

## 1 机制的来源 ---- 哲学

- 想出来的，协议或规定，特别是‘恰当（中庸的思想），极端就是毁灭’

就像 TDD 没有沿用 3G 的上下行随便配置的方法，但也不能只有一种配置，这样太死板，所以折中之后提取出了七种比较有意义的帧结构模型。

- 具体问题具体分析。不能生搬硬套，要根据具体的情况订出具体的策略。后面介绍每种信道的时候就能看出来，每种信道的处理几乎都不一样，没有一种完全统一的方式。
- 就像数学推论一样，当问一个为什么，不断问下去的时候？最后要不是规定或者设计思想；就要不是‘公理，定理’，根本没法证明。
- 任何事情都没有完美的，有利有弊，只是看你有没有发现而已。
- 配置出来的
- 潜规则，这是一种规则但并没有显示表示（在代码中也有同样的。由于潜规则不容易发现而且难于理解，最好少用）

注：也许这些看起来比较空洞，但当你看完了后面的信道实现再反过来看的时候，就能很好的感觉这些思想的意义了。

## 2 后面讨论的一些限制

- 只涉及 TDD-LTE，TDD 比较复杂些，想清楚了它，FDD 自然也好理解
- 只涉及子载波是 15kHz 的情况
- 只讨论‘一个时隙有 7 个 symbol 的情况’，也就是 normal 循环前缀（Normal cyclic prefix）的情况。不讨论 Extended cyclic prefix 的情况
- 不讨论半静态调度，也许偶尔会涉及到
- 不讨论 MIMO 的情况
- 看的都是 860 的协议，分别是 36211-860, 36212-860, 36213-860
- 注：调制之后也产生符号，而一个资源块 RB 也是时域上也是有符号的概念。所以为了两者区别，‘调制符号’就是指‘调制之后也产生符号’；而正常的‘符号’就是指‘时域的符号’的概念。

## 3 LTE 整体理解

### 3.1 生活交流就是 LTE -- 设计思想

让我们从生活的角度来简单理解下‘通讯’，自己想出来的，有些也可能不太准确，只是想表达一种意思。假设 eNodeB，UE 都是人，是一个 eNodeB 同时和多个 UE 进行交流。

- **加扰**：由于 eNodeB 和每个 UE 谈话的时候，都不想别人听得懂它们之间的谈话的内容。所以 eNodeB 和每个 UE 谈话的时候，都用一种不同的语言，这也就相当于别的人虽然听到了，但是听不懂。相当于通讯中加扰。
- **功控**：由于 eNodeB 和多个 UE 都在一个环境谈话。如果一个 UE 讲得太小，eNodeB 听不到，eNodeB 就会让那个 UE 说话声音大点；如果 UE 说话声音太大了，又吵着了 eNodeB 和其他人谈话，所以太大了又会让那个 UE 说话小声点。就这样不停的根据环境变化说话声音的大小，这也就是‘通讯中的功控了’，当然 eNodeB 肯定也会控制自己说话的音量的。
- **编码率（CQI 决定）**：eNodeB 和 UE 之间谈话，觉得 UE 说话太快了，听不清楚，就会跟 UE 说，你说话慢点；这样 UE 每一个分钟说的话也就少了，表达的意思就少了，当然这也是根据环境不断变化的；反过来也一样。这也就是通讯中‘编码率’，表达了选择到的那块资源（时间+频域）所能携带的，由 CQI（channel quality indication）决定的。由于只能让听的人来决定说话是否快慢，所以：通讯中下行就是通过 UE 上报的 CQI—channel quality indication 决定下行编码率，因为 UE 是听者；上行 eNodeB 自己来判断 CQI—channel quality indication 决定上行编码率，因为 eNodeB 是听者。
- **ASN 编码方式**：就像人说话是否精练一样。同样的字数能传递的信息数是不一样的，像电报就要求比较精炼。无线侧的 ASN 编码就像人说话很干练；而有线侧 TLV 的 ASN 编码模式就相当于说话比较啰嗦。
- **资源位置的选择（CQI 决定）**：eNodeB 可以让 UE 站在不同的地方，看看它听 eNodeB 说话的效果怎么样，或者让 UE 站在各个地方说‘事先订好大家都知道的话’。哪里 eNodeB 听得最清楚，最后 eNodeB 就说你就站在那里说话吧，那里说话听得最清楚。这也就是通讯中‘资源位置的选择’，就是通过‘不同资源上返回的 CQI，去选择 CQI 最好的资源进行分配，当然这只是理想情况’。此时说话的内容都是事先订好的，这也就是通讯中的 RS（参考信号的作用），RS 还有个作用‘相干解调’，后面会介绍。
- **资源数目的选择**：用说话不好做比喻。就用货物运输吧。UE 说我有很多货要送。eNodeB 说我就给你多拍几辆车来送货把。这就是资源数目的意思了。
- **调度**：一个 eNodeB 和多个 UE 之间对话，每个人都有话要说，每个人可能要讲好几件事，每件事重要程度也是不一样的（这也就是通讯中 DRB 的优先级），每件事说多少话也是不一样。而且有些 UE 的话重要，有些不太重要（这也就是 UE 的调度优先级）。但 eNodeB 又忙不过来，它就去决定什么时候和某个 UE 对话，什么时候又听 UE 说话，分配多少时间给某个 UE，分配多少车辆给 UE 送货（因为总的车辆数是一定的，也就是上下行带宽），最后调度就决定最后怎么去做。
- **正交**：想到一个比喻但不是太恰当。就像一盘有各种颜色的珠子混在一起，然后你用自己对应的颜色，就能从混在一起的珠子中选出你自己想要的颜色的珠子。颜色就相当于正交码；用想要的颜色去匹配的动作就是正交运算。

### 3.2 一些设计基本原则 -- 设计思想

- 为了防止小区间干扰，通常通用的会通过 PCI (physical cell id) 进行偏移计算或者‘参与加扰计算’来防止干扰；如果和时间（时隙 0~19）的变化相关，还加上‘时间’参与加扰。
- 为了防止小区内不同 UE 的干扰或者~~决定~~ UE 的资源分配位置，通过一个与无线侧 UE 相关的唯一标识--‘RNTI’进行加扰或者定位资源分配的位置。考虑到，如果资源分配的位置还有冲突，可能还会加入一个系统内相对的子帧号（0~9）或者时隙号（0~19）来解决这种资源冲突，让这种冲突再下一个时间点能得到解决，也就是资源分配的位置由 RNTI 和子帧号/时隙号共同决定。当然也会加上 PCI 来区分不同小区之间的不同 UE。
- 为了‘离散化’数据，一般喜欢‘横放列取’的方法。
- 由于‘空口最大的一个缺陷就是资源少’，所以为了尽量节省资源，产生了很多潜规则，而且也有时会‘1bit 当 2bit 用，就是说不同的外部条件下，该 1bit 代表不同的意思’。这样虽然节省了资源，但这样的不利就是‘算法和限制条件太多了太烦了’。
- 要是‘没有了 TDD’，也许思路该清静/清晰很多了。看物理层协议，TDD 由于上下行配置的多样性和不对称性，产生了非常多的额外的处理问题，特别是 HARQ ACK/NACK 的处理。

### 3.3 基准时间单位 --- 规定

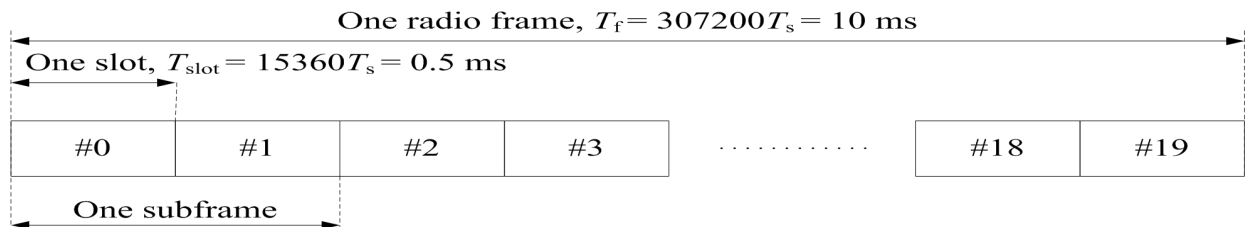
$$T_s = 1/30\ 720\ 000\ S$$

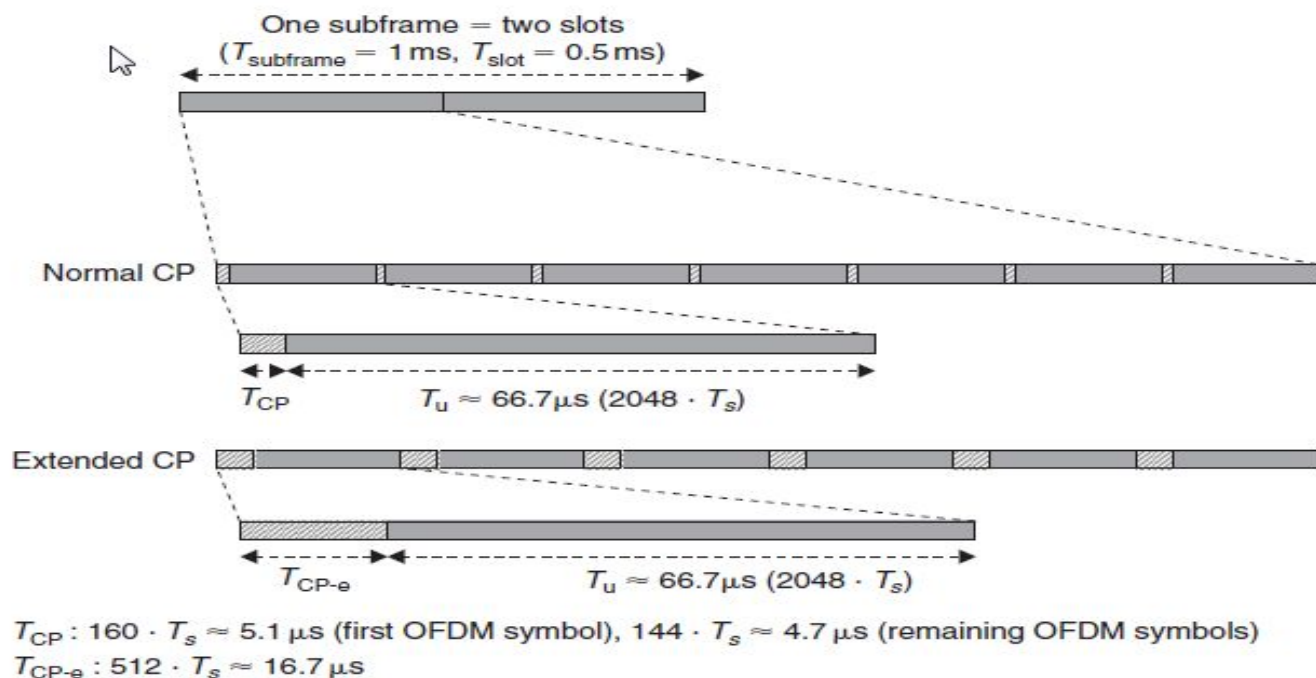
这个的意思就是说‘每 1 秒，每个天线端口都会发送出 30 720 000 个‘调制符号’出去’。

### 3.4 FDD 和 TDD 的帧结构 -- 规定

#### 3.4.1 FDD 帧的结构

FDD 的配置，对称的(上下行不同的频点)





系统帧，子帧，时隙，符号（symbol）与时间单位的关系

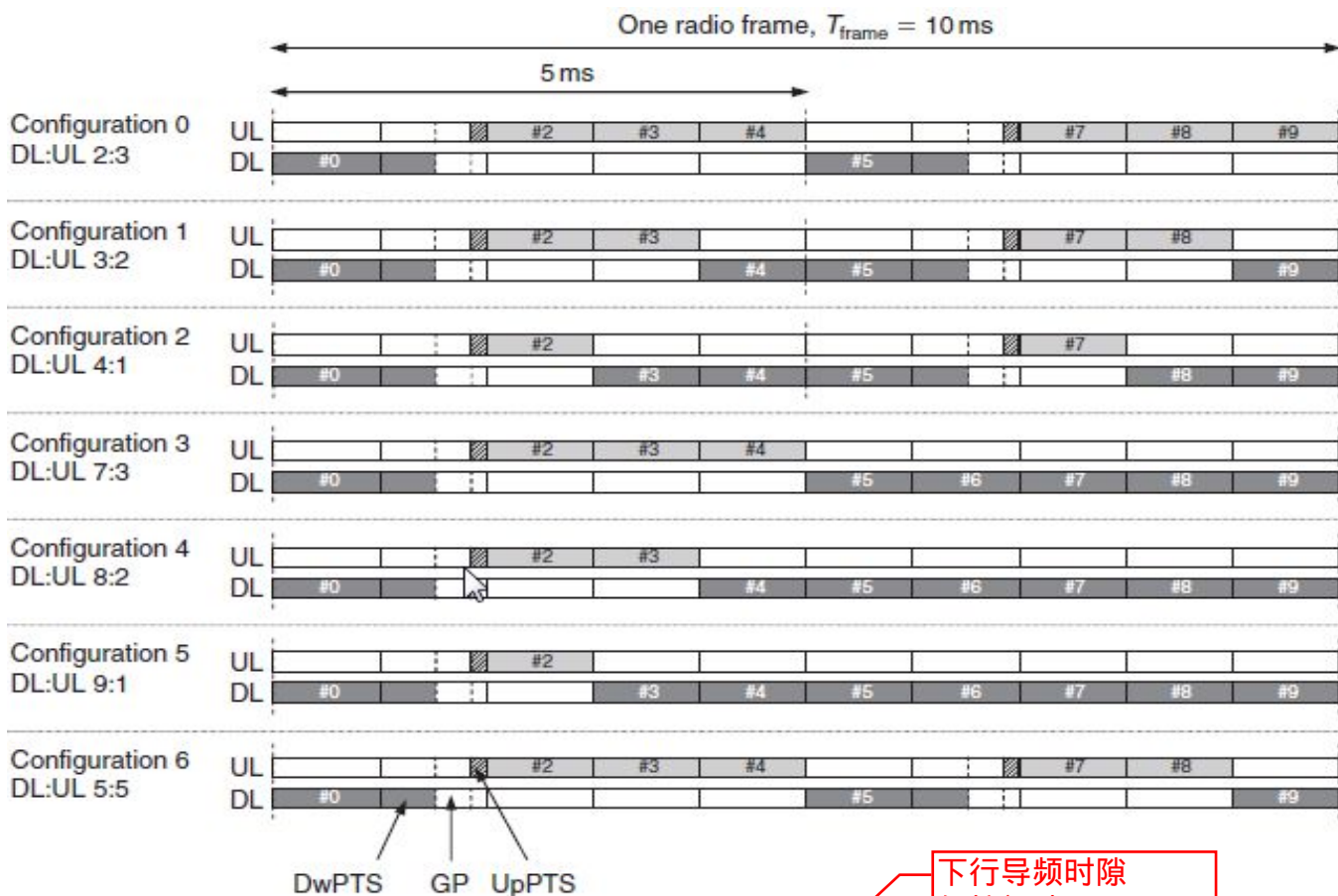
$T_{\text{frame}} (307200 \cdot T_s = 10 \text{ ms}) \rightarrow 10 \cdot T_{\text{subframe}} (30720 \cdot T_s = 1 \text{ ms}) \rightarrow$   
 $2 \cdot T_{\text{slot}} (15360 \cdot T_s = 0.5 \text{ ms}) \rightarrow 7/6 \text{ symbol} (2048 \cdot T_s = 66.7 \mu\text{s}).$

### 3.4.2 TDD 帧的结构

#### 3.4.2.1 思想

TDD 的几种配置，可以不对称

- 思想（折中）：就像 TDD 没有沿用 3G 的上下行随便配置的方法，但也不能只有一种配置，这样太死板，所以折中之后提取出了七种比较有意义的帧结构模型。
- 参看：36211 的 Table 4.2-2



下行导频时隙  
保护间隙  
上行导频时隙

- 0 和 5 这两个子帧都必须是下行，2 必须是上行。
- 帧结构的配置可以改变，但不能改变得太快，不能每个系统帧都变一下
- 为了防止小区间干扰，相邻小区的上下行配置最好一样
- 特殊子帧只有下行转换到上行之间才有
- 帧结构和特殊指针的 DWPTS/GP/UPPTS 的时长都是由系统信息通知给手机的
- 使用那种时隙结构，是基于每个子帧都可以变化的。一般'扩展的 CP'就是给 MBMS 子帧用的。
- 后面就能知道由于'一个帧内的上下行子帧的数目不一样'这种不对称的配置，最后导致很多特殊的处理出来。也许现在还不太了解，看完后面的说明应该就了解了。

### 3.4.2.2 配置

- RRC::SystemInformationBlockType1 → TDD-Config → subframeAssignment

### 3.4.3 TDD 特殊子帧的结构

- RRC::SystemInformationBlockType1 → TDD-Config → specialSubframePatterns  
决定特殊子帧的配置。

36211 的 Table 4.2-1: Configuration of special subframe (lengths of DwPTS/GP/UpPTS).

Special subframe configuration	Normal cyclic prefix in downlink		Extended cyclic prefix in downlink		UpPTS	
	DwPTS	Normal cyclic prefix in uplink	Extended cyclic prefix in uplink	DwPTS	Normal cyclic prefix in uplink	Extended cyclic prefix in uplink
0	(3symbol + 16ts) 6592 · T <sub>s</sub>	2192 · T <sub>s</sub>	2560 · T <sub>s</sub>	7680 · T <sub>s</sub>	2192 · T <sub>s</sub>	2560 · T <sub>s</sub>
1	(9symbol) 19760 · T <sub>s</sub>			20480 · T <sub>s</sub>		
2	(10symbol) 21952 · T <sub>s</sub>			23040 · T <sub>s</sub>		
3	(11symbol) 24144 · T <sub>s</sub>			25600 · T <sub>s</sub>		
4	(12symbol) 26336 · T <sub>s</sub>			7680 · T <sub>s</sub>	4384 · T <sub>s</sub>	5120 · T <sub>s</sub>
5	6592 · T <sub>s</sub>	4384 · T <sub>s</sub>	5120 · T <sub>s</sub>	20480 · T <sub>s</sub>		
6	19760 · T <sub>s</sub>			23040 · T <sub>s</sub>		
7	21952 · T <sub>s</sub>			-	-	-
8	24144 · T <sub>s</sub>			-	-	-

- 注意上表的红色部分，对应的符号 symbol 数，因为 PDCCH 要占用 1~3 (normal) 符合，所以 ‘也就会明白，后面提到的为什么特殊子帧配置为 0,5 的时候，为什么不能传输下行数据了，因为如果 PDCCH 占 3 个符号就没有资源给 PDSCH 用了（设计的人也是以 PDCCH 占最大情况来考虑的，一刀切。没有根据 PCFICH 来判断，如果根据 PCFICH 来判断算法会复杂。两种方法各有利弊）’

### 3.4.4 问题

#### 3.4.4.1 问题 1：既然说 GP 是为了上下行转换提供空余时间减少干扰，那为什么说上行到下行转换得地方都没有 GP 呢？

因为下行到上行转换时，UE 根本不知道和 enodeb 之间的距离，如果提前量太早了，UE 发送上行数据而 enodeb 还在发送下行数据，就会发生干扰，所以需要 GAP。当上行到下行转换的时候，如果 UE 没有 TA（时间提前量），它肯定是在 PRACH 上发送，preamble 占用的时间比较短，不会完全占满上行子帧，所以后面还是留了点时间，不会发送上下行冲突；而当 UE 已经有 TA 的时候，时间已经



对齐了，即使发送有点误差也是落在了 cyclic prefix（每个时域上 symbol 前面的空白）里面了，所以不会发生上下行干扰。

==》也进一步推出：为什么 PRACH 的资源在时域上，为什么在特殊子帧上要以‘特殊子帧’的尾部进行对齐，而在正常的上行子帧上，要以‘正常上行子帧的’开头对齐了。因为特殊子帧后面肯定是上行子帧，所以要向后对齐；而正常的上行子帧后面可能是下行子帧，所以要向前对齐。

#### 3.4.4.2 问题 2：为什么要有扩展的 CP？

- 覆盖范围大的小区，可以解决延迟长的问题
- MBMS 广播，对于多个小区同时广播一套节目给终端，必须考虑不同小区到终端的时间延迟不同，所以用扩展的长的 CP 比较好。

### 3.5 一些基本概念 -- 规定

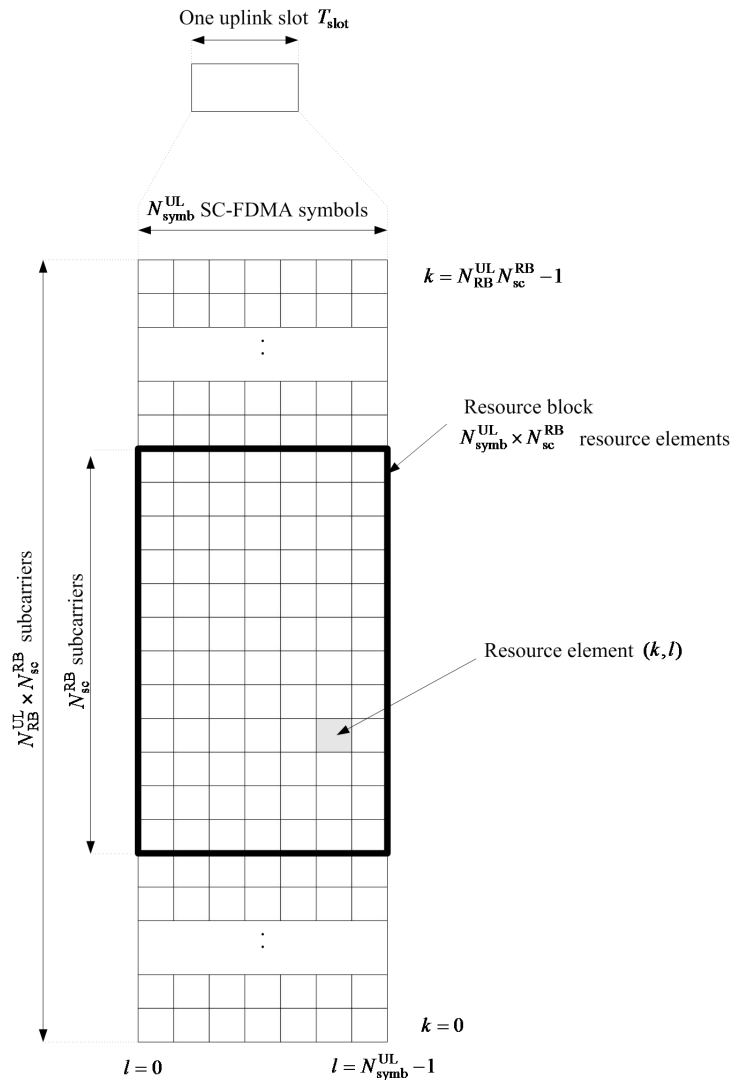
#### 3.5.1 公式 - 拉斯变换

$$1 + j = \sqrt{2} \cdot e^{-j\frac{\pi}{4}}$$

- 变换的目的就是：让乘法变得很简单了。

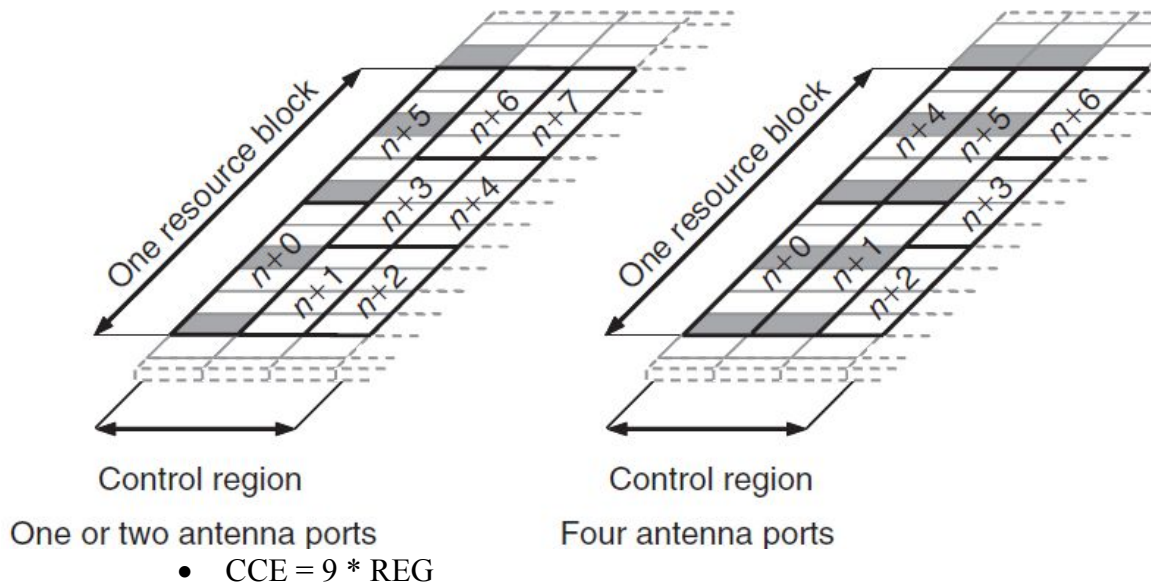
#### 3.5.2 资源块的描述 -- 规定

- 1 个资源块（RB）= 12 subcarrier \* 1 slot(正常 7 个符号)



- 1 subcarrier = 15kHz  $\rightarrow$  也就是说一秒钟的发射载波频率是 15k
- RE = (频域)15KZ \* 1 symbol (时域)，就是上面的一个 ‘最小的方框’。
- **REG** = 4 个频域挨着的但不一定连续的，时域上相同的 RE 的集合。





注意：CCE 只是一个逻辑上的概念，也就是说它物理上只是等于 9 个 REG, 并没有实际的对应关系。为了 PDCCH 盲检测用的。它和 REG 的顺序不一样，它的顺序是先时域，再频域的。

### 3.5.2.1 问题 1：为什么 CCE 要先时域后频域？

因为这样可以获得时域分集（就是把一组完整的数据分在不连续的时间上发送），跟后面提到的交织一样，都是为了错误随机化。因为‘射频单元’会以  $(1/T_s = 30\ 720\ 000\ S)$  的频率‘按照先频域后时域发送‘调制符号’。

### 3.5.3 调度的单位 -- 规定(个人觉得也是一种恰当不极端的思想)

- 时间上：一个 TTI(1ms)，即 2 个 TS 调度一次
- 频域上：调度的最小资源单位却是由一个 subframe 中的两个资源块为最小调度单位（一个时隙一个 RB，但这两个 RB 可能载频不一样），也就是所谓的时隙间跳频，跳即‘变化，不同的’意思。

### 3.5.3.1 问题 1：为什么要不同时隙间的使用的载频可能不一样？

这样应该是为了获得良好的接收效果。如果在某个频点的信号不好，而 1 个 TTI 内上下时隙的频点不一样，这样另外一个频点对应的信息还是能很好的解出来。

一个很特别的例子就是 PUCCH 资源回应 HARQ ACK/NACK 的时候：它对应的上下时隙的频点就不一样，但是它们传输的数据是有关联的，只要一个时隙能解

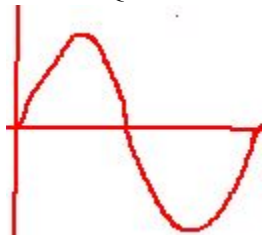
出来就行了，所以某个频点的信号不好也不会受影响。具体我们后面谈到 PUCCH 的时候再解释。

### 3.5.4 符号和真实的 BIT 数据的对应关系

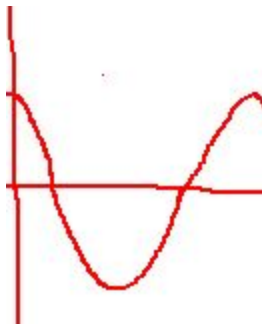
- 我们可以简单的把符号理解成电磁波，接收端接收到的电磁波然后根据不同的相位可以认为代表不同的 BIT.

记住：记住接收是指接收一个时间段的波形，而不是一个时间点的波形。

例如 QPSK：1 个符号代表 2bit 的情况。



这种波形代表 00

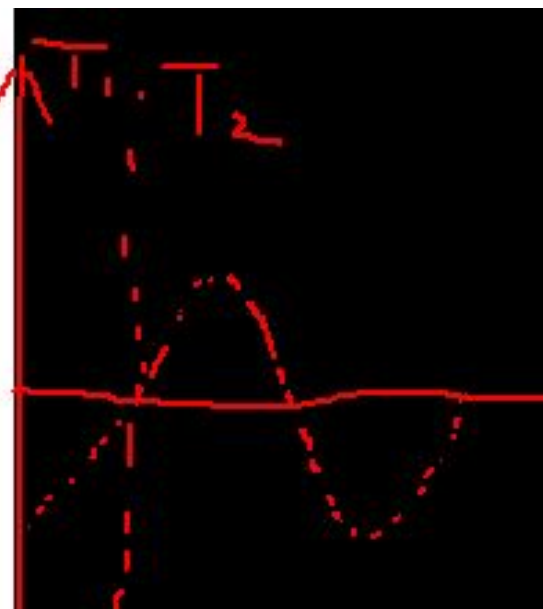
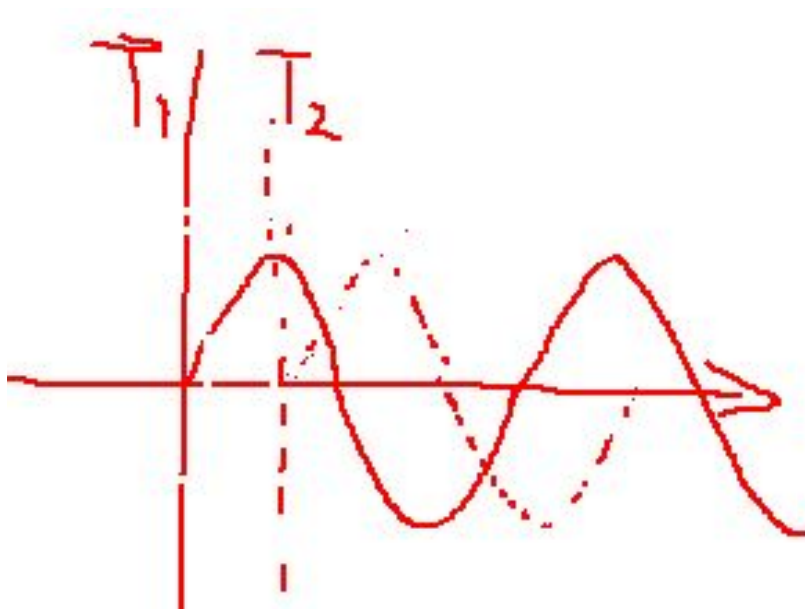


这种波形代表 01

- 参考 36211 的 7.1。注意：64QAM 有些手机是不支持的，所以要从 UE 的信息中获取是否支持，才能决定是否对该手机使用 64QAM（RRC::UE-EUTRA-Capability->ue-Category 能查到）

### 3.5.5 时域延迟等同于频率相位偏移如何理解

T1 时间点应该发送波形，推迟到 T2 点发送，所以相对于接收端它不知道推迟，所以它还是在 T1 时间点进行接收，接收到的就是 T2 时间点的波形。所以相位不一样，就相当于偏移。



## 4 物理信道与传输信道

### 4.1 上下行信道--- 规定

- 上行

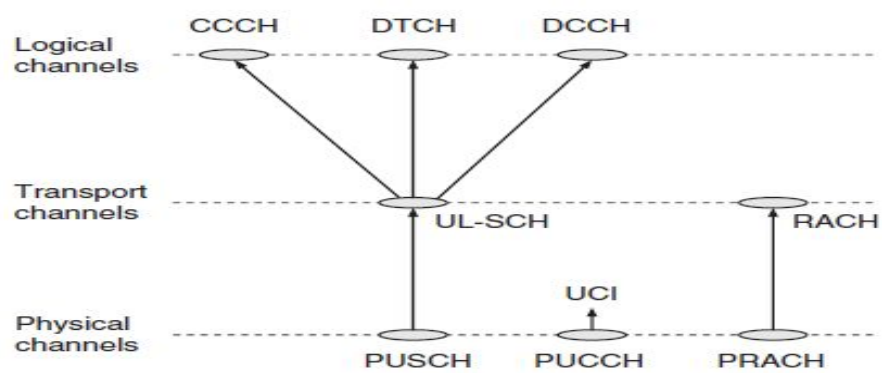


Figure 15.4 Uplink channel mapping.

