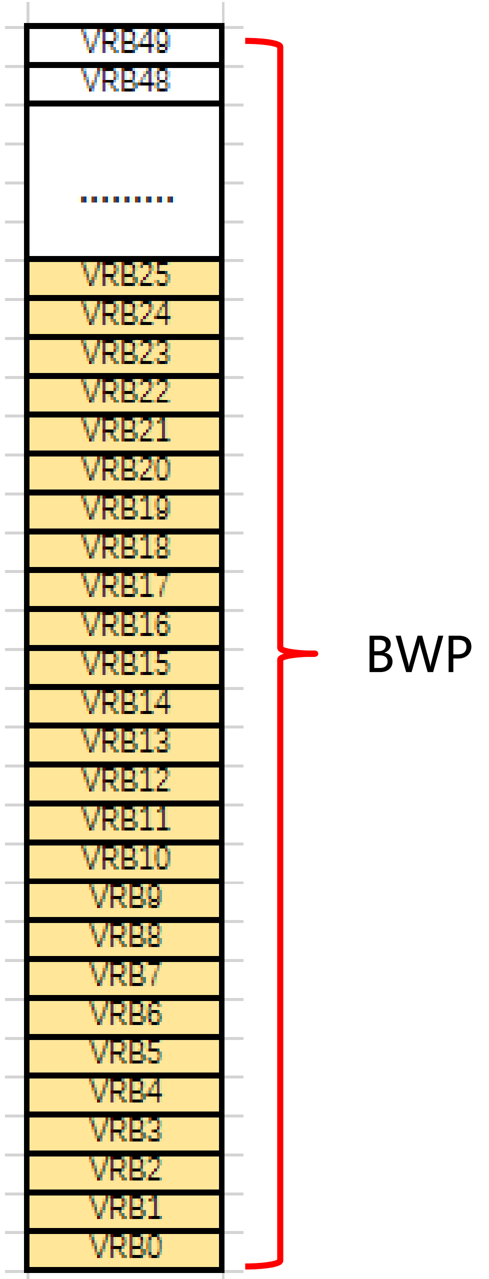
RRC信令获取BWP带宽：N=50RB

DCI0-1中：

Frequency domain resource assignment:10011100010

110011转换成十进制RIV=1250

反推得到S=0，L=26



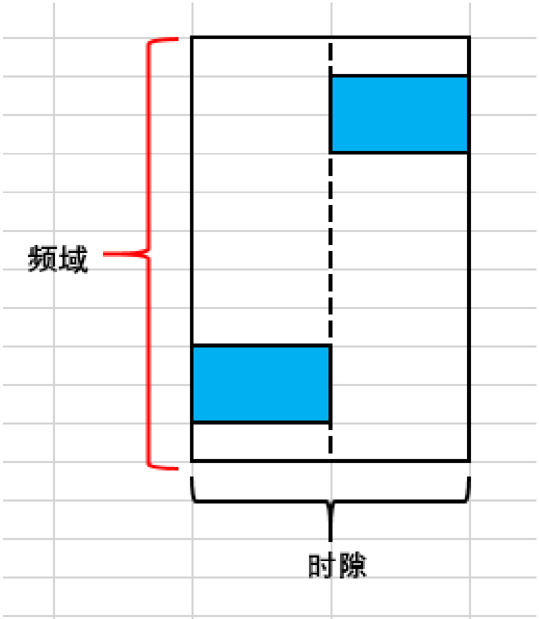
PUSCH跳频（hopping）

PUSCH跳频（hopping）模式

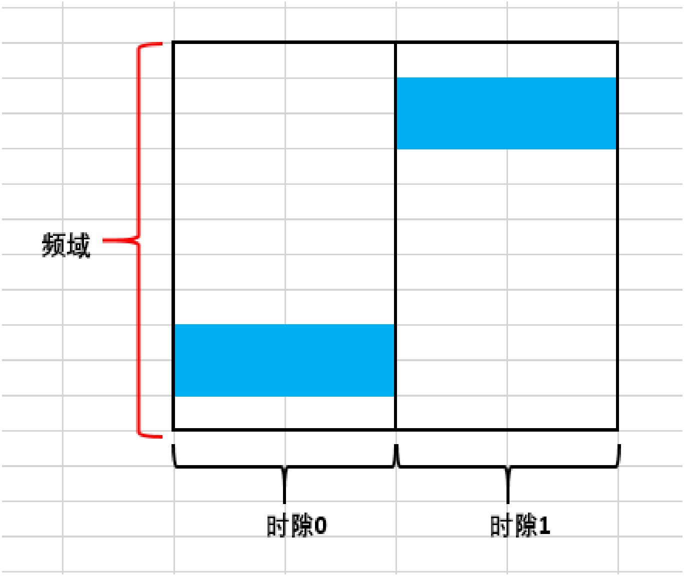
跳频的主要作用就是降低干扰。

5G PUSCH 支持两种跳频模式(下行PDSCH不支持跳频)：

时隙内跳频



时隙间跳频



PUSCH频域资源分配type1才会使用跳频，type0模式下不会使用跳频。本身type0 的RB资源就是离散分布的，已经对干扰有一定的频率分集的作用。

UE怎么知道是否开启跳频?

在PUSCH使用频域分配类型type1的时候，有以下3中情况会开启跳频：

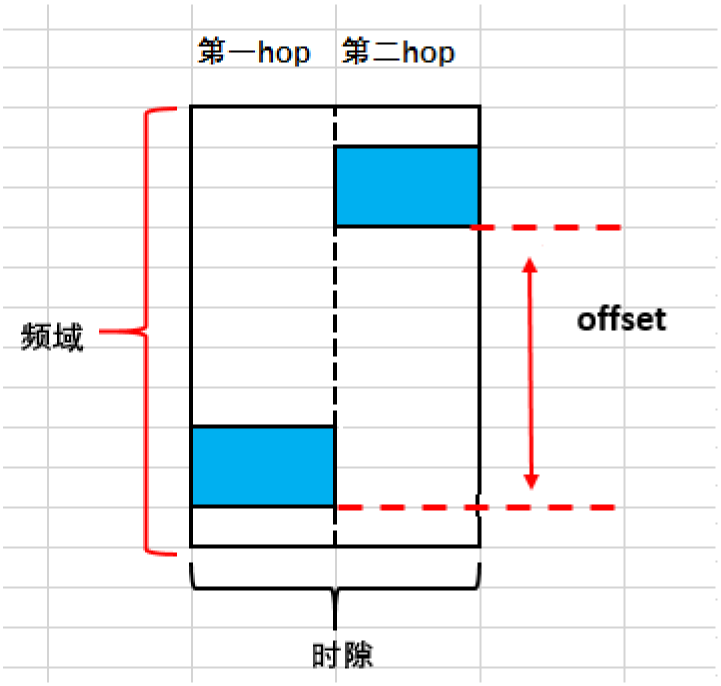
1、DCI格式当中被检测到跳频Frequency hopping flag字段值为1

2、高层信令配置了frequencyHoppingOffset（跳频偏移）

3、随机接入信令MSG2中Frequency hopping flag字段的值为1

时隙内跳频

在时隙内跳频中，一个时隙中前一半符号为第一hop，后一半符号为第二hop



$$RB_{start} = \begin{cases} RB_{start} & i = 0 \\ \left(RB_{start} + RB_{offset}\right) \bmod N_{BWP}^{size} & i = 1 \end{cases}$$

第一hop起始位置 RB_{start}

第二hop起始位置 $RB_{start} + RB_{offset}$

Number of PRBs in initial UL BWP	Value of $N_{UL,hop}$ Hopping Bits	Frequency offset for 2nd hop
$N_{BWP}^{size} < 50$	0	$\lfloor N_{BWP}^{size}/2 \rfloor$
	1	$\lfloor N_{BWP}^{size}/4 \rfloor$
$N_{BWP}^{size} \geq 50$	00	$\lfloor N_{BWP}^{size}/2 \rfloor$
	01	$\lfloor N_{BWP}^{size}/4 \rfloor$
	10	$-\lfloor N_{BWP}^{size}/4 \rfloor$
	11	Reserved

核心参数就是偏移量 $N_{UL,hop}$

此处视频中画的有问题，更正了

举个例子

Number of PRBs in initial UL BWP	Value of $N_{UL,hop}$ Hopping Bits	Frequency offset for 2nd hop
$N_{BWP}^{size} < 50$	0	$\lfloor N_{BWP}^{size}/2 \rfloor$
	1	$\lfloor N_{BWP}^{size}/4 \rfloor$
$N_{BWP}^{size} \geq 50$	00	$\lfloor N_{BWP}^{size}/2 \rfloor$
	01	$\lfloor N_{BWP}^{size}/4 \rfloor$
	10	$-\lfloor N_{BWP}^{size}/4 \rfloor$
	11	Reserved

得知：offset=10

假设：

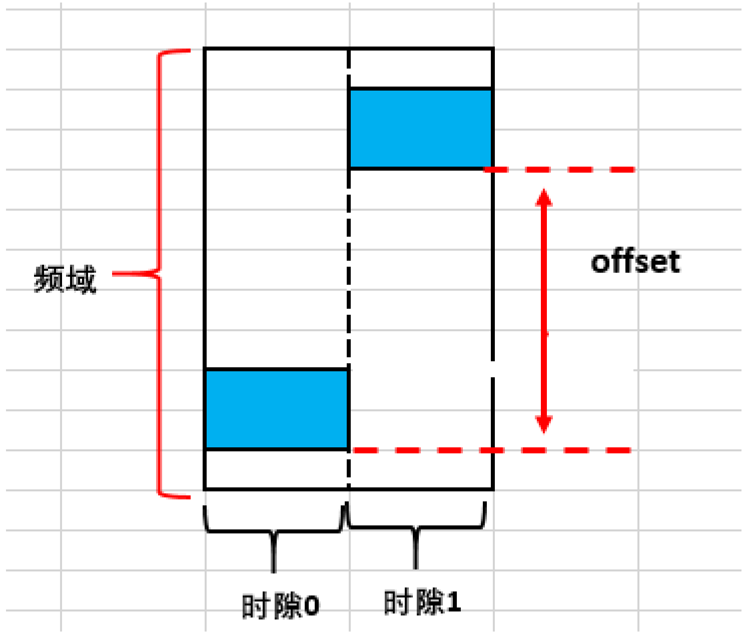
BWP=20PRB

$N_{UL,hop}$ =0

第一hop RB start=2

时隙间跳频

n_s^μ 是时隙编号



$$\mathrm{RB}_{\mathrm{start}}(n_s^\mu) = \begin{cases} \mathrm{RB}_{\mathrm{start}} & n_s^\mu \bmod 2 = 0 \\ (\mathrm{RB}_{\mathrm{start}} + \mathrm{RB}_{\mathrm{offset}}) \bmod N_{BWP}^{size} & n_s^\mu \bmod 2 = 1 \end{cases},$$

偶数时隙起始位置 $\mathrm{RB}_{\mathrm{start}}$

奇数时隙起始位置 $\mathrm{RB}_{\mathrm{start}} + \mathrm{RB}_{\mathrm{offset}}$

Number of PRBs in initial UL BWP	Value of $N_{\mathrm{UL,hop}}$ Hopping Bits	Frequency offset for 2nd hop
$N_{\mathrm{BWP}}^{\mathrm{size}} < 50$	0	$\lfloor N_{\mathrm{BWP}}^{\mathrm{size}}/2 \rfloor$
	1	$\lfloor N_{\mathrm{BWP}}^{\mathrm{size}}/4 \rfloor$
$N_{\mathrm{BWP}}^{\mathrm{size}} \geq 50$	00	$\lfloor N_{\mathrm{BWP}}^{\mathrm{size}}/2 \rfloor$
	01	$\lfloor N_{\mathrm{BWP}}^{\mathrm{size}}/4 \rfloor$
	10	$-\lfloor N_{\mathrm{BWP}}^{\mathrm{size}}/4 \rfloor$
	11	Reserved

核心参数就是偏移量 $N_{\mathrm{UL,hop}}$

此处视频中画的有问题，更正了

UE如何获取偏移量参数

Frequency hopping flag：是否支持跳频

Frequency domain resource assignment：offset选项在这里

Format 0_0

This is used for the scheduling of PUSCH in one cell.