Table 4.1-1

TrCH	Physical Channel
UL-SCH	PUSCH
RACH	PRACH

Table 4.1-2

Control information	Physical Channel
UCI	PUCCH, PUSCH

• 下行

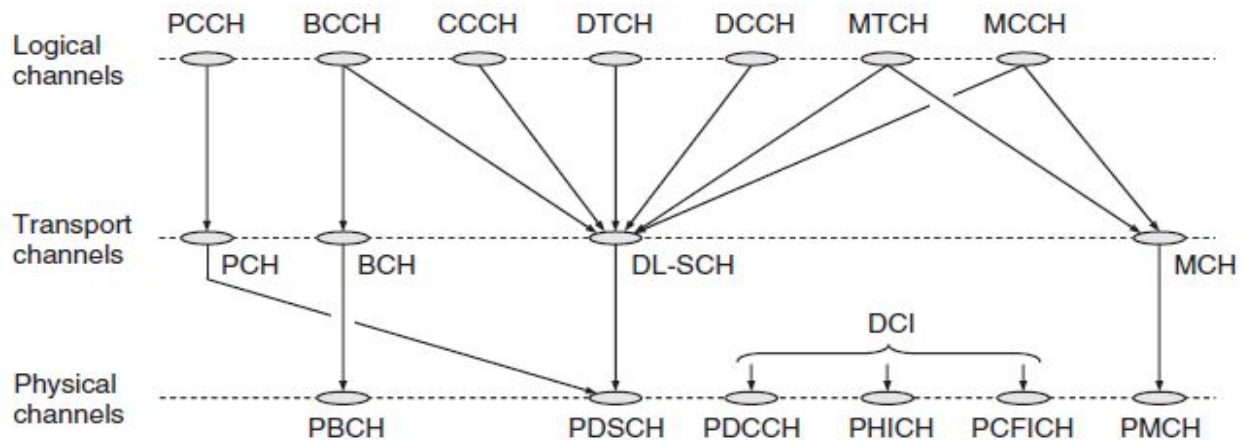


Table 4.2-1

TrCH	Physical Channel
DL-SCH	PDSCH
BCH	PBCH
PCH	PDSCH
MCH	PMCH

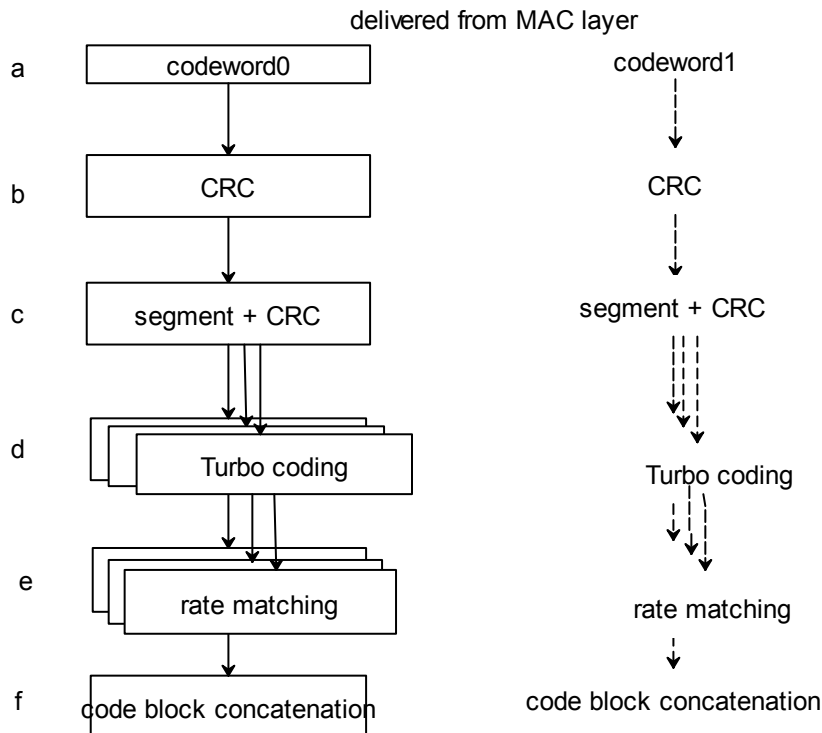
Table 4.2-2

Control information	Physical Channel
CFI	PCFICH
HI	PHICH
DCI	PDCCH

## 4.2 传输信道到物理信道的基本处理流程 (不分上下行)

- **输入:** TBS(transport block size), 也叫码字, 可能有一个或者两个码字-----调度决定给 UE 多少个 RB, 让然后根据 ‘CQI 或者加上其他因素 ‘算出  $I_{MCS}$ , 根据  $I_{MCS}$  算出  $I_{TBS}$ , 最后根据  $I_{TBS}$  和分配的 RB 数查表 36213---Table 7.1.7.2.1 得到能传多少个 bit, 这也就是 MAC 最后组成的 PDU 的大小。

- 流程



注：（1）每个分段自己去做 turbo 编码和速率匹配，最后才串联在一起。

（2）对于某些信道可能增加过程，也可能有些过程没有。总之是恰当的思想，不需要就不要；不足的就加。从该图也看出了有时只有一个码字 **codeword**, **codeword1** 不一定有。

#### 4.2.1 加 CRC (a → b) --- 一路输入，一路输出

##### 4.2.1.1 作用

- 校验,检错.

##### 4.2.1.2 参看

- 参看 36212 的 5.1.1

The bits after CRC attachment are denoted by  $b_0, b_1, b_2, b_3, \dots, b_{B-1}$ , where  $B = A + L$ . The relation between  $a_k$  and  $b_k$  is:

$$b_k = a_k \quad \text{for } k = 0, 1, 2, \dots, A-1$$

$$b_k = p_{k-A} \quad \text{for } k = A, A+1, A+2, \dots, A+L-1.$$

## 4.2.2 Segment + CRC (b→cr) ---- 一路输入，多路输出。 cr 可能带有<NIL>

### 4.2.2.1 作用

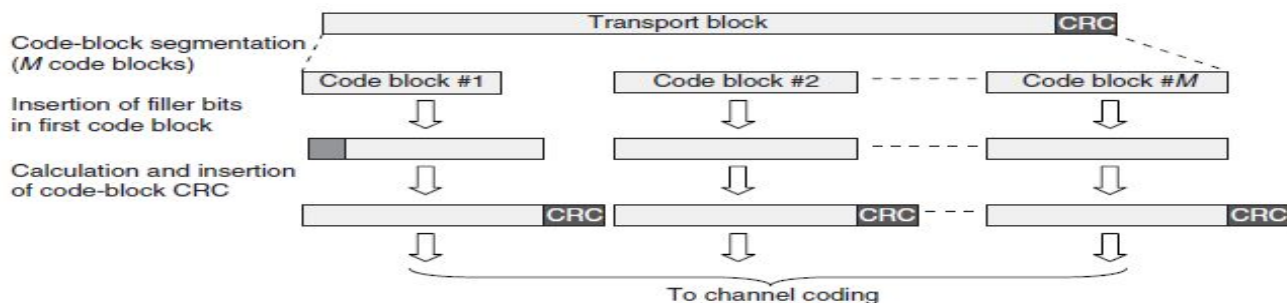
- 控制住 code block 的最大长度(算是分段之后的 CRC 长度).为什么需要控制，可以参看问题 3.

### 4.2.2.2 参看

- 参看：36212 的 5.1.2

### 4.2.2.3 思想及实现

- 思想：尽量保证‘分的每段的长度保持均匀’，长度在 36212 的 5.1.3-3 有规定(恰当的思想)。<NIL>比特是加在第一段的前面.



注意：只有分段之后才能再加 CRC24A

- 举个例子：

假设第一次加 CRC 之后是 13000bit(也就是 B，未分段前加 CRC 之后的 bit 数)，显然大于允许的最大长度  $Z=6144\text{bit}$ ，所以要分段

(1) 分段的段数为： $C = \lceil B / (Z - L) \rceil = \lceil 13000 / (6144 - 24) \rceil = 3$ .  $Z-L$  是代表分段之后每段还得加个 24bit 的 CRC24A

(2) 分段之后最后的有效的总 bit 数就为

$$B' = B + C \cdot L = 13000 + 3 \cdot 24 = 13072\text{bit}$$

(3) 去查表 36212 的 5.1.3-3 要满足  $C \cdot K_+ \geq 13072$ , 查表得到  $K_+ = 4416$ 。算出  $K_- = 4352$ ,  $\Delta_K = 64$ 。根据协议公式

$$C_- = \left\lfloor \frac{C \cdot K_+ - B'}{\Delta_K} \right\rfloor = 2, \text{ 就说明长度为 } K_- = 4352 \text{ 的有 } 2 \text{ 段;}$$

$C_+ = C - C_- = 3 - 2 = 1$ , 就说明长度为  $K_+ = 4416$  的有 1 段;

$$F = C_+ \cdot K_+ + C_- \cdot K_- - B' = 48, \text{ 说明第一段长度为 } K_- = 4352 \text{ 需要加 } 48\text{bit}$$

的<NIL>.

(4) 最后可以算一算，2 段 4352 + 1 段 4416 - 填充的 48bit<NIL>正好 = 13072.

- 协议里面的算法比较复杂，可以用另外一种算法来理解。

$C = \lceil B/(Z-L) \rceil$  ,  $B' = B + C \cdot L$  .  $Z=6144$ ,  $L=24$ (分段之后加 CRC24A 的长度)。

总共分 C 段。

$K_+$  是 5.1.3-3 中最小的满足 “ $C \cdot K \geq B'$ ” 的 K 值，K-就是 5.1.3-3 中  $K_+$  的一个值，比  $K_+$  小。

设  $C \cdot K_+ = B' + a \cdot \Delta K + b$  ( $b < \Delta K$ , 其实根据后面的说明  $\Delta K$  肯定是 64)

$\rightarrow C_- = a$ ;  $C_+ = C - a$ ;

通过协议的算法也是一样的：

$$\text{Number of segments of size } K_- : C_- = \left\lfloor \frac{C \cdot K_+ - B'}{\Delta K} \right\rfloor .$$

$$\text{Number of segments of size } K_+ : C_+ = C - C_- .$$

也就是说明：长度为  $K_-$  的有 a 个；长度为  $K_+$  的有  $C - a$  个。第一段 code block 的填充 b 个 <NULL>

$$\begin{aligned} \rightarrow F &= C_+ \cdot K_+ + C_- \cdot K_- - B' \\ &= C_+ \cdot K_+ + C_- \cdot (K_+ - \Delta K) - B' \\ &= (C_+ + C_-) \cdot K_+ - a \cdot \Delta K - B' \\ &= C \cdot K_+ - a \cdot \Delta K - B' \\ &= b \end{aligned}$$

#### 4.2.2.4 问题

##### 4.2.2.4.1 问题1：为什么分段之前要加CRC？分段之后还需要再加CRC？

- 因为分段之前加 CRC 就是对 ‘transport block’ 的校验，如果不加的话，分段要是丢了就没办法查出来了。一个 ‘transport block’ 可以被分为多个 ‘code block’，分段之后加的 CRC 是对 ‘code block’ 的校验。第一次的 CRC 可能是：CRC24A, CRC24B, CRC16, CRC8. 而分段之后的 CRC 都是 CRC24A.

##### 4.2.2.4.2 问题2：可不可能出现上面 $a \geq C$ 的情况？

- 不可能, 因为要分段, 每段的大小不可能小于 6144/2, 根据 5.13-3 表, 对应的  $\Delta K=64$  的, 如果出现  $a \geq C$  的情况的话,  $K_+$  应该对应的再往上跳一级。



#### 4.2.2.4.3 问题3：为什么需要分段，为什么分段的长度限制在 6144？

- 后面提到的 turbo 编码有限制是 6144bit，所以需要分段。

**4.2.3 信道编码( $c_r \rightarrow d_k^{(0)} / d_k^{(1)} / d_k^{(2)}$ )**。一路输入,三路输出. $d_k^{(0)}$  可能带有< NULL>

#### 4.2.3.1 作用

- 前向纠错编码。应该是重复编码，冗余来保证解码的成功率的。

#### 4.2.3.2 参看

- 参看 36212 的 5.1.3

**Table 5.1.3-1: Usage of channel coding scheme and coding rate for TrCHs**

TrCH		Coding scheme	Coding rate
UL-SCH (PUSCH)	Turbo coding	1/3	
DL-SCH(PDSCH)			
PCH(PDSCH)			
MCH(PMCH)			
BCH(PBCH)	Tail biting convolutional coding	1/3	

**Table 5.1.3-2: Usage of channel coding scheme and coding rate for control information**

Control Information		Coding scheme	Coding rate
DCI(PDCCH)	Tail biting convolutional coding	1/3	
CFI(PCFICH)	Block code	1/16	
HI(PHICH)	Repetition code	1/3	
UCI(PUCCH/PUSCH)	Block code	variable	
	Tail biting convolutional coding	1/3	

The values of  $D$  in connection with each coding scheme: ( $D$  代表  $d_r$  的 bit 数)

- tail biting convolutional coding with rate 1/3:  $D = K$ ;
- turbo coding with rate 1/3:  $D = K + 4$ .

#### 4.2.3.3 重要特性

- Turbo 编码有个很重要的特性，就是短暂的记忆性。也就是某个 bit 可以根据前面的  $n$  个 bit 算出来。所以它能进行些纠错的处理。

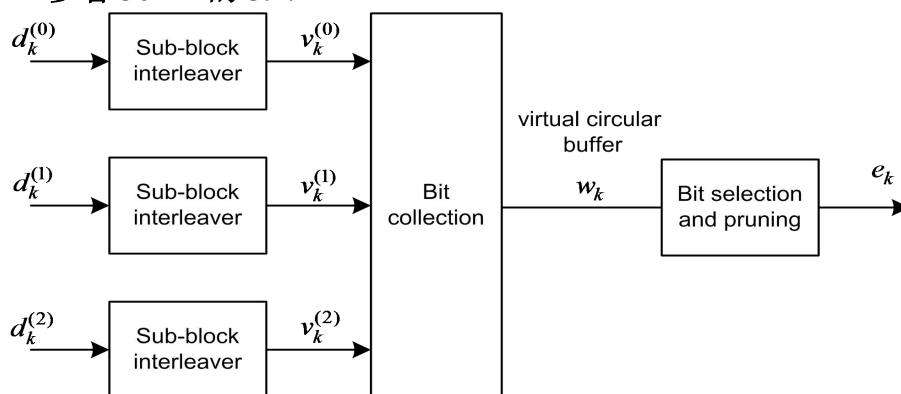
#### 4.2.4 交织, 速率匹配( $d_k^{(0)} / d_k^{(1)} / d_k^{(2)} \rightarrow e_k$ )。三路输入,一路输出. $e_k$ 不带有< NULL>

##### 4.2.4.1 作用

- (1) 交织作用: 使错误随机化 (离散化);
- (2) 速率匹配作用: 然后通过 ‘bit 选择和打孔’ 来选择以实现---传下去的 bit 数达到要求的实际的可用的 RE 对应数据量 (RB 中真正可用的 RE 和  $Q_m$  共同决定的--- 例如要除去 ‘参考信号’ 占用的 RE), 同时去掉多余或者<NIL>的数据以达到数据量与真正传输物理资源的 RE 对应的数据量一样。

##### 4.2.4.2 参看

- 参看 36212 的 5.1.4



##### 4.2.4.3 思想及实现

- 思想:
  - (1) 交织的思想: 横放列取。横放→列交换→按列读取。
  - (2) 速率匹配比特选择 (bit collection) 的思想: 根据 rvidx 得到起始的选择位置, 然后再把交织的数据列读的方式来选择 ‘物理层需要的真实分配的 bit 数’, 同时把<NIL>给去掉。
- 举个例子: 配置的 rvidx 决定选择的起始位置。

假设信道编码的结果为: a,b,NIL, NIL,c,d,e,f, NIL,g,h, NIL,i, j, NIL,k

注意 a.....k 都是代表 bit0 或者 1, 只是为了区别不同位置的二进制数这样写

$$\begin{pmatrix} a & b & NIL & NIL \\ c & d & e & f \\ NIL & g & h & NIL \\ i & j & NIL & k \end{pmatrix} \quad \text{----横放}$$

第 2 列和第 4 列交换

$$\begin{pmatrix} a & NIL & NIL & b \\ c & f & e & d \\ NIL & NIL & h & g \\ i & k & NIL & j \end{pmatrix} \quad \text{‘第 2 和第 4 列交换’ 之后的矩阵,}$$

然后，假设根据 ‘rvidx’ 算出起始位置为 d，最后要映射到能物理上能传送 ‘20bit’ 的资源上去。

按列读取，跳过 NIL。结果也就为：d,g,j,a,c, i,f,k,e,h, b, d,g,j,a, c,i,f,k,e.

注：从这里也就能看出物理层打孔和重复的意思了。就是如果资源对应能传得数据大于 turbo 编码后的数据，就会 turbo 编码后的数据会重复传送；如果就是如果资源对应能传得数据小于 turbo 编码后的数据，就会 turbo 编码后的有些数据会丢掉不传送，也就是 ‘打孔’ 的意思了。上面这个例子对应的是重复的情况。下面这个例子对应的是 ‘打孔’ 的例子。

然后，假设根据 ‘rvidx’ 算出起始位置为 d，最后要映射到能传送 ‘8bit’ 的资源上去。

按列读取，跳过 NIL。结果也就为：d,g,j,a,c, i,f,k.

#### 4.2.4.4 难理解的地方

- 看看 36211 的 5.1.4.1.2 节，有一些难理解的地方

(1) Denote the soft buffer size for the transport block by  $N_{IR}$  bits and the soft buffer size for the  $r$ -th code block by  $N_{cb}$  bits. The size  $N_{cb}$  is obtained as follows, where  $C$  is the number of code blocks computed in subclause 5.1.2:

$$\begin{aligned} - N_{cb} &= \min\left(\left\lfloor \frac{N_{IR}}{C} \right\rfloor, K_w\right) && \text{for downlink turbo coded transport channels} \\ - N_{cb} &= K_w && \text{for uplink turbo coded transport channels} \end{aligned}$$

where  $N_{IR}$  is equal to:

$$N_{IR} = \left\lfloor \frac{N_{soft}}{K_{MIMO} \cdot \min(M_{DL\_HARQ}, M_{limit})} \right\rfloor$$

为何有这种上下行的区别：因为下行是手机接收，必须考虑手机的 buffersize 的大小的能接受的程度；而上行是 eNodeB 接收，默认就认为它的 buffersize 肯定比 UE 大很多，所以不必担心，这也是一种潜规则。所以也就知道手机上报的 buffersize 有什么用了（RRC::UE-EUTRA-Capability-> ue-Category 能查到）。控制信息就没有这样的区别，因为默认都比较小。

$N_{IR} = \left\lfloor \frac{N_{soft}}{K_{MIMO} \cdot \min(M_{DL\_HARQ}, M_{limit})} \right\rfloor$ ：每个 HARQ 中的每个码字分一些‘缓存空间’；然后  $N_{cb} = \min\left(\left\lfloor \frac{N_{IR}}{C} \right\rfloor, K_w\right)$  下行的时候每个分段又继续分到对应的‘缓存空间’。

(2) 从获取  $e_k$  就知道  $rv_{idx}$ （上行是固定的，下行的时候在 DCI 中带有）有什么用了，就是用来决定实际的获取数据的位置的。

$$k_0 = R_{subblock}^{TC} \cdot \left( 2 \cdot \left\lfloor \frac{N_{cb}}{8 R_{subblock}^{TC}} \right\rfloor \cdot rv_{idx} + 2 \right)$$

(3)  $G$  和  $N_{IR}, N_{soft}$  是什么关系

$G$  是实际上要传输的‘transport block’的长度，是由“分配的 PRB 中有效的 RE 的个数, 调制模式  $Q_m$ , 以及时间长度决定”决定的(例如：BCH 最后算下来就是 1920bit，所以  $e_k$  也就会有 1920bit，因为此时考虑到它的时间长度是 40ms。参看 36212 的 5.3.1. 一般的正常调度应该为 1ms—1 个 TTI)，也是所有‘code block’合起来的长度。而  $N_{soft}$  是指手机实际的 buffersize 最大能容纳多少， $N_{IR}$  也就是对应到每个 HARQ PROCESS 能的每个码字分配多少的 buffersize，简单来做就是‘取平均’。

$$N_{IR} = \left\lfloor \frac{N_{soft}}{K_{MIMO} \cdot \min(M_{DL\_HARQ}, M_{limit})} \right\rfloor$$

注： $N_{cb}$  只是能决定选择‘bit collection’后的 bit 流的范围，并不能决定实际传输的数据量。实际还是由  $G$  来决定，然后映射到每个分段的实际传输 bit 流的长度就是  $E$ 。

#### 4.2.4.5 问题

##### 4.2.4.5.1 问题 1：为什么 $N_L$ 在 4 层的时候不是 4 而是 2 呢？

Set  $G' = G / (N_L \cdot Q_m)$  where  $Q_m$  is equal to 2 for QPSK, 4 for 16QAM and 6 for 64QAM, and where

- $N_L$  is equal to 1 for transport blocks mapped onto one transmission layer, and
- $N_L$  is equal to 2 for transport blocks mapped onto two or four transmission layers.

因为此时 2, 3, 4 层，最多可能有 2 个码字，也有可能是 1 个码字。所以这里是按默认最大 2 个码字算出来的。

##### 4.2.4.5.2 问题 2： $G$ 是实际上要传输的‘transport block’的长度，计算的时候需要考虑到空分和发射分集吗？

只需要考虑空分，不需要考虑发射分集。

#### 4.2.4.5.3 问题3：为什么 5.1.4.2.2 节的对于 BCH 的公式直接是

$$e_k = w_{j \bmod K_w}, \text{ 没有 } N_{cb} \text{ 呢?}$$

- 因为‘BCH 和上行才会用 convolutionally coded 编码方式’。BCH 数据很少，UE 的 buffer 肯定够。而上行的时候接收端是 enodeb，默认觉得它的 buffer size 也是肯定足够的。而控制信息数据很少，肯定 buffersize 也是够的。所以这里直接就用了  $K_w$ 。

#### 4.2.5 Codeblock 串联 ( $e_k \rightarrow f$ )。多路输入,一路输出

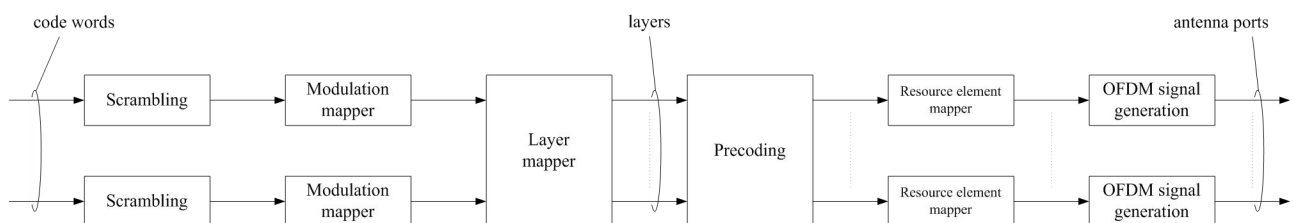
- 参看 36211 的 5.1.5

#### 4.2.6 总结

- 对于接收端：“CRC, Turbo 编码，交织”组成了一个几乎完美的防错网，一层一层的过滤防错纠错。  
>>>首先：交织是数据离散化了，所以一串连续出错的数据最后被离散化了  
>>>然后，通过 turbo 编码短暂的记忆性，又能对这种离散化了的错误进行一些恢复。如果不离散化，一串错误的 bit 紧靠在一起，turbo 编码是无法利用它的‘短暂记忆性’进行纠错的。  
>>>最后利用 CRC 在整体进行错误的检查，和简单错误的恢复。

### 4.3 下行物理信道的基本处理流程

- 参看 36211 的 6.3



注：（1）因为下行可以空分，上行没有。所以可以看到‘下行’最多可能有两个码字。而且下行有层映射，上行也没有。

（2）输入就是上面提到的最后 code block 串联的结果

（3）加扰和调制是针对一个码字的，层映射的时候它们才合在一起。所以后面能明白为什么说到 DCI 格式的时候，有些 DCI 格式的‘对于 MCS,新数据只是 new data indication, RV’有两个，而 PMI 始终只有一个了。

- 输入：  
这里的输入就是 4.2 节中最后 codeblock 串联在一起的结果作为输入的。

- 码字和传输块

- (1) 一个传输块对应一个码字
- (2) 单天线只能有一个传输块，多天线至多两个传输块

#### 4.3.1 加扰 ( $b \rightarrow \tilde{b}$ ) -----一路输入,一路输出

##### 4.3.1.1 作用

- 作用：加扰的目的除了打散用户信息外，最主要的目的就是让相应的信息白噪声化，相对于其它 UE，小区都是随机噪声了，那么处理起来就简单很多。加扰的目的是为了避免长连零或者长连一的出现，由于在 ofdm 系统中，数据要进行快速傅立叶变换，如果系统中存在长连零或者长连一的话，ifft 后的数据会在某个频率上能量超高，即造成严重的 papr 问题，此时接收端 agc 会对信号起到 clipping 的效果，从而是数据信息损失，因此 randomization 在系统中还是相当重要的，一般加扰码的作用无非也就是为了避免出现过长的 0 或 1，以便于时钟信号的提取。就像开始比喻的‘虽然别人听到了，但是由于语言不通，让别的 UE 听不懂’

##### 4.3.1.2 参看

- 参看 36211 的 6.3.1

$$\tilde{b}^q(i) = (b^q(i) + c^q(i)) \bmod 2$$

#### 4.3.2 调制 ( $\tilde{b} \rightarrow d$ ) -----一路输入,一路输出

##### 4.3.2.1 作用

- 作用：把 bit 转换成调制符号，好对应到 RE 上去传送。

##### 4.3.2.2 参看

- 参看：36211 的 6.3.1，具体的调制相位 36211 的 7.1

**Table 6.3.2-1: Modulation schemes**

Physical channel	Modulation schemes
PD SCH	QPSK, 16QAM, 64QAM
PMCH	QPSK, 16QAM, 64QAM
PBCH	QPSK
PCFICH	QPSK
PDCCH	QPSK
PHICH	BPSK

### 4.3.3 层映射 layer mapping (d→x) ----- m 路输入,n 路输出

#### 4.3.3.1 作用

- 作用：可以多传数据。可以计算一下，如果‘一个码字两层’，对应传送的数据量是‘一个码字一层’的两倍。

#### 4.3.3.2 参看

- 参看 36211 的 6.3.2

#### 4.3.3.3 思想及实现

- 思想：均匀映射到层上去，奇偶相错开。

(1) Layer mapping for spatial multiplexing

Table 6.3.3.2-1: Codeword-to-layer mapping for spatial multiplexing

Number of layers	Number of code words	Codeword-to-layer mapping $i = 0, 1, \dots, M_{\text{symp}}^{\text{layer}} - 1$	
1	1	$x^{(0)}(i) = d^{(0)}(i)$	$M_{\text{symp}}^{\text{layer}} = M_{\text{symp}}^{(0)}$
2	2	$x^{(0)}(i) = d^{(0)}(i)$ $x^{(1)}(i) = d^{(1)}(i)$	$M_{\text{symp}}^{\text{layer}} = M_{\text{symp}}^{(0)} = M_{\text{symp}}^{(1)}$
2	1	$x^{(0)}(i) = d^{(0)}(2i)$ $x^{(1)}(i) = d^{(0)}(2i+1)$	$M_{\text{symp}}^{\text{layer}} = M_{\text{symp}}^{(0)} / 2$
3	2	$x^{(0)}(i) = d^{(0)}(i)$ $x^{(1)}(i) = d^{(1)}(2i)$ $x^{(2)}(i) = d^{(1)}(2i+1)$	$M_{\text{symp}}^{\text{layer}} = M_{\text{symp}}^{(0)} = M_{\text{symp}}^{(1)} / 2$
4	2	$x^{(0)}(i) = d^{(0)}(2i)$ $x^{(1)}(i) = d^{(0)}(2i+1)$ $x^{(2)}(i) = d^{(1)}(2i)$ $x^{(3)}(i) = d^{(1)}(2i+1)$	$M_{\text{symp}}^{\text{layer}} = M_{\text{symp}}^{(0)} / 2 = M_{\text{symp}}^{(1)} / 2$

(2) Layer mapping for transmit diversity --- 发射分集的层数与物理上能发送的 port 数应该是一样的。



**Table 6.3.3.3-1: Codeword-to-layer mapping for transmit diversity**

Number of layers	Number of code words	Codeword-to-layer mapping $i = 0, 1, \dots, M_{\text{symb}}^{\text{layer}} - 1$
2	1	$x^{(0)}(i) = d^{(0)}(2i)$ $x^{(1)}(i) = d^{(0)}(2i + 1)$ $M_{\text{symb}}^{\text{layer}} = M_{\text{symb}}^{(0)} / 2$
4	1	$x^{(0)}(i) = d^{(0)}(4i)$ $x^{(1)}(i) = d^{(0)}(4i + 1)$ $x^{(2)}(i) = d^{(0)}(4i + 2)$ $x^{(3)}(i) = d^{(0)}(4i + 3)$ $M_{\text{symb}}^{\text{layer}} = \begin{cases} M_{\text{symb}}^{(0)} / 4 & \text{if } M_{\text{symb}}^{(0)} \bmod 4 = 0 \\ (M_{\text{symb}}^{(0)} + 2) / 4 & \text{if } M_{\text{symb}}^{(0)} \bmod 4 \neq 0 \end{cases}$ If $M_{\text{symb}}^{(0)} \bmod 4 \neq 0$ two null symbols shall be appended to $d^{(0)}(M_{\text{symb}}^{(0)} - 1)$

• 举个例子:

如果调制后的调制符号的序列为 a,b,c,d 传，enodeb 是 4 天线 port。

(1) 如果层数为 1，那层映射的结果也就为  $[a, b, c, d]$ ，最后要映射到 4 根天线 port 上去传送，需要一个矩阵变化，也就是我们说的 PMI（预编码矩阵 4\*1）

假设  $\text{PMI} = \begin{bmatrix} 1 \\ 2 \\ 1 \\ 1 \end{bmatrix}$ ，那最后的结果也就是  $\text{PMI} * \text{层映射的结果}$

$$\begin{bmatrix} 1 \\ 2 \\ 1 \\ 1 \end{bmatrix} * [a, b, c, d] = \begin{bmatrix} a & b & c & d \\ 2a & 2b & 2c & 2d \\ a & b & c & d \\ a & b & c & d \end{bmatrix} = \begin{bmatrix} p^{(0)}(i) \\ p^{(1)}(i) \\ p^{(2)}(i) \\ p^{(3)}(i) \end{bmatrix}$$

最后：可以看出每个 port 上需要发送 4 个调制符号。

(2) 如果层数为 2，那层映射的结果也就为  $\begin{bmatrix} a & c \\ b & d \end{bmatrix}$ ，最后要映射到 4 根天线 port 上去传送，也需要一个 PMI（预编码矩阵 4\*2）

假设  $\text{PMI} = \begin{bmatrix} 1 & 0 \\ 1 & 2 \\ 2 & 1 \\ 1 & 1 \end{bmatrix}$ ，那最后的结果也就是  $\text{PMI} * \text{层映射的结果}$

$$\begin{bmatrix} 1 & 0 \\ 1 & 2 \\ 2 & 1 \\ 1 & 1 \end{bmatrix} * \begin{bmatrix} a & c \\ b & d \end{bmatrix} = \begin{bmatrix} a & c \\ a + 2b & c + 2d \\ 2a + b & 2c + d \\ a + b & c + d \end{bmatrix} = \begin{bmatrix} p^{(0)}(i) \\ p^{(1)}(i) \\ p^{(2)}(i) \\ p^{(3)}(i) \end{bmatrix}$$

最后：可以看出每个 port 上需要发送 2 个调制符号。

最多是是双流

结论：使用层映射可以多传数据。

- 最多两路输入，也就是  $m$ ，也就是两个码字，两个 transport block；输出的路数  $n$  跟‘层数一致，和上报的 RI (rank indication) 数相关。为什么只是相关，后面讲 TM3，TM4 的时候会提到’。

#### 4.3.3.4 问题

##### 4.3.3.4.1 问题1：为什么层数 ( $n$ ) 不能大于天线的 port 数 ( $m$ ) 呢？

因为从矩阵算法通过预编码来看  $n$  个向量（层数）变成  $m$  个向量（port 数），UE 收到的是  $m$  个向量。如果  $n > m$  的时候，无法反解确切的  $n$  个向量出来，或者说反解出来的  $n$  个向量的解不唯一。也就是数学的算法。

##### 4.3.3.4.2 问题2：为什么最多 2 个码字呢？

个人认为是因为码字越多越难解，每个码字要对应回 ACK/NACK，一是从后面了解 PUCCH 资源回 ACK/NACK 来看，码字太多很难编码，如果还是用一个 symbol 来表示 ACK/NACK 回应，三个码字就必须用 16QAM，这样对信道条件要求高一个码字对应一个 ACK 回应，如果把码字增加，对应的 ACK 回应就得增多，增加算法复杂度。二是随着码字的增加，出错的概率也会增加，如果出错一个，所有的多个 TTI 的码字都要重传，也许会导致大量的重传。所以也是折中和恰当的思想。

#### 4.3.4 预编码 ( $x \rightarrow y$ ) ----- $m$ 路输入, $n$ 路输出

##### 4.3.4.1 作用

- 作用：把对应层数的向量（调制符号组合），通过矩阵运算对应到能发送数据的 port 端口上，然后在这些 port 端口上发送。

##### 4.3.4.2 参看

- 参看 36211 的 6.3.4

##### 4.3.4.3 思想及实现

- 只有一个 port。
$$y^{(p)}(i) = x^{(0)}(i)$$

- 有两个或多个 port

(1) 空分复用

- a. 带有 CDD 的空分复用 --- 不需要上报 PMI

$$\begin{bmatrix} y^{(0)}(i) \\ \vdots \\ y^{(P-1)}(i) \end{bmatrix} = W(i)D(i)U \begin{bmatrix} x^{(0)}(i) \\ \vdots \\ x^{(v-1)}(i) \end{bmatrix}$$

注：对于 D(I)和 U 矩阵可以根据 Table 6.3.4.2.2-1 得到，i 代表符号位置。此时预编码是固定的，所以只需要上报 RI (layer 数) 即可，也就知道为什么后面的一些传输模式配置上报的 CQI/PMI 格式的时候为什么有些不需要上报 PMI 了，因为已经固定了。

\*\*\*\*对于 2 天线的 2 层用 Table 6.3.4.2.3-1 对应的 (codebook index=0, v=2) 的。

36211---Table 6.3.4.2.3-1: Codebook for transmission on antenna ports {0,1}.

Codebook index	Number of layers $v$	
	1	2
0	$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$	$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
1	$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix}$	$\frac{1}{2} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$
2	$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ j \end{bmatrix}$	$\frac{1}{2} \begin{bmatrix} 1 & 1 \\ j & -j \end{bmatrix}$
3	$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -j \end{bmatrix}$	-

\*\*\*\*\*对于 4 天线的 1,2, 3, 4 层的根据公式 (i 表示的是层映射之后某一层的调制第几个的符号从  $0 \sim M_{\text{sybm}}^{\text{ap}} = M_{\text{sybm}}^{\text{layer}}$ ，v 为分层的层数)，

k=1,2,3,4 分别去

Table 6.3.4.2.3-2 的 codebook index 为 12,13,14 and 15 的地方去找。

$$k = \left( \left\lfloor \frac{i}{v} \right\rfloor \bmod 4 \right) + 1$$

$$W_n = I - 2u_n u_n^H / u_n^H u_n。$$

举个例子：

假设 i=5, v=2.那么 k=3，对应的就是 codebook index=14，然后只有两层 v=2.最后选择的预编码矩阵 PMI 也就是  $w_{14}^{\{13\}} / \sqrt{2}$ 。

下图中的  $w_0^{\{14\}} / \sqrt{2}$  中的 W 的上标 14 表示取的是第几列的数据。14，就是取第 1 列和第 4 列的数据构成一个 4\*2 的矩阵。

36211----Table 6.3.4.2.3-2: Codebook for transmission on antenna ports {0,1,2,3}.

Codebook index	$u_n$	Number of layers $\nu$			
		1	2	3	4
0	$u_0 = [1 \quad -1 \quad -1 \quad -1]^T$	$W_0^{\{1\}}$	$W_0^{\{14\}}/\sqrt{2}$	$W_0^{\{124\}}/\sqrt{3}$	$W_0^{\{1234\}}/2$
1	$u_1 = [1 \quad -j \quad 1 \quad j]^T$	$W_1^{\{1\}}$	$W_1^{\{12\}}/\sqrt{2}$	$W_1^{\{123\}}/\sqrt{3}$	$W_1^{\{1234\}}/2$
2	$u_2 = [1 \quad 1 \quad -1 \quad 1]^T$	$W_2^{\{1\}}$	$W_2^{\{12\}}/\sqrt{2}$	$W_2^{\{123\}}/\sqrt{3}$	$W_2^{\{3214\}}/2$
3	$u_3 = [1 \quad j \quad 1 \quad -j]^T$	$W_3^{\{1\}}$	$W_3^{\{12\}}/\sqrt{2}$	$W_3^{\{123\}}/\sqrt{3}$	$W_3^{\{3214\}}/2$
4	$u_4 = [1 \quad (-1-j)/\sqrt{2} \quad -j \quad (1-j)/\sqrt{2}]^T$	$W_4^{\{1\}}$	$W_4^{\{14\}}/\sqrt{2}$	$W_4^{\{124\}}/\sqrt{3}$	$W_4^{\{1234\}}/2$
5	$u_5 = [1 \quad (1-j)/\sqrt{2} \quad j \quad (-1-j)/\sqrt{2}]^T$	$W_5^{\{1\}}$	$W_5^{\{14\}}/\sqrt{2}$	$W_5^{\{124\}}/\sqrt{3}$	$W_5^{\{1234\}}/2$
6	$u_6 = [1 \quad (1+j)/\sqrt{2} \quad -j \quad (-1+j)/\sqrt{2}]^T$	$W_6^{\{1\}}$	$W_6^{\{13\}}/\sqrt{2}$	$W_6^{\{134\}}/\sqrt{3}$	$W_6^{\{1324\}}/2$
7	$u_7 = [1 \quad (-1+j)/\sqrt{2} \quad j \quad (1+j)/\sqrt{2}]^T$	$W_7^{\{1\}}$	$W_7^{\{13\}}/\sqrt{2}$	$W_7^{\{134\}}/\sqrt{3}$	$W_7^{\{1324\}}/2$
8	$u_8 = [1 \quad -1 \quad 1 \quad 1]^T$	$W_8^{\{1\}}$	$W_8^{\{12\}}/\sqrt{2}$	$W_8^{\{124\}}/\sqrt{3}$	$W_8^{\{1234\}}/2$
9	$u_9 = [1 \quad -j \quad -1 \quad -j]^T$	$W_9^{\{1\}}$	$W_9^{\{14\}}/\sqrt{2}$	$W_9^{\{134\}}/\sqrt{3}$	$W_9^{\{1234\}}/2$
10	$u_{10} = [1 \quad 1 \quad 1 \quad -1]^T$	$W_{10}^{\{1\}}$	$W_{10}^{\{13\}}/\sqrt{2}$	$W_{10}^{\{123\}}/\sqrt{3}$	$W_{10}^{\{1324\}}/2$
11	$u_{11} = [1 \quad j \quad -1 \quad j]^T$	$W_{11}^{\{1\}}$	$W_{11}^{\{13\}}/\sqrt{2}$	$W_{11}^{\{134\}}/\sqrt{3}$	$W_{11}^{\{1324\}}/2$
12	$u_{12} = [1 \quad -1 \quad -1 \quad 1]^T$	$W_{12}^{\{1\}}$	$W_{12}^{\{12\}}/\sqrt{2}$	$W_{12}^{\{123\}}/\sqrt{3}$	$W_{12}^{\{1234\}}/2$
13	$u_{13} = [1 \quad -1 \quad 1 \quad -1]^T$	$W_{13}^{\{1\}}$	$W_{13}^{\{13\}}/\sqrt{2}$	$W_{13}^{\{123\}}/\sqrt{3}$	$W_{13}^{\{1324\}}/2$
14	$u_{14} = [1 \quad 1 \quad -1 \quad -1]^T$	$W_{14}^{\{1\}}$	$W_{14}^{\{13\}}/\sqrt{2}$	$W_{14}^{\{123\}}/\sqrt{3}$	$W_{14}^{\{3214\}}/2$
15	$u_{15} = [1 \quad 1 \quad 1 \quad 1]^T$	$W_{15}^{\{1\}}$	$W_{15}^{\{12\}}/\sqrt{2}$	$W_{15}^{\{123\}}/\sqrt{3}$	$W_{15}^{\{1234\}}/2$

b. 没 CDD，也就是 close-loop 的空分复用

注：首先 enodeb 配置决定哪些 PMI 可以上报

RRC:PhysicalConfigDedicated-> antennaInfo-> explicitValue->

codebookSubsetRestriction，然后 UE 再根据能上报的 PMI 选择上报，

eNodeb 决定下行的时候用哪一个，也许并不用上报的那一个。由于层用 Table 6.3.4.2.3-1 对应的（codebook index=0，v=2）已经被 CDD 用了，所以不能用，其实个人认为还是可以用的。

(2) 发射分集 --- 参考 36211 的 6.3.4.3

多根天线发送针对一个对象的数据，各发各的。也是固定的预编码矩阵 PMI。注意发射分集虽然在多根天线上，但只能传一根天线对应的数据量。发射分集对应的层数是和 port 数一样的。

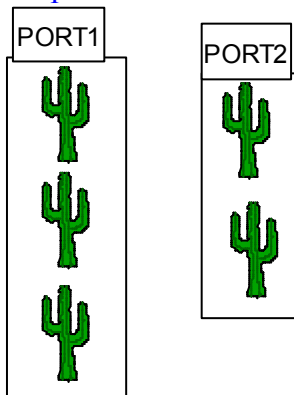
#### 4.3.4.4 问题

##### 4.3.4.4.1 问题1：要是发射分集的layer数与实际的enodeb的天线端口port数不一样该怎么算？

应该不考虑这种情况。潜规则就是发射分集时的层数与天线端口的数目是一样的。

##### 4.3.4.4.2 问题2：天线与天线port之间是什么关系？

- 是多对一的关系。也就是说多根物理上的天线（至少一根）对应于一个天线port，而且一根天线不能让多个port共用。主要靠配置决定。

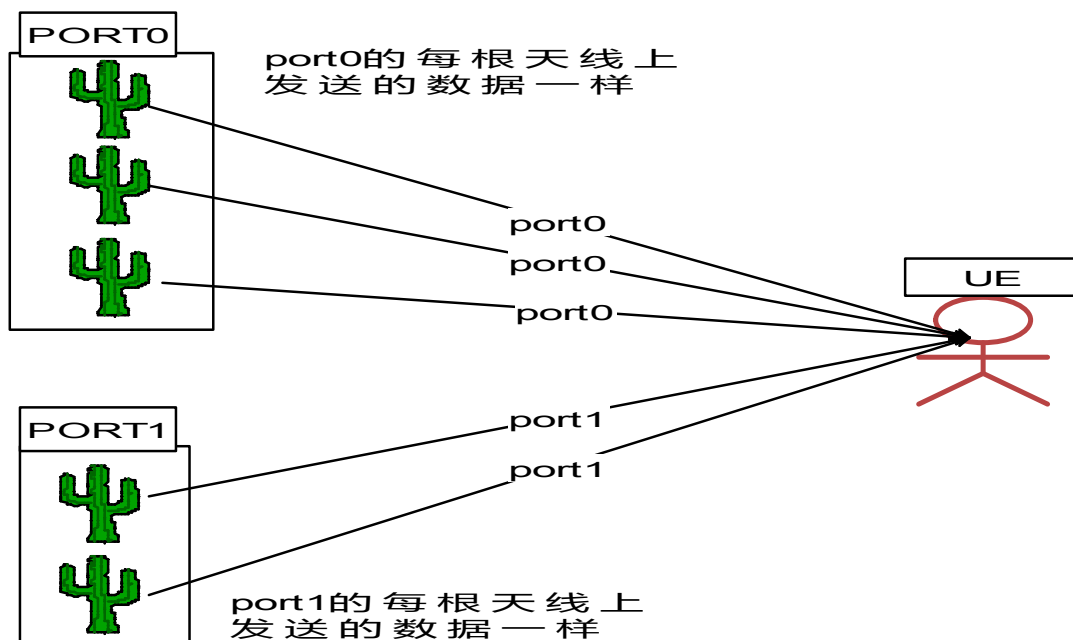


- 一个port对应的所有天线物理上发送的数据一般都是一样的（非基于码字的波束赋型除外），所以UE接收端根本不关心一个port上的数据是几根天线发送出来的。而且UE根据RS参考信号的位置能知道是port0，port1，port2还是port3

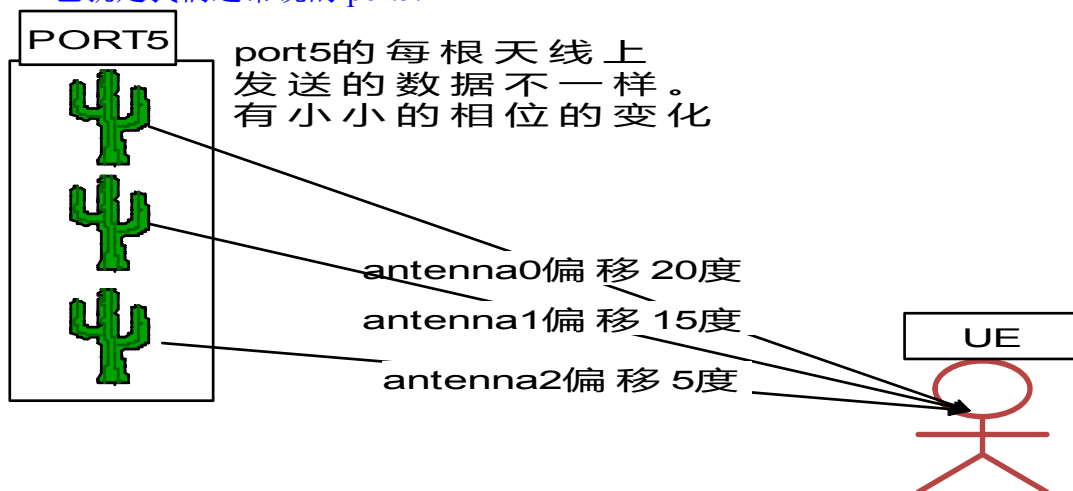
##### 4.3.4.4.3 问题3：不是很理解波束赋型，能不能解释一下？

波束赋型分为两类，一是基于码本的波束赋型；还有一种是不基于码本的波束赋型

- 基于码本的波束赋型：其实就是RI=1的空分。此时一个port对应的物理上的每根天线发射的是一样的东西。-----其实就相当于多个port的波束赋型，产生累加效果。

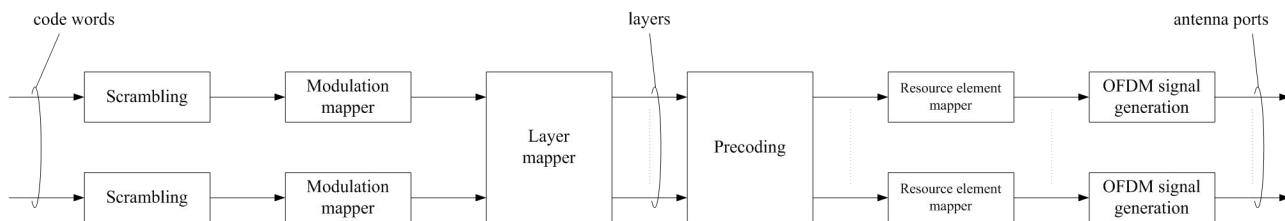


- 不基于码本的波束赋型：enodeb 的一个 port 里面的所有物理天线，根据 UE 发送来的 SRS 或者上行的数据，来判断每根天线的‘相位的偏移量’。由于 TDD 上下行同频，所以该 port 下行发数据的时候，该 port 里面的每根天线都根据‘自己的相位偏移量做反向偏移’之后，再把数据发送给 UE，所以 UE 收到的数据理论上都是一样的，而且有个很好的累加效果，因为该 port 上的每根天线都考虑到了‘相位偏移量’-----其实就相当于一个 port 中所有天线的波束赋型，理论上是‘该 port 对应的天线数越多越好’，其实也就是我们通常说的 port5.



#### 4.3.4.4.4 问题 4:: 怎么理解 port, RI, layer 之间的关系?

- 层数是不能大于 PORT 的，前面已经提到了这个规定。所以，UE 上报的 RI 不能大于手机本身的用作正常传输的 PORT 数（注：这里为什么提到正常，像 port5 对应的天线有特殊用途就不能算进来）。
- 从下图可以看出。enodeb 选择的层数也不能大于上报的 RI 数，否则跟前面提到的一样，层数大于 port 数之后，UE 收到了信息无法反解。



- 虽然 enodeb 的 port 数可能会比 UE 的 port 数多。假设 enodeb 为 4，UE 为 2；那最多只会发 2 层的数据，虽然 UE 只有两个 port，只收到了 enodeb 为 4 个 port 发送出来的 2 路数据。但是由于只有两层，也就是 2 个层的向量，而 UE 也收到了 2 个 port 的数据，层数没有大于 port 数，所以同样能反解出来每层的数据。

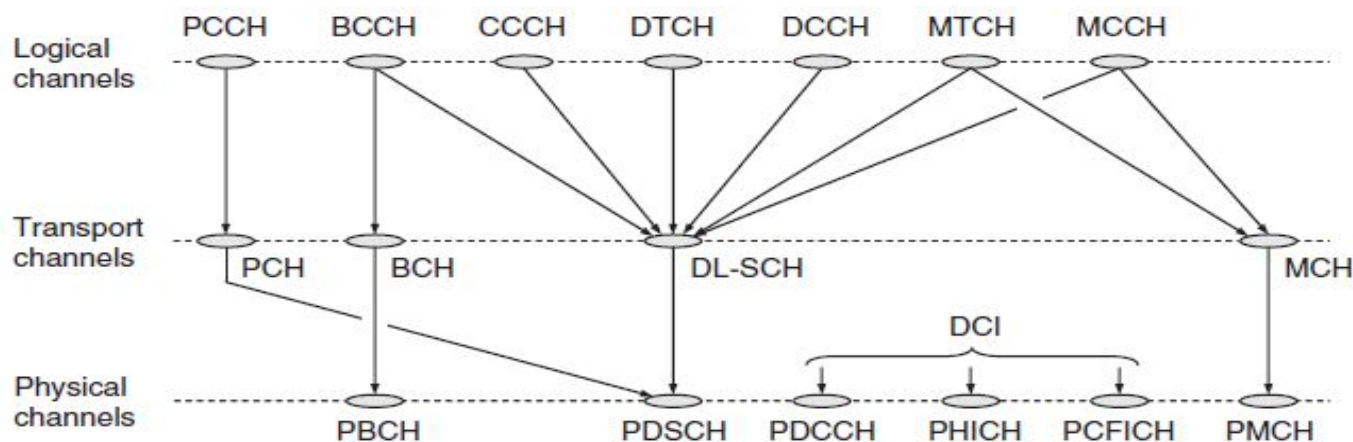
### 4.3.5 映射到资源原子 RE

#### 4.3.5.1 参看

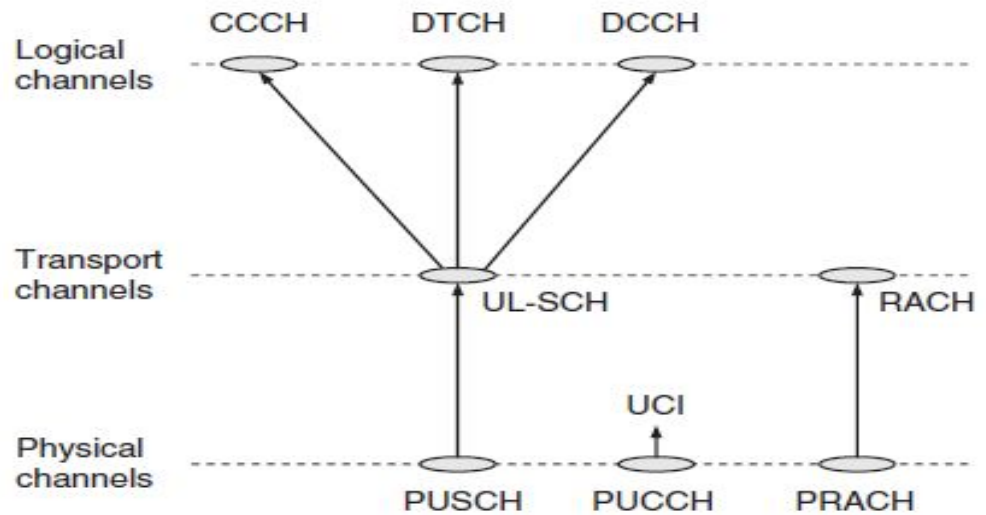
- 参看 36211 的 6.3.5,一般都是先频后时

## 4.4 下行信道与控制信令 -- 规定

### 4.4.1 下行信道的简介







**Figure 15.4** Uplink channel mapping.


- *Broadcast Control Channel (BCCH)*, used for transmission of *system information* from the network to all mobile terminals in a cell. Prior to accessing the system, a mobile terminal needs to acquire the system information to find out how the system is configured and, in general, how to behave properly within a cell.
- *Paging Control Channel (PCCH)*, used for paging of mobile terminals whose location on cell level is not known to the network. The paging message therefore needs to be transmitted in multiple cells.
- *Common Control Channel (CCCH)*, used for transmission of control information in conjunction with random access.
- *Dedicated Control Channel (DCCH)*, used for transmission of control information to/from a mobile terminal. This channel is used for individual configuration of mobile terminals such as different handover messages.
- *Multicast Control Channel (MCCH)*, used for transmission of control information required for reception of the MTCH, see below.
- *Dedicated Traffic Channel (DTCH)*, used for transmission of user data to/from a mobile terminal. This is the logical channel type used for transmission of all uplink and non-MBSFN downlink user data.
- *Multicast Traffic Channel (MTCH)*, used for downlink transmission of MBMS services.

- *Broadcast Channel* (BCH) has a fixed transport format, provided by the specifications. It is used for transmission of parts of the BCCH system information, more specifically the so-called *Master Information Block* (MIB), as described in Chapter 18.
- *Paging Channel* (PCH) is used for transmission of paging information from the PCCH logical channel. The PCH supports *discontinuous reception* (DRX) to allow the mobile terminal to save battery power by waking up to receive the PCH only at predefined time instants. The LTE paging mechanism is described in Chapter 18.
- *Downlink Shared Channel* (DL-SCH) is the main transport channel used for transmission of downlink data in LTE. It supports key LTE features such as dynamic rate adaptation and channel-dependent scheduling in the time and frequency domains, hybrid ARQ with soft combining, and spatial multiplexing. It also supports DRX to reduce mobile-terminal power consumption while still providing an always-on experience. The DL-SCH is also used for transmission of the parts of the BCCH system information not mapped to the BCH and for single-cell MBMS services.
- *Multicast Channel* (MCH) is used to support MBMS. It is characterized by a semi-static transport format and semi-static scheduling. In case of multi-cell transmission using MBSFN, the scheduling and transport format configuration is coordinated among the cells involved in the MBSFN transmission.
- *Uplink Shared Channel* (UL-SCH) is the uplink counterpart to the DL-SCH, that is the uplink transport channel used for transmission of uplink data.

#### 4.4.2 同步信号 ---- 规定

##### 4.4.2.1 P-SS(primary 主同步信号)

###### 4.4.2.1.1 作用

- 作用：获得帧的 5ms 的下行时间同步，同时获得 CELL 组内的 ID (0~2)，不同的 PSS 的 ZC 序列决定了不同组内 ID。LTE 的 PCI (physical cell id) 分 168 个组，每个组内有 3 个 ID。 

物理小区标识

###### 4.4.2.1.2 参看

- 参看：36211 的 6.11.1

###### 4.4.2.1.3 思想及实现

- 思想：考虑不同系统带宽下的兼容性，只是把信号放在中心频带，这样就和下行带宽没关系了。

- **原始 Bit 数:** ZC 序列构成, 序列长 63. 实际上只使用 62 个, 因为 63 个子载波里面有个 DC 子载波不用。

$N_{ID}^{(2)}$	Root index $u$
0	25
1	29
2	34

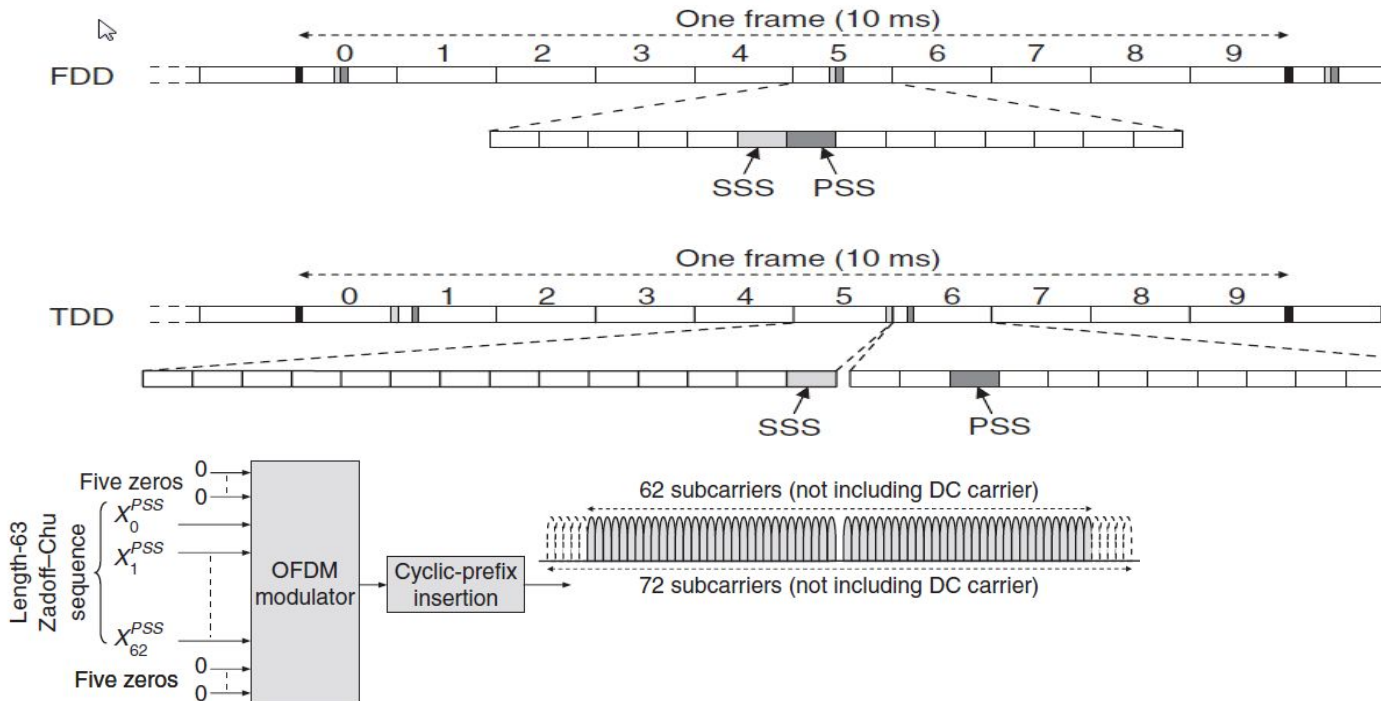
$$d_u(n) = \begin{cases} e^{-j \frac{\pi u n (n+1)}{63}} & n = 0, 1, \dots, 30 \\ e^{-j \frac{\pi u (n+1)(n+2)}{63}} & n = 31, 32, \dots, 61 \end{cases}$$

从这里除以 63 就能明白开始说了。当  $n$  为 62 的时候, 能整除 63. 所以是 DC。

- 对应的资源分配的基本单位: RE
- 位置:

**频域:** 占中心频率的 72 个 RE。ZC 序列总共 63 个符号, 但中间的第 32 个符号打掉 (从上面除以 63 就能看出来), 同时两边预留 10 个子载波做保护带。这样的设计主要是考虑不同系统带宽下的兼容性, 无论在那种带宽下, UE 都只需要接频带中心的 6 个 RB 上的这些 ZC 序列就可以了

**时域:** 从下图中可以看出 TDD 是在第 1 和第 6 个子帧的第三个符号上。



注: (1) PSS 在系统帧内的不同时间隙传输的序列都是一样的, 所以只能得到 5ms 的同步, 得到它并不能知道这是在系统帧第 1 号子帧, 还是第 6 号子帧。



(2) 通过不同的 PSS,SSS 的相对位置, 还可以决定是 TDD 还是 FDD 模式, 所以为什么广播信息里面并没有指定是 TDD 或者 FDD 的 eNodeB。

具体发送数据的位置:  $k$  是频域,  $l$  是时域

$$a_{k,l} = d(n), \quad n = 0, \dots, 61$$

$$k = n - 31 + \frac{N_{RB}^{DL} N_{sc}^{RB}}{2}$$

保留的不发送 10 个子载波的位置:

$$k = n - 31 + \frac{N_{RB}^{DL} N_{sc}^{RB}}{2}$$

$$n = -5, -4, \dots, -1, 62, 63, \dots, 66$$

- UE 做法: 就是尝试着用不同的值去匹配收到的信号, 来判断具体的值。

#### 4.4.2.2 S-SS(辅同步信号)

##### 4.4.2.2.1 作用

- 作用: 10ms 同步, 获得 CELL 组 ID(0~167), 根据不同的 ZC 序列获得。

##### 4.4.2.2.2 参看

- 参看: 36211 的 6.11.2

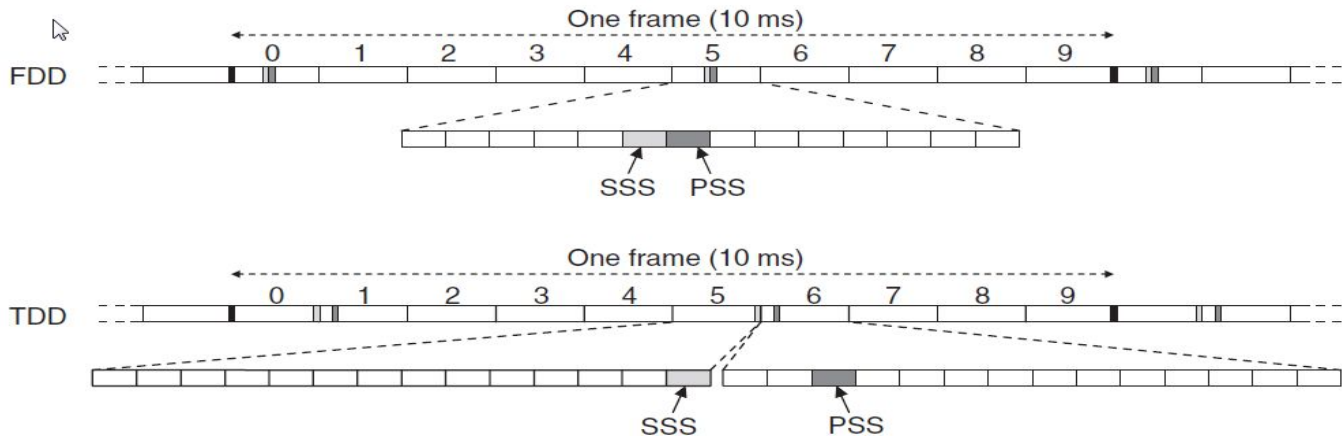
##### 4.4.2.2.3 思想及实现

- 思想: 考虑不同系统带宽下的兼容性, 只是把信号放在中心频带, 这样就和下行带宽没关系了。
- 原始 Bit 数: ZC 序列构成, 序列长 63. 实际上只使用 62 个, 因为 63 个子载波里面有个 DC 子载波不用。参考: 36211 的 6.11.2.1
- 对应的资源分配的基本单位: RE
- 位置:

频域: 占中心频率的 72 个 RE。中间的第 32 个符号打掉, 同时两边预留 10 个子载波做保护带。这样的设计主要是考虑不同系统带宽下的兼容性, 无论在那种带宽下, UE 都只需要接频带中心的 6 个 RB 上的这些 ZC 序列就可以了。

TDD 时域是在第 5 个子帧的最后一个符号上

时域:



具体发送数据的位置：k 是频域，l 是时域

$$a_{k,l} = d(n), \quad n = 0, \dots, 61$$

$$k = n - 31 + \frac{N_{RB}^{DL} N_{sc}^{RB}}{2}$$

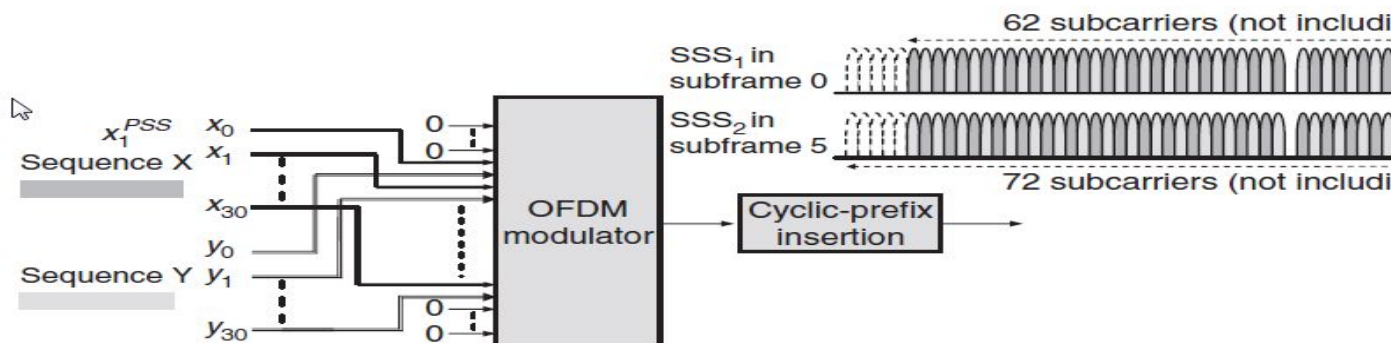
$$l = \begin{cases} N_{symb}^{DL} - 2 & \text{in slots 0 and 10 for frame structure type 1} \\ N_{symb}^{DL} - 1 & \text{in slots 1 and 11 for frame structure type 2} \end{cases}$$

保留的不发送 10 个子载波的位置：

$$k = n - 31 + \frac{N_{RB}^{DL} N_{sc}^{RB}}{2}$$

$$l = \begin{cases} N_{symb}^{DL} - 2 & \text{in slots 0 and 10 for frame structure type 1} \\ N_{symb}^{DL} - 1 & \text{in slots 1 and 11 for frame structure type 2} \end{cases}$$

$$n = -5, -4, \dots, -1, 62, 63, \dots, 66$$



**Figure 18.3** Definition and structure of SSS.

注：（1）SSS 在系统帧内的不同时间隙传输的序列都是不一样的，X,Y 序列交换了位置。所以可以根据那个和预先定义的 ZC 序列是否一样来判断是 1 子帧还是 6 子帧

#### 4.4.2.3 问题

- 问题 1：为什么要分‘主辅同步信号’??