Field (Item)	Bits	
Identifier for DCI formats	1	
Frequency domain resource assignment	4	Variable with Indicate PRB I The number o as described I
Time domain resource assignment	X	Carries the ro in pusch_alloc
Frequency Hopping Flag	1	
Modulation and coding scheme	5	38.214 - 6.1.4
New data indicator	1	

Format 0_1

This is used for the scheduling of PUSCH in one cell.

Field (Item)	Bits	Reference
Identifier for DCI formats	1	
Carrier indicator	0 or 3	
UL/SUL Indicator	0,1	0 - bit for UE not configured with SUL in the cell 1 - bit for UEs configured with SUL in the cell
Bandwidth part indicator	0,1,2	Determined by BandwidthPart-Config in higher layer message and 38.212 - Table 7.3.1.1.2-1
Frequency domain resource assignment	Variable	Variable with Resource Allocation Type
Time domain resource assignment	4	Carries the row index of the items in pusch_allocationList in RRC Number of Bit Length is determined by log(I,2), where I is the number of elements in pusch_allocationList in RRC
Frequency Hopping Flag	0,1	
Modulation and coding scheme	5	38.214 - 6.1.4 (See this table)
New data indicator	1	

UE如何获取偏移量参数

从高层信令获取支持哪一种跳频方式以及offset列表

```
PUSCH-Config ::= SEQUENCE {
    dataScramblingIdentityPUSCH              INTEGER (0..1023) OPTIONAL,
    txConfig                                 ENUMERATED {codebook, nonCodebook}
    dmrs-UplinkForPUSCH-MappingTypeA         SetupRelease { DMRS-UplinkConfig }
    dmrs-UplinkForPUSCH-MappingTypeB         SetupRelease { DMRS-UplinkConfig }
    pusch-PowerControl                       PUSCH-PowerControl
    frequencyHopping                         ENUMERATED {intraSlot, interSlot}
    frequencyHoppingOffsetLists              SEQUENCE (SIZE (1..4)) OF
                                                 INTEGER (1.. maxNrofPhysicalResourceBlocks-1)
    resourceAllocation                       ENUMERATED { resourceAllocationType0,
                                                          resourceAllocationType1
```

总的来说，首先，由DCI决定是否支持跳频，如果支持跳频，再由高层信令字段PUSCH-config决定支持哪种跳频，以及跳频offset列表，最终由DCI字段决定offset列表中选择哪一个offset。

特殊情况，随机接入的时候，msg2会决定是否支持跳频，以及相应的字段会决定offse列表中选择哪一种offset。

随机接入流程中的跳频

随机接入简单来说就是：UE从空闲态转移到与基站之间连接态的过程
在随机接入的第三部信令需要PUSCH来承载，因此第二步骤msg2里面会携带是否跳频以及Offset的选项。

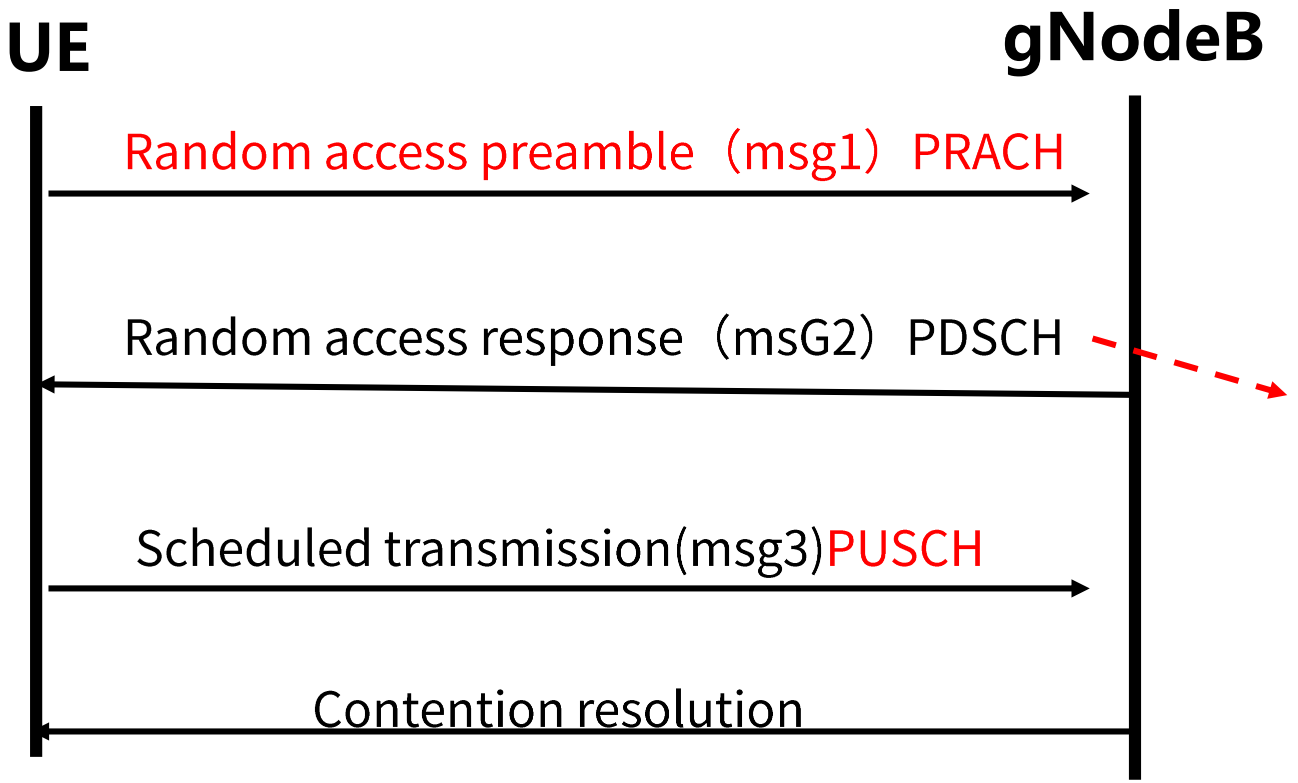


Table 5-1: Random Access Response Grant Content field size

RAR grant field	Number of bits
Frequency hopping flag	1
PUSCH frequency resource allocation	14
PUSCH time resource allocation	4
MCS	4
TPC command for PUSCH	3
CSI request	1

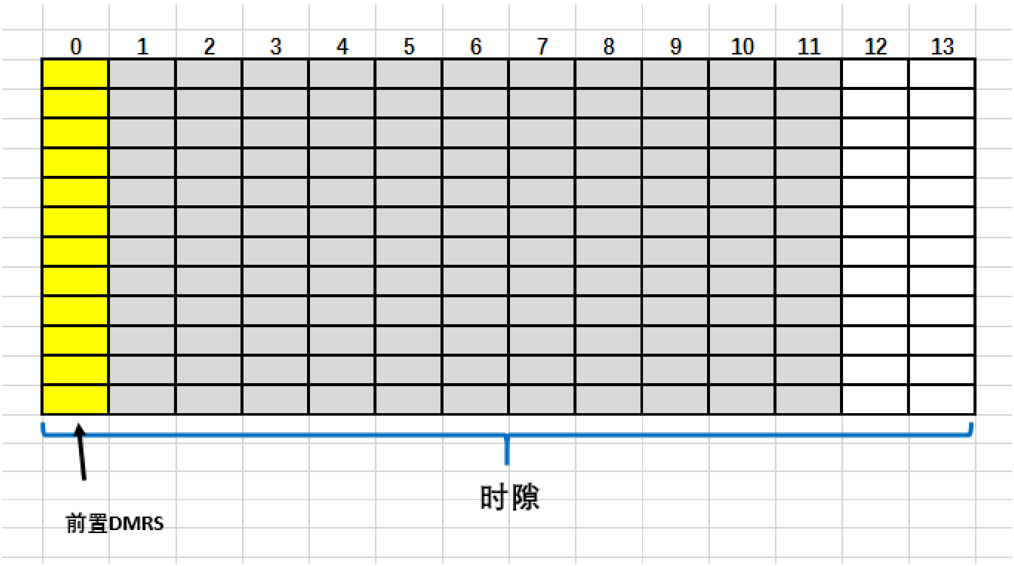
PUSCH中的DMRS

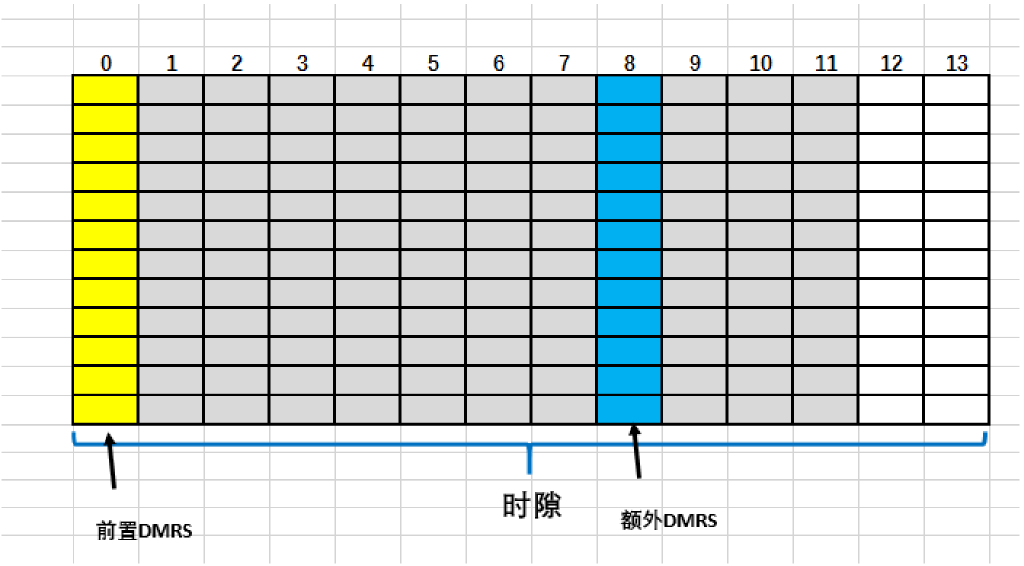
PUSCH的DMRS分类

与PDSCH类似，PUSCH也支持两种DMRS：

Front Loaded DMRS，前置DMRS，1~2符号，默认需要配置

Additional DMRS，额外DMRS，1~3符号，由高层配置；UE高速移动场景下进行更精准的信道估计





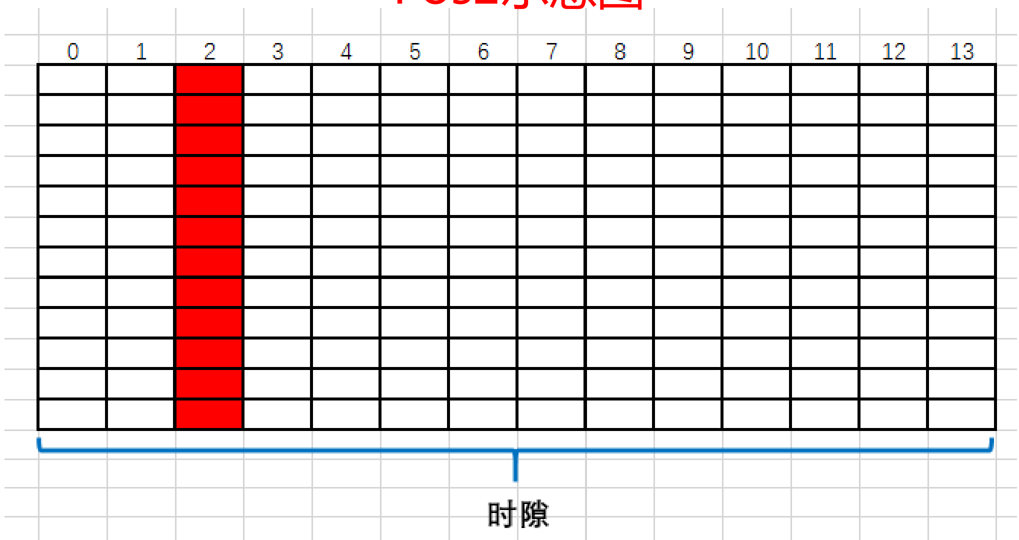
Front Loaded DMRS时域符号起始位

PUSCH 时域分布采用Type A的时候（现网）：

Front Loaded DMRS（前置DMRS）的时域符号起始位置，由MIB里面的dmrs-Type A-Position字段决定可以是符号2或者符号3（pos2，pos3）。这一点与PDSCH是一样的。

```
MIB ::= SEQUENCE {
    systemFrameNumber                   BIT STRING (SIZE (6)),
    subCarrierSpacingCommon             ENUMERATED {scs15or60, scs30or120},
    ssb-SubcarrierOffset                INTEGER (0..15),
    dmrs-TypeA-Position                 ENUMERATED {pos2, pos3},
    pdcch-ConfigSIB1                   INTEGER (0..255),
    cellBarred                          ENUMERATED {barred, notBarred},
    intraFreqReselection                ENUMERATED {allowed, notAllowed},
    spare                               BIT STRING (SIZE (1))
}
```

Pos2示意图



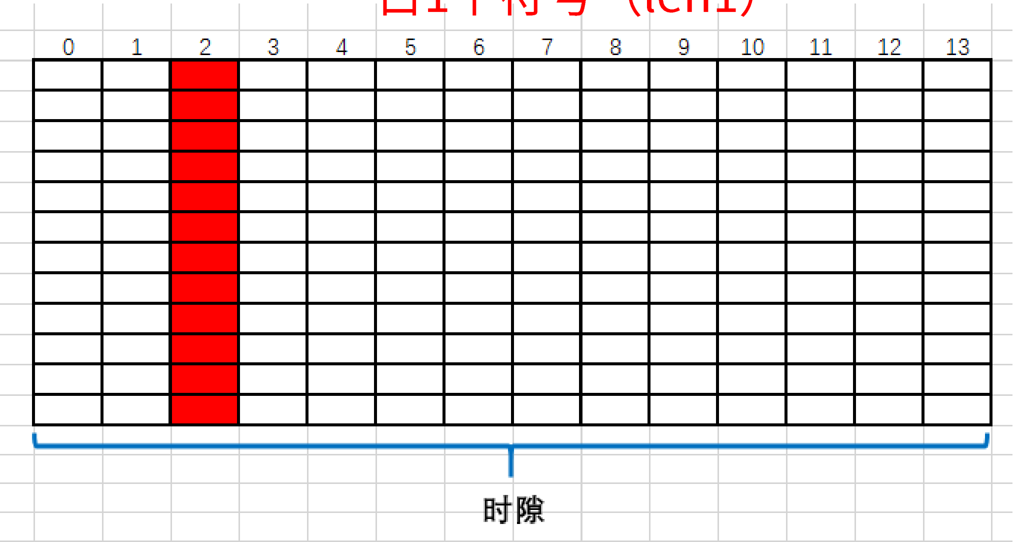
Pos3示意图

Front Loaded DMRS时域占用符号数

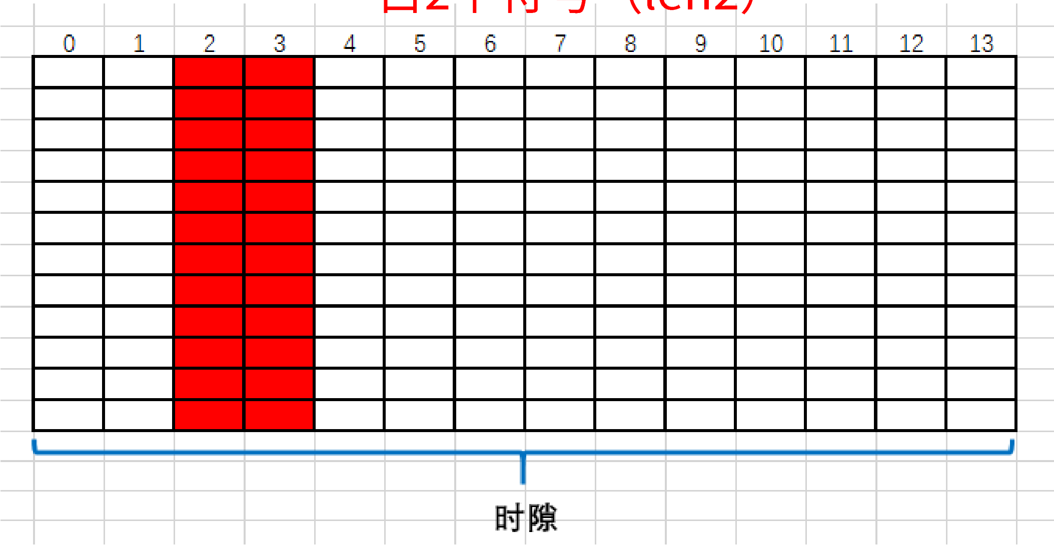
Front Loaded DMRS（前置DMRS）的时域占用符号数，由高层信令PUSCH-Config => DMRS-UplinkConfig => maxLength决定（len1、len2）

```
DMRS-UplinkConfig ::=                  SEQUENCE {
    dmrs-Type                              ENUMERATED {type2}
    dmrs-AdditionalPosition                ENUMERATED {pos0, pos1, pos3}
    phaseTrackingRS                        SetupRelease { PTRS-UplinkConfig }
    maxLength                              ENUMERATED {len2}
    transformPrecodingDisabled             SEQUENCE {
        scramblingID0                          INTEGER (0..65535)
        scramblingID1                          INTEGER (0..65535)
```

占1个符号（len1）



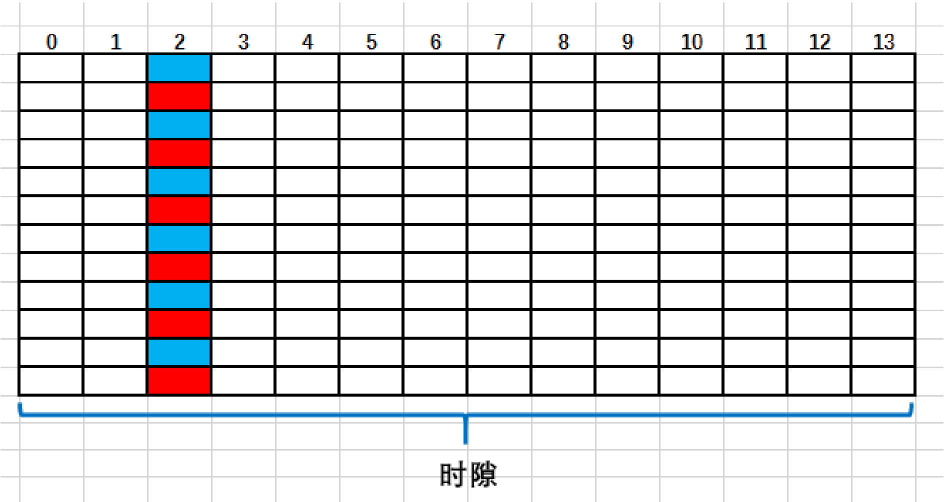
占2个符号（len2）



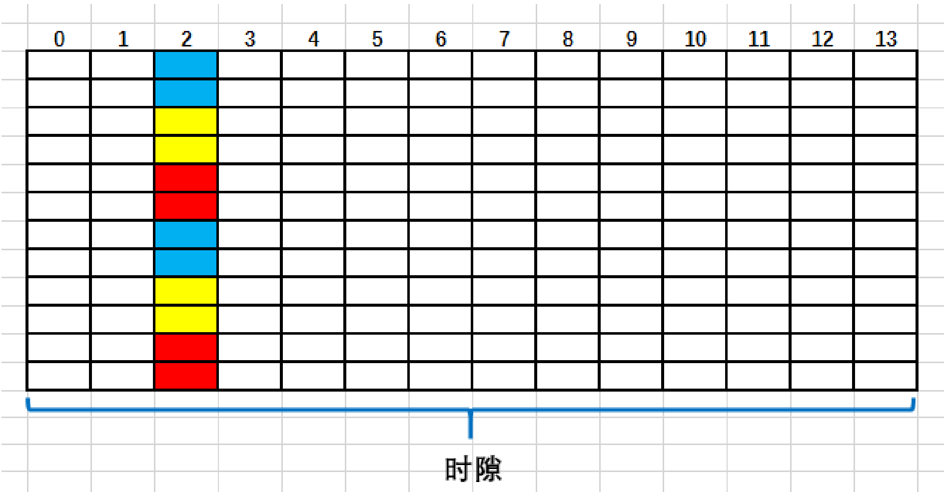
Front Loaded DMRS频域分布

DMRS频域映射方式分为 type 1和 type 2两种，与PDSCH一样。由高层信令PUSCH-Config => DMRS-UplinkConfig => drms-Type指示.如果该field未配置，则默认为 type 1。

```
DMRS-UplinkConfig ::=                    SEQUENCE {
    dmrs-Type                                ENUMERATED {type2}
    dmrs-AdditionalPosition                  ENUMERATED {pos0, pos1, pos3}
    phaseTrackingRS                          SetupRelease { PTRS-UplinkConfig }
    maxLength                                ENUMERATED {len2}
    transformPrecodingDisabled               SEQUENCE {
        scramblingID0                            INTEGER (0..65535)
        scramblingID1                            INTEGER (0..65535)
```