答：个人理解，从公式来看，主同步信号只有三种可能，所以检测比较快。把主辅同步信号分开是为了‘手机开机的时候’检测的时间缩短。首先，主同步只有三种可能，检测出来之后；然后再通过‘主辅同步信号的相对位置’来检测‘辅同步信号’，不用慢慢的去试‘辅同步信号在哪里了’。如果，主辅同步信号合并在一起，手机不仅要‘慢慢试同步信号的位置，而且每个位置还得试 503 次（PCI 的取值范围），显然比试 3 次慢得多。

- 问题 2：为什么‘辅同步信号要放在主同步信号的前面’，而不是按照一般的思想‘辅同步’放在后面？？

### 4.4.3 下行参考信号 -- 规定

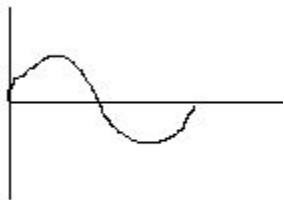
#### 4.4.3.1 CELL 相关的参考信号

##### 4.4.3.1.1 作用

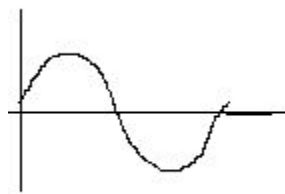
- 作用：为了用来做信道评估，方便相干解调用。

##### 4.4.3.1.2 相干解调

个人猜想，就是根据接收到的参考信号的变化，来看参考信号附近的‘RE 对应的数据应该进行怎样的变化’。



期望收到的参考信号的样子；



实际收到的参考信号的样子。

假设实际收到的参考信号比期望的相位偏转了 15 度，振幅也提高了 1.3 倍。那么该参考信号附近的 RE 对应的‘调制符号’，也应该进行对应的反变化：相位反方向偏转 15 度，振幅减少到 1/1.3 倍。这就叫相干解调。然后再把‘调制符号’反向对应成 bit。

##### 4.4.3.1.3 参看

- 参看：36211 的 6.10.1

##### 4.4.3.1.4 思想及实现

- 思想：

- (1) **频域上**: 为了能有好的相干解调的效果, 而又不占太多的资源, 频域的跨度是 6 个子载波。
- (2) **时域上**: 为了保证参考信号的在一个子帧的时域上的跨度比较均匀,  $14/4=3$ ; 而且不能和 PSS 和 SSS 相冲突, 所以就设计成了 ‘第一个符号, 和倒数第三个符号’

注: 个人理解这些设计应该是经过大量模拟实验的结果。

- **周期**: 每一个子帧
- **长度**: 参考信号的序列长度都是按下行最大的 110 个资源块做预算算出来的, 然后再根据实际的 CELL 下行带宽进行截取。注意: 是向后对齐截取。
- **位置**: 非 MBSFN 子帧, 由 cell ID 决定参考信号的起始位置 (6 种 shift), 一般是每个时隙的第一个符号和倒数第三个符号。

**\*\* 6 种频域的 Shift 的情况 --- 为什么有 shift?? 防止相邻小区同位置的参考信号干扰, 而参考信号在一个 symbol 下面, 它们频域上的跨度正好是 6 个子载波, 所以这里最多也就只有 6 种 shift 的情况。由 PCI 决定。**

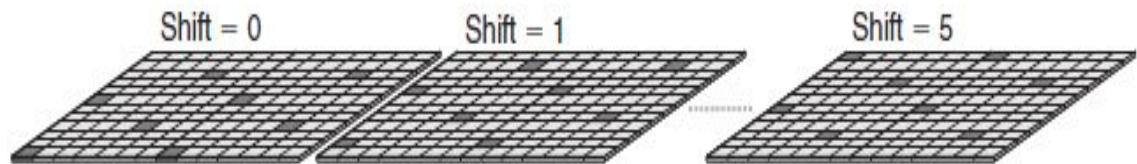
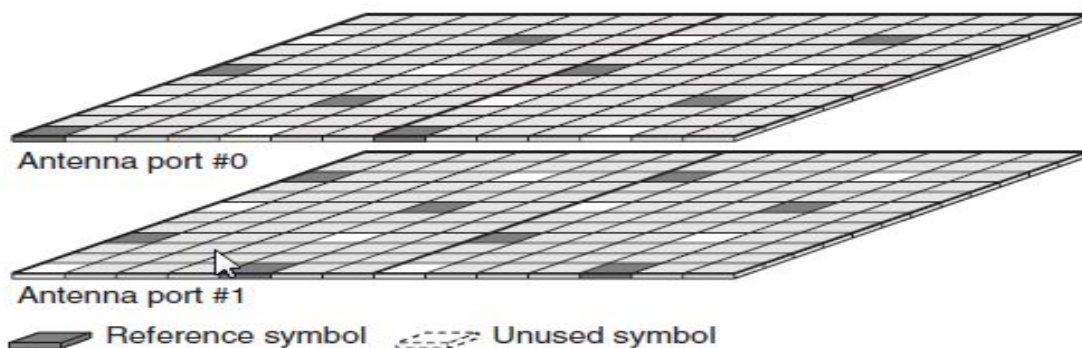


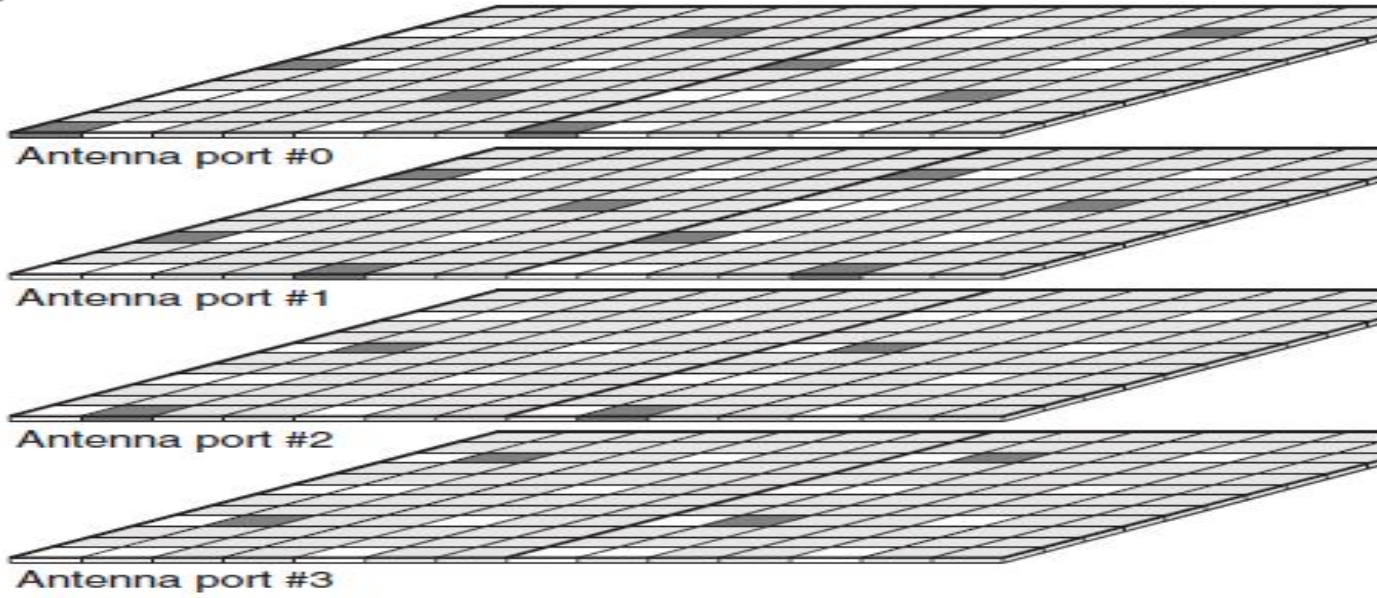
Figure 16.9 Different reference-signal frequency shifts.

**\*\* 不同物理 port 数参考信号的位置**





5



注：参考信号是个很有势力的恶霸，一根天线的参考信号占用的资源也不能被其他天线的作为任何目的使用。因为参考信号太关键了，怕干扰。

• 公式：

(1) 算出最大的序列长度，显然会加上时隙,PCI，还有符号的位置：

$$r_{l,n_s}(m) = \frac{1}{\sqrt{2}}(1 - 2 \cdot c(2m)) + j \frac{1}{\sqrt{2}}(1 - 2 \cdot c(2m + 1)), \quad m = 0, 1, \dots, 2N_{RB}^{\max, DL} - 1$$

$$c_{init} = 2^{10} \cdot (7 \cdot (n_s + 1) + l + 1) \cdot (2 \cdot N_{ID}^{cell} + 1) + 2 \cdot N_{ID}^{cell} + N_{CP}, \quad N_{CP} = \begin{cases} 1 & \text{for normal CP} \\ 0 & \text{for extended CP} \end{cases}$$

$n_s$  时隙号：0~19,  $l$ ：0~6.

注：每个时域的 symbol 上对应的参考信号的序列是不一样的。

(2) 向后对齐截取

符号映射：注意映射的序列的序号是从 “ $N_{RB}^{\max, DLI} - N_{RB}^{DL}$ ” 到

$N_{RB}^{\max, DLI} + N_{RB}^{DL} - 1$ ，正好让 cell 之间错开

$$a_{k,l}^{(p)} = r_{l,n_s}(m')$$

$$k = 6m + (v + v_{\text{shift}}) \bmod 6$$

$$l = \begin{cases} 0, N_{\text{symp}}^{DL} - 3 & \text{if } p \in \{0, 1\} \\ 1 & \text{if } p \in \{2, 3\} \end{cases} \quad v = \begin{cases} 0 & \text{if } p = 0 \text{ and } l = 0 \\ 3 & \text{if } p = 0 \text{ and } l \neq 0 \\ 3 & \text{if } p = 1 \text{ and } l = 0 \\ 0 & \text{if } p = 1 \text{ and } l \neq 0 \\ 3(n_s \bmod 2) & \text{if } p = 2 \\ 3 + 3(n_s \bmod 2) & \text{if } p = 3 \end{cases}$$

$$m = 0, 1, \dots, 2 \cdot N_{RB}^{DL} - 1$$

$$m' = m + N_{RB}^{\max, DL} - N_{RB}^{DL}$$

$$v_{\text{shift}} = N_{ID}^{cell} \bmod 6$$

为什么 mod 6? 是因为频域上的 CELL-RS 的间隔也就是 6 个子载波。

注意: port2, port3 上面的参看信号是根据时隙号  $n_s$  变化而交替变化的。

#### 4.4.3.1.5 问题

##### 4.4.3.1.5.1 问题 1: 为什么 CELL 参考信号要有 shift?

防止相邻小区同位置的参考信号干扰

##### 4.4.3.1.5.2 问题 2: 为什么四个 port 的时候, 第三, 四两个 port 只有一个符号作为参考信号呢?

减少参考信号导致的 overhead, 负面影响也就是不能适应变化过快的信道情况。例如高速移动的时候, 也是一种折中的思想。

##### 4.4.3.1.5.3 问题 3: 为什么不同天线 port 对应的参考信号不做成正交的呢, 这样不是可以节约空口资源吗, 因为不同天线 port 可以占用同样的 RE 了?

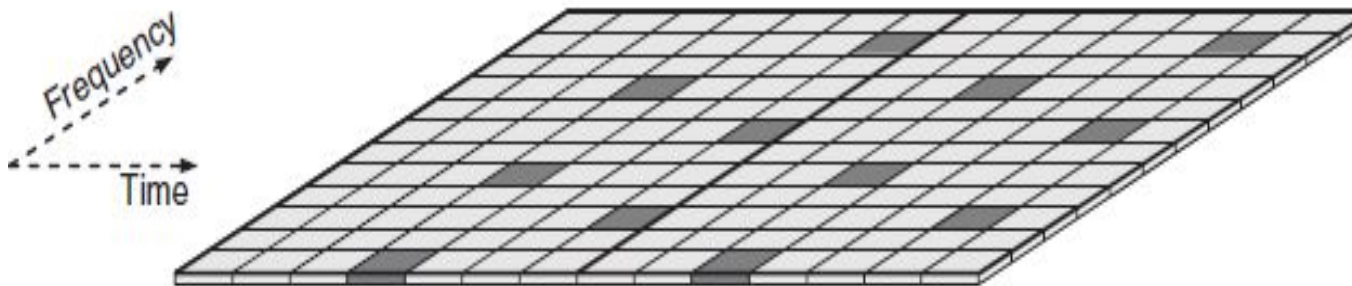
个人认为, 由于无线环境变化不定, 虽然每个 port 发送出去的是正交的, 等到 UE 接收到的时候也许就不会正交了, 这样解出的某个 port 上的原始的参看信号就不准确, 就会影响到相干解调, 进而影响到 CQI 的上报, 最后会对整个系统产生影响。

#### 4.4.3.2 特定 UE 相关的参考信号

##### 4.4.3.2.1 作用

- 作用:

个人理解: 一般只有在信道条件很差的情况下才能用它, 因为正常的情况可以用 cell 的参考信号做相干解调, 用它只能说明用 cell 的参考信号已经解不出来的或者解得很差了, 信道条件很差了, 用‘特定 UE 相关的参考信号’增加相干解调的成功率 (你可以看从‘ue-RS 的频域间隔 3 个子载波, 而不是正常的 cell-RS 的间隔 6 个子载波; ue-RS 的时域间隔 2 个符号, 而不是正常 3 个符号’推论出来)。



#### 4.4.3.2.2 参看

- 参看：36211 的 6.10.3

#### 4.4.3.2.3 思想及实现

- 思想来源：

从作用也能大概了解到设计者的思想的行程。就是想：如果正常的参考信号不能很好的做相干解调了，CQI 很低了怎么办？？？于是就想了这种方法，产生一种新的 RS，UE-RS，它要比 CELL-RS 之间的间隔小，这样提高相干解调成功率，提高 CQI，能弥补一些 UE-RS 占用资源带来的一些缺陷。

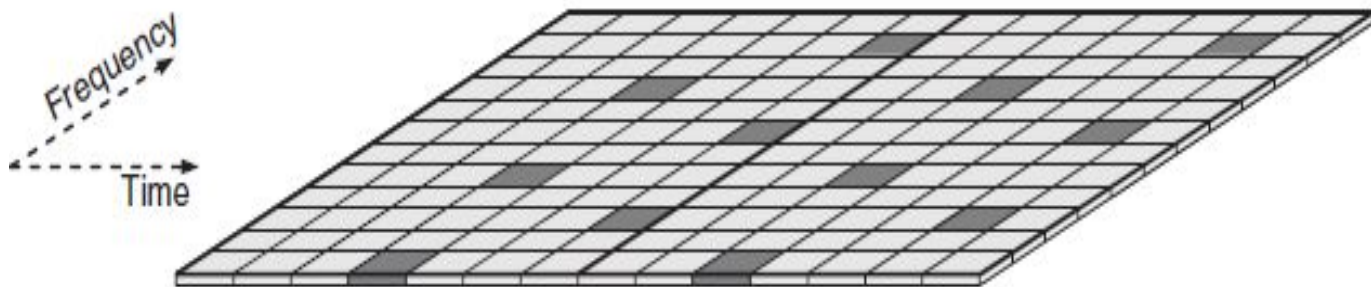
- 缺点：

缺点也显然一见：可用的资源少了，一般情况下吞吐量会降下来。为什么说是一般情况？？后面我们说到 PDCCH 对应的 UE 的 TM 模式的时候会详细来说，现在说很多还理解不了。

- 周期：一个子帧

- 长度：UE 分配的下行数据的带宽

- 位置：就是和 UE 的下行数据一起传送，所以在变化



- 公式：

$$r(m) = \frac{1}{\sqrt{2}}(1 - 2 \cdot c(2m)) + j \frac{1}{\sqrt{2}}(1 - 2 \cdot c(2m + 1)), \quad m = 0, 1, \dots, 12 N_{\text{RB}}^{\text{PDSCH}} - 1$$

为什么为 12？？看图就能知道一对 RB pair 有 12 个特定 UE 相关的参考信号。

$$c_{\text{init}} = (\lfloor n_s / 2 \rfloor + 1) \cdot (2 N_{\text{ID}}^{\text{cell}} + 1) \cdot 2^{16} + n_{\text{RNTI}}$$

$$a_{k,l}^{(p)} = r(3 \cdot l' \cdot N_{\text{RB}}^{\text{PDSCH}} + m')$$

$$\begin{aligned}
 k &= (k') \bmod N_{sc}^{RB} + N_{sc}^{RB} \cdot n_{PRB} \\
 k' &= \begin{cases} 4m' + v_{\text{shift}} & \text{if } l \in \{2, 3\} \\ 4m' + (2 + v_{\text{shift}}) \bmod 4 & \text{if } l \in \{5, 6\} \end{cases} \\
 l &= \begin{cases} 3 & l' = 0 \\ 6 & l' = 1 \\ 2 & l' = 2 \\ 5 & l' = 3 \end{cases} \\
 l' &= \begin{cases} 0, 1 & \text{if } n_s \bmod 2 = 0 \\ 2, 3 & \text{if } n_s \bmod 2 = 1 \end{cases} \\
 m' &= 0, 1, \dots, 3N_{RB}^{PDSCH} - 1 \\
 v_{\text{shift}} &= N_{ID}^{\text{cell}} \bmod 3
 \end{aligned}$$

为什么 mod 3? 是因为频域上的 UE-RS 的间隔也就是 3 个子载波

#### 4.4.3.2.4 问题

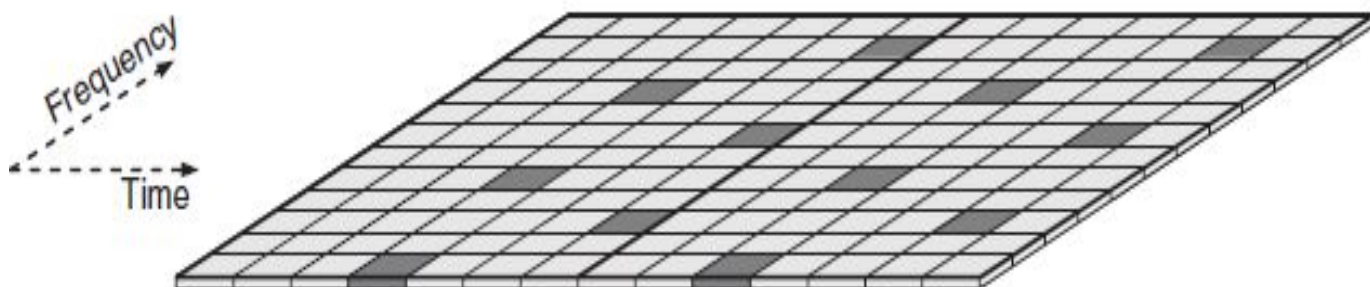
##### 4.4.3.2.4.1 问题 1: 为什么要有针对特定 UE 的参考信号呢?

因为信道条件非常差, 用 cell 的参考信号都不能解码成功的情况下, 只能通过增加参考信号的密度用作相干解调增加解码成功率, 所以想出一种办法就是‘特定 UE 的参考信号’。所以特定 UE 的参考信号是适用在‘信道条件非常差的情况下’。

##### 4.4.3.2.4.2 问题 2: UE 怎么知道自己有没有 UE 相关的参考信号呢? 又怎么知道在 UE 接收的资源块的什么位置呢?

通过配置的传输模式 TM7 (后面讲 PDCCH 的时候会提到), 还有就是盲检测对应的 DCI format 1 知道的。而且 UE 相关的参考信号针对某个资源块对的位置是固定的。

##### 4.4.3.2.4.3 问题 3: UE-RS 参考信号为什么时域上‘第一个子帧是在 3,6symbol’的位置, 而‘第二个子帧却是在 2,5symbol’的位置, 不对称?



答：因为第一个子帧 ‘PDCCH 可能占前 3 个 symbol’，所以 0,1,2 就不能用。时域上的跨度 2 个 symbol，所以就是 ‘第一个时隙就是 3,6’，第二个时隙里面又要和第一个时隙的参考信号保持距离上的均匀，所以选择 ‘2,5’。

#### 4.4.4 PBCH(Physical Broadcast Channel) --- 规定

##### 4.4.4.1 作用

- 作用：周期性的发送 MIB 消息. 通过解调 PBCH，可以得到系统帧号和带宽信息，以及 PHICH 的配置以及天线配置。

##### 4.4.4.2 参看

- 参看：36212 的 5.3.1 和 36211 的 6.6

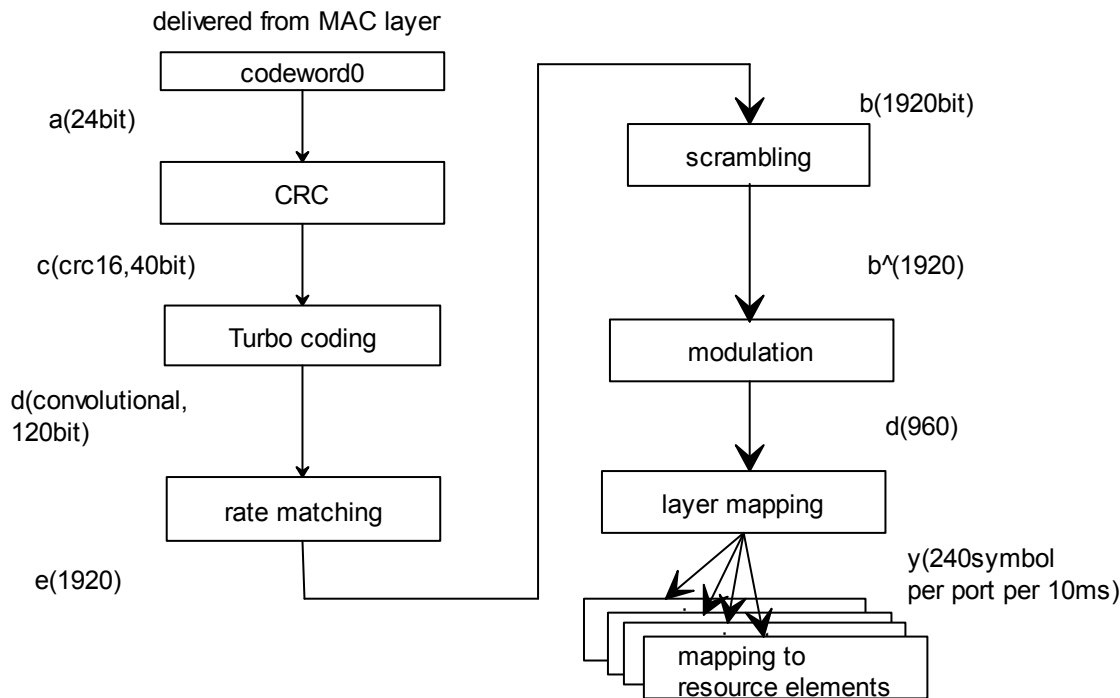
##### 4.4.4.3 思想来源

- 任何一套通讯系统开始都必须通过一种 ‘潜规则’ 去获得一些最基本的信息，例如： ‘带宽’，LTE 中的 ‘PHICH’ 的配置情况。然后通过这些基本信息和一些规则不断的推论出其他的東西。PBCH 携带的 MIB 信息就是 ‘最基本的信息’。
- 主辅同步信号携带的也是 ‘最基本的信息’。

##### 4.4.4.4 数据来源

- 原始 BIT 数： 24bit (downlink cell bandwidth 3bit + PHICH config 3bit + SFN 的前 8 个 bit 8bit + 10bit spare)

4.4.4.5 流程



注：（1）因为‘BCH 的一个 transport block’对应的 40ms 的周期，所以会图中加扰对应到了 4 次资源的映射。也就是通过四次发送。具体发送时间，频域可以看后面

（2）enodeb 的天线端口的数目是‘算在 crc16 中的’。

$$\begin{aligned} c_k &= a_k && \text{for } k = 0, 1, 2, \dots, A-1 \\ c_k &= (p_{k-A} + x_{ant,k-A}) \bmod 2 && \text{for } k = A, A+1, A+2, \dots, A+15. \end{aligned}$$

Table 5.3.1.1-1: CRC mask for PBCH

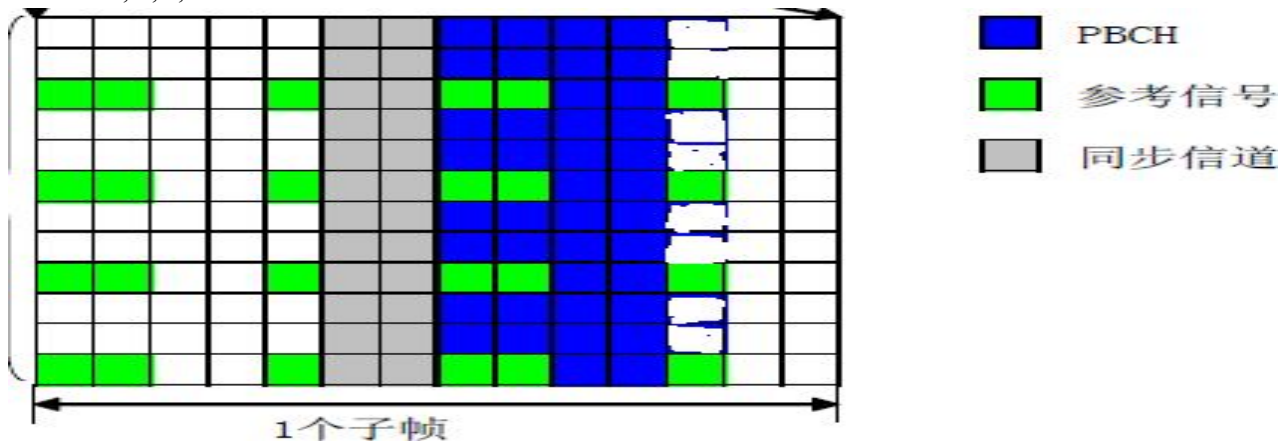
Number of transmit antenna ports at eNode-B	PBCH CRC mask $\langle x_{ant,0}, x_{ant,1}, \dots, x_{ant,15} \rangle$
1	$\langle 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 \rangle$
2	$\langle 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1 \rangle$
4	$\langle 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1 \rangle$

- **速率匹配：**和 Rvidx（redundancy version）没有关系。由于没有分段不可能有<NIL>数据，其实就相当与把‘turbo 编码后的 120bit’的数据重复了‘16 次’。
- **扰码的初始化：** $c_{init} = N_{ID}^{cell}$ 。
- **调制模式：**QPSK
- **发射分集：**层映射的层数应该与实际物理的天线端口数一样。不管多少根天线，层映射之后虽然每层的 symbol 符号数变成了 240/portNum，但后面发射分集之后每个 port 对应的 symbol 数又恢复到了 240。

4.4.4.6 时频位置 ---- 潜规则

- 频域(k): 与 CELL 带宽无关, 只分配在中心频点的 72 子载波上
- 时域(l 符号位置):  
PBCH 传输的周期从无线帧 (满足  $n_f \bmod 4 = 0$ ) 开始, 在子帧#0 的时隙#1 的中间 72 个 RE

$$k = \frac{N_{RB}^{DL} N_{sc}^{RB}}{2} - 36 + k', \quad k' = 0, 1, \dots, 71$$
$$l = 0, 1, \dots, 3$$



注:

(1) 显然如果以后天线 port 增加了, CELL 对应的下行参考信号增加了。上图中有些‘蓝色’放 MIB 消息的地方就得去掉了, 算法也就需要修改。

(2) 这个图的同步信号只是对 FDD(帧格式 1 有效)。TDD(帧格式 2)对应的同步信号的位置在不同的位置。

4.4.4.7 协议结构体

- RRC:: MasterInformationBlock

4.4.4.8 特殊性

- 频域位置和 CELL 带宽无关, 只分配在中心频点的 72 子载波上。
- 由于 UE 开始没有收到 MIB 信息, 并不知道天线数, 所以只能假设最多的天线数 4, 这样对应的参考信号的位置就不能放置数据。
- 小区对应的天线数隐性放在 CRC 中的。根据 CRC 不同的序列决定小区不同的天线数

eNode-B 使用的天线端口数	PBCH CRC mask $\langle x_{ant,0}, x_{ant,1}, \dots, x_{ant,15} \rangle$
1	$\langle 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 \rangle$
2	$\langle 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1 \rangle$
4	$\langle 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1 \rangle$



- 在 PBCH 的 MIB 广播中只广播系统帧号的前 8 个 bit，因为系统帧号是 0~1023 所以只需要 10 个 bit 表示，剩下的两位根据该帧在 PBCH 40ms 周期窗口的位置确定，第一个 10ms 帧为 00，第二帧为 01，第三帧为 10，第四帧为 11。PBCH 的 40ms 窗口手机可以通过盲检确定。开始我也认为‘剩下的两位根据该帧在 PBCH 40ms 周期窗口的位置确定’，但其实有更简单直接的方法：解码 PBCH 的时候，由于扰码是用  $c_{init} = N_{ID}^{cell}$  初始化，你知道用生成的扰码的那一部分解码成功，其实就可以推断出系统帧的后面两位了，例如：如果生成的扰码是 0000 1111 0011 1100，如果我用 0000 解码成功就知道‘对应的系统帧号的后两位是 00，如果用 0011 解码成功就知道是 10 了’。
- 可靠性接收，主要通过 FEC 前向纠错机制，时间分集与天线分集来实现。时间分集是，让 PBCH 在 40ms 里面重复 4 次，每一次都是自解码的，当然也可以合并解码，因此在 40ms 里面都丢失的可能性就非常低了，因为 MIB 消息比较小，前向纠错码采用卷积码实现，编码率是 1/3，编码以后，在对系统 bit 与校验 bit 重复，这样下来 40ms 相应的编码率只有 1/48，相当的低。这样做的目的显然是 MIB 消息太重要了，绝对不允许错，所以宁愿浪费点资源。

#### 4.4.4.9 问题

**4.4.4.9.1 问题 1：**其实个人觉得没必要只广播系统帧号的前 8 个 bit，后面又 10bit 的预留，完全可以把后面 2bit 加进去？

#### 4.4.5 PCFICH(Physical Control Format Indicator Channel)

##### 4.4.5.1 作用

- 指示子帧控制域的符号数，即一个 TTI 里面 PDCCH 占用的符号数。

##### 4.4.5.2 作用

- 参看：36212 的 5.3.4 和 36211 的 6.7

##### 4.4.5.3 思想来源

- 显然是为了节省资源，因为有时调度的时候 UE 数目不多，占用的资源也不多，这样有一个选择的空间。PDCCH 资源少了，那么 PDSCH 的资源也就多出来了，能多传真实的数据。后面讲 PDCCH 和 PDSCH 的时候就能理解了。

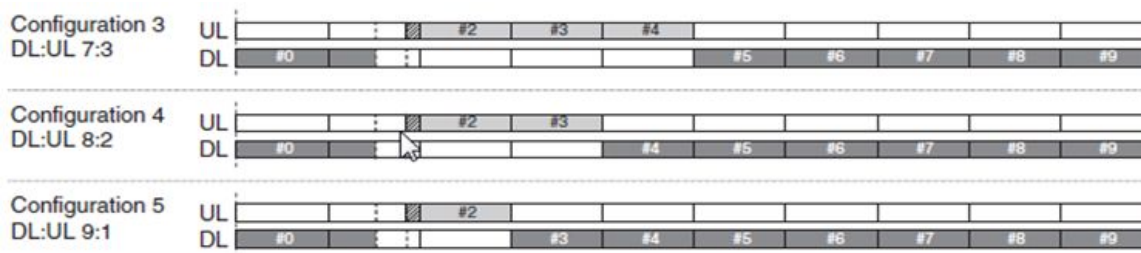


#### 4.4.5.4 数据来源

36211 Table 6.7-1: Number of OFDM symbols used for PDCCH.

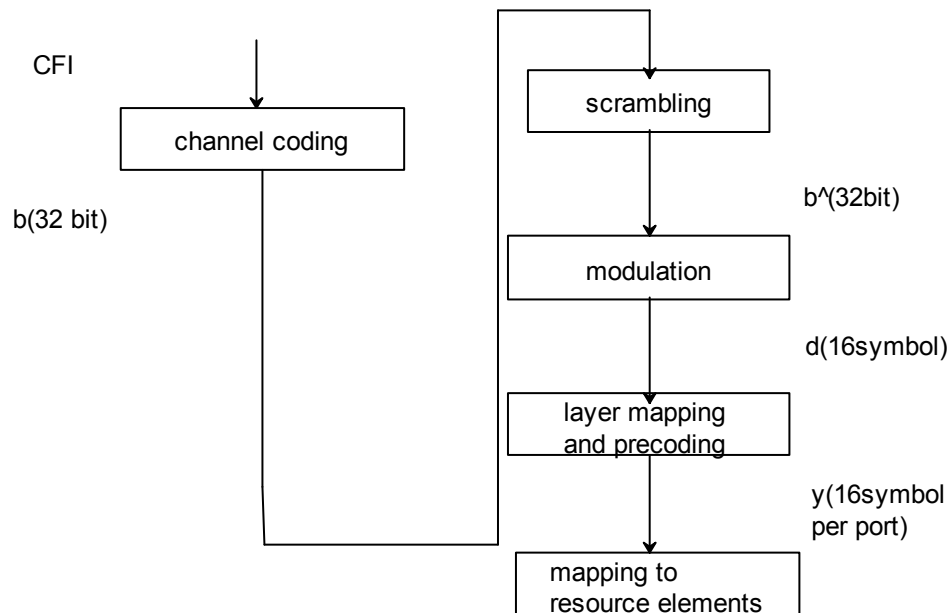
Subframe	Number of OFDM symbols for PDCCH when $N_{RB}^{DL} > 10$	Number of OFDM symbols for PDCCH when $N_{RB}^{DL} \leq 10$
Subframe 1 and 6 for frame structure type 2	1, 2	2
MBSFN subframes on a carrier supporting both PMCH and PDSCH for 1 or 2 cell specific antenna ports	1, 2	2
MBSFN subframes on a carrier supporting both PMCH and PDSCH for 4 cell specific antenna ports	2	2
MBSFN subframes on a carrier not supporting PDSCH	0	0
All other cases	1, 2, 3	2, 3, 4

- For system bandwidths  $N_{RB}^{DL} > 10$ , the span of the DCI in units of OFDM symbols, 1, 2 or 3, is given by the CFI. For system bandwidths  $N_{RB}^{DL} \leq 10$ , the span of the DCI in units of OFDM symbols, 2, 3 or 4, is given by CFI+1.
- 应该是出于‘帧结构 2 的子帧 1 和 6 是特殊子帧的考虑，所以 PCFICH 配置不能配置 3’。但其实对于帧结构 2（TDD）的 config3,4,5 这几种配置情况，6 子帧并不是特殊子帧，其实可以配置此时 CFI 可以配置为 3 的。从这里也可以看出设计得太复杂，最后协议自己也有点不能‘不厌其烦’了，就好像再说‘太繁琐了，我自己都有点受不了了，一刀切简单点’。



- 显然决定 PCFICH 取值就是看这次‘对应的调度的 UE 总共会占多个 CCE 也就是 PDCCH 占用的范围来决定’，后面介绍 PDCCH 的时候再介绍。

### 4.4.5.5 流程



- 编码是固定的：

36212 ---Table 5.3.4-1: CFI codewords

CFI	CFI codeword < b <sub>0</sub> , b <sub>1</sub> , ..., b <sub>31</sub> >
1	<0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1>
2	<1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0>
3	<1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1>
4 (Reserved)	<0,0>

- 加扰： $c_{init} = (\lfloor n_s/2 \rfloor + 1) \cdot (2N_{ID}^{cell} + 1) \cdot 2^9 + N_{ID}^{cell}$ 。跟 cell 相关，而且跟时间相关，所以扰码的初始化与 PCI 和  $n_s$  相关。但 PCFICH 一次 TTI（1MS）一次，对于的一个子帧（0~9），所以这里用  $n_s/2$ ,  $n_s$  表示的一个帧内时隙号（0~19）。
- 调制模式：QPSK（1 个调制符号可以表示 2bit）
- 层映射和预编码：都是和 PBCH 一样 ‘层数和 port 数一样，而且预编码使用发射分集’。
- 对应的资源分配的基本单位: REG
- 编码之后符号数：32bit --> 16 QPSK 符号 -> 4 个资源组（REG）

#### 4.4.5.6 时频位置

- 频域 ( $k$ ) : 4 个 REG 均匀分布在 cell 的下行带宽的上
- 时域( $l$ ): 每个子帧的第一个符号

$z^{(p)}(0)$  is mapped to the resource - element group represented by  $k = \bar{k}$

$z^{(p)}(1)$  is mapped to the resource - element group represented by  $k = \bar{k} + \left\lfloor \frac{N_{RB}^{DL}}{2} \right\rfloor \cdot \frac{N_{sc}^{RB}}{2}$

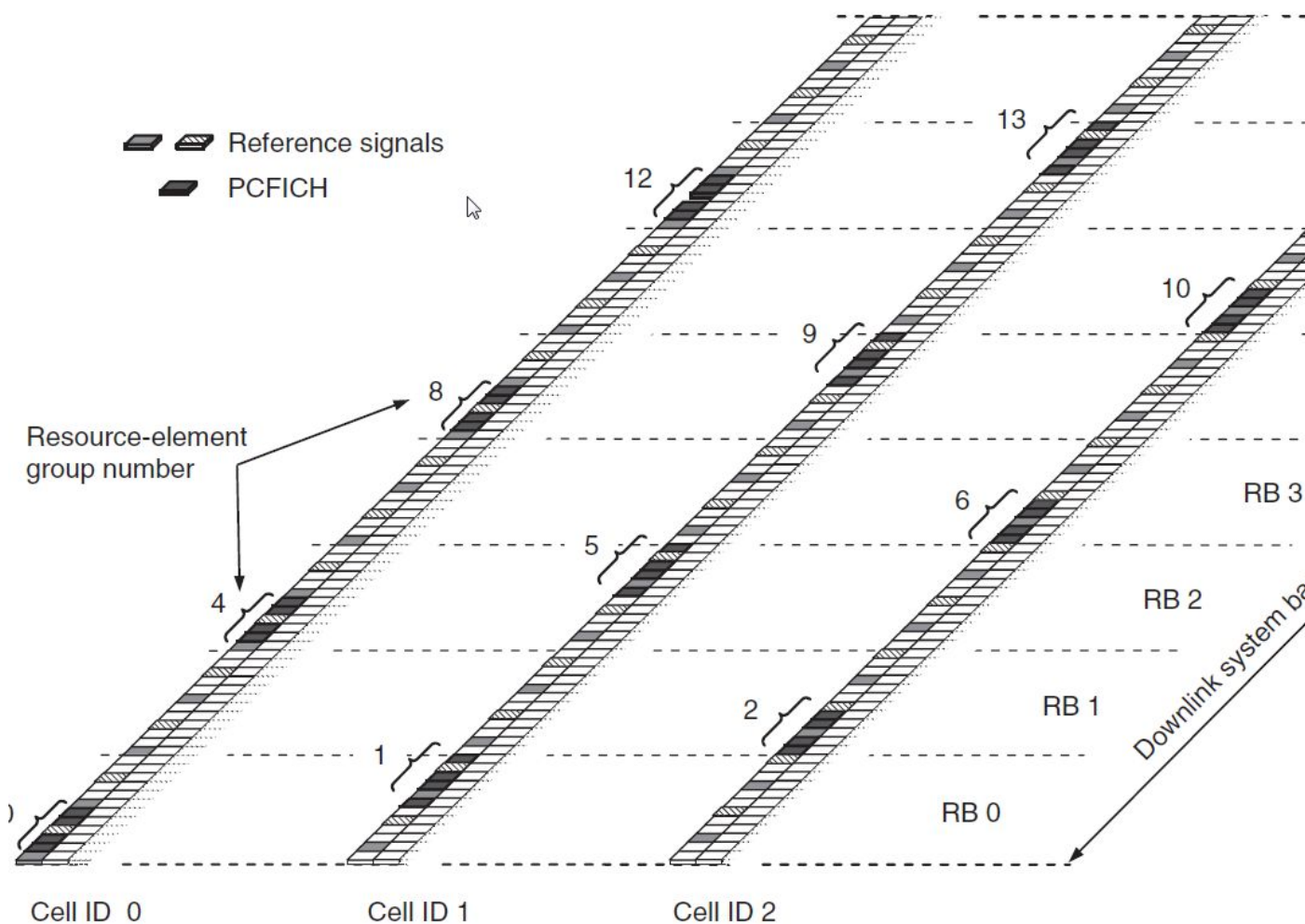
$z^{(p)}(2)$  is mapped to the resource - element group represented by  $k = \bar{k} + \left\lfloor \frac{2N_{RB}^{DL}}{2} \right\rfloor \cdot \frac{N_{sc}^{RB}}{2}$

$z^{(p)}(3)$  is mapped to the resource - element group represented by  $k = \bar{k} + \left\lfloor \frac{3N_{RB}^{DL}}{2} \right\rfloor \cdot \frac{N_{sc}^{RB}}{2}$

注：上面的结果都要再对  $N_{RB}^{DL} N_{sc}^{RB}$  取模。

$\bar{k} = \left( \frac{N_{sc}^{RB}}{2} \right) \cdot \left( N_{ID}^{cell} \bmod 2N_{RB}^{DL} \right)$  --- PCI 决定偏移。

注：从 CELL-RS 的分布就可以看出，为什么  $\bmod 2N_{RB}^{DL}$ ，因为第一个符号中下行带宽对应的除去 CELL-RS 剩下的 REG 数也就是  $2N_{RB}^{DL}$  个。



#### 4.4.5.7 协议结构体

无

#### 4.4.5.8 特殊性

无

#### 4.4.5.9 问题

##### 4.4.5.9.1 问题1: 为什么 PCFICH 的加扰要和时间有关

$c_{\text{init}} = (\lfloor n_s/2 \rfloor + 1) \cdot (2N_{\text{ID}}^{\text{cell}} + 1) \cdot 2^9 + N_{\text{ID}}^{\text{cell}}$ , 而 PBCH 的加扰却不和时间有关能  $c_{\text{init}} = N_{\text{ID}}^{\text{cell}}$  ?

因为 PBCH 始终都是在子帧 0 的第一个时隙上。也就是  $n_s$  固定=1, 没有变化, 所以没有必要加进加扰中; 而 PCFICH 是有子帧的变化的, 所以需要把子帧号加入进行加扰。

### 4.4.6 PHICH(Physical Hybrid-ARQ Indicator Channel)

#### 4.4.6.1 作用

- 对上行 (uplink UP-SCH) UE 传送的数据进行应答确认。

#### 4.4.6.2 参看

- 参看: 36212 的 5.3.5, 36211 的 6.9, 36213 的 9.1.2

#### 4.4.6.3 思想来源

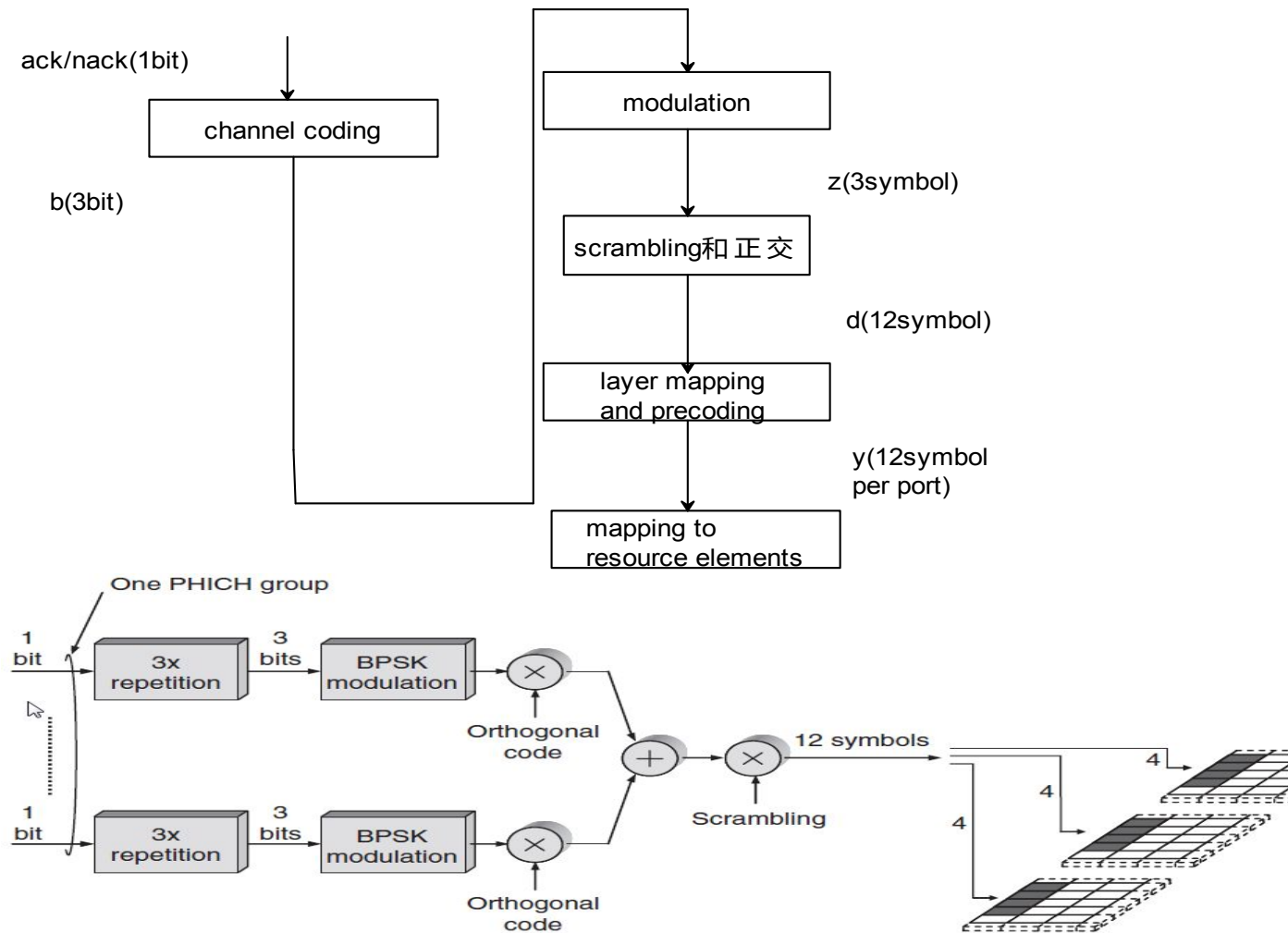
- 思想演变过程:
  - (1) 如果针对一个 UE 就对应一次回应就对应叫做一个 PHICH, 由于上行没有空分, 所以只有 1bit, 对应与一个 RE 就够了。
  - (2) 但如果每个 UE 的上行回应都对应一个 RE。一是容易出错, 二是 eNodeB 对于不同 UE 发送 PHICH 就需要的功率不一样, 这种变化太快了, 对 RF (射频处理) 是一个难度。所以就设计把多个 PHICH 通过码分正交的方式合成一个, 形成了 PHICH GROUP 以减小 RF 的处理难度。
- 协议规定: (思想定下来了, 具体实现的结果就可以有多种可能了) 一般情况也是用 PHICH GROUP 去完成 ACK/NACK 响应的。
  - (1) Normal cyclic prefix 一个 PHICH GROUP 有 8 个 PHICH, 码分的长度是 4;
  - (2) 码分之后产生 12 个调制符号, 最后对应到 3 个 REG 上面去。这 3 个 REG 均匀分布在下行带宽上
- 好处:
  - (1) 提高了数据接收的可靠性

(2) 减少了射频处理的难度

4.4.6.4 数据来源

- 对于上行由于没有空分，所以只有是一个码字。所以只有 1bit 的 ACK/NACK (0: NACK; 1: ACK)。

4.4.6.5 流程



注：最后多个 HICH（一个 HICH GROUP）会映射到 3 个 REG（12symbol），每个 TTI, 一般情况是均匀分布在第一个符号的子载波，靠配置决定。

- 信道编码：固定的，1bit 重复成 3bit

63212 ---- Table 5.3.5-1: HI codewords

HI	HI codeword < b <sub>0</sub> , b <sub>1</sub> , b <sub>2</sub> >
0	< 0,0,0 >

1	$\langle 1, 1, 1 \rangle$
---	---------------------------

- 调制模式: BPSK(1 个调制符号表示 1bit)
- 加扰和正交:  $c_{\text{init}} = (\lfloor n_s/2 \rfloor + 1) \cdot (2N_{\text{ID}}^{\text{cell}} + 1) \cdot 2^9 + N_{\text{ID}}^{\text{cell}}$   

$$d(i) = w(i \bmod N_{\text{SF}}^{\text{PHICH}}) \cdot (1 - 2c(i)) \cdot z(\lfloor i/N_{\text{SF}}^{\text{PHICH}} \rfloor)$$

$$i = 0, \dots, M_{\text{symb}} - 1$$

$$M_{\text{symb}} = N_{\text{SF}}^{\text{PHICH}} \cdot M_s$$

$$N_{\text{SF}}^{\text{PHICH}} = \begin{cases} 4 & \text{normal cyclic prefix} \\ 2 & \text{extended cyclic prefix} \end{cases}$$

Normal 情况下  $M_{\text{symb}} = 12$ ,  $M_s = 3$

注意: 如下表, 选择哪一个 walsh 序列(正交码)是由  $n_{\text{PHICH}}^{\text{seq}}$  决定的, 后面介绍。

**36211 --- Table 6.9.1-2: Orthogonal sequences  $[w(0) \dots w(N_{\text{SF}}^{\text{PHICH}} - 1)]$  for PHICH**

Sequence index $n_{\text{PHICH}}^{\text{seq}}$	Orthogonal sequence	
	Normal cyclic prefix $N_{\text{SF}}^{\text{PHICH}} = 4$	Extended cyclic prefix $N_{\text{SF}}^{\text{PHICH}} = 2$
0	$[+1 \ +1 \ +1 \ +1]$	$[+1 \ +1]$
1	$[+1 \ -1 \ +1 \ -1]$	$[+1 \ -1]$
2	$[+1 \ +1 \ -1 \ -1]$	$[+j \ +j]$
3	$[+1 \ -1 \ -1 \ +1]$	$[+j \ -j]$
4	$[+j \ +j \ +j \ +j]$	-
5	$[+j \ -j \ +j \ -j]$	-
6	$[+j \ +j \ -j \ -j]$	-
7	$[+j \ -j \ -j \ +j]$	-

- 层映射和预编码: 都是和 PBCH 一样 ‘层数和 port 数一样, 而且预编码使用发射分集’  
**注:** 对于发射分集如果是 4 天线, 对应的矩阵有两个。
- 对应的资源分配的基本单位: REG。3 个 REG 均匀分布在下行带宽上。

#### 4.4.6.6 时频位置

- 什么时候对上行的数据回 ACK---- 设计思想:  
 (1) 距离上行发送数据子帧  $n \geq 4$  的最近的下行子帧回 ACK/NACK, 尽量保持均匀。也就是  $n + k_{\text{PHICH}}$  的那个下行子帧回,  $n$  代表的是有上行数据发送的上行子帧。

注：不管上下行回 ACK/NACK，还是上行的调度时间间隔，一般都是  $n \geq 4$  的算法。从理论上应该马上就可以的，但是由于硬件软件的处理时间和电磁波传送也是需要时间的，所以现在只能达到‘最小间隔为 4’的情况，也应该是一种‘模拟仿真’的结果。

$k_{PHICH}$  的值由下表决定

36213 --- Table 9.1.2-1:  $k_{PHICH}$  for TDD

TDD UL/DL Configuration	UL subframe index $n$									
	0	1	2	3	4	5	6	7	8	9
0			4	7	6			4	7	6
1			4	6				4	6	
2			6					6		
3			6	6	6					
4			6	6						
5			6							
6			4	6	6			4	7	

(2) 一看协议中的表，发现好多都不是“ $n \geq 4$  的最近的下行子帧回 ACK/NACK”，好多都是 6 个子帧之后才回 ACK/NACK。为什么这么做呢？

答：如果你不了解 LTE 的上行调度，你是回答不出这个问题的。原因是 LTE 为了简单和提高效率，尽量把‘对某个上行子帧  $n$  回 ACK/NACK 的下行子帧’和‘给下一个帧上的该上行子帧  $n$  分配上行资源的 UL-grant (DCI0) 的下行子帧’放在一起。

也许听起来绕口。举个例子：

对于 TDD 的 config3

Configuration 3	UL	#0	#1	#2	#3	#4	#5	#6	#7	#8	#9
DL:UL 7:3	DL	#0	#1	#2	#3	#4	#5	#6	#7	#8	#9

对某个 UE，本来 2 子帧发送的上行数据的 ACK/NACK 应该在 6 子帧回的；但是 8 子帧是‘有可能该 UE 在下一个系统帧的 2 子帧有上行调度的分配(也就是 UL grant, 或者 DCI0)，因为‘调度和 ACK/NACK’可能是针对同一个 UE 的，所以为了一起处理，所以把 2 子帧对应的 ACK/NACK 推迟到 8 子帧回了。

结论：

(1) LTE 很多信道都是和其他信道以及其实现相关的，也就是紧耦合，只有完全了解了所有信道及其实现方式才能完全把它们弄明白。

(2) 上行的数据回 ACK/NACK，也就有两个原则：

a. 距离上行发送数据子帧  $n \geq 4$  的最近的下行子帧回 ACK/NACK，尽量保持均匀。

b. 尽量把‘对某个上行子帧  $n$  回 ACK/NACK 的下行子帧’和‘给该上行子帧  $n$  分配上行资源的 UL-grant (DCI0) 的下行子帧’放在一起。

## • Normal 情况下一个子帧有多个 PHICH group

(1) 对于 FDD --- 思想演变过程

>>>>因为分配的最小单位是一对 RB，算下来的话：‘如果一个 TTI 调度的时候，每个 RB 分配给一个 UE，最后算出来的需要的 PHICH group 数就是  $\lceil N_{RB}^{DL} / 8 \rceil$ ’

>>>>但是每个厂商可能对应的一个 TTI 对应的能调度的 UE 数不一样，而且有时为了节省资源的角度，所以再乘以了  $N_g$ ，由配置

RRC::MasterInformationBlock→PHICH-Config→phich-Resource 值为 (ENUMERATED {oneSixth, half, one, two}) 决定。

所以，最后公式就成了：

$$N_{PHICH}^{group} = \begin{cases} \lceil N_g (N_{RB}^{DL} / 8) \rceil & \text{for normal cyclic prefix} \\ 2 \cdot \lceil N_g (N_{RB}^{DL} / 8) \rceil & \text{for extended cyclic prefix} \end{cases}$$

注：按理应该是算成每个 TTI 能调度的 UE 个数来算是最精确的，可以通过它进行配置。协议最后为什么没采取这种方式呢？？？

我也不知道。没采取这种方式也许是怕两个邻小区是不同厂家，配置不一样，产生干扰码？？

## (2)对于 TDD

>>>>> 对于 TDD 由于‘上下行的不对称性’，有些下行子帧可能需要对多个上行子帧进行 ACK/NACK 回应，有些可能都没有上行子帧需要回应。所以需要乘以个系数。

$$N_{PHICH}^{group} = m_i \cdot N_{PHICH}^{group}$$

而  $m_i$  就是代表：该下行子帧有多少个上行子帧数回 PHICH。FDD 都是一对一，所以  $m_i$  为 1.  $m_i$  对应到下表，其实可用由上面的 36213 --- Table 9.1.2-1 回 ACK/NACK (HICH) 的时机推导出来。

36211--- Table 6.9-1: The factor  $m_i$  for frame structure type 2.

Uplink-downlink configuration	Subframe number $i$									
	0	1	2	3	4	5	6	7	8	9
0	2	1	-	-	-	2	1	-	-	-
1	0	1	-	-	1	0	1	-	-	1
2	0	0	-	1	0	0	0	-	1	0
3	1	0	-	-	-	0	0	0	1	1
4	0	0	-	-	0	0	0	0	1	1
5	0	0	-	0	0	0	0	0	1	0
6	1	1	-	-	-	1	1	-	-	1

- UE 怎么选择对应的  $n_{PHICH}^{group}$  和  $n_{PHICH}^{seq}$  呢？？

(1) 选择哪个关键字来计算？

计算的时候 UE 有两个单独的关键字，一是  $r_{nti}$ ，一是 PUSCH 对应的 PRB 的起始位置。但是  $r_{nti}$  的随机性太大，很难控制，范围也大。所以最后根据另外一个唯一性的数据：‘UE 分配的上行的第一个 PRB 对应的位置来决



定，先按‘PHICH 的 group’ 进行分配  $n_{\text{PHICH}}^{\text{group}}$ ，然后再按‘PHICH group 的不同正交偏移  $n_{\text{PHICH}}^{\text{seq}}$ ’ 来进行分配。

这种情况的话： $n_{\text{PHICH}}^{\text{group}} = I_{\text{PRB\_RA}}^{\text{lowest\_index}} \bmod N_{\text{PHICH}}^{\text{group}};$

$$n_{\text{PHICH}}^{\text{seq}} = \left\lfloor I_{\text{PRB\_RA}}^{\text{lowest\_index}} / N_{\text{PHICH}}^{\text{group}} \right\rfloor \bmod 8$$

举个例子：

对应的总共的 PHICH GROUP 的数目只有 3；8 个 UE 分配的 PUSCH 的 PRB 对应的起始位置分别为 0,1,2,3,4,5,6,7；那它们对应的

( $n_{\text{PHICH}}^{\text{group}}, n_{\text{PHICH}}^{\text{seq}}$ ) 组就是

0 ---- (0, 0)

1 ---- (1, 0)

2 ---- (2, 0)

3 ---- (0, 1)

4 ---- (1, 1)

5 ---- (2, 1)

6 ---- (0, 2)

7 ---- (1, 2)

## (2) 为什么会有 $n_{\text{DMRS}}$ ?

但 PHICH GROUP 配置得比较少 (phich-Resource 值为 oneSixth, 或者 half 的情况下)，会产生了一个问题。假设 UE 分配的时候 PRB 的跨度比较大，而 PHICH GROUP 配置得比较少 (phich-Resource 值为 oneSixth, 或者 half 的情况下)，有可能‘不同的 PUSCH 的 PRB 对应的起始位置算出来的  $n_{\text{PHICH}}^{\text{group}}$  和  $n_{\text{PHICH}}^{\text{seq}}$  是一样的’。所以就产生了  $n_{\text{DMRS}}$ ， $n_{\text{DMRS}}$  为什么为 3bit，也是因为一个 PHICH GROUP 能容纳 8 个 PHICH 正交码，对应 8 个 UE 的 ACK/NACK 回应。

举个例子：

而对应的总共的 PHICH GROUP 的数目只有 3；2 个 UE 分配的 PUSCH 的 PRB 对应的起始位置分别为 0,24 (3\*8+0)；

>> 那 0,24 对应的 PHICH GROUP 就分别为  $0 \bmod 3=0, 24 \bmod 3=0$

>> 那 0,24 对应的 PHICH GROUP 内的正交码的索引也就是

$$(0/3) \bmod 8 = 0; (24/3) \bmod 8 = 0;$$

结论：算出来的  $n_{\text{PHICH}}^{\text{group}}$  和  $n_{\text{PHICH}}^{\text{seq}}$  是一样的，所以产生了一个  $n_{\text{DMRS}}$  来进行区分。这完全靠调度来进行区分，算法不可能有办法解决。协议最后把  $n_{\text{DMRS}}$  也用来‘计算 PHICH GROUP’，不知道为什么？？。

## • 最后协议具体公式，参考 36213 的 9.1.2，：

$$n_{\text{PHICH}}^{\text{group}} = (I_{\text{PRB\_RA}}^{\text{lowest\_index}} + n_{\text{DMRS}}) \bmod N_{\text{PHICH}}^{\text{group}} + I_{\text{PHICH}} N_{\text{PHICH}}^{\text{group}}$$

$$n_{\text{PHICH}}^{\text{seq}} = (\left\lfloor I_{\text{PRB\_RA}}^{\text{lowest\_index}} / N_{\text{PHICH}}^{\text{group}} \right\rfloor + n_{\text{DMRS}}) \bmod 2 N_{\text{SF}}^{\text{PHICH}}$$

$$N_{SF}^{PHICH} = \begin{cases} 4 & \text{normal cyclic prefix} \\ 2 & \text{extended cyclic prefix} \end{cases}$$

(1)  $I_{PRB\_RA}^{lowest\_index}$  表示 format 0 分配最低的 PRB 的 index, 所以其实  $I_{PRB\_RA}^{lowest\_index}$  作为一种索引,  $n_{DMRS}$  作为偏移来找到对应的  $n_{PHICH}^{group}$ 。

(2)  $n_{DMRS}$  也是在 format 0 中配置的。但如果是半静态调度 (只有 PDSCH, 没有 PDCCH, 或者 a random access response grant: msg3) 的时候,  $n_{DMRS}$  为 0

(3)  $n_{PHICH}^{seq}$  为什么  $\bmod 2 N_{SF}^{PHICH}$  是因为 ‘正交码’ 正好有  $2 N_{SF}^{PHICH}$  个, 8 个。

(4) 对于  $I_{PHICH}$  为什么会有下面这样的特殊处理, 是因为 ‘只有在 TDD configuration0 的 3/4,8/9 上行子帧 ‘对应到同一个 ‘下行子帧回 ACK/NACK’. 所以才此时可能  $I_{PRB\_RA}^{lowest\_index}$  和  $n_{DMRS}$  是一样的, 所以为了回 ACK/NACK 不冲突, 才产生了这种特殊处理。也就是前面提到的  $mi=2$ , 也就是上面表 36211--- Table 6.9-1 中对应值为 2 的情况, 有  $2 * N_{PHICH}^{group}$  个 group, 前半给 3,8 时隙用, 后半给 4,9 时隙回 ACK/NACK 用。

$$I_{PHICH} = \begin{cases} 1 & \text{for TDD UL/DL configuration 0 with PUSCH transmission in subframe } n = 4 \text{ or } 9 \\ 0 & \text{otherwise} \end{cases}$$

(5)  $n_{PHICH}^{group}$  决定频域的位置,  $n_{PHICH}^{seq}$  决定正交码。

• 时域 (l) : 正常配置下是第一个符号

$$l'_i = \begin{cases} 0 & \text{normal PHICH duration, all subframes} \\ \lfloor m'/2 \rfloor + i + 1 \bmod 2 & \text{extended PHICH duration, MBSFN subframes} \\ \lfloor m'/2 \rfloor + i + 1 \bmod 2 & \text{extended PHICH duration, subframe 1 and 6 in frame structure type 2} \\ i & \text{otherwise} \end{cases}$$

Normal PHICH duration 或者 extended PHICH duration 由配置决定:  
RRC::MasterInformationBlock → PHICH-Config → phich-Duration

• 频域 (k) : 3 个 REG (i=0,1,2) 均匀分布在下行的频带上, 由  $n_{PHICH}^{group}$  决定。

$$\bar{n}_i = \begin{cases} \left( \left\lfloor N_{ID}^{cell} \cdot n_{l'_i} / n_1 \right\rfloor + m' \right) \bmod n_{l'_i} & i = 0 \\ \left( \left\lfloor N_{ID}^{cell} \cdot n_{l'_i} / n_1 \right\rfloor + m' + \left\lfloor n_{l'_i} / 3 \right\rfloor \right) \bmod n_{l'_i} & i = 1 \\ \left( \left\lfloor N_{ID}^{cell} \cdot n_{l'_i} / n_1 \right\rfloor + m' + \left\lfloor 2 n_{l'_i} / 3 \right\rfloor \right) \bmod n_{l'_i} & i = 2 \end{cases}$$

肯定先除去 cell 的参考信号和 PCFICH 占用的资源。l' 表示第几个符号 (正常的 TTI 是 0~13),

$n_{l'}$  表示该符号上对应有多少个除去 PCFICH, cell-RS 之后有多少可用的 REG 个数。

$m' = n_{PHICH}^{group}$  也就是按照 ‘三分之一’ 的间隔均匀分布在整個下行频带上。

#### 4.4.6.7 协议结构体

RRC::MasterInformationBlock→PHICH-Config

#### 4.4.6.8 特殊性

- 对于发射分集如果是 4 天线，对应的矩阵有两个。根据  $(i + n_{\text{PHICH}}^{\text{group}}) \bmod 2 = 0$  还是  $! = 0$  来判断。其实  $i=0,1,2$  代表‘层映射’之后的第几个符号。由于 12symbol，层映射之后每层都是‘3 个符号’。具体参考 36211 的 6.9.2

#### 4.4.6.9 问题

##### 4.4.6.9.1 问题1：为什么编码要重复3次？

个人认为重复 3 次只是一种恰当的思想，重复 3 次，如果有一个出错可以纠正；重复 5 次也可以，但这样会浪费资源。也是一种折中的想法。

##### 4.4.6.9.2 问题2：要是实际使用到的 PHICH 资源并没有占到 $m_i \cdot N_{\text{PHICH}}^{\text{group}}$ 组，没用到的是否能被 PDCCH 使用？

显然不能。因为 UE 根本就不可能知道哪些没有用。所以即使没有用，也只能空着。所以“MasterInformationBlock→PHICH-Config→phich-Resource”的配置也会影响到‘PDCCH’的能使用的资源情况。

##### 4.4.6.9.3 问题3：为什么要有 $n_{\text{DMRS}}$ ？

PHICH GROUP 配置得比较少（phich-Resource 值为 oneSixth,或者 half 的情况下），有可能‘不同的 PUSCH 的 PRB 对应的起始位置算出来的  $n_{\text{PHICH}}^{\text{group}}$  和  $n_{\text{PHICH}}^{\text{seq}}$  是一样的’。所以就产生了  $n_{\text{DMRS}}$ ， $n_{\text{DMRS}}$  为什么为 3bit，也是因为一个 PHICH GROUP 能容纳 8 个 PHICH 正交码，对应 8 个 UE 的 ACK/NACK 回应。

##### 4.4.6.9.4 问题4：为什么需要配置 PHICH 的资源数

RRC::MasterInformationBlock→PHICH-Config→phich-Resource？

因为每个厂家的处理一个 TTI 能调度的 UE 数目不一样，所以一个下行 TTI 需要回应的 UE 的数目不一样。所以需要配置。

##### 4.4.6.9.5 问题5：为什么需要配置 PHICH 的配置要放在 MIB 当中，不放在 SIB 当中？

因为 PDCCH 的解码对应的需要除去 PCFICH 和 PHICH 占用的资源数之后才能算出 CCE 的个数。所以如果 PHICH 放在 SIB 里面，PDCCH 根本解不对。

#### 4.4.6.9.6 问题6：协议为什么最后把 $n_{DMRS}$ 也用来‘计算PHICH GROUP’??

不知道为什么??只能说‘ $n_{DMRS}$ 用来计算PHICH GROUP’只能说把冲突的解决更加离散化了，不仅从PHICH不同正交码的角度进行了解决，而且还从‘更改回应UE的PHICH GROUP进行解决’。但冲突本身发生的概率不变。

例如：110RB，但是只有3个PHICH GROUP,总共就只有24个选择，100/24的向上取整为5.所以冲突是肯定的，不管怎么算法都没有用，所以只能靠调度自己去规避冲突。

#### 4.4.6.9.7 问题7：按理应该是算成每个TTI能调度的UE个数来算是最精确的，可以通过它进行配置。协议最后为什么没采取这种方式呢??

我也不知道，个人觉得这样显然能节省PHICH资源，给PDCCH更多的资源；当然不利条件就是PHICH GROUP和PHICH SEQ算法就得修改了。没采取这种方式也许是怕两个邻小区是不同厂家，配置不一样，产生干扰吗??

#### 4.4.6.9.8 问题8：为什么算频域的对应哪个REG的时候，要除以 $n_1$ 呢，而不是 $n_0$ 呢??

$$\overline{n}_i = \begin{cases} \left( \left\lfloor N_{ID}^{cell} \cdot n_{l_i'} / n_1 \right\rfloor + m' \right) \bmod n_{l_i'} & i = 0 \\ \left( \left\lfloor N_{ID}^{cell} \cdot n_{l_i'} / n_1 \right\rfloor + m' + \left\lfloor n_{l_i'} / 3 \right\rfloor \right) \bmod n_{l_i'} & i = 1 \\ \left( \left\lfloor N_{ID}^{cell} \cdot n_{l_i'} / n_1 \right\rfloor + m' + \left\lfloor 2 n_{l_i'} / 3 \right\rfloor \right) \bmod n_{l_i'} & i = 2 \end{cases}$$

### 4.4.7 PDCCH(Physical Downlink Control Channel)

#### 4.4.7.1 作用

- 指示对应UE的资源在哪里。下行资源调度分配，上行准入资源分配，HARQ info，功控信息

#### 4.4.7.2 参看

- 36212的5.3.3, 36211的6.8, 36213的7.1; 36213的9.1.1

### 4.4.7.3 思想来源

- 可以通过信令通知给 UE 它占用哪些资源，一直为它保留。这是 3G 的做法，显然这种方法对资源利用率不高，不能再选了。
- 那就可以时时的控制信息通知给 UE，而周期后面订成了 1ms (1TTI)，不订成 0.5ms (1TS) 显然是因为这种情况下，留给 PDSCH 的资源不多了----也是一种折中了，其实也可以规定 2ms 作为一次调度单位时间，不过后面的算法就得全改了。此时 UE 只有唯一的一个关键字  $r_{nti}$ ，可以（潜规则）事先规定好‘一片区域让所有的 UE 用自己的  $r_{nti}$  去盲检测’，但这片区域很难规定，‘太大了 UE 盲检测的数目会多，而太小了又不适合一次调度太多的 UE’。
- 最后还是规定了一片区域，这就是 PDCCH，规定了一个 PDCCH(一次针对 UE 的控制信息的原子分配单位为一个 CCE)。但是为了解决上面的问题. UE 要用 UE 自己的  $r_{nti}$  去算出自己的‘盲检测区域’，按照某种格式来解，预先规定的格式也就叫做 DCI(downlink control information)。
- 但不管算法怎么好，调度的 UE 个数，肯定会大于‘PDCCH 占用的 CCE 数’的个数，就像在 PHICH 中举的例子一样。但此时 UE 还没有获得任何专有的信息，所以不能像 PHICH 那样给个  $n_{DMRS}$  来解决冲突。最后，就通过两种手段来解决

(1) 盲检测的 CCE 数目比实际传送的数据要多。这样虽然两个  $r_{nti}$  算出来 CCE 的起始位置一样，但由于给 UE 的 CCE 数目比较多，可以给两个  $r_{nti}$  分配‘在盲检测区域内，但不同的 CCE’来解决冲突。

例如：2 个 UE 的  $r_{nti}$  算出来的 CCE 的起始位置都是 12. 但它们每个实际占用的 CCE 的个数为 2；但盲检测的范围却是 6.