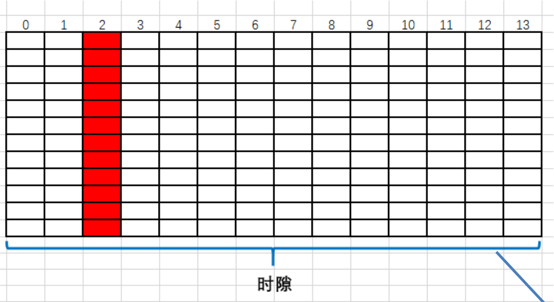
Type1，频域上间隔一个符号



Type2频域上连续2个符号做间隔

PUSCH有几个天线端口？

与PDSCH一样，不同的天线端口，对应不同的DMRS分布位置。



对于DMRS type1，天线端口有4或者8端口
对于DMRS type2，天线端口有6或者12端口
编号方式与PDSCH不同

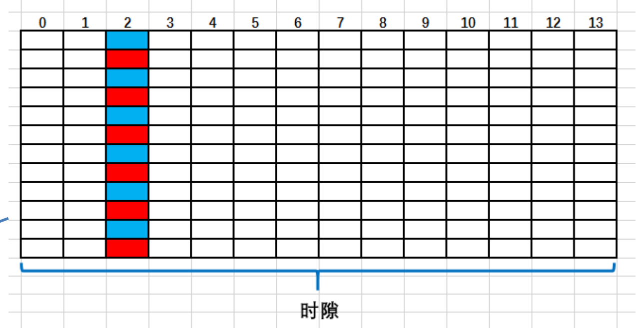
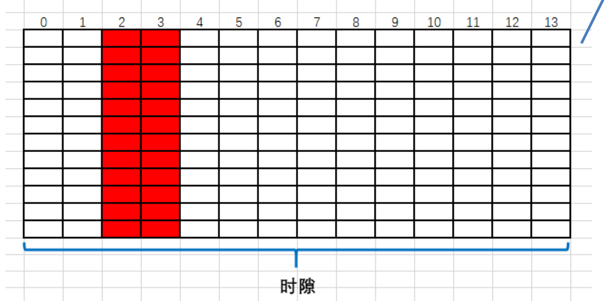


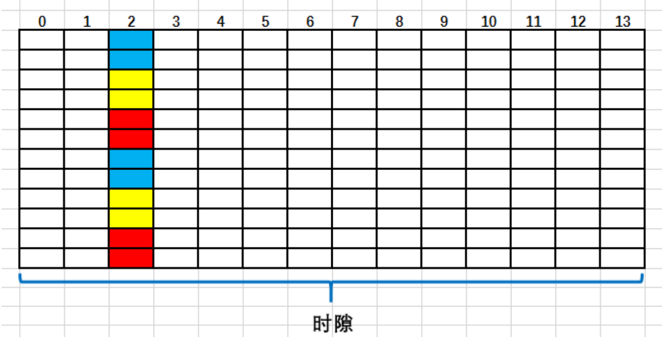
Table 6.4.1.1.3-5: PUSCH DM-RS time index l'.

DM-RS duration	l'	Supported antenna ports $\tilde{p}$	
		Configuration type 1	Configuration type 2
single-symbol DM-RS	0	0 – 3	0 – 5
double-symbol DM-RS	0, 1	0 – 7	0 – 11



PDSCH

Single or double symbol DM-RS	l'	Supported antenna ports p	
		Configuration type 1	Configuration type 2
single	0	1000 – 1003	1000 – 1005
double	0, 1	1000 – 1007	1000 – 1011



DMRS频域映射公式

DMRS频域分布公式有两种，与是否使用转换预编码有关

不使用转换预编码的时候，公式与PDSCH是一样的

$$k = \begin{cases} 4n + 2k' + \Delta & \text{Configuration type 1} \\ 6n + k' + \Delta & \text{Configuration type 2} \end{cases}$$

$$k' = 0,1$$

$$n = 0,1,\ldots$$

K值就是频域位置

使用转换预编码

$$k = 4n + 2k' + \Delta$$

$$k' = 0,1$$

$$n = 0,1,\ldots$$

K值就是频域位置

我们发现，
使用转换预编码
的时候，相当于
DMRS频域映射
只使用type1

是否使用转换预编码，高层信令PUSCH-Config => DMRS-UplinkConfig当中配置

```
DMRS-UplinkConfig ::=                    SEQUENCE {
    dmrs-Type                                ENUMERATED {type2}
    dmrs-AdditionalPosition                  ENUMERATED {pos0, pos1, pos3}
    phaseTrackingRS                          SetupRelease { PTRS-UplinkConfig }
    maxLength                                ENUMERATED {len2}
    transformPrecodingDisabled               SEQUENCE {
        scramblingID0                            INTEGER (0..65535)
        scramblingID1                            INTEGER (0..65535)
```

DMRS频域映射公式举例

不使用转换预编码的时候，DMRS映射与PDSCH一模一样

$$k = \begin{cases} 4n + 2k' + \Delta & \text{Configuration type 1} \\ 6n + k' + \Delta & \text{Configuration type 2} \end{cases}$$

$$k' = 0,1$$

$$n = 0,1,\ldots$$

K值就是频域位置

Table 6.4.1.1.3-1: Parameters for PUSCH DM-RS configuration type 1.

$\tilde{p}$	CDM group	Δ	$w_f(k')$ $k'=0$	$w_f(k')$ $k'=1$	$w_t(l')$ $l'=0$	$w_t(l')$ $l'=1$
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: Parameters for PUSCH DM-RS configuration type 2.

$\tilde{p}$	CDM group	Δ	$w_f(k')$ $k'=0$	$w_f(k')$ $k'=1$	$w_t(l')$ $l'=0$	$w_t(l')$ $l'=1$
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

举例子

假设使用Type1，天线端口4，

查表得知 Δ=0

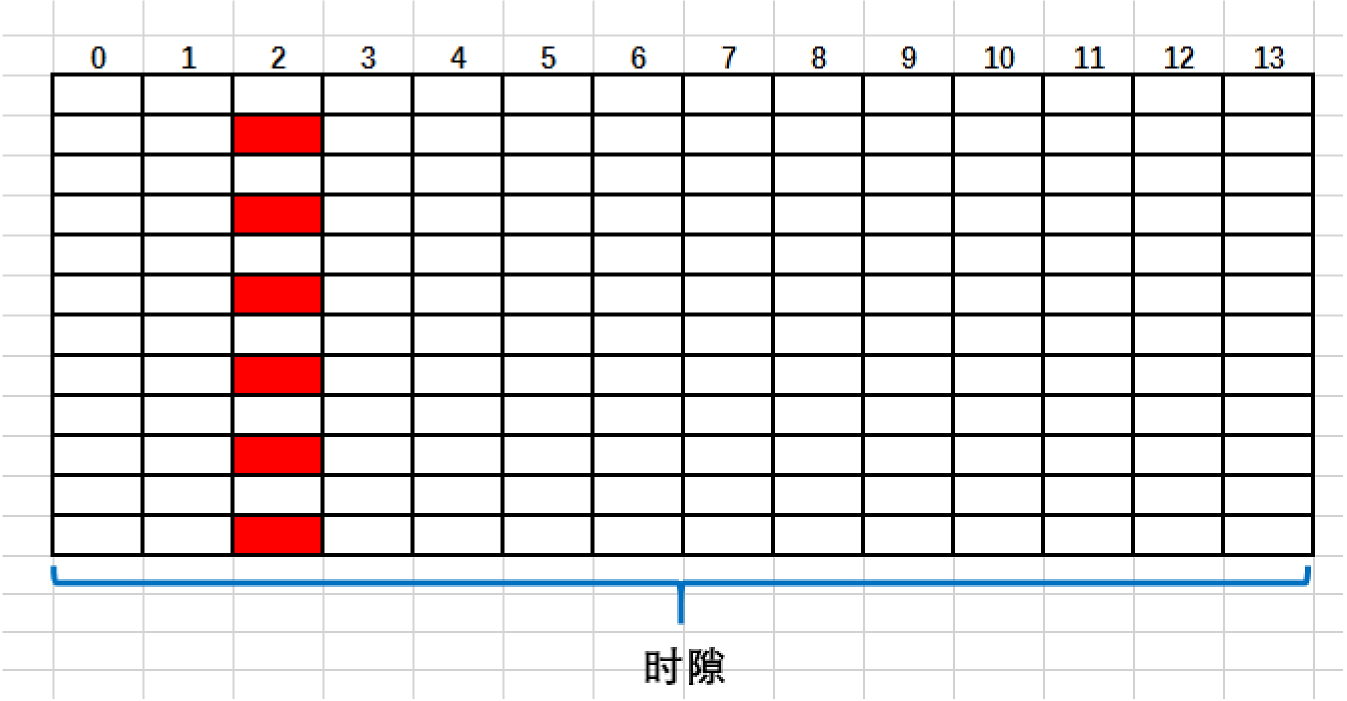
当K’=0 K=4n，因为n=0,1,2…得到K=0,4,8,12,16…

当K’=1 K=4n+2, 因为n=0,1,2…得到K=2,6,10,14…

所以，天线端口4,的DMRS频域位置就是：0.2.4.6…

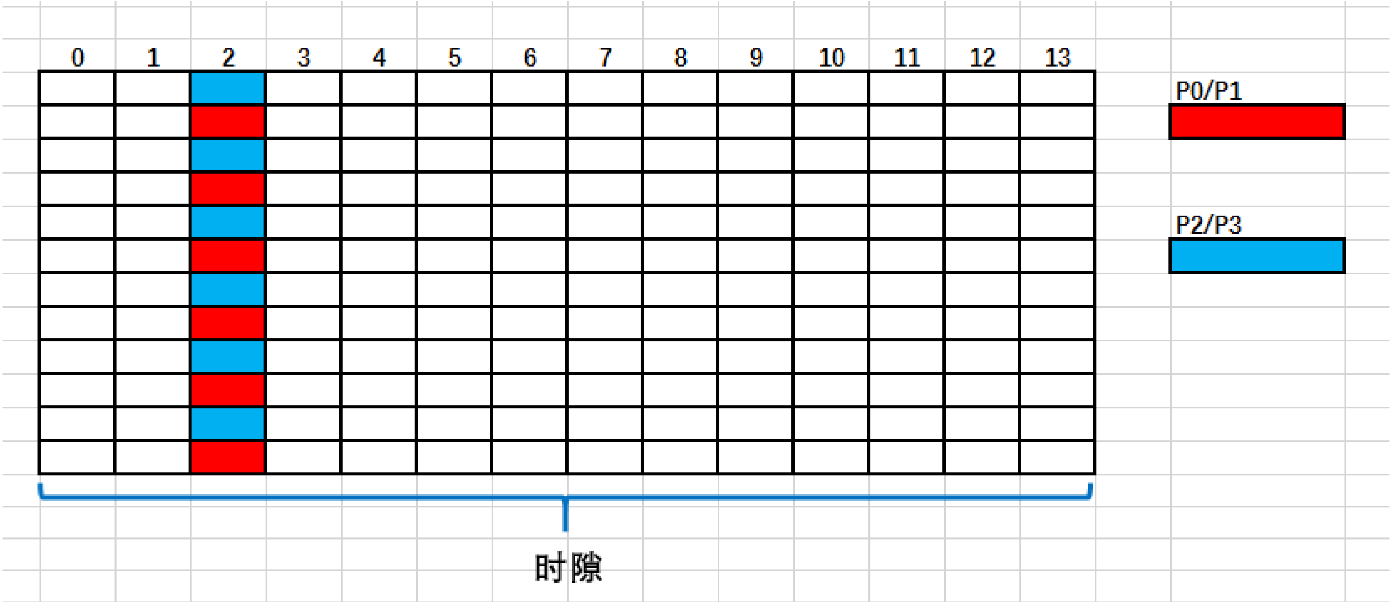
天线端口4的频域分布图

频域分布Type1,天线端口4,的DMRS频域位置就是：0.2.4.6.8.10…

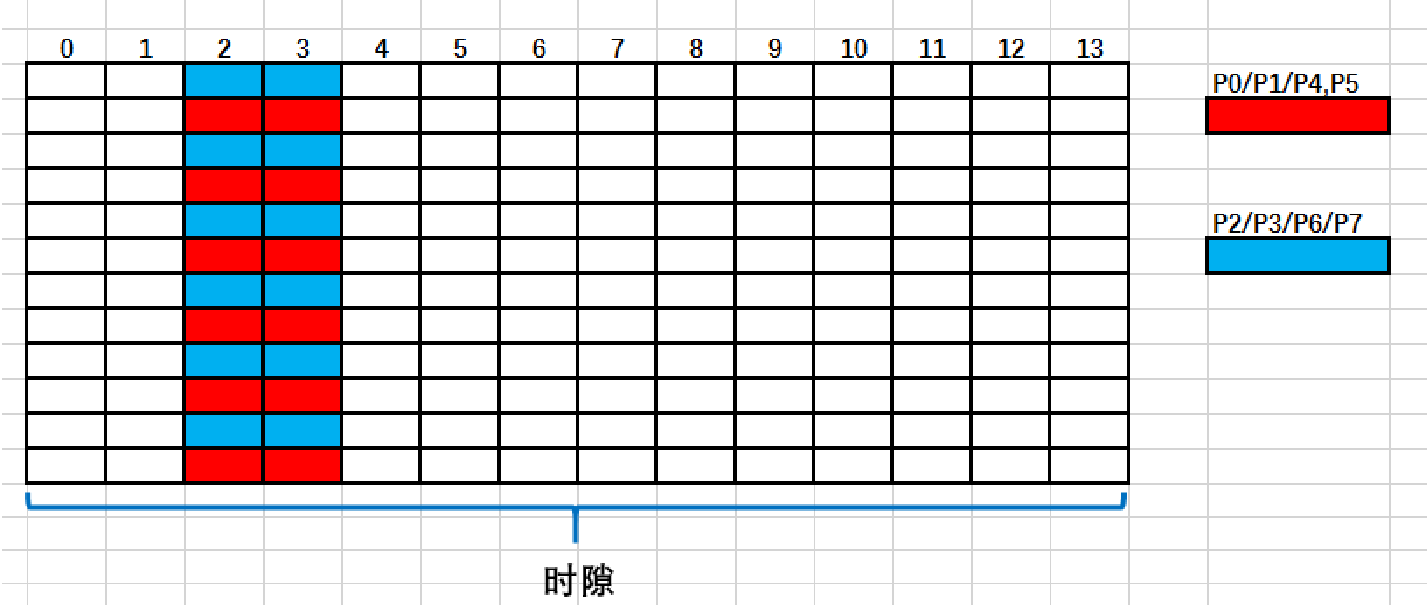


接下来我们把所有的天线端口对应的DMRS图全部画出来

DMRS type1

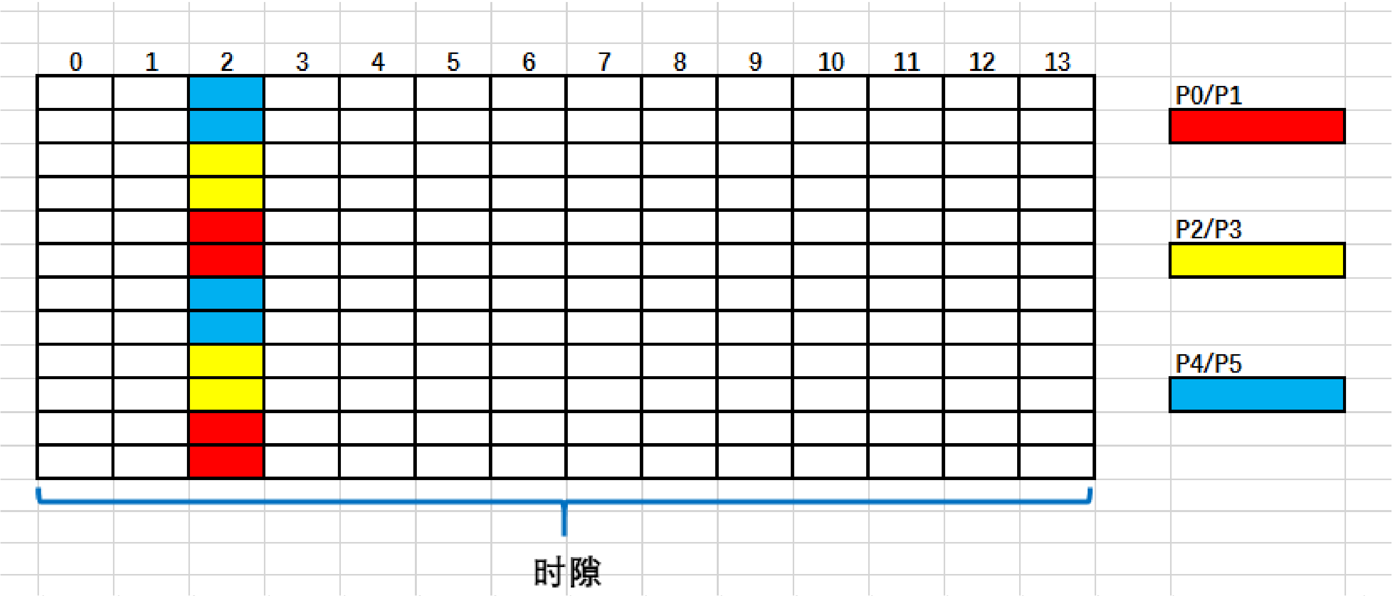


- 端口0,1码分复用，属于CDM组0
- 端口2,3码分复用，属于CDM组1

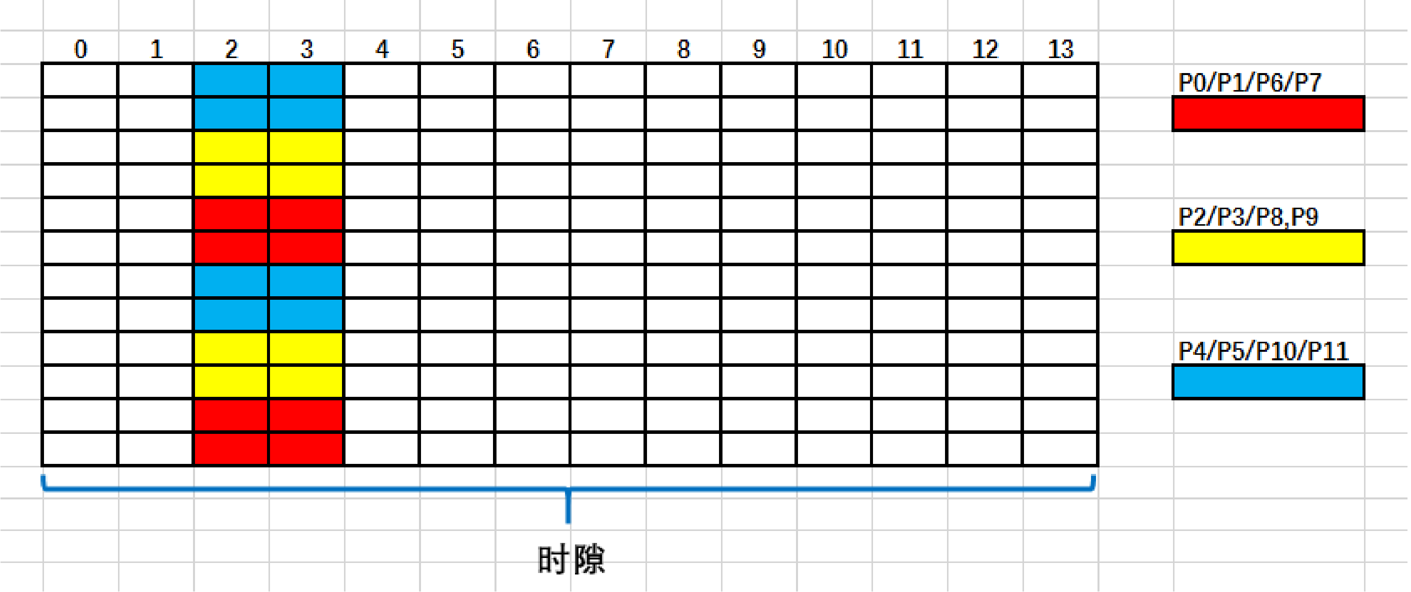


- 端口0,1,4,5码分复用，属于CDM组0
- 端口2,3,6,7码分复用，属于CDM组1

DMRS type2



- 端口0,1码分复用，属于CDM组0
- 端口2,3码分复用，属于CDM组1
- 端口4,5码分复用，属于CDM组2

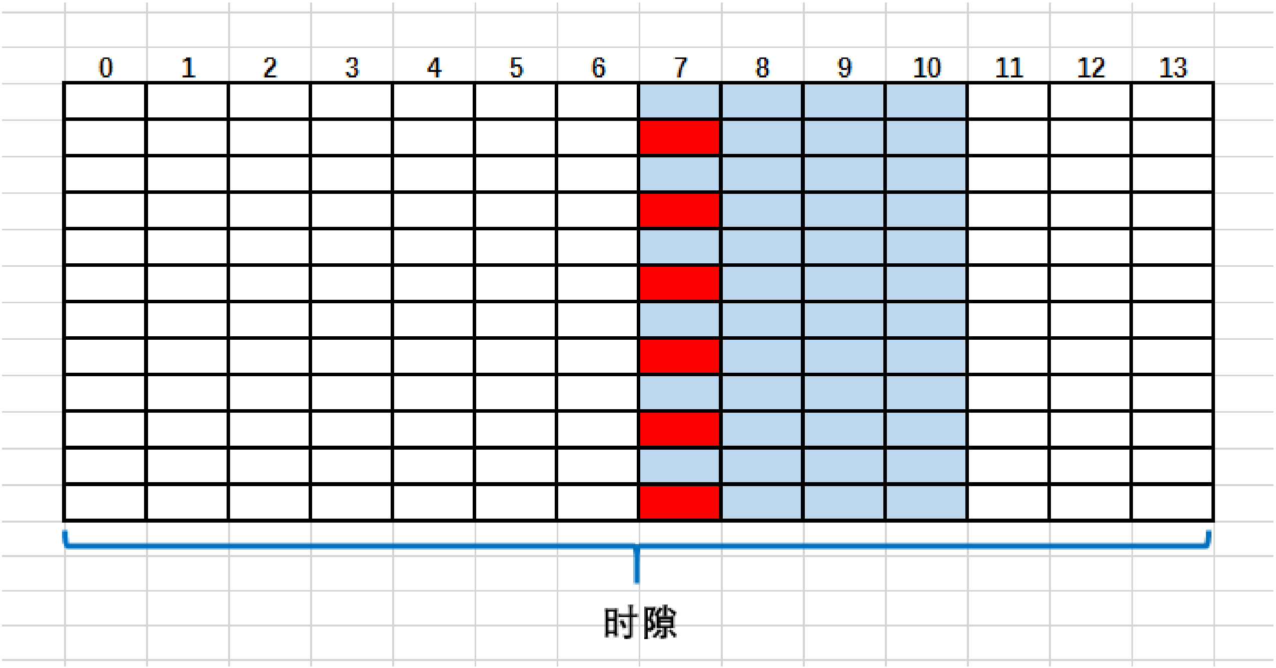


- 端口0,1,6,7码分复用，属于CDM组0
- 端口2,3,8,9码分复用，属于CDM组1
- 端口4,5,10,11码分复用，属于CDM组2

Front Loaded DMRS时域符号起始位

PUSCH 时域分布采用Type B的时候（mini slot）：

时域位置一般在PUSCH的第一个符号开始。



Additional DMRS

Additional DMRS（额外DMRS）为UE高速移动场景下进行更精准的信道估计。

由高层参数PUSCH-Config=>DMRS-uplinkConfig=>dmrs-Additional Position配置。

dmrs-Additional Position取值：pos0，pos1，pos2，pos3（代表数值0,1,2,3）

```
DMRS-UplinkConfig ::=                SEQUENCE {
    dmrs-Type                            ENUMERATED {type2}
    dmrs-AdditionalPosition              ENUMERATED {pos0, pos1, pos3}
    phaseTrackingRS                      SetupRelease { PTRS-UplinkConfig }
    maxLength                            ENUMERATED {len2}
    transformPrecodingDisabled           SEQUENCE {
        scramblingID0                        INTEGER (0..65535)
        scramblingID1                        INTEGER (0..65535)
```