所以，可以把 12,13 这两个 CCE 分给第一个 UE; 14,15 分给第二个 UE。

(2) 把‘子帧号 (0~9)’参与计算‘盲检测’，这样就在某个时间点‘算 CCE 起始位置冲突的两个 RNTI’再下一个时间点就‘不要冲突’了。否则一直冲突下去的话，极端情况有个 RNTI 一直会调度不到。

- 最后，考虑到  $e_{nodeb}$  和 UE 硬件及处理效率的问题。为了让 UE 盲检测的数目少，协议引进了‘传输模式的概念 (TM)’；让 UE 在某种信道条件下，只处于‘某一种传输模式’，在这种‘传输模式’下只有‘两种可能的 DCI 格式需要盲检测。显然：‘ $e_{nodeb}$  和 UE 硬件及处理效率的问题’是‘TM’的基石，当‘ $e_{nodeb}$  和 UE 硬件及处理效率的提高的时候’，也许‘TM’就不需要了，UE 能很快的盲检测所有的 DCI 格式。
- LTE 最后还保留些 3G 的一些固定分配的影子：那就是半静态调度。

### 4.4.7.4 UE 传输模式 --- 规定

- 配置：RRC::radioResourceConfigDedicated-->PhysicalConfigDedicated-->AntennaInfoDedicated-->transmissionMode

36213-----Table 7.1-5: PDCCH and PDSCH configured by C-RNTI

Transmission mode	DCI format	Search Space	Transmission scheme of PDSCH corresponding to PDCCH
Mode 1	DCI format 1A	Common and UE specific by C-RNTI	Single-antenna port, port 0 (see subclause 7.1.1)
	DCI format 1	UE specific by C-RNTI	Single-antenna port, port 0 (see subclause 7.1.1)
Mode 2	DCI format 1A	Common and UE specific by C-RNTI	Transmit diversity (see subclause 7.1.2)
	DCI format 1	UE specific by C-RNTI	Transmit diversity (see subclause 7.1.2)
Mode 3 (CDD的空分)	DCI format 1A	Common and UE specific by C-RNTI	Transmit diversity (see subclause 7.1.2)
	DCI format 2A	UE specific by C-RNTI	Large delay CDD (see subclause 7.1.3) or Transmit diversity (see subclause 7.1.2)
Mode 4 (close-loop的空分)	DCI format 1A	Common and UE specific by C-RNTI	Transmit diversity (see subclause 7.1.2)
	DCI format 2	UE specific by C-RNTI	Closed-loop spatial multiplexing (see subclause 7.1.4) or Transmit diversity (see subclause 7.1.2)
Mode 5	DCI format 1A	Common and UE specific by C-RNTI	Transmit diversity (see subclause 7.1.2)
	DCI format 1D	UE specific by C-RNTI	Multi-user MIMO (see subclause 7.1.5)
Mode 6 (基于码本的波束赋型)	DCI format 1A	Common and UE specific by C-RNTI	Transmit diversity (see subclause 7.1.2)
	DCI format 1B	UE specific by C-RNTI	Closed-loop spatial multiplexing (see subclause 7.1.4) using a single transmission layer
Mode 7 (非基本码本的波束赋型)	DCI format 1A	Common and UE specific by C-RNTI	If the number of PBCH antenna ports is one, Single-antenna port, port 0 is used (see subclause 7.1.1), otherwise Transmit diversity (see subclause 7.1.2)
	DCI format 1	UE specific by C-RNTI	Single-antenna port; port 5 (see subclause 7.1.1)

- 从前面的介绍和图，也能大致了解到传输模式 TM 的变化过程。

(1) 如果单 port，肯定是 TM1。

(2) 如果多 port，不知道信号的好坏，开始应该驻留在 TM2，后面根据 UE 的能力和上报的 CQI 条件变好了（也许还加上回应的 ACK/NACK 的情况），就变化到 TM3 或者 TM4（理论上 TM4 上报 PMI，应该更好的反应信道条件，它的传输效率比 TM3 更高些），然后信道条件变差了，又回到 TM6 或者 TM2；CQI 实在太差了，经常回 NACK，就变化到 TM7，这也是为什么 UE 在小区边缘会迁到 TM7 的原因，信道条件太差了(原因前面已经解释过)。如此这般循环往复。所以：‘信道条件’是所有 TM 变换的根本。

注：TM5 不太理解，暂时不讨论。

#### 4.4.7.4.1 问题



- 问题 1：同样的频点位置，同样的 RB 数的情况下。各种 TM 模式的吞吐量关系是什么？是不是由于 TM7 有 UE-RS，TM2 的吞吐量永远比 TM7 好呢？

答：同样的频点位置，同样的 RB 数的情况下，根据前面的介绍，由于 turbo 编码都是一样的，所以觉得吞吐量的因素有三个：速率匹配的重复率（由上报的 CQI 决定），调制方式  $Q_m$ （由上报的 CQI 决定），层数（由上报的 RI 决定）。

所以

（1）TM3/TM4 是比其他 TM 模式好的。TM3 和 TM4 就很难说了，理论上由于 TM4 可以根据 UE 根据实际情况上报的 PMI 进行‘层---port’之间的映射，TM4 应该比 TM3 要好些。

（2）TM2 是一般的发射分集；TM6 是‘基于码字的波束赋型’；它们都只有 CELL-RS，所以一般 TM6 会比 TM2 好。TM7 是‘非基于码本的波束赋型’，虽然 TM7 有 UE-RS，会占用一些资源，但是当‘信道条件变差到一定程度---也就是 CELL-RS 已经不能进行很好的相干解调的时候’，TM7 由于有 UE-RS，它们的距离比 CELL-RS 更近，所以相干解调成功率也会比 TM2/TM6 好，也就体现在‘上报的 CQI’更好，这样会“减小速率匹配的重复率和提高调制方式，会弥补 UE-RS 在占用一些资源的缺陷”。当然信道条件再坏下去，最后只能是掉线了。

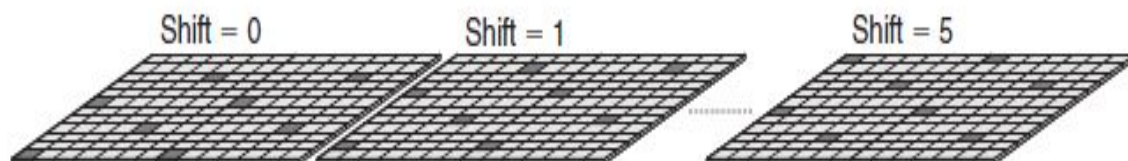
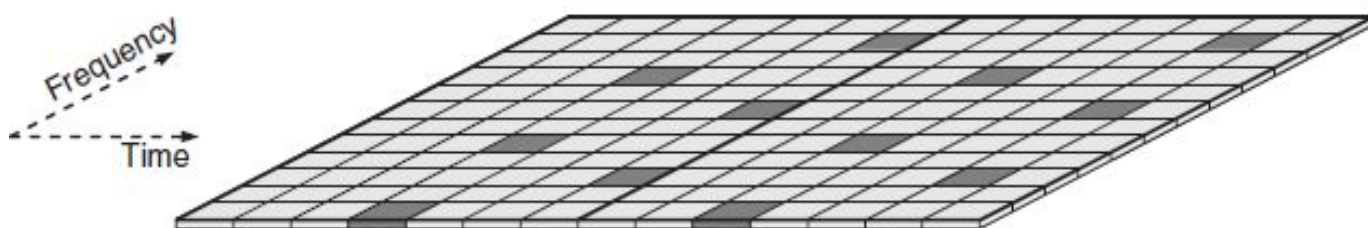


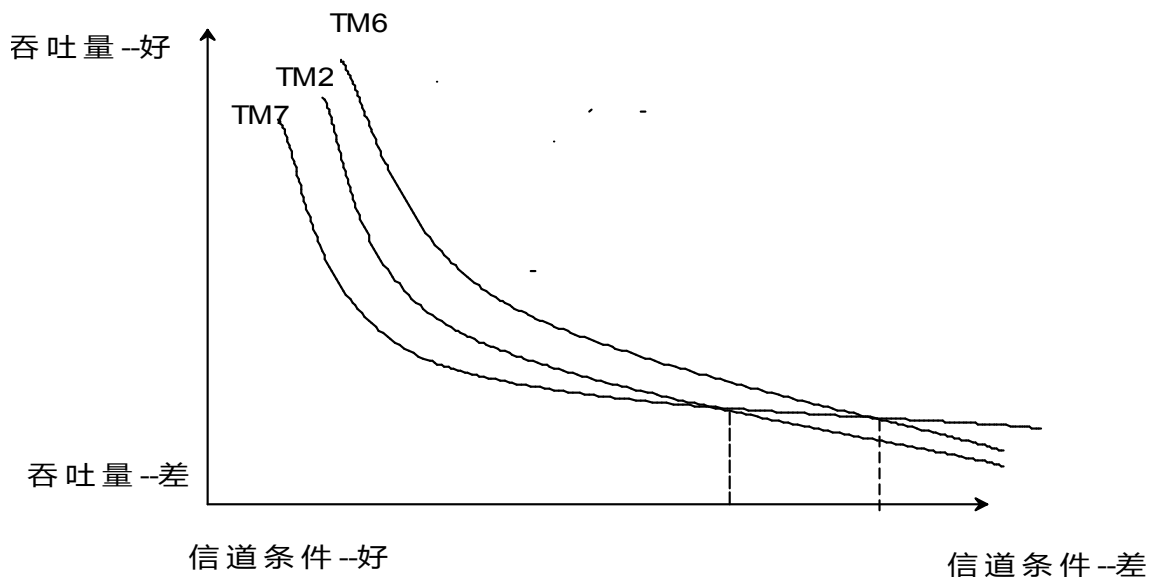
Figure 16.9 Different reference-signal frequency shifts.

\*\*\* CELL-RS



\*\*\* UE-RS

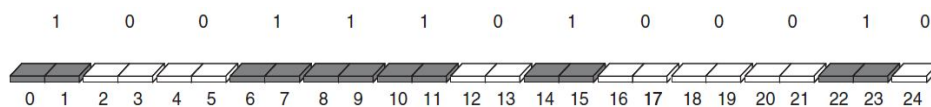
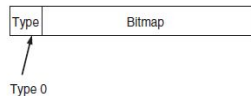
总结：TM2/TM6/TM7 的吞吐量的关系就是，如下图



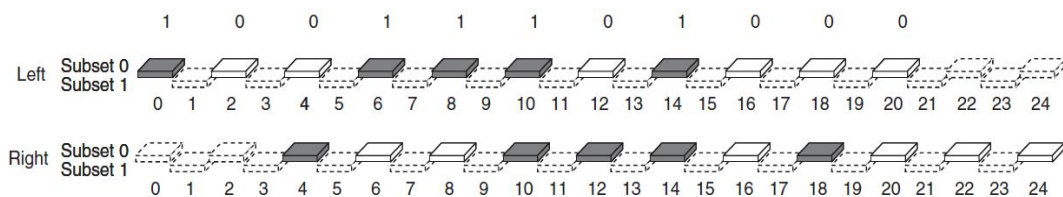
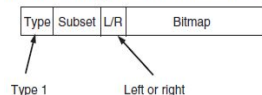
#### 4.4.7.5 资源分配的格式

- 配置: DCI 模式里面规定

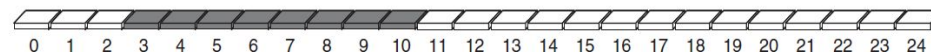
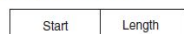
Resource allocation type 0  
downlink only, DCI formats 1 and 2



Resource allocation type 1  
downlink only, DCI formats 1 and 2



Resource allocation type 2  
downlink, DCI formats 1A and 1B  
uplink, DCI format 0

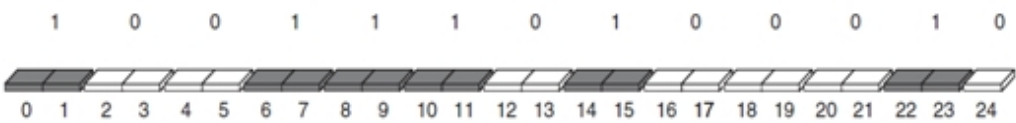
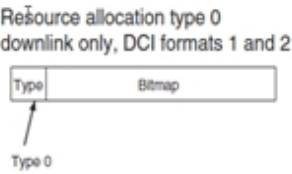


注: 由于上行分配 RB 必须连续, 所以只能选择资源分配 2 的方式。

##### 4.4.7.5.1 TYPE0 --- RB 分配不连续

- 参看: 36213 的 7.1.6.1
- 思想: 1bit 代表多个 RB (P 个), 具体是根据下行带宽决定的。分配的 RB 不连续。





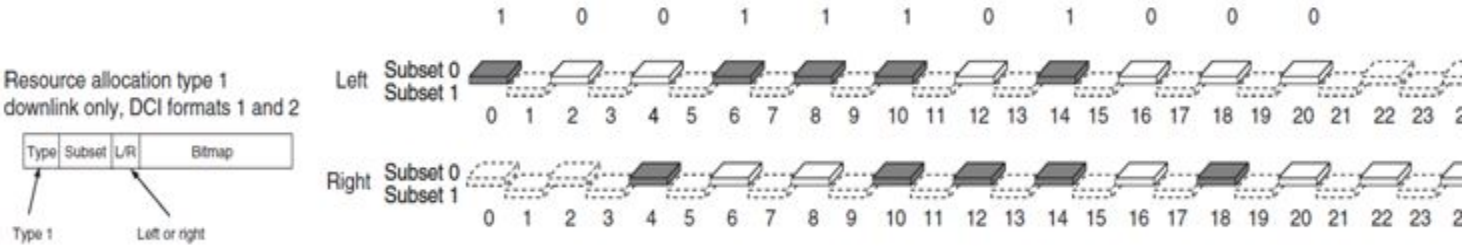
36213---- Table 7.1.6.1-1: Type 0 Resource Allocation RBG Size vs. Downlink System Bandwidth

System Bandwidth $N_{RB}^{DL}$	RBG Size ( $P$ )
$\leq 10$	1
11 – 26	2
27 – 63	3
64 – 110	4

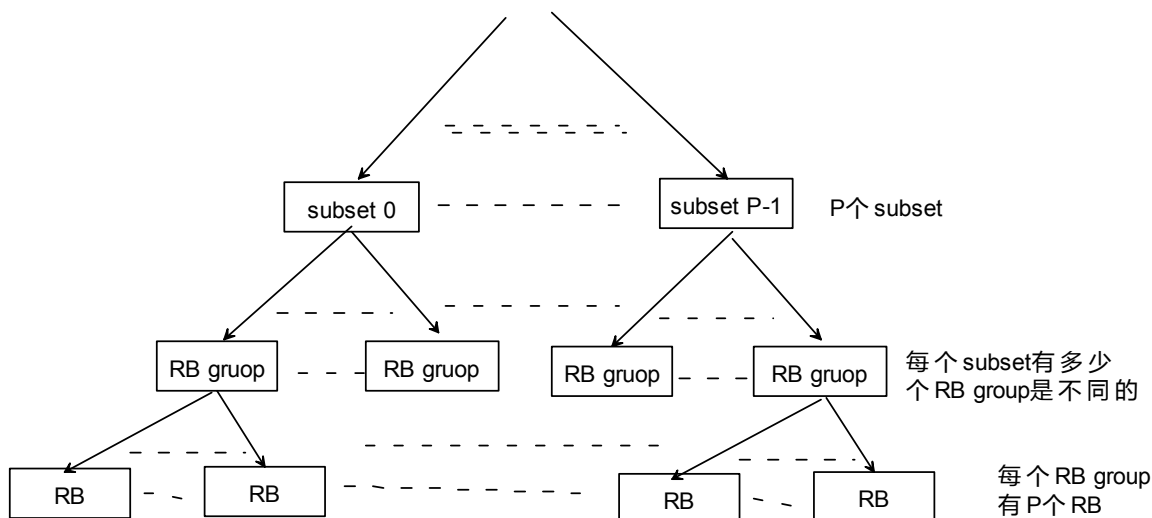
- 所以总共需要的 bit 数是  $N_{RBG} = \lceil N_{RB}^{DL} / P \rceil$ 。如果不能整除（ $N_{RB}^{DL} \bmod P > 0$ ），前面每个 bit 表示的 RB 数为  $P$ , 最后一个 bit 表示的 RB 数是  $N_{RB}^{DL} - P \cdot \lfloor N_{RB}^{DL} / P \rfloor$

#### 4.4.7.5.2 TYPE1 --- RB 分配不连续

- 参看：36213 的 7.1.6.2。
- Subset 思想：**下行带宽又均匀分成  $P$  的 subset，每个 subset 又由多个 'RB group' 组成，而一个 'RB group' 又由  $P$  个 RB 组成，也许最后一个 'RB group' 没有  $P$  个 RB。分配的 RB 不连续。（ $P$  跟 TYPE0 的  $P$  是一样的）。



$$N_{RB}^{DL}$$



- **分配思想：**简单点说就是‘以 RBG(RB group)为基本分配单位’，‘把 RBG 按顺序先按照 subset 分，等每个 subset 都分到之后再给从头来给每个 subset 在分配 RBG’的方式，协议的算法也就是这个意思。

举个例子：

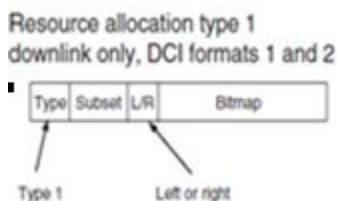
假设  $N_{RB}^{DL}=30$ ，那么  $P=3$ ，就会有 3 个 subset，RB group 就是 (0,1,2)，(3,4,5)，(6,7,8)，(9,10,11) ..... 。先按照 subset 分，把 (0,1,2) 这个 REG 分给 subset0，然后把(3,4,5)分给 subset1，(6,7,8)分给 subset2；每个 subset 都分到之后再重新来，把 (9,10,11) 分给 subset0，这样一直延续下去，最后的结果就是。

Subset0: (0,1,2) , (9,10,11) , (18,19,20) ,(27,28,29)

Subset1: (3,4,5), (12,13,14), (21,22,23)

Subset2: (6,7,8), (15,16,17), (24,25,26)

所以决定哪个 subset 的 bit 数是： $\lceil \log_2(P) \rceil$  ,也就是分配图中的 subset 的 bit 数；总的 RBG 数是：  $N_{RBG} = \lceil N_{RB}^{DL} / P \rceil$  。

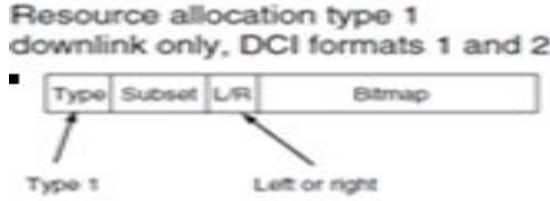


同时，你也就会明白为什么协议公式里面会除以  $P^2$  了，应该有  $P$  个 subset，每个 RBG 都有  $P$  个 RB，所以每个 subset 至少能分到  $\lceil N_{RB}^{DL} / P^2 \rceil$  个 RB。

(1) 从上面就看出来由于每个 subset 对应的 RB 数不一样，所以就**规定了**

‘只用较少的 bit 数来表示 RB，也就是  $N_{RB}^{TYPE1} = \lceil N_{RB}^{DL} / P \rceil - \lceil \log_2(P) \rceil - 1$ ’，也就是

分配图中的 bitmap 的 bit 数。其实是可以按照最小的那个也就是最后一个 subset 的 RB 数来作为 bitmap 数的，不知道为什么协议没有这么算??



(2) 但这样有些 RB 又分配不了，怎么办呢？于是又想出一种方法：让一个标志来表示 ‘bitmap 对应的 RB 是从某个 subset 头开始数呢(向前对齐)，还是从中间某一个开始数（向后对齐）’，这个标志也就是也就是上面分配图中的 bitmap 的 L/R，只占 1bit，那后一种情况（向后对齐）究竟从第几个开始数呢？显然就是  $\Delta_{\text{shift}}(p) = N_{\text{RB}}^{\text{RBGsubset}}(p) - N_{\text{RB}}^{\text{TYPE1}}$

• 协议的算法看起来太复杂，我们可以用另外一种思路来理解：

设  $N_{\text{RB}}^{\text{DL}} - 1 = a * P^2 + b * P + c$  来计算就得到，根据分配思想：能得到每个 subset 能分配到 a 个完整的 RB group（根据  $a * P^2$ ）；前面 b 个 subset 每个能再多分配到 1 个完整的 RB group（根据  $b * P$ ）；而第 b+1 个 subset 也就能分配到最后一个 RB group 可能完整也可能不完整（RB 数为 c+1）。根据这些你也能很好的理解协议中的公式的意义了。

也能得到  $N_{\text{RB}}^{\text{TYPE1}} = aP + b - \lceil \log_2(P) \rceil$ ，也可以得到  $\left\lfloor \frac{N_{\text{RB}}^{\text{DL}} - 1}{P} \right\rfloor \bmod P = b$ 。

每个 RB subset 含有多少个用  $N_{\text{RB}}^{\text{RBGsubset}}(p)$  表示。

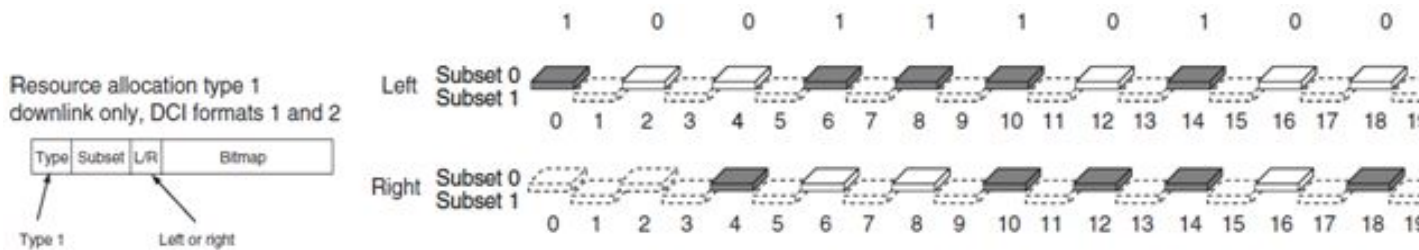
$$N_{\text{RB}}^{\text{RBGsubset}}(p) = \begin{cases} \left\lfloor \frac{N_{\text{RB}}^{\text{DL}} - 1}{P^2} \right\rfloor \cdot P + P = (aP + P) & , p < \left\lfloor \frac{N_{\text{RB}}^{\text{DL}} - 1}{P} \right\rfloor \bmod P \text{ (有 } b \text{ 个, } \Delta_{\text{shift}}(p) = P - b + \lceil \log_2(P) \rceil) \\ \left\lfloor \frac{N_{\text{RB}}^{\text{DL}} - 1}{P^2} \right\rfloor \cdot P + (N_{\text{RB}}^{\text{DL}} - 1) \bmod P + 1 = (aP + c + 1) & , p = \left\lfloor \frac{N_{\text{RB}}^{\text{DL}} - 1}{P} \right\rfloor \bmod P \text{ (有 } 1 \text{ 个, } \Delta_{\text{shift}}(p) = c + 1 - b + \lceil \log_2(P) \rceil) \\ \left\lfloor \frac{N_{\text{RB}}^{\text{DL}} - 1}{P^2} \right\rfloor \cdot P = (aP) & , p > \left\lfloor \frac{N_{\text{RB}}^{\text{DL}} - 1}{P} \right\rfloor \bmod P \text{ (有 } P - b - 1 \text{ 个, } \Delta_{\text{shift}}(p) = -b + \lceil \log_2(P) \rceil) \end{cases}$$

可以算一下，正好算出来所有的 RB 个数最后是  $N_{\text{RB}}^{\text{DL}}$

举例：

例如：  $N_{\text{RB}}^{\text{DL}} = 30$ ，那么  $P = 3$ ，推断出  $a = 3, b = 0, c = 2$ 。

多少 bit 来表示 RB 数  $N_{\text{RB}}^{\text{TYPE1}} = aP + b - \lceil \log_2(P) \rceil = 3 * 3 + 0 - 2 = 7$



Subset0: (0,1,2), (9,10,11), (18,19,20), (27,28,29)。  $\Delta_{shift}(0) = 5$

Subset1: (3,4,5), (12,13,14), (21,22,23)。  $\Delta_{shift}(1) = 2$

Subset2: (6,7,8), (15,16,17), (24,25,26)。  $\Delta_{shift}(2) = 2$

所以  $N_{RB}^{TYPE1} = 7$  的 BIT 位对应的 RB 序号就是。

$n_{PRB}^{RBsubset}(0)$  没有 shift = 0, 1, 2, 9, 10, 11, 18

$n_{PRB}^{RBsubset}(0)$  有 shift = 11, 18, 19, 20, 27, 28, 29

$n_{PRB}^{RBsubset}(1)$  没有 shift = 3, 4, 5, 12, 13, 14, 21

$n_{PRB}^{RBsubset}(1)$  有 shift = 5, 12, 13, 14, 21, 22, 23

$n_{PRB}^{RBsubset}(2)$  没有 shift = 6, 7, 8, 15, 16, 17, 24

$n_{PRB}^{RBsubset}(2)$  有 shift = 8, 15, 16, 17, 24, 25, 26

注：最后理解得到也是一种折中的思想，RB 的位置既不能完全离散，也不能完全连续。翻译过来也就是：一个 subset 中的 RB group 之间的 RB 是不连续的，但是一个 RB group 中的 RB 是连续的。

- 最后计算的时候对应的 PRB 公式也是表达了这种思想：

$$n_{PRB}^{RBsubset}(p) = \left\lfloor \frac{i + \Delta_{shift}(p)}{P} \right\rfloor P^2 + p \cdot P + (i + \Delta_{shift}(p)) \bmod P$$

#### 4.4.7.5.2.1 问题

- 问题 1：其实是可以按照最小的那个也就是最后一个 subset 的 RB 数来作为 bitmap 数的，不知道为什么协议没有这么做？

#### 4.4.7.5.3 TYPE2 --- RB 分配连续，上下行都可用

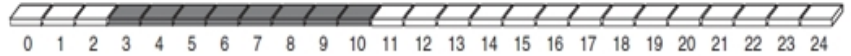
- 参看：36213 的 7.1.6.3
- 思想：从哪个 RB 开始，后面的多少个 RB 都归我。按理是需要 bit 数是：

表示哪个 RB 开始  $RB_{start}$  需要  $\lceil \log_2(N_{RB}) \rceil$  bit 表示，多少个 RB 是  $L_{CRBs}$  需要  $\lceil \log_2(N_{RB}) \rceil$  bit，合起来也就是  $2 * \lceil \log_2(N_{RB}) \rceil$ 。但是为了节省资源，从理

论上某个 RB 开始，分配的 RB 个数  $L_{CRBs}$  不可能超过  $N_{RB} - RB_{start}$ 。所以采取 n 进制（ $N_{RB}^{DL}$  决定）的方法，其实最后合起来的 bit 数不会超过  $\lceil \log_2(N_{RB}^{DL}(N_{RB}^{DL} + 1)/2) \rceil$ 。

Resource allocation type 2  
downlink, DCI formats 1A and 1B  
uplink, DCI format 0

Start	Length
-------	--------



>>>>何为 n 进制，觉个例子：如果 5 进制的符号为 A,B,C,D,E。那么十进制对应的 6 用 5 进制表示就是 BA.

• 协议公式就是：

if  $(L_{CRBs} - 1) \leq \lfloor N_{RB}^{DL} / 2 \rfloor$  then

$$RIV = N_{RB}^{DL} (L_{CRBs} - 1) + RB_{start}$$

else

$$RIV = N_{RB}^{DL} (N_{RB}^{DL} - L_{CRBs} + 1) + (N_{RB}^{DL} - 1 - RB_{start})$$

where  $L_{CRBs} \geq 1$  and shall not exceed  $N_{VRB}^{DL} - RB_{start}$ .

注：（1）显然通过 RIV 给  $N_{RB}^{DL}$  取模就能得到对应的  $RB_{start}$  和  $L_{CRBs}$ 。

（2）协议为了更加节省资源，对于  $(L_{CRBs} - 1) > \lfloor N_{RB}^{DL} / 2 \rfloor$  还进行了特殊处理。这样最后算出来的 RIV 不论在什么情况，都不可能大于  $N_{RB}^{DL} * \lfloor N_{RB}^{DL} / 2 \rfloor + \lfloor N_{RB}^{DL} / 2 \rfloor$ 。

从这些地方我们也看出无限资源的宝贵，协议真实不遗余力的去节省资源。

• 资源分配占用 bit 数为：  $1 (\text{loc/dis}) + \lceil \log_2 (N_{RB}^{DL} (N_{RB}^{DL} + 1) / 2) \rceil$  （对于 DCI1A,DCI1B,DCI1D），  $1 (\text{loc/dis}) + \lceil \log_2 (N_{RB}^{DL} / N_{RB}^{step} (N_{RB}^{DL} / N_{RB}^{step} + 1) / 2) \rceil$  \*\*\*协议最后（对于 DCI1C）进行了特殊处理。1bit 代表  $N_{RB}^{step}$  个 RB。为什么用这种算法，应该是因为 DCI1C 只是给公共信息用，如果对应分配资源的跨度比较大，这样可以节约 bit 数。

注意：该资源分配方式给 DCI1A,1B,1D 用时。1bit 表示 1RB 给 format1C 用的时候，1bit 不是 1RB，而是由下行带宽决定。

36213 --- Table 7.1.6.3-1:  $N_{RB}^{step}$  values vs. Downlink System Bandwidth

System BW ( $N_{RB}^{DL}$ )	$N_{RB}^{step}$
	DCI format 1C
6-49	2
50-110	4

• 怎么区分  $(L_{CRBs} - 1) \leq \lfloor N_{RB}^{DL} / 2 \rfloor$  还是  $(L_{CRBs} - 1) > \lfloor N_{RB}^{DL} / 2 \rfloor$  呢？？

>>>> 如果  $(L_{CRBs} - 1) \leq \lfloor N_{RB}^{DL} / 2 \rfloor$ ，则  $(RIV / N_{RB}^{DL}) + (RIV \bmod N_{RB}^{DL}) = L_{CRBs} - 1 + RB_{start} < N_{RB}^{DL}$ ；

>>>> 如果  $(L_{CRBs} - 1) > \lfloor N_{RB}^{DL} / 2 \rfloor$  , 则  $(RIV / N_{RB}^{DL}) + (RIV \bmod N_{RB}^{DL}) = (N_{RB}^{DL} - L_{CRBs} + 1) + (N_{RB}^{DL} - 1 - RB_{start}) = 2 * N_{RB}^{DL} - (L_{CRBs} + RB_{start}) > N_{RB}^{DL}$  ;

结论：所以靠  $(RIV / N_{RB}^{DL}) + (RIV \bmod N_{RB}^{DL})$  的值就能区分这两种情况了。

- 那会不会出现分配两种不同的 RB 的时候  $((L_{CRBs} - 1) \leq \lfloor N_{RB}^{DL} / 2 \rfloor$  和  $(L_{CRBs} - 1) > \lfloor N_{RB}^{DL} / 2 \rfloor$ ) , 导致的 RIV 值一样呢？

答：不会。假设  $N_{RB}^{DL} (L_{CRBs}^1 - 1) = N_{RB}^{DL} (N_{RB}^{DL} - L_{CRBs}^2 + 1)$  , 也就是

$L_{CRBs}^2 = N_{RB}^{DL} - L_{CRBs}^1 + 2$  。对应的  $RB_{start}^1$  范围就是从 0 到  $N_{RB}^{DL} - L_{CRBs}^1$  ; 而对应的  $RB_{start}^2$  的范围也是 0 到  $N_{RB}^{DL} - L_{CRBs}^2$  , 所以  $N_{RB}^{DL} - 1 - RB_{start}^2$  的范围也就是  $(N_{RB}^{DL} - 1, L_{CRBs}^2 - 1)$  , 也就是  $(N_{RB}^{DL} - 1, N_{RB}^{DL} - L_{CRBs}^1 + 1)$  , 所以不会一样。

也以此得到结论就是：当  $(L_{CRBs} - 1) \leq \lfloor N_{RB}^{DL} / 2 \rfloor$  的时候，RIV 的值随着  $RB_{start}$  的增加而增加，相当于从 0 开始往前数；而当  $(L_{CRBs} - 1) > \lfloor N_{RB}^{DL} / 2 \rfloor$  的时候，RIV 的值随着  $RB_{start}$  的增加而减少，相当于从  $N_{RB}^{DL} - 1$  开始往后数。

#### 4.4.7.5.3.1 问题

- 问题 1：为什么 DCI1C 的资源分配方式要特殊化出来呢？

### 4.4.7.6 下行资源分配跳频（跳-变化）模式

#### 4.4.7.6.1 下行 PDSCH 资源的跳频

- 参看：36211 的 6.2.3.2 节
- 设计思想：横放列读。就是配置 VRB 的是多少，然后放入一个矩阵当中，按照列的顺序去数，数到第几个对应的值为配置的 VRB, PRB 的值就是几。举个例子：

0	1	2	3
4	5	6	7
8	9	10	11
12	*	13	*
14	*	15	*
16	*	18	*

如果配置的 VRB 为 14，那实际对应的 PRB，按照列的顺序数：0,4,8,12,14。正好是第 5 个，因为从 0 开始计数，所以对应的 PRB=4。VRB=13 对应的 PRB=12。

注：数的话，\*不能算在内，为什么有\*，代表什么意思后面再讲。

- 协议规定：

(1) 有两种跳频方式 gap1, gap2.  $6 \leq N_{RB}^{DL} \leq 49$ , 只能为 gap1;  $50 \leq N_{RB}^{DL} \leq 110$ , 能通过配置知道是 gap1 或者 gap2.