参数代表additional DMRS的位置，需要查表来查询。

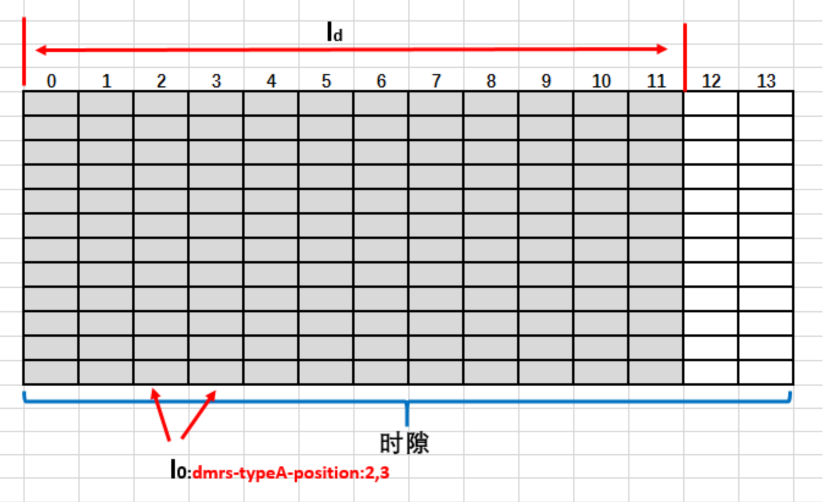
整体逻辑跟PDSCH是一模一样

DMRS单符号表

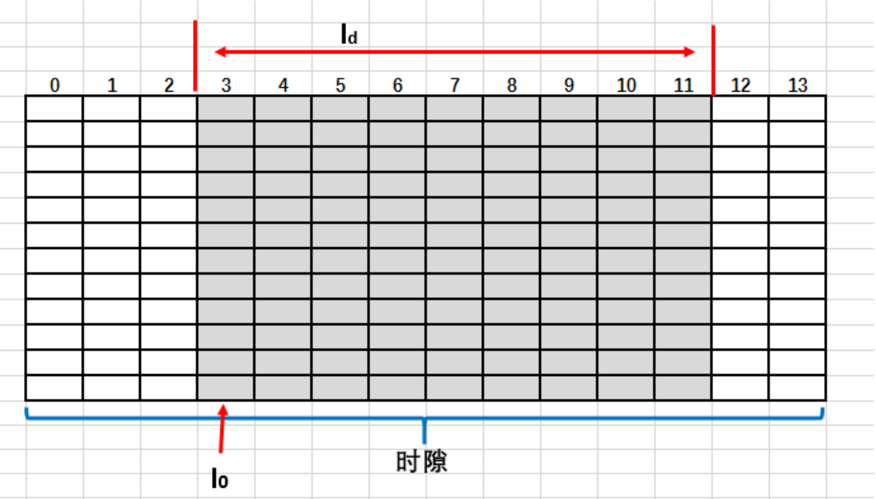
非时隙内跳频模式

Table 6.4.1.1.3-3: PUSCH DM-RS positions $\bar{l}$ within a slot for single-symbol DM-RS and intra-slo frequency hopping disabled.

l_d in symbols	DM-RS positions $\bar{l}$							
	PUSCH mapping type A				PUSCH mapping type B			
	dmrs-AdditionalPosition				*dmrs-AdditionalPosition*			
	pos0	*pos1*	*pos2*	*pos3*	*pos0*	*pos1*	*pos2*	*pos3*
<4	-	-	-	-	l_0	l_0	l_0	l_0
4	l_0	l_0	l_0	l_0	l_0	l_0	l_0	l_0
5	l_0	l_0	l_0	l_0	l_0	l_0, 4	l_0, 4	l_0, 4
6	l_0	l_0	l_0	l_0	l_0	l_0, 4	l_0, 4	l_0, 4
7	l_0	l_0	l_0	l_0	l_0	l_0, 4	l_0, 4	l_0, 4
8	l_0	l_0, 7	l_0, 7	l_0, 7	l_0	l_0, 6	l_0, 3, 6	l_0, 3, 6
9	l_0	l_0, 7	l_0, 7	l_0, 7	l_0	l_0, 6	l_0, 3, 6	l_0, 3, 6
10	l_0	l_0, 9	l_0, 6, 9	l_0, 6, 9	l_0	l_0, 8	l_0, 4, 8	l_0, 3, 6, 9
11	l_0	l_0, 9	l_0, 6, 9	l_0, 6, 9	l_0	l_0, 8	l_0, 4, 8	l_0, 3, 6, 9
12	l_0	l_0, 9	l_0, 6, 9	l_0, 5, 8, 11	l_0	l_0, 10	l_0, 5, 10	l_0, 3, 6, 9
13	l_0	l_0, 11	l_0, 7, 11	l_0, 5, 8, 11	l_0	l_0, 10	l_0, 5, 10	l_0, 3, 6, 9
14	l_0	l_0, 11	l_0, 7, 11	l_0, 5, 8, 11	l_0	l_0, 10	l_0, 5, 10	l_0, 3, 6, 9



PUSCH
typeA
Slot



PUSCH
typeB
minislot

举个例子

Table 6.4.1.1.3-3: PUSCH DM-RS positions $\bar{l}$ within a slot for single-symbol DM-RS and intra-slot frequency hopping disabled.

l_d in symbols	DM-RS positions $\bar{l}$							
	PUSCH mapping type A				PUSCH mapping type B			
	dmrs-AdditionalPosition				*dmrs-AdditionalPosition*			
	pos0	*pos1*	*pos2*	*pos3*	*pos0*	*pos1*	*pos2*	*pos3*
<4	-	-	-	-	l_0	l_0	l_0	l_0
4	l_0	l_0	l_0	l_0	l_0	l_0	l_0	l_0
5	l_0	l_0	l_0	l_0	l_0	l_0, 4	l_0, 4	l_0, 4
6	l_0	l_0	l_0	l_0	l_0	l_0, 4	l_0, 4	l_0, 4
7	l_0	l_0	l_0	l_0	l_0	l_0, 4	l_0, 4	l_0, 4
8	l_0	l_0, 7	l_0, 7	l_0, 7	l_0	l_0, 6	l_0, 3, 6	l_0, 3, 6
9	l_0	l_0, 7	l_0, 7	l_0, 7	l_0	l_0, 6	l_0, 3, 6	l_0, 3, 6
10	l_0	l_0, 9	l_0, 6, 9	l_0, 6, 9	l_0	l_0, 8	l_0, 4, 8	l_0, 3, 6, 9
11	l_0	l_0, 9	l_0, 6, 9	l_0, 6, 9	l_0	l_0, 8	l_0, 4, 8	l_0, 3, 6, 9
12	l_0	l_0, 9	l_0, 6, 9	l_0, 5, 8, 11	l_0	l_0, 10	l_0, 5, 10	l_0, 3, 6, 9
13	l_0	l_0, 11	l_0, 7, 11	l_0, 5, 8, 11	l_0	l_0, 10	l_0, 5, 10	l_0, 3, 6, 9
14	l_0	l_0, 11	l_0, 7, 11	l_0, 5, 8, 11	l_0	l_0, 10	l_0, 5, 10	l_0, 3, 6, 9

假设：

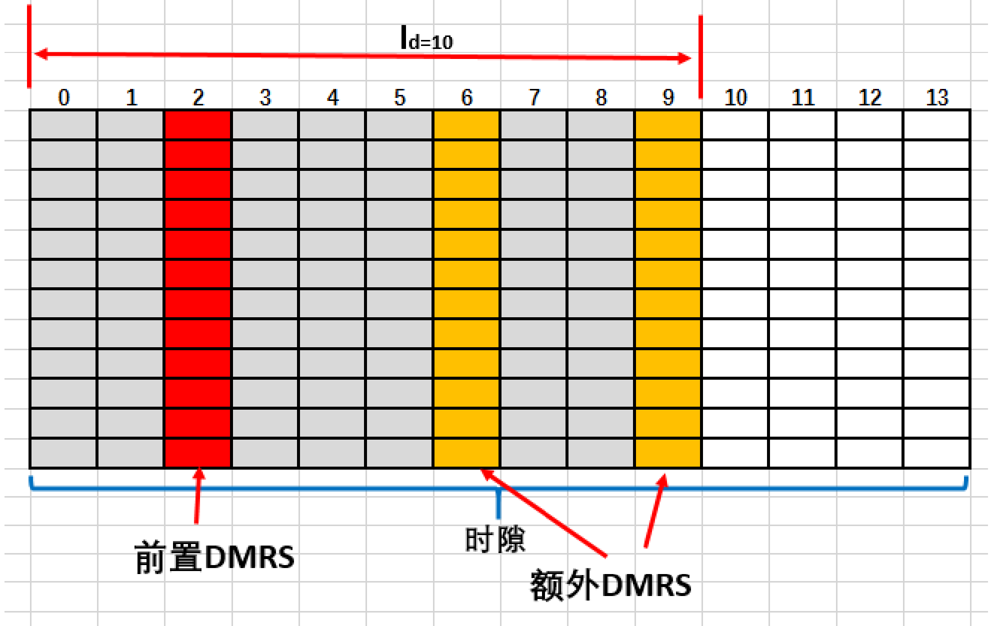
PUSCH使用typeA模式（时隙调度模式）

Dmrs时域使用的是单符号

dmrs-TypeA-Position =2

PUSCH占用symbol = 0~9

dmrs-Additional Position：2，结果如图所示



DMRS双符号表

非时隙内跳频模式

Table 6.4.1.1.3-4: PUSCH DM-RS positions $\bar{l}$ within a slot for double-symbol DM-RS and intra-slot frequency hopping disabled.

l_d in symbols	DM-RS positions $\bar{l}$							
	PUSCH mapping type A				PUSCH mapping type B			
	dmrs-AdditionalPosition				dmrs-AdditionalPosition			
	pos0	*pos1*	*pos2*	*pos3*	*pos0*	*pos1*	*pos2*	*pos3*
<4	-	-			-	-		
4	l_0	l_0			-	-		
5	l_0	l_0			l_0	l_0		
6	l_0	l_0			l_0	l_0		
7	l_0	l_0			l_0	l_0		
8	l_0	l_0			l_0	l_0, 5		
9	l_0	l_0			l_0	l_0, 5		
10	l_0	l_0, 8			l_0	l_0, 7		
11	l_0	l_0, 8			l_0	l_0, 7		
12	l_0	l_0, 8			l_0	l_0, 9		
13	l_0	l_0, 10			l_0	l_0, 9		
14	l_0	l_0, 10			l_0	l_0, 9		

DMRS单符号表

时隙内跳频模式　　对于时隙内跳频模式，DMRS仅支持单符号。

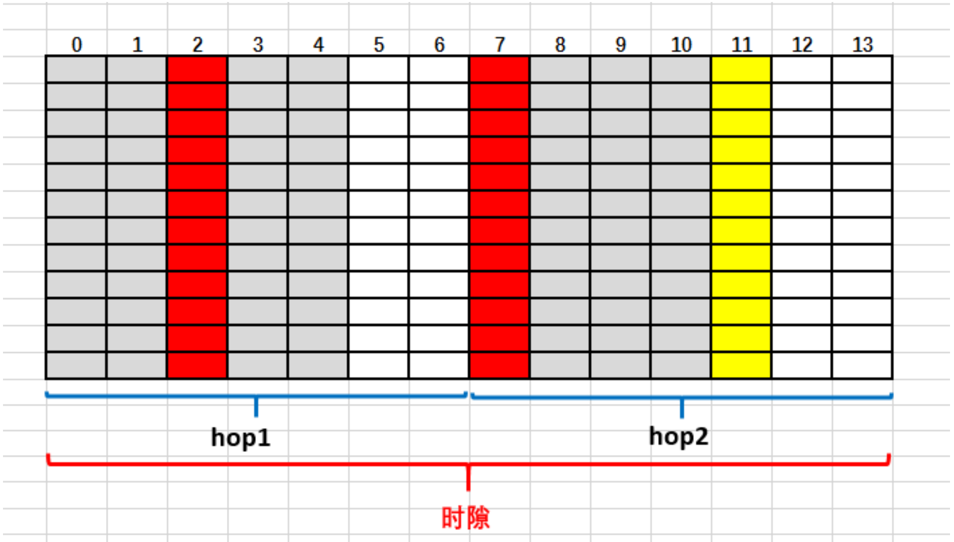
Table 6.4.1.1.3-6: PUSCH DM-RS positions $\bar{l}$ within a slot for single-symbol DM-RS and intra-slot frequency hopping enabled.

l_d in symbols	DM-RS positions $\bar{l}$											
	PUSCH mapping type A								PUSCH mapping type B			
	$l_0 = 2$				$l_0 = 3$				$l_0 = 0$			
	dmrs-AdditionalPosition				*dmrs-AdditionalPosition*				*dmrs-AdditionalPosition*			
	pos0		*pos1*		*pos0*		*pos1*		*pos0*		*pos1*	
	1st hop	2nd hop	1st hop	2nd hop	1st hop	2nd hop	1st hop	2nd hop	1st hop	2nd hop	1st hop	2nd hop
≤3	-	-	-	-	-	-	-	-	0	0	0	0
4	2	0	2	0	3	0	3	0	0	0	0	0
5, 6	2	0	2	0, 4	3	0	3	0, 4	0	0	0, 4	0, 4
7	2	0	2, 6	0, 4	3	0	3	0, 4	0	0	0, 4	0, 4

假设：
PUSCH使用typeA模式（时隙调度模式）
Dmrs时域使用的是单符号

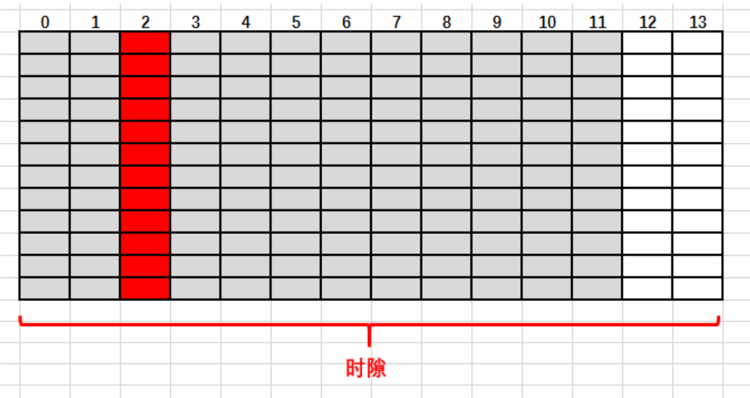
dmrs-TypeA-Position =2

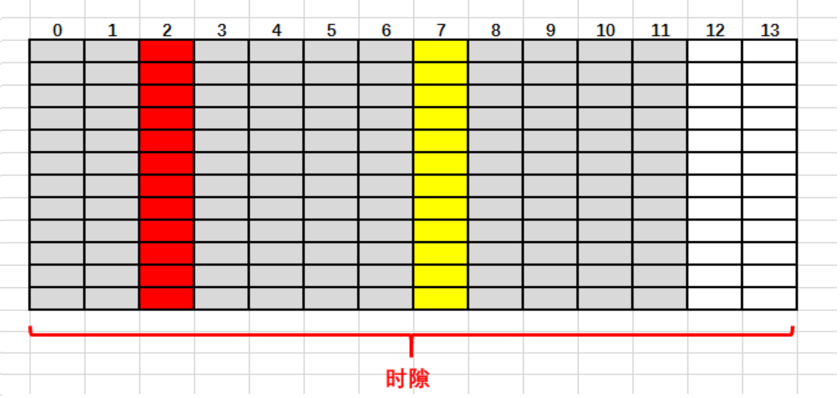
PUSCH占用symbol = 0~4
dmrs-Additional Position：1，结果如图所示