(2) 而且能配置的 VRB 并不是和下行带宽一样, 下行带宽从  $0 \sim N_{RB}^{DL} - 1$ ; 而是 VRB 从  $0 \sim N_{VRB}^{DL} - 1$ , 对应的 PRB 也就是从  $0 \sim N_{VRB}^{DL} - 1$ 。  $N_{VRB}^{DL}$  由下列公式决定:

$$N_{VRB}^{DL} = N_{VRB, gap1}^{DL} = 2 \cdot \min(N_{gap}, N_{RB}^{DL} - N_{gap}) \quad \text{for} \quad N_{gap} = N_{gap,1}$$

$$N_{VRB}^{DL} = N_{VRB, gap2}^{DL} = \left\lfloor N_{RB}^{DL} / 2N_{gap} \right\rfloor \cdot 2N_{gap} \quad \text{for} \quad N_{gap} = N_{gap,2}$$

**36211-----Table 6.2.3.2-1: RB gap values.**

System BW ( $N_{RB}^{DL}$ )	Gap ( $N_{gap}$ )	
	1st Gap ( $N_{gap,1}$ )	2nd Gap ( $N_{gap,2}$ )
6-10	$\left\lceil N_{RB}^{DL} / 2 \right\rceil$	N/A
11	4	N/A
12-19	8	N/A
20-26	12	N/A
27-44	18	N/A
45-49	27	N/A
50-63	27	9
64-79	32	16
80-110	48	16

$N_{VRB}^{DL}$  其实就是能够跳频的范围。从公式可以看出--设计的思想:

(1) 对于 gap1, 就是在整个  $N_{VRB}^{DL}$  里面跳频, 也就是  $\tilde{N}_{VRB}^{DL} = N_{VRB}^{DL}$  for  $N_{gap} = N_{gap,1}$ ;

(2) 而 gap2, 就是把  $N_{RB}^{DL}$  按照  $2N_{gap}$  等分, 也就是  $\tilde{N}_{VRB}^{DL} = 2N_{gap}$  for  $N_{gap} = N_{gap,2}$ , 每一等分的 VRB 在自己的等分里面进行跳频。最后一个‘小于  $2N_{gap}$ ’个 RB 的等分不能进行‘跳频 distribute’的配置。

举个 gap2 的例子:

假设:  $N_{RB}^{DL} = 50$ , 配置 gap2。

计算得到:  $N_{gap} = 9$ ; 共  $\left\lfloor N_{RB}^{DL} / 2N_{gap} \right\rfloor = 2$  个等分; 每个等分  $\tilde{N}_{VRB}^{DL} = 2N_{gap} = 18$  个 RB。也就是配置的 0~17 的 VRB, 最后算出来对应的 PRB 肯定还是在 0~17 之间; 18~35 算出来的 PRB 也是在 18~35 之间。而 36~49 是不能配置为 VRB 的, 因为算法会超出  $N_{RB}^{DL}$  的范围, 但这也从一定程度上反映了 VRB 算法的不太完整。

#### • 算法: ----- 行放列读

(1) 矩阵的列数是固定的 4 列; 行数由  $N_{row} = \left\lceil \tilde{N}_{VRB}^{DL} / (4P) \right\rceil \cdot P$  决定。前面提到了, P 是一个 RBG 的大小, 从前面资源分配的角度来看一个 RBG 必须在一起, 也就是在‘VRB 的跳频范围内’。所以这里用了 P, 又有 4 列, 所以公式里面有 4。



**36213-----Table 7.1.6.1-1: Type 0 Resource Allocation RBG Size vs. Downlink System Bandwidth**

System Bandwidth $N_{RB}^{DL}$	RBG Size (P)
$\leq 10$	1
11 – 26	2
27 – 63	3
64 – 110	4

(2) 有可能矩阵能容纳的数目  $4 * N_{row}$  大于一个等分的 RB 数目  $\tilde{N}_{VRB}^{DL}$ ，怎么办呢？协议规定如果多出来的话，就在第 2,4 列的最后插入空的数据（也就是前面举例矩阵里面的\*），显然插入多少个  $4 * N_{row} - \tilde{N}_{VRB}^{DL}$ ，也就是

$N_{null} = 4N_{row} - \tilde{N}_{VRB}^{DL}$ ，第 2,4 列分别插入  $N_{null} / 2$  个空的数据\*。所以，前面的例子里面有\*号。看看公式：-- 最后协议偶数时隙与奇数时隙之间又进行了一定的偏移。

偶数时隙  $n_s$ ：

$$\tilde{n}_{PRB}(n_s) = \begin{cases} \tilde{n}'_{PRB} - N_{row} & , N_{null} \neq 0 \text{ and } \tilde{n}_{VRB} \geq \tilde{N}_{VRB}^{DL} - N_{null} \text{ and } \tilde{n}_{VRB} \bmod 2 = 1 \\ \tilde{n}'_{PRB} - N_{row} + N_{null} / 2 & , N_{null} \neq 0 \text{ and } \tilde{n}_{VRB} \geq \tilde{N}_{VRB}^{DL} - N_{null} \text{ and } \tilde{n}_{VRB} \bmod 2 = 0 \\ \tilde{n}''_{PRB} - N_{null} / 2 & , N_{null} \neq 0 \text{ and } \tilde{n}_{VRB} < \tilde{N}_{VRB}^{DL} - N_{null} \text{ and } \tilde{n}_{VRB} \bmod 4 \geq 2 \\ \tilde{n}''_{PRB} & , \text{otherwise} \end{cases}$$

$$\text{where } \tilde{n}'_{PRB} = 2N_{row} \cdot (\tilde{n}_{VRB} \bmod 2) + \lfloor \tilde{n}_{VRB} / 2 \rfloor + \tilde{N}_{VRB}^{DL} \cdot \lfloor n_{VRB} / \tilde{N}_{VRB}^{DL} \rfloor,$$

$$\text{and } \tilde{n}''_{PRB} = N_{row} \cdot (\tilde{n}_{VRB} \bmod 4) + \lfloor \tilde{n}_{VRB} / 4 \rfloor + \tilde{N}_{VRB}^{DL} \cdot \lfloor n_{VRB} / \tilde{N}_{VRB}^{DL} \rfloor,$$

where  $\tilde{n}_{VRB} = n_{VRB} \bmod \tilde{N}_{VRB}^{DL}$  and  $n_{VRB}$  就是前面提到的资源分配的结果。而  $\tilde{n}_{VRB}$  显然也就是‘在对应等分的偏移量’

奇数时隙  $n_s$ ：

$$\tilde{n}_{PRB}(n_s) = (\tilde{n}_{PRB}(n_s - 1) + \tilde{N}_{VRB}^{DL} / 2) \bmod \tilde{N}_{VRB}^{DL} + \tilde{N}_{VRB}^{DL} \cdot \lfloor n_{VRB} / \tilde{N}_{VRB}^{DL} \rfloor$$

最后对所有时隙再计算一下，才是最终的 PRB 的位置：

$$n_{PRB}(n_s) = \begin{cases} \tilde{n}_{PRB}(n_s) & , \tilde{n}_{PRB}(n_s) < \tilde{N}_{VRB}^{DL} / 2 \\ \tilde{n}_{PRB}(n_s) + N_{gap} - \tilde{N}_{VRB}^{DL} / 2 & , \tilde{n}_{PRB}(n_s) \geq \tilde{N}_{VRB}^{DL} / 2 \end{cases}$$

(3) 从公式可以看出：

- a. 奇数时隙又在偶数时隙算出的基础上，进行了半个等分的跳频。也就是  $+ \tilde{N}_{VRB}^{DL} / 2$

b. 偶数时隙看起来复杂，我们可以来推导一下。用个例子来说明：

$N_{RB}^{DL}=50$ ，配置 gap2，查表 36211----6.2.3.2-1 得到  $N_{gap}=9$ ， $P=3$ 。

计算得到： $\lfloor N_{RB}^{DL} / 2N_{gap} \rfloor = 2$  个等分；每个等分  $\tilde{N}_{VRB}^{DL} = 2N_{gap} = 18$ ；行数

$N_{row} = \lceil \tilde{N}_{VRB}^{DL} / (4P) \rceil \cdot P = 6$ ； $N_{null} = 4N_{row} - \tilde{N}_{VRB}^{DL} = 4 \cdot 6 - 18 = 6$ ，第 2,4 列需要插入的空数据\*也就各为 3 个。所以 0~17 按照行优先放入矩阵的结果也就为

$$N_{row} \left\{ \begin{array}{c} \text{列数4} \\ \begin{bmatrix} 0 & 1 & 2 & 3 \\ 4 & 5 & 6 & 7 \\ 8 & 9 & 10 & 11 \\ 12 & * & 13 & * \\ 14 & * & 15 & * \\ 16 & * & 17 & * \end{bmatrix} \end{array} \right\} N_{null} / 2$$

结果的：

(1) 当  $\tilde{n}_{VRB} \geq \tilde{N}_{VRB}^{DL} - N_{null} = 4(N_{row} - N_{null} / 2)$  且为偶数时，也就是  $N_{null} \neq 0$  and  $\tilde{n}_{VRB} \geq \tilde{N}_{VRB}^{DL} - N_{null}$  and  $\tilde{n}_{VRB} \bmod 2 = 0$  的时候，此时对应的 12,14, 16.

它们对应的 PRB 是 ‘一个个往上加的’，分别为按列数的 3,4,5（从 0 开始），12 → 3, 14 → 4; 16 → 5。。所以，最小的 12 对应 PRB 也就是 3。从图中就可以看出 12,14,16 之间隔 2；13,15,17 也是隔 2 的。所以公式

$\tilde{n}'_{PRB} = 2N_{row} \cdot (\tilde{n}_{VRB} \bmod 2) + \lfloor \tilde{n}_{VRB} / 2 \rfloor + \tilde{N}_{VRB}^{DL} \cdot \lfloor n_{VRB} / \tilde{N}_{VRB}^{DL} \rfloor$  可以看出  $\tilde{n}_{VRB}$  还得  $\lfloor \tilde{n}_{VRB} / 2 \rfloor$ ，因为后面的  $N_{null} / 2$  行都只有第 1, 3 列有数据

所以，（第一个带有\*的行，第一列）对应的 VRB 的配置就是  $4(N_{row} - N_{null} / 2) / 2$ ，它对应的 PRB 也就是  $N_{row} - N_{null} / 2$ 。可以列出一个方程式  $4(N_{row} - N_{null} / 2) / 2 + x = N_{row} - N_{null} / 2$  》

推导出  $x = -N_{row} + N_{null} / 2$ ，所以就得出了公式

$$\tilde{n}'_{PRB} - N_{row} + N_{null} / 2, N_{null} \neq 0 \text{ and } \tilde{n}_{VRB} \geq \tilde{N}_{VRB}^{DL} - N_{null} \text{ and } \tilde{n}_{VRB} \bmod 2 = 0$$

其实对于这种情况，公式可以改为：

第一个位置对应的  $\tilde{n}_{VRB} = 4(N_{row} - N_{null} / 2)$ ，对应的  $\tilde{n}_{PRB} = N_{row} - N_{null} / 2$ 。所以得到： $\tilde{n}_{PRB}$  = 相对位置（ $\tilde{n}_{VRB} - 4(N_{row} - N_{null} / 2)$ ）/ 2 + 起始位置  $N_{row} - N_{null} / 2$

最后就能再把在第几段加上  $\tilde{N}_{VRB}^{DL} \cdot \lfloor n_{VRB} / \tilde{N}_{VRB}^{DL} \rfloor$ ，

$$\tilde{n}_{PRB}(n_s) = \tilde{n}_{VRB} / 2 - N_{row} + N_{null} / 2 + \tilde{N}_{VRB}^{DL} \cdot \lfloor n_{VRB} / \tilde{N}_{VRB}^{DL} \rfloor$$

(2) 同理,当  $\tilde{n}_{\text{VRB}} \geq \tilde{N}_{\text{VRB}}^{\text{DL}} - N_{\text{null}} = 4(N_{\text{row}} - N_{\text{null}}/2)$  且为奇数是, 也就是 ( $N_{\text{null}} \neq 0$  and  $\tilde{n}_{\text{VRB}} \geq \tilde{N}_{\text{VRB}}^{\text{DL}} - N_{\text{null}}$  and  $\tilde{n}_{\text{VRB}} \bmod 2 = 1$ ) 对于也可以推论出公式:

推导出  $x = N_{\text{row}}$

$$\tilde{n}'_{\text{PRB}} - N_{\text{row}}, N_{\text{null}} \neq 0 \text{ and } \tilde{n}_{\text{VRB}} \geq \tilde{N}_{\text{VRB}}^{\text{DL}} - N_{\text{null}} \text{ and } \tilde{n}_{\text{VRB}} \bmod 2 = 1$$

>>>>其实对于这种情况, 公式可以改为:

第一个位置对应的  $\tilde{n}_{\text{VRB}} = 4(N_{\text{row}} - N_{\text{null}}/2) + 1$ , 对应的  $\tilde{n}_{\text{PRB}} = 3N_{\text{row}} - N_{\text{null}}$ 。所以得到:  $\tilde{n}_{\text{PRB}} = \text{相对位置} (\tilde{n}_{\text{VRB}} - 4(N_{\text{row}} - N_{\text{null}}/2) - 1) / 2 + \text{起始位置 } 3N_{\text{row}} - N_{\text{null}}$   
最后就能再把在第几段加上  $\tilde{N}_{\text{VRB}}^{\text{DL}} \cdot \lfloor n_{\text{VRB}} / \tilde{N}_{\text{VRB}}^{\text{DL}} \rfloor$ ,  
 $\tilde{n}_{\text{PRB}}(n_s) = \tilde{n}_{\text{VRB}} / 2 + N_{\text{row}} + \tilde{N}_{\text{VRB}}^{\text{DL}} \cdot \lfloor n_{\text{VRB}} / \tilde{N}_{\text{VRB}}^{\text{DL}} \rfloor$

(3) 当  $\tilde{n}_{\text{VRB}} < \tilde{N}_{\text{VRB}}^{\text{DL}} - N_{\text{null}} = 4(N_{\text{row}} - N_{\text{null}}/2)$  就比较简单了。

由于前面的  $N_{\text{row}} - N_{\text{null}}/2$  行中, 每列都有数据, 就可以算出  $\tilde{n}_{\text{VRB}}$  对应的列数为  $(\tilde{n}_{\text{VRB}} \bmod 4)$ , 行数  $\lfloor \tilde{n}_{\text{VRB}} / 4 \rfloor$ 。从图中可以看出, 如果是在第 3,4 列, 就要减去  $N_{\text{null}}/2$  个 NULL 的数据 (也就是\*号), 前面 1,2 列就不用减了。最后就得到了公式:

$$\tilde{n}_{\text{PRB}}(n_s) = \begin{cases} \tilde{n}_{\text{PRB}}'' - N_{\text{null}}/2 & N_{\text{null}} \neq 0 \text{ and } \tilde{n}_{\text{VRB}} < \tilde{N}_{\text{VRB}}^{\text{DL}} - N_{\text{null}} \text{ and } \tilde{n}_{\text{VRB}} \bmod 4 \geq 2 \\ \tilde{n}_{\text{PRB}}'' & , \text{otherwise} \end{cases}$$

最后就能再把在第几段加上  $\tilde{N}_{\text{VRB}}^{\text{DL}} \cdot \lfloor n_{\text{VRB}} / \tilde{N}_{\text{VRB}}^{\text{DL}} \rfloor$ ,

$$\tilde{n}_{\text{PRB}}'' = N_{\text{row}} \cdot (\tilde{n}_{\text{VRB}} \bmod 4) + \lfloor \tilde{n}_{\text{VRB}} / 4 \rfloor + \tilde{N}_{\text{VRB}}^{\text{DL}} \cdot \lfloor n_{\text{VRB}} / \tilde{N}_{\text{VRB}}^{\text{DL}} \rfloor$$

#### 4.4.7.7 DCI 格式

- 参看协议: 36212 的 5.3.3.1, 36213 的 7.1
- **DCI 来源:** 由于要满足在不同环境下发送不同格式需要不同信息的数据, 所以产生的 DCI format。Format0 代表上行; format1 代表下行一个码字, 即没有空分; format2 代表下行至多两个码字, 即有空分; format3 代表的功控。
- **RNTI:** RNTI 并不是显示发送的, 而是放在 CRC 中进行发送的
- **DCI format 的区别:**

Relative DCI size	Uplink grant	Downlink assignment	Power control
Small	–	1C Small contiguous allocations	–
	0	1A Contiguous allocations only	3, 3A
...	–	1B Contiguous allocations with spatial multiplexing	–
	–	1 Flexible allocations, no spatial multiplexing	–
Large	–	2 Flexible allocations, full spatial multiplexing	–

**36213 ---- Table 7.1-1: PDCCH and PDSCH configured by SI-RNTI**

DCI format	Search Space	Transmission scheme of PDSCH corresponding to PDCCH
DCI format 1C	<b>Common</b>	If the number of PBCH antenna ports is one, Single-antenna port, port 0 is used, otherwise Transmit diversity.
DCI format 1A	Common	If the number of PBCH antenna ports is one, Single-antenna port, port 0 is used, otherwise Transmit diversity

**36213-----Table 7.1-2: PDCCH and PDSCH configured by P-RNTI**

DCI format	Search Space	Transmission scheme of PDSCH corresponding to PDCCH
DCI format 1C	Common	If the number of PBCH antenna ports is one, Single-antenna port, port 0 is used, otherwise Transmit diversity
DCI format 1A	Common	If the number of PBCH antenna ports is one, Single-antenna port, port 0 is used, otherwise Transmit diversity

**36213-----Table 7.1-3: PDCCH and PDSCH configured by RA-RNTI**

DCI format	Search Space	Transmission scheme of PDSCH corresponding to PDCCH
DCI format 1C	Common	If the number of PBCH antenna ports is one, Single-antenna port, port 0 is used, otherwise Transmit diversity
DCI format 1A	Common	If the number of PBCH antenna ports is one, Single-antenna port, port 0 is used, otherwise Transmit diversity

36213-----Table 7.1-5: PDCCH and PDSCH configured by C-RNTI

Transmission mode	DCI format	Search Space	Transmission scheme of PDSCH corresponding to PDCCH
Mode 1	DCI format 1A	Common and UE specific by C-RNTI	Single-antenna port, port 0 (see subclause 7.1.1)
	DCI format 1	UE specific by C-RNTI	Single-antenna port, port 0 (see subclause 7.1.1)
Mode 2	DCI format 1A	Common and UE specific by C-RNTI	Transmit diversity (see subclause 7.1.2)
	DCI format 1	UE specific by C-RNTI	Transmit diversity (see subclause 7.1.2)
Mode 3	DCI format 1A	Common and UE specific by C-RNTI	Transmit diversity (see subclause 7.1.2)
	DCI format 2A	UE specific by C-RNTI	Large delay CDD (see subclause 7.1.3) or Transmit diversity (see subclause 7.1.2)
Mode 4	DCI format 1A	Common and UE specific by C-RNTI	Transmit diversity (see subclause 7.1.2)
	DCI format 2	UE specific by C-RNTI	Closed-loop spatial multiplexing (see subclause 7.1.4) or Transmit diversity (see subclause 7.1.2)
Mode 5	DCI format 1A	Common and UE specific by C-RNTI	Transmit diversity (see subclause 7.1.2)
	DCI format 1D	UE specific by C-RNTI	Multi-user MIMO (see subclause 7.1.5)
Mode 6	DCI format 1A	Common and UE specific by C-RNTI	Transmit diversity (see subclause 7.1.2)
	DCI format 1B	UE specific by C-RNTI	Closed-loop spatial multiplexing (see subclause 7.1.4) using a single transmission layer
Mode 7	DCI format 1A	Common and UE specific by C-RNTI	If the number of PBCH antenna ports is one, Single-antenna port, port 0 is used (see subclause 7.1.1), otherwise Transmit diversity (see subclause 7.1.2)
	DCI format 1	UE specific by C-RNTI	Single-antenna port; port 5 (see subclause 7.1.1)

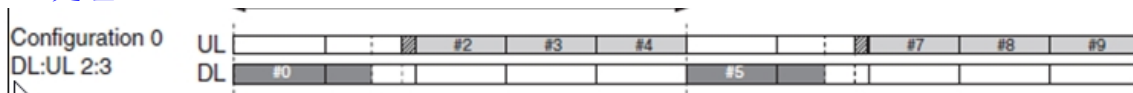
name	用途	format长度bit数 (仅仅是真实的长度)	是否空分	资源分配	TPC对象	有无指示PMI	支持的UE的TM模式	有无power offset
format0	分配PUSCH资源	$18 + \lceil \log_2(N_{RB}^{UL}(N_{RB}^{UL} + 1)/2) \rceil$	否	TYPE2	PUSCH	无	无	无
format1A	分配PDSCH资源	$16/18 + \lceil \log_2(N_{RB}^{UL}(N_{RB}^{UL} + 1)/2) \rceil$	否	TYPE2	PUCCH	无	所有TM模式	无
format1	分配PDSCH资源	$16/17 + \lceil N_{RB}^{DL} / P \rceil$	否	TYPE0/TYPE1	PUCCH	无	TM1/TM2/TM7	无
format1B	分配PDSCH资源	$20/22 + \lceil \log_2(N_{RB}^{UL}(N_{RB}^{UL} + 1)/2) \rceil$	否	TYPE2	PUCCH	有	TM6	无
format1C	分配PDSCH资源	$5/6 + \lceil \log_2(\lfloor N_{RB, gap1}^{UL} / N_{RB}^{step} \rfloor \lfloor N_{RB, gap1}^{UL} / N_{RB}^{step} \rfloor + 1) / 2 \rceil$	否	TYPE2	无	无	无	无
format1D	分配PDSCH资源	$20/22 + \lceil \log_2(N_{RB}^{UL}(N_{RB}^{UL} + 1)/2) \rceil$	否	TYPE2	PUCCH	有	TM5	有
format2	分配PDSCH资源	$28/29/31/32 + \lceil N_{RB}^{DL} / P \rceil$	是	TYPE0/TYPE1	PUCCH	有	TM4	无
format2A	分配PDSCH资源	$25/26/27/28 + \lceil N_{RB}^{DL} / P \rceil$	是	TYPE0/TYPE1	PUCCH	有	TM3	无
format3	PUCCH/PUSCH功控	31bit	否	无	无	无	无	无
format3A	PUCCH/PUSCH功控	31bit	否	无	无	无	无	无

#### 协议规定:

- format0,format1A 必须保持 bit 数长度一样; 用 1bit 来区分它们。
- 由于一个 RNTI 在一种 TM 下, 可以对应两个或多个 DCI 格式, 所以你可以想象到为了区分它们, 协议规定用它们含有消息的长度来区分, 而 format0 与 format1A 的区别是用 1bit 区分的。从上面的总结的各种 format 的长度来看, format1A 只可能和 format1B, format1 长度可能一样, format1C 比 format1A 短, format2/2A 又比 format1A 长, 所以明白协议为什么 “format1,format1B 才需要去和 format1A 进行比较, 如果一样就对 format1,format1B 加 bit, 而其他的 format 不需要”。
- format3/3A 与 format0/format1 是用不同的 RNTI 区分开的, 但长度相同
- 从 36213-----Table 7.1-5 了解到 format2A 和 2 的用途的区别, 你也能大概估计到它们上报的 ‘PMI’ 的不同了。
- 你也可以了解到为什么 ‘format1A 的时候为什么要对 RA-RNTI,P-RNTI,SI-RNTI 有特殊处理了’? 因为 format1A 可能被正常的 ue 特定的 RNTI, TC-RNTI 使用, 也有可能被 RA-RNTI,P-RNTI,SI-RNTI 使用, 而 RA-RNTI,P-RNTI,SI-RNTI 对应的数据是公共的, 不需要重传,和某个具体的 UE 没关系, 所以对于 RA-RNTI,P-RNTI,SI-RNTI 在 format1A 中的 HARQ process number 和 downlink Assignment index 也就没用了。
- 由于 format1A 给 RA-RNTI,P-RNTI,SI-RNTI; 出于让 RA-RNTI,P-RNTI,SI-RNTI 能更可靠的接收, 采取了 ‘用增加资源来增加冗余’ 的方法。所以, 你才会明白为什么对 format1A 中的 TPC 字段的有个关于 ‘RA-RNTI,P-RNTI,SI-RNTI’ 的特殊处理。

#### 4.4.7.7.1 DCI 0

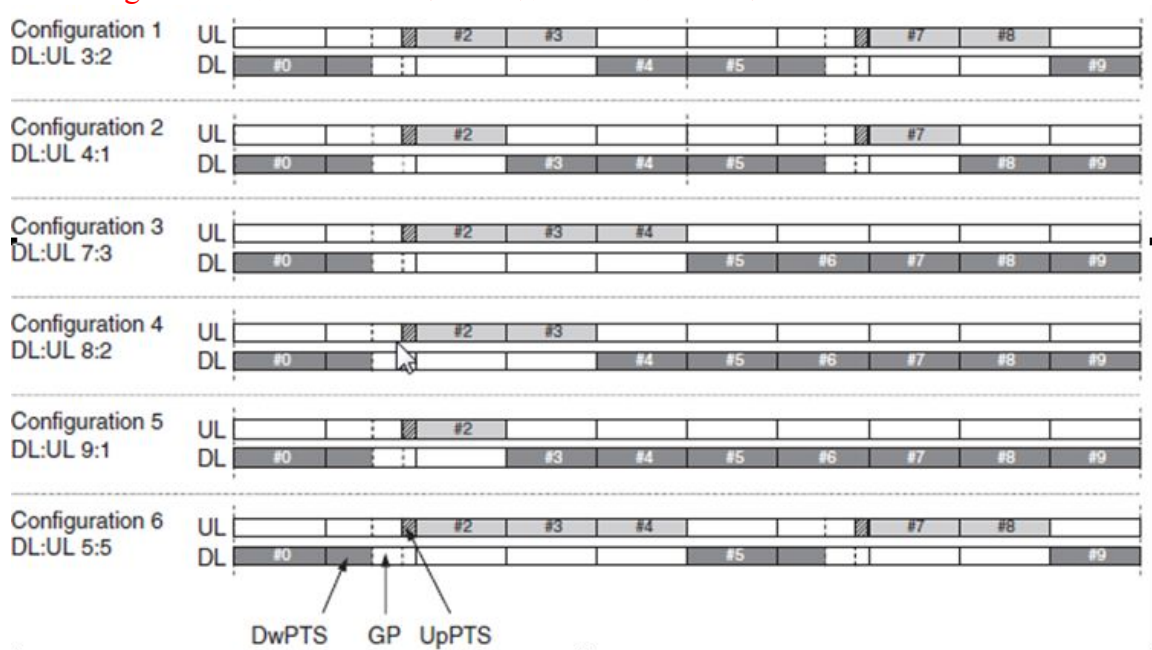
- **相关 TM 模式:** 和 TM 模式没有关系, 因为它是上行调度的分配。TM 只是对下行有区别。
- **参看:** 36212 的 5.3.3.1.1
- **资源分配:** 参看 ‘资源分配的 TYPE2’
- $N_{UL\_hop}$ : 下行带宽小于等于 49, 占用 1bit; 否则占用 2 比特。
- **DMRS;** 因为是给 ‘PUSCH’ 分配资源, 所以需要 ‘DMRS’ 来决定对应的 PHICH 资源的位置, 只有上行资源分配才需要 ‘DMRS’。具体参看 ‘本文档的对 PHICH 的描述’
- **UL index (2bit):** 为什么只在 TDD uplink-downlink configuration 0 中才需要是因为: 这种配置中 10ms 中, 上行子帧比下行多, 所以就想到怎么让调度到所有的上行 subframe, 就通过这个指示。MSB (高比特位为 1) 就表示更近一些, 子帧 0/1 发送的 format0 将在 4/7 子帧起效; LSB (低比特位为 1) 就表示更远一些, 子帧 0/1 发送的 format0 将在 7/8 子帧起效。如果 MSB 和 LSB 都是 1, 那么在 4/7,7/8 子帧都会有数据的发送。子帧 5/6 也类似的处理。





注：其实这里有个问题，就是‘即使两个上行时隙都有调度，分配的资源都是一样’。应该不一样比较灵活吧。也可以考虑以后扩展。

- DAI (2bit)：为什么只在 TDD uplink-downlink configuration 1-6 中才需要？因为下行的子帧比上行多，必然会产生一个上行子帧对多个下行子帧同时回 ACK/NACK 的情况。所以该字段的作用是指示对于一个 UE 下行已经调用了几次了，这样让 UE 知道它收到的调用是否和 eNodeB 发送的一样，是否有漏掉的。所以 config0 上行 subframe 比下行的多，肯定一次下行调度就对应一次 ACK,所以不需要；而 config1-6 下行的子帧大于等于上行，所以可能多次的 PDSCH 需要再某一个上行 subframe 上回，所以需要。个人感觉其实 config6，上下行 subframe 数目一样的，不需要该字段。



- 因为上行传数据相对的 HARQ process 是同步的，也就是说在这个 subframe 从 UE 传给 eNodeB 的数据归哪个 HARQ process 处理是潜规则定好的，所以上行不需要传 HARQ process；而下行是异步的，需要传 HARQ process 号。
- CQI request 表示‘PUSCH 传输数据的时候，是否上报 CQI/PMI/RI’信息，具体参看后面上行的说明。

#### 4.4.7.7.1.1 问题

- 问题 1：DCI0 里面又不是下行调度，为什么需要带有 DAI 呢？

答：的确按理“DCI0”是不需要带 DAI。但是‘发送 DCI0 的下行时隙’时间点比较特殊，一定是‘DCI0 调度的上行时隙往前数 $\geq 4$ 的最近的一个下行时隙’，也就是说‘发送 DCI0 的下行时隙’一定是‘DCI0 调度的上行时隙需要回 HARQ ACK/NACK 的最后一个时隙’。所以，要带下去‘以防止中间数据的丢失’。具体后面谈到 PUCCH ACK/NACK 再讲。

\



#### 4.4.7.7.2 DCI 1A

- **相关 TM 模式：**所有情况，包括所有 TM 模式 + 公共的消息。所以为什么长度都要与它做比较，就是它再任何模式下都可能有。
- **参看：**36212 的 5.3.3.1.3
- **资源分配：**TYPE2
- 由于 1A 又能给‘公共消息用’又能针对‘某个 UE’用，所以看到里面有些特殊处理

(1) RA-RNTI, P-RNTI, or SI-RNTI 的资源分配，由于它们传送的数据都是新数据，不可能重发，所以 new data indicator 这个其实没有什么意义，所以就用它来表示‘跳频’的类型是 gap1 还是 gap2（当然是  $N_{RB}^{DL} \geq 50$  才有意义，因为跳频的时候， $N_{RB}^{DL} \geq 50$  才有 gap1，gap2 的区别；否则只有 gap1），这样做得好处是在‘distributed VRB’即跳频的时候多了 1bit 去分配资源。

(2) 对于 RA-RNTI, P-RNTI, or SI-RNTI 由于比较重要，所以为了提高传输的可靠性，使用了提高冗余来达到传输可靠性的目的。TPC command 对于它们也没用，因为不知道具体的 UE，所以利用 TPC command 字段。TBS 的大小判断不是根据‘资源分配里面的 RB 数来决定的’得到的，而是根据 If least significant bit is 0 then  $N_{PRB}^{1A} = 2$  else  $N_{PRB}^{1A} = 3$  得到的，去查 36213 的 Table 7.1.7.2.1-1。但映射到多少实际的物理资源上去还是按照‘资源分配里面的 RB 数来决定的’。但调制方式还是根据 DCI1A 带有的 MCS 决定的。

举个例子：

如果传输 80bit 的数据， $N_{PRB}^{1A} = 3$  去查表得到 88bit，所以按理 3 个 RBpair 就够了，为了增加冗余性提高解码成功率。资源分配的时候，实际分配了 5 个 RBpair。所以传输的 TBS 就是 88bit，‘TPC command 的低 bit 设置为 1，而资源分配对应到了实际的 5 个 RB，然后按照正常的去‘编码交织到 5 个 RB pair’上去，这样就更加冗余了，可靠性也提高了。

##### 4.4.7.7.2.1 问题

- **问题 1: 怎么区分 DCI1A 里面的 PDCCH order 和其他情况呢？PDCCH order 一般式什么时候发送呢？**

答：一般是 enodeb 认为 UE 上行失步的时候，就会发起 PDCCH order。

DCI1A 里面，由于 PDCCH order 不需要分配 RB，所以通过‘资源分配是的位置是否全部设置为 1’来区分是 DCI 1A 的‘PDCCH order 还是其他情况’。36212 协议里面有 PDCCH order 的描述里面有这样一句话

Resource block assignment –  $\lceil \log_2(N_{RB}^{DL}(N_{RB}^{DL} + 1)/2) \rceil$  bits, where all bits shall be set to 1

#### 4.4.7.7.3 DCI 1

- **相关 TM 模式：**TM1，TM2，TM7.