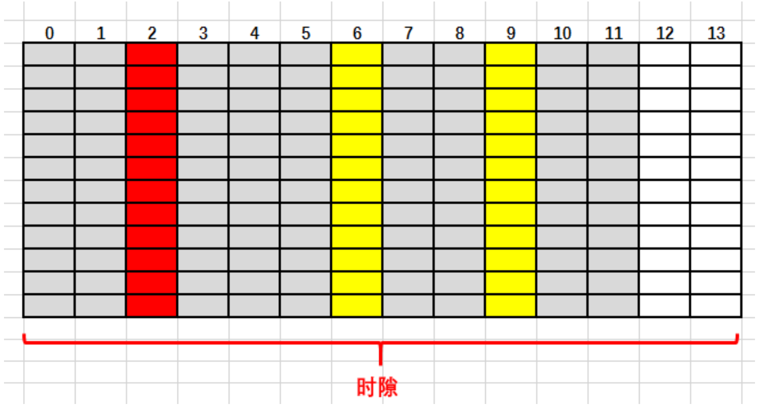


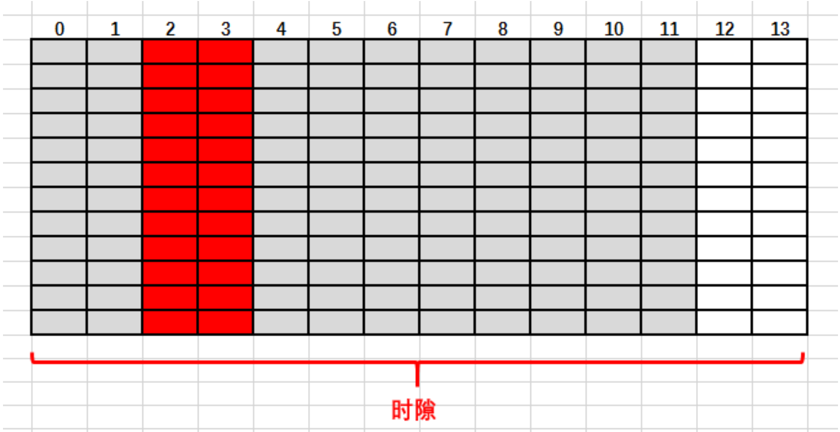
PUSCH typeA DMRS全景图

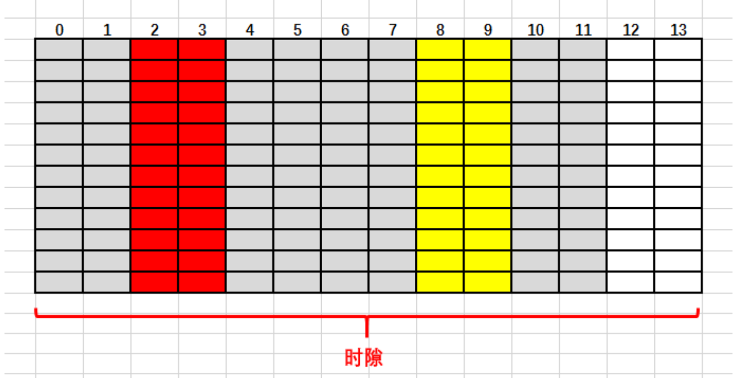
Front Loaded DMRS　　Additional DMRS　　非跳频









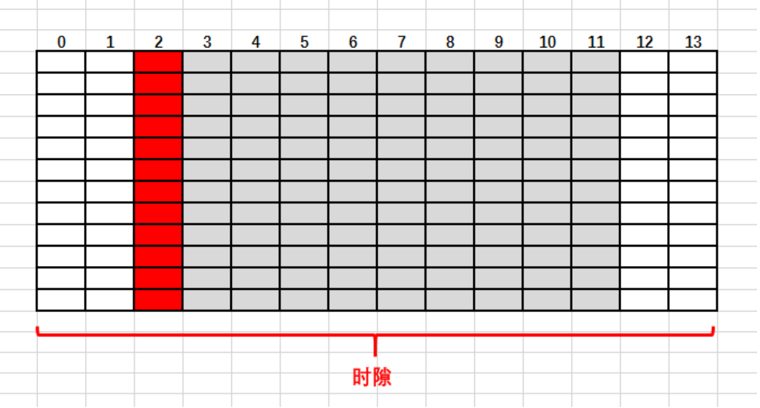


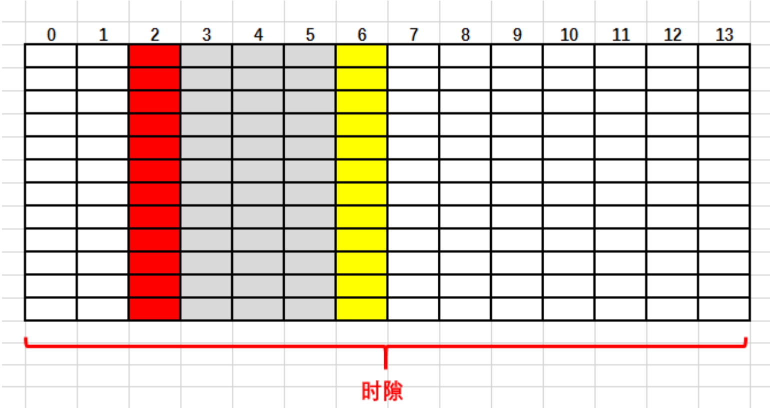
PUSCH typeB DMRS全景图

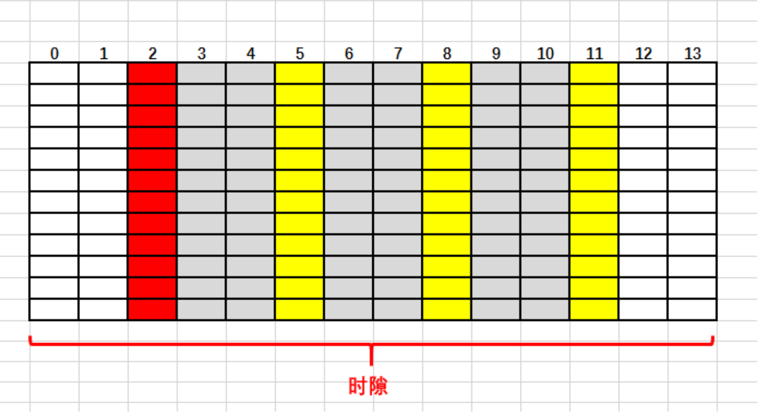
Front Loaded DMRS

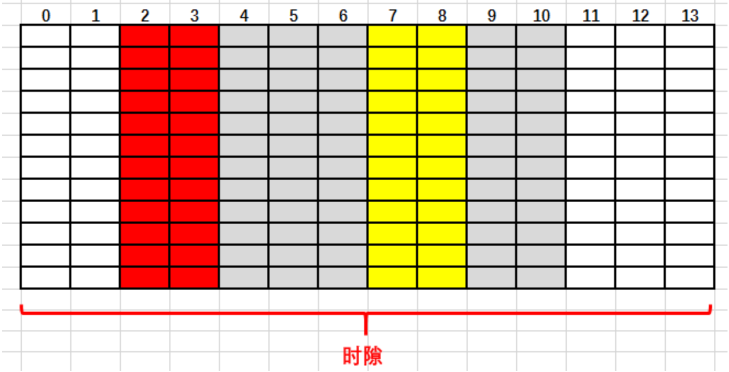
Additional DMRS

非跳频



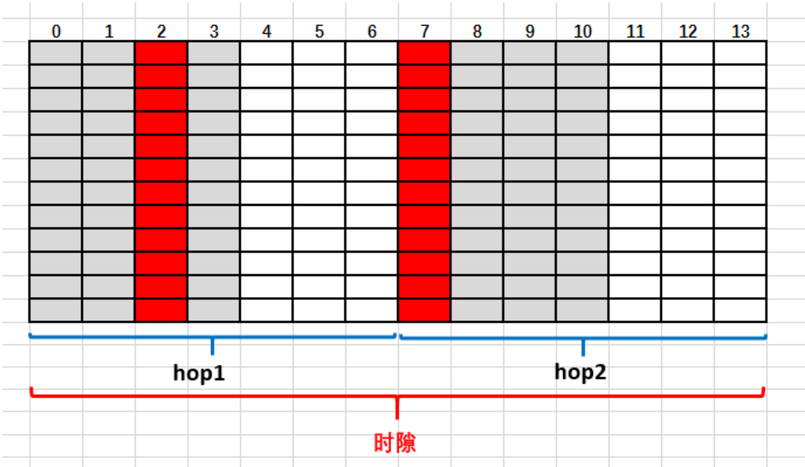


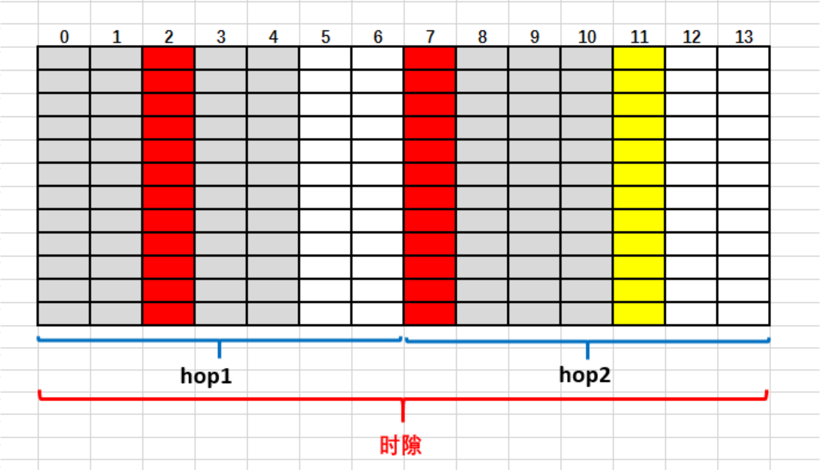


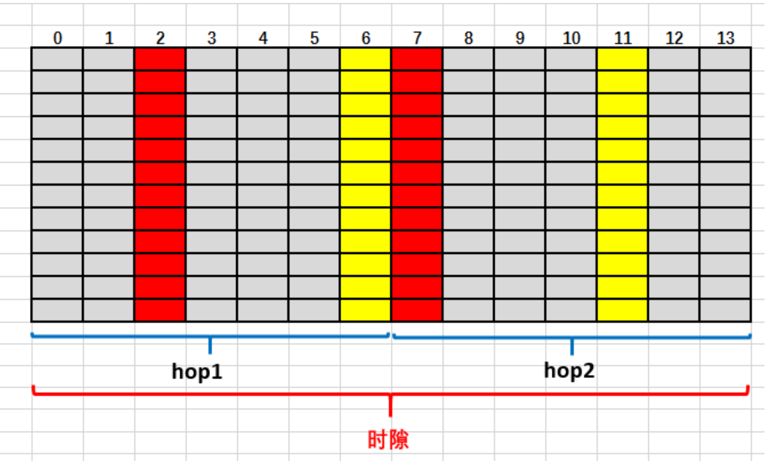


PUSCH typeA DMRS全景图

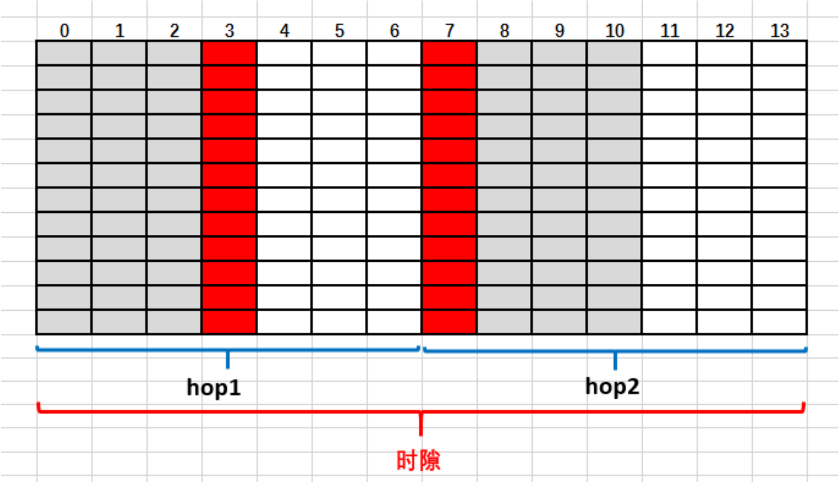
Front Loaded DMRS   Additional DMRS   跳频

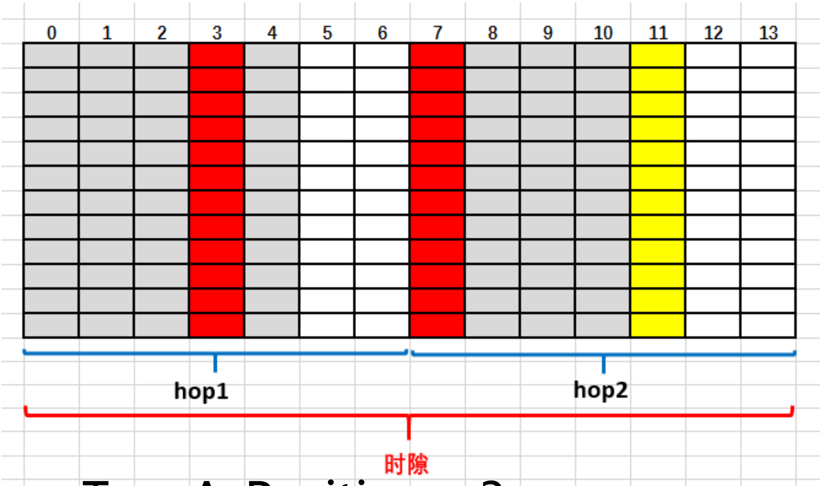






dmrs-TypeA-Position =2





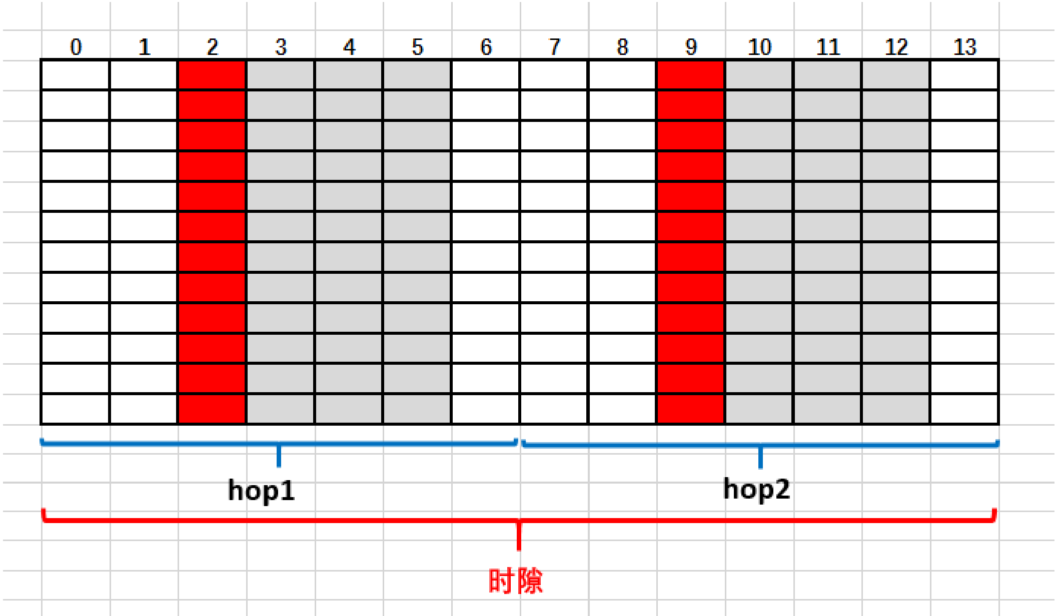
dmrs-TypeA-Position =3

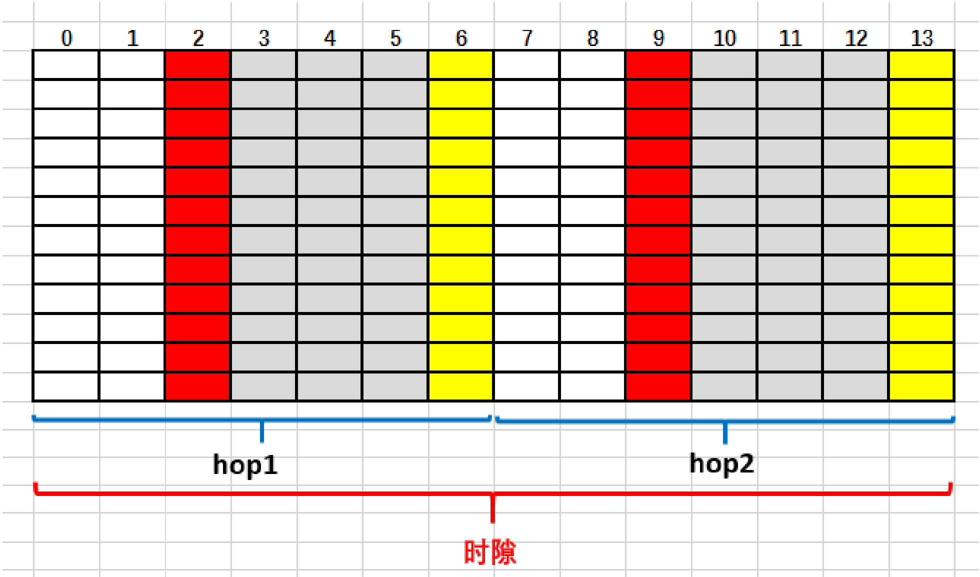
PUSCH typeB DMRS全景图

Front Loaded DMRS

Additional DMRS

跳频





感谢观看