- 参看：36212 的 5.3.3.1.2
- 资源分配：TYPE0/1
- 适用场景：用于传输 DL-SCH 的 SIMO 操作调度分配信息。非连续的资源分配。提高灵活性，带来了更大的控制负载。

#### 4.4.7.7.4 DCI 1B

- 相关 TM 模式：TM6.
- 参看：36212 的 5.3.3.1.3A
- 资源分配：TYPE2
- 适用场景：codebook-based beam-forming. 因为要上报预编码矩阵（precoding matrix）和 RI（多少层指示信息）。所以要比 1A 大一些。

用于闭环 MIMO **rank=1** 时的调度分配，它可以支持连续的资源分配或者基于分布式虚拟资源块的连续资源分配

- **PMI**:表示 ‘是用 UE 最近上报的 PMI 还是用 enodeb 指定的 PMI’,所以只有 1bit。
- **TPMI**:表示 ‘用 enodeb 指定的 PMI 的时候，是哪一个 PMI’。因为此时只有一层，所以对于天线数 PMI 天线数为 2，4 分别只有以下 PMI 可选。

**36211--- Table 6.3.4.2.3-1: Codebook for transmission on antenna ports {0,1}, 2 天线用 2bit 表示.**

Codebook index	Number of layers $\nu$	
	1	2
0	$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$	
1	$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix}$	
2	$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ j \end{bmatrix}$	
3	$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -j \end{bmatrix}$	

36211----Table 6.3.4.2.3-2: Codebook for transmission on antenna ports {0,1,2,3} .—4 天线用 4bit 表示

Codebook index	$u_n$	Number of layers $\nu$
0	$u_0 = [1 \quad -1 \quad -1 \quad -1]^T$	$W_0^{\{1\}}$
1	$u_1 = [1 \quad -j \quad 1 \quad j]^T$	$W_1^{\{1\}}$
2	$u_2 = [1 \quad 1 \quad -1 \quad 1]^T$	$W_2^{\{1\}}$
3	$u_3 = [1 \quad j \quad 1 \quad -j]^T$	$W_3^{\{1\}}$
4	$u_4 = [1 \quad (-1-j)/\sqrt{2} \quad -j \quad (1-j)/\sqrt{2}]^T$	$W_4^{\{1\}}$
5	$u_5 = [1 \quad (1-j)/\sqrt{2} \quad j \quad (-1-j)/\sqrt{2}]^T$	$W_5^{\{1\}}$
6	$u_6 = [1 \quad (1+j)/\sqrt{2} \quad -j \quad (-1+j)/\sqrt{2}]^T$	$W_6^{\{1\}}$
7	$u_7 = [1 \quad (-1+j)/\sqrt{2} \quad j \quad (1+j)/\sqrt{2}]^T$	$W_7^{\{1\}}$
8	$u_8 = [1 \quad -1 \quad 1 \quad 1]^T$	$W_8^{\{1\}}$
9	$u_9 = [1 \quad -j \quad -1 \quad -j]^T$	$W_9^{\{1\}}$
10	$u_{10} = [1 \quad 1 \quad 1 \quad -1]^T$	$W_{10}^{\{1\}}$
11	$u_{11} = [1 \quad j \quad -1 \quad j]^T$	$W_{11}^{\{1\}}$
12	$u_{12} = [1 \quad -1 \quad -1 \quad 1]^T$	$W_{12}^{\{1\}}$
13	$u_{13} = [1 \quad -1 \quad 1 \quad -1]^T$	$W_{13}^{\{1\}}$
14	$u_{14} = [1 \quad 1 \quad -1 \quad -1]^T$	$W_{14}^{\{1\}}$
15	$u_{15} = [1 \quad 1 \quad 1 \quad 1]^T$	$W_{15}^{\{1\}}$

所以为什么会产生这个表

36212---Table 5.3.3.1.3A-1: Number of bits for TPMI information

Number of antenna ports at eNode-B	Number of bits
2	2
4	4

#### 4.4.7.7.5 DCI 1C

- 相关 TM 模式：和 TM 模式没关，只用于公共消息(SI-RNTI, P-RNTI, RA-RNTI,).
- 参看:36212 的 5.3.3.4
- 资源分配： TYPE2
- 适用场景： random-access response, paging, system information。主要用于下行调度呼叫，RAR 以及广播消息指示

- 因为 1C 只用于 RA-RNTI, P-RNTI, or SI-RNTI, 所以可以看到很多字段都没有。例如: - Modulation and coding scheme, 因为固定是 QPSK, HARQ process number。

#### 4.4.7.7.6 DCI 1D

- 相关 TM 模式: TM5.
- 参看:36212 的 5.3.3.1.4A
- 资源分配: TYPE2
- 适用场景: MIMO.

注: 没太了解过 MIMO 的过程, 不讨论。

#### 4.4.7.7.7 DCI 2

- 相关 TM 模式: TM4, close-loop 的空分 或者 发射分集.参看 36212 的 5.3.3.1.5
  - 资源分配: TYPE0/1
  - 适用场景: 用于 DL-SCH MIMO 调度。非连续资源分配+ 空分。 DCI 1 + 空分
  - 怎么判断一个 transport block 是 disable 的. Modulation and coding scheme=0(Imcs=0)并且 Redundancy version=1 (RVidx=1)
  - 由于 DCI 2 对应的传输模式中得空分为 ‘close-loop’, 需要指定 PMI, 所以能明白 Table 5.3.3.1.5-4 and Table 5.3.3.1.5-5 表为什么要指明具体的 PMI 了。而 DCI 2A 对应的传输模式用的空分为 CDD,所以不需要指定具体的 PMI 了, PMI 是 ‘潜规则’ 可以算出来的。
  - 从 PMI 的分配情况能看出很多限制
- (1) 2 个 port

36212---Table 5.3.3.1.5-4: Content of precoding information field for 2 antenna ports

One codeword: Codeword 0 enabled, Codeword 1 disabled		Two codewords: Codeword 0 enabled, Codeword 1 enabled	
Bit field mapped to index	Message	Bit field mapped to index	Message
0	2 layers: Transmit diversity	0	2 layers: Precoding corresponding to precoder matrix $\frac{1}{2} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$
1	1 layer: Precoding corresponding to precoding vector $[1 \quad 1]^T / \sqrt{2}$	1	2 layers: Precoding corresponding to precoder matrix $\frac{1}{2} \begin{bmatrix} 1 & 1 \\ j & -j \end{bmatrix}$
2	1 layer: Precoding corresponding to precoder vector $[1 \quad -1]^T / \sqrt{2}$	2	2 layers: Precoding according to the latest PMI report on PUSCH, using the precoder(s) indicated by the reported PMI(s)
3	1 layer: Precoding corresponding to precoder vector $[1 \quad j]^T / \sqrt{2}$	3	reserved
4	1 layer: Precoding corresponding to precoder vector $[1 \quad -j]^T / \sqrt{2}$	4	reserved
5	1 layer: Precoding according to the latest PMI report on PUSCH, using the precoder(s) indicated by the reported PMI(s), if RI=2 was reported, using 1 <sup>st</sup> column  multiplied by $\sqrt{2}$ of all precoders implied by the reported PMI(s)	5	reserved
6	1 layer: Precoding according to the latest PMI report on PUSCH, using the precoder(s) indicated by the reported PMI(s), if RI=2 was reported, using 2 <sup>nd</sup> column  multiplied by $\sqrt{2}$ of all precoders implied by the	6	reserved

	reported PMI(s)		
7	reserved	7	reserved

限制：

<1> 上报的 RI=2，如果最后 PDSCH ‘只传一个码字’，只能用一层，不能用两层。很奇怪为什么不用 ‘一个码字两层的情况’ 呢？？从限制好像是 ‘用两层肯定就是两个码字’。

(2) 4 个 port

**Table 5.3.3.1.5-5: Content of precoding information field for 4 antenna ports**

<b>One codeword: Codeword 0 enabled, Codeword 1 disabled</b>		<b>Two codewords: Codeword 0 enabled, Codeword 1 enabled</b>	
<b>Bit field mapped to index</b>	<b>Message</b>	<b>Bit field mapped to index</b>	<b>Message</b>
0	4 layers: Transmit diversity	0	2 layers: TPMI=0
1	1 layer: TPMI=0	1	2 layers: TPMI=1
2	1 layer: TPMI=1	⋮	⋮
⋮	⋮	15	2 layers: TPMI=15
16	1 layer: TPMI=15	16	2 layers: Precoding according to the latest PMI report on PUSCH using the precoder(s) indicated by the reported PMI(s)
17	1 layer: Precoding according to the latest PMI report on PUSCH using the precoder(s) indicated by the reported PMI(s)	17	3 layers: TPMI=0
18	2 layers: TPMI=0	18	3 layers: TPMI=1
19	2 layers: TPMI=1	⋮	⋮
⋮	⋮	32	3 layers: TPMI=15
33	2 layers: TPMI=15	33	3 layers: Precoding according to the latest PMI report on PUSCH using the precoder(s) indicated by the reported PMI(s)
34	2 layers: Precoding according to the latest PMI report on PUSCH using the precoder(s) indicated by the reported PMI(s)	34	4 layers: TPMI=0
35 – 63	reserved	35	4 layers: TPMI=1
		⋮	⋮
		49	4 layers: TPMI=15
		50	4 layers: Precoding according to the latest PMI report on PUSCH using the precoder(s) indicated by the reported PMI(s)
		51 – 63	Reserved

限制：



<1> 当选择 UE 在 PUSCH 上报的 PMI 的时候，此时上报的 RI 必须和 enodeb 选择的层数一样。否则，如果 UE 上报的是 3 层的 PMI，而 enodeb 指定 2 层而又使用 PUSCH 上报的 PMI（例如一个码字，选择 index=34 的情况），这样 PMI 根本不可能匹配得上

<2> enodeb 选择两层的时候，可以是一个码字或者两个码字。如果大于两层，就肯定是 2 个码字。没有一个码字对应‘三层或者四层’的情况。

- 根据限制总结：

- （1） 如果 enodeb 的 port 数 $\geq 2$ ，选择两层必须为 2 个码字；
- （2） 1 个码字最多 2 层
- （3） 1 个码字用 2 层的情况，只可能是重传的情况。

For a single enabled codeword, indices 18 to 34 inclusive in Table 5.3.3.1.5-5 are only supported for retransmission of the corresponding transport block if that transport block has previously been transmitted using two layers with closed-loop spatial multiplexing

从协议这句话可以看出。

#### 4.4.7.7.1 问题

- 问题 1：36212 的表 Table 5.3.3.1.5-4 and Table 5.3.3.1.5-5 表示有时用的 PMI 是 UE 在 PUSCH 上报的，但 UE 可能上报多个 PMI，怎么知道哪一个呢？

using the precoder(s) indicated by the reported PMI(s). 由于最多上报 2 个 PMI，所以 UE 必须去试用哪个 PMI 才行。

- 问题 2：为什么只接受 PUSCH 上报的 PMI 呢，不接受 PUCCH 上报的 PMI 呢，因为很多时候 PUSCH 上报的 PMI 就是把 PUCCH 上报的 PMI 移到 PUSCH 上上报而已？？？

- 问题 3：enodeb 的 port 数为 2 的情况，很奇怪为什么不用‘一个码字两层的情况’呢？

按道理是可以的，但不利于资源的利用。从限制好像是‘用两层肯定就是两个码字’。

#### 4.4.7.7.8 DCI 2A

- 相关 TM 模式：TM3，CDD 的空分 或者 发射分集.
- 参看：36212 的 5.3.3.1.5A
- 资源分配：TYPE0/1
- 适用场景：用于 DL-SCH MIMO 调度。非连续资源分配+ 空分。 DCI 1 + 空分，CDD.

- 协议规定：enodeb 为 2 个 port 的时候，如果传一个码字，用发射分集，不会用空分，所以 PMI 是订下来的；如果传两个码字，就用空分，PMI 也是固定的，如下图，所以也不需要说明。所以，最后 enodeb 为 2 个 port 的时候，不需要通知 UE 关于 PMI 的信息，也就是 Number of bits for precoding information = 0.

**36211---Table 6.3.4.2.3-1: Codebook for transmission on antenna ports {0,1}.**

Codebook index	Number of layers $\nu$	
	1	2
0		$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

**36212---Table 5.3.3.1.5A-1: Number of bits for precoding information**

Number of antenna ports at eNode-B	Number of bits for precoding information
2	0
4	2

- 看 PMI 图也能知道 DCI2A 和 DCI2 有同样的限制：
  - (1) 如果 enodeb 的 port 数  $\geq 2$ ，选择两层必须为 2 个码字；
  - (2) 一个码字最多两层
  - (3) 1 个码字用 2 层的情况，只可能是重传的情况

#### 4.4.7.7.9 DCI3、3A

- 相关 TM 模式：和 TM 模式无关，主要用于功控。
- 参看：36212 的 5.3.3.1.6 和 5.3.3.1.7
- 资源分配：特有的分配方式
- 适用场景：用于传输 PUCCH 以及 PUSCH 的 TPC 控制信息，采用 1/2 bit 表示的功率调整
- 配置：

RadioResourceConfigDedicated-->PhysicalConfigDedicated-->tpc-PDCCH-ConfigPUCCH

RadioResourceConfigDedicated-->PhysicalConfigDedicated-->tpc-PDCCH-ConfigPUSCH

- 思想：就是想针对一些 UE 的上行 PUCCH/PUSCH 进行功控，由于功控占用的资源比较少，为了节省资源。通过：配置的给多个 UE 分配给同样的“tpc-PUCCH-RNTI 和 tpc-PUSCH-rnti，各个 UE 通过不同的 tpc-index 来区分”，这样 UE 就知道它对应的是第几个 TPC command number 了，注意 tpc-index 是从 1 开始的，不是从 0 开始的。

例如：如果 UE 的 index 为 1，就知道 TPC command number1 是针对自己的。

The following information is transmitted by means of the DCI format 3A:

- TPC command number 1, TPC command number 2,..., TPC command number M

- 要与 format1A 的长度尽量对齐，为了在不同带宽的情况下的长度不一样，所以只能选择带宽最大的 100RB。Format1A 的长度为：  

$$16/18 + \left\lceil \log_2(N_{RB}^{UL}(N_{RB}^{UL} + 1)/2) \right\rceil$$
按照最大的 100RB 算出就是 31bit。而 DCI3 用 2BIT 表示对一个 UE 的功控，DCI3A 用 1BIT，所以针对一个 tpc-PUCCH-RNTI 和 tpc-PUSCH-rnti，DCI 3 的 tpc-index 显然就只有 3A 的一半。从 ASN 文件中就能看出来。
- 显然 DCI3 能针对更多的 UE 进行功控；而 DCI3A 由于功控为 2bit，所以相对 DCI3 来说，功控更加精准。

#### 4.4.7.8 时频位置 ---- 盲检测

- 为什么要有盲检测：因为 UE 在每一个下行子帧不知道自己对应的 PDCCH 是位置，所以必须去检测每个看是不是给自己的，前面‘思想来源’已经提到了。关键字有两个：RNTI 和子帧号。
- 参看：36213 的 9.1.1
- 如果减少 UE 的盲检测的次数：（1）规定 CCE 的 level，只有 1,2,4,8，也就是多少个 CCE 为一个检测的对应的 PDCCH 的起点位置。因为 PDCCH 的格式不一样，所以对应的需要的 CCE 的资源大小也不一样。（2）同时也规定了每个 UE 的在每个 CCE LEVEL 对应的搜索空间的长度，也就是说每个 UE 只在某些资源上进行搜索，而不是所有 PDCCH 资源；（3）为了不增加系统的控制负荷，所以盲检测空间不是 eNodeB 告诉手机的，而是通过公式计算的，告诉也就不叫盲检测了。
- 搜索空间分类：公共搜索空间和 UE 专属的搜索空间--- 规定
  - （1）由于有些消息是针对多个 UE 的，所以有公共搜索空间。这些消息是：系统消息，paging 寻呼消息，功控消息。对应的 RNTI 也就为 SI-RNTI, RA-RNTI, P-RNTI, TC-RNTI。这些 RNTI 它们有个共同的特点：就是这些 RNTI 都是给‘不定数’的多个 UE 的。特别强调为什么用‘不定数’而不用‘多个’，因为协议后面的‘TM8’是给两个定下来的 UE 的，要放到‘UE 专属的搜索空间’才行。
  - （2）公共搜索空间只支持 CCE LEVEL 4 和 8
    - i. 公共搜索空间只支持 0/1A/3/3A, 1C 这些 DCI 格式
    - ii. 公共搜索空间也可以给单个的 UE 调度用，但从上面看出明显受到格式和 LEVEL 的限制（TC-RNTI）

36213-----Table 9.1.1-1: PDCCH candidates monitored by a UE.

Type	Search space $S_k^{(L)}$		Number of PDCCH candidates $M^{(L)}$
	Aggregation level $L$	Size [in CCEs]	
UE-specific	1	6	6
	2	12	6
	4	8	2
	8	16	2
Common	4	16	4
	8	16	2

由于在一种下行传输模式（参考36.213 7.1节），只有两种DCI长度的格式，因此用户专属搜索空间的盲检次数为：（6+6+2+2）\*2=32。\*2代表公共空间只能有两种DCI format，或者UE专用空间在某一种传输模式(TM)下也只能有两种DCI format；公共搜索空间也只支持两种DCI format

也即是每个子帧处于LTE-ACTIVE的UE需要盲检的次数为：

公共搜索空间 12 次+用户专属搜索空间 32 次=44 次

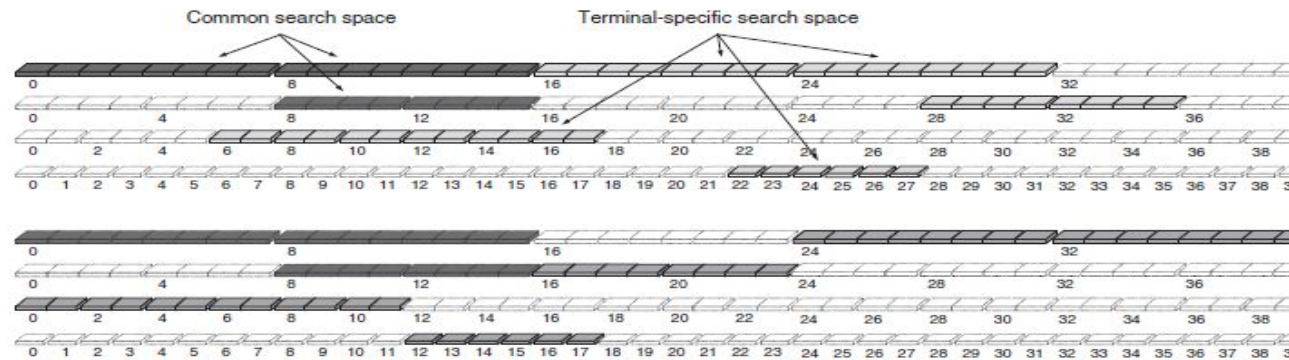


Figure 16.25 Principal illustration of search spaces in two terminals.

- 盲检测的公式的思想：为了让 UE 尽量错开搜索空间，所以公式会根据 UE 的 C-RNTI 来进行计算；同时万一在一个子帧不同 UE 的搜索空间有冲突了，希望在下一个子帧里面冲突能解决或者随机化，所以又引入了子帧的时隙号来计算。

$$L \cdot \left\{ (Y_k + m) \bmod \left\lfloor N_{\text{CCE},k} / L \right\rfloor \right\} + i$$

$i = 0, \dots, L-1$ ，代表了一次盲检测对应的 CCE 的相对位置

$m = 0, \dots, M^{(L)} - 1$ 。  $M^{(L)}$  代表 level 为 L 的搜索空间的 PDCCH 的数目

$L \in \{1, 2, 4, 8\}$ ，CCE 的 level

它们都能从表中获得：

36213-----Table 9.1.1-1: PDCCH candidates monitored by a UE.

Type	Search space $S_k^{(L)}$		Number of PDCCH candidates $M^{(L)}$
	Aggregation level $L$	Size [in CCEs]	
UE-specific	1	6	6
	2	12	6
	4	8	2
	8	16	2
Common	4	16	4
	8	16	2

$$Y_k = (A \cdot Y_{k-1}) \bmod D, \quad Y_{-1} = n_{\text{RNTI}} \neq 0, \quad A = 39827, \quad D = 65537 \quad \text{and}$$

$$k = \lfloor n_s / 2 \rfloor$$

注：如果是公共搜索空间的时候，用  $Y_k = 0$  来计算。

举个例子：

假如  $n_s=1$ ，则  $k = \lfloor n_s/2 \rfloor = 0$ ， $n_{\text{RNTI}}=100$ ， $Y_{-1} = 100$  那么就可以推算出

$$\begin{aligned} Y_k &= (A \cdot Y_{k-1}) \bmod D \\ &= (39827 \cdot 100) \bmod 65537 \\ &= 50480 \end{aligned}$$

假如  $N_{\text{CCE},k} = 88$ ，可以得到相应的 CCE 索引：

$i = 0$  并且  $m = 0, \dots, M^{(1)} - 1$ ，对于  $L=1$ ， $M^{(L)} = M^{(1)} = 6$ ，每个候选 PDCCH 占用 1 个 CCE：

L=1

0

56

57

58

59

60

61

62

63

87

56

PDCCH 1

57

PDCCH 2

58

PDCCH 3

59

PDCCH 4

60

PDCCH 5

61

PDCCH 6

$$L \cdot \{(Y_k + m) \bmod \lfloor N_{\text{CCE},k} / L \rfloor\} + i$$
$$= 1 \cdot \{(50480 + 0) \bmod \lfloor 88 / 1 \rfloor\} + i$$
$$= 1 \cdot 56 + i$$

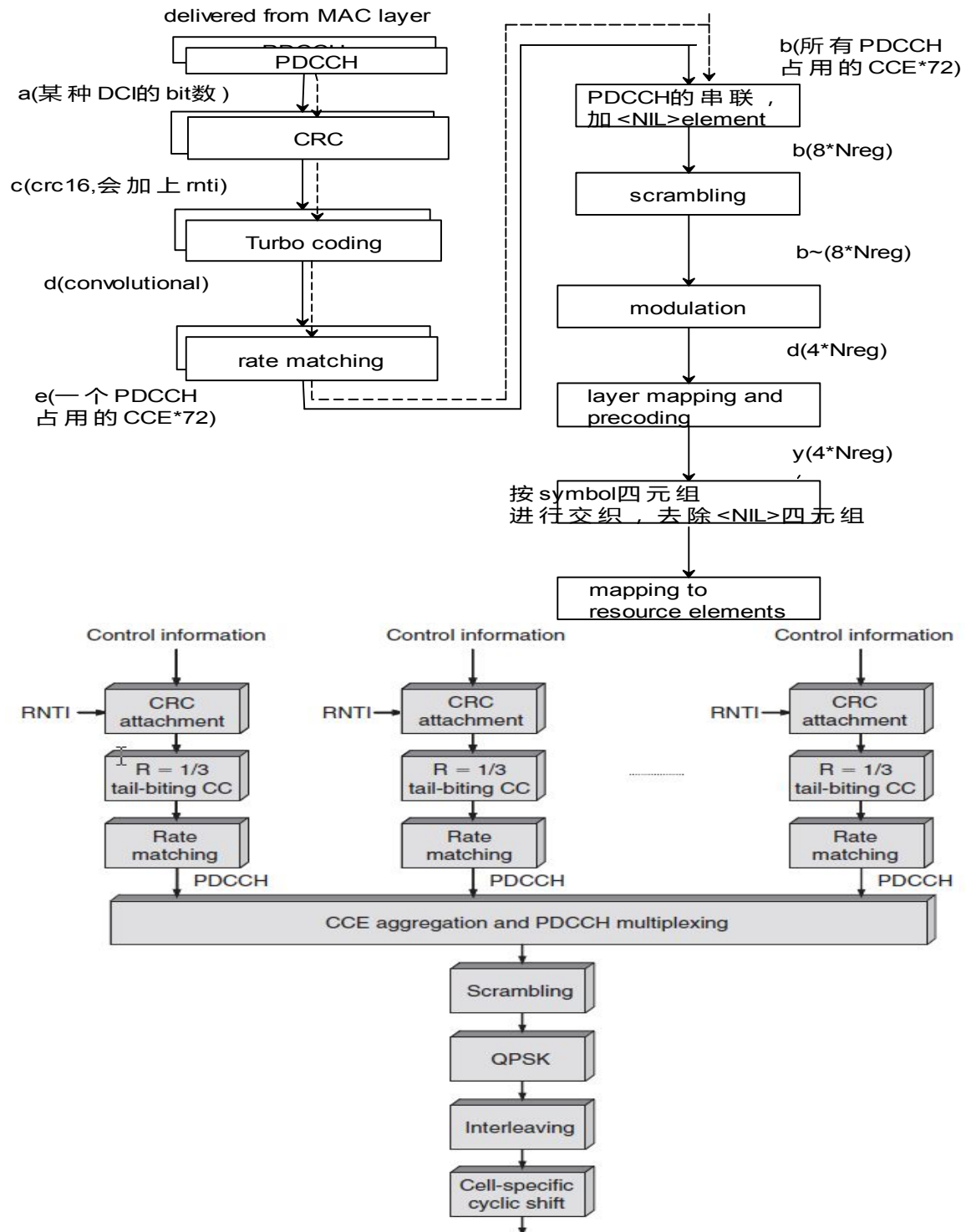
4.4.7.9 数据来源

- 上层调度分配‘信息’，也就是前面面提到的各种 DCI 的格式。

4.4.7.10 流程

- 参看: 36212 的 5.3.3,和 36211 的 6.8.1

注：PDCCH 是把所有的不同 UE 的 PDCCH 合在一起交织之后放在所有的 PDCCH 资源上发送的；而 PDSCH 则不同，只是针对一个 UE 的数据进行发送，所以每个 UE 对应的资源上只是发送自己的 PDSCH 的数据。



- 输入的 bit 数: 对应不同的情况 (SI-RNTI, C-RNTI), 会有不同的 DCI format 格式需要的 bit 数

36321----- Table 7.1-1: RNTI values.

Value (hexa-decimal)	RNTI
0000	N/A
0001-003C	RA-RNTI, C-RNTI, Semi-Persistent Scheduling C-RNTI, Temporary C-RNTI, TPC-PUCCH-RNTI and TPC-PUSCH-RNTI (see note)
003D-FFF3	C-RNTI, Semi-Persistent Scheduling C-RNTI, Temporary C-RNTI, TPC-PUCCH-RNTI and TPC-PUSCH-RNTI
FFF4-FFFD	Reserved for future use
FFFE	P-RNTI
FFFF	SI-RNTI

NOTE: The values corresponding to the RA-RNTI values of a cell's PRACH configuration are not used in the cell for any other RNTI (C-RNTI, Semi-Persistent Scheduling C-RNTI, Temporary C-RNTI, TPC-PUCCH-RNTI or TPC-PUSCH-RNTI).

- **CRC:**加 CRC16，也会把 RNTI 加入进去。如果 UE 传输天线选择也配置了，还得加入天线选择。

$$c_k = b_k \quad \text{for } k = 0, 1, 2, \dots, A-1$$

$$c_k = (b_k + x_{rnti,k-A}) \bmod 2 \quad \text{for } k = A, A+1, A+2, \dots, A+15.$$

$$c_k = b_k \quad \text{for } k = 0, 1, 2, \dots, A-1$$

$$c_k = (b_k + x_{rnti,k-A} + x_{AS,k-A}) \bmod 2 \quad \text{for } k = A, A+1, A+2, \dots, A+15.$$

Table 5.3.3.2-1: UE transmit antenna selection mask

UE transmit antenna selection	Antenna selection mask $\langle x_{AS,0}, x_{AS,1}, \dots, x_{AS,15} \rangle$
UE port 0	$\langle 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 \rangle$
UE port 1	$\langle 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1 \rangle$

- **速率匹配:** 和 RVidx 没有关系
- **串联加<NIL>:** 由于 PDCCH 对应到固定的 CCE 位置，所以中间有些 CCE 可能没有分配，所以这里要把所有的 PDCCH 的 bit 数据按照 CCE 的顺序来安排，对于没有分配的 CCE，进行<NIL>填充处理，一直要充填满整个给 PDCCH 留用的所有 REG 为止 ( $N_{REG}$ ，除去给 PCFICH, PHICH 的所有的 REG 的个数)，总 bit 数  $8N_{REG}$ 。这样才能达到“36211 的 6.8.2 中写到的 CCE number  $n$  corresponds to bits  $b(72n), b(72n+1), \dots, b(72n+71)$ ”，

$M_{\text{tot}} = 8N_{REG} \geq \sum_{i=0}^{n_{\text{PDCCH}}-1} M_{\text{bit}}^{(i)}$  ‘。这里为什么是  $8N_{REG}$  是因为 ‘每个 REG 对应 4 个符号，而 QPSK 编码方式一个符号对应 2bit，所以算下来，占满所有的  $N_{REG}$  对应的 bit 数  $M_{\text{tot}} = 8N_{REG}$ ’

- **扰码:**  $c_{\text{init}} = \lfloor n_s / 2 \rfloor 2^9 + N_{\text{ID}}^{\text{cell}}$
- **调制模式:** QPSK
- 每个发送端口，按 ‘symbol 四元组为单位（以前我们都是按照每个符号进行交织，现在相当于用 REG 作为单位）’ 进行交织，去除<NIL>四元组：因为有<NIL>的 symbol，所以这里必须在进行一次交织来去掉<NIL>的四元组，然后真正达到 ‘有效的四元组的数目 =  $N_{REG}$ ’，然后再循环移位一下：



$$\overline{w}^{(p)}(i) = w^{(p)}((i + N_{\text{ID}}^{\text{cell}}) \bmod M_{\text{quad}})。$$

注：PDCCH 是最后一步才进行交织的，而 PDSCH 是在加扰前进行交织的。

- 资源映射：按照四元组为单位映射，注意是‘先时域，再频域’的思想。

#### 4.4.7.11 协议结构体

- 传输模式：RRC:: radioResourceConfigDedicated-->PhysicalConfigDedicated-->AntennaInfoDedicated-->transmissionMode
- 功控： RRC:: RadioResourceConfigDedicated-->PhysicalConfigDedicated-->tpc-PDCCH-ConfigPUCCH  
RRC:: RadioResourceConfigDedicated-->PhysicalConfigDedicated-->tpc-PDCCH-ConfigPUSCH

#### 4.4.7.12 问题

4.4.7.12.1 **问题1：PDCCH 把所有的UE 的调度交织在一起？那UE 怎么解呢？怎么知道哪些地方插入的<NIL>最后又被打掉了呢？**

### 4.4.8 PDSCH

#### 4.4.8.1 作用

- 传送上层的控制信息（RRC 层，MAC 层）或者下行业务数据，不过对于物理层来看，它们都是数据。

#### 4.4.8.2 参看

- 参看：36212 的 5.3.2； 36211 的 6.4，36213 的

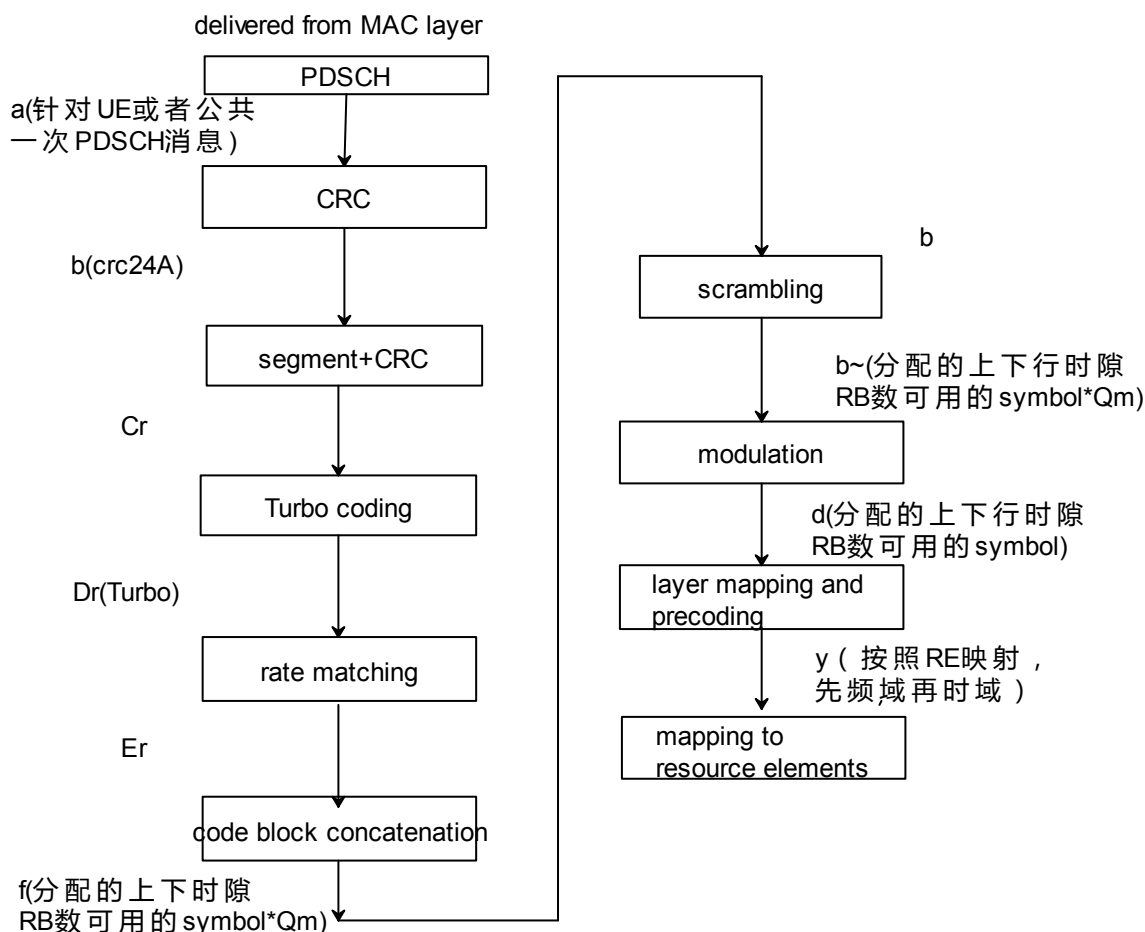
#### 4.4.8.3 数据来源

- 上层：调度决定给某个 UE 分配多少 RB pair，然后通过 CQI 得到判断需要的  $I_{\text{MCS}}$ ，通过  $I_{\text{MCS}}$  查表 36213---Table 7.1.7.1-1 就能得到‘调制模式， $I_{\text{TBS}}$ ’，然后再‘用  $I_{\text{TBS}}$  分配的 RB 数，通过 Table 7.1.7.2.1-1 得到最好 MAC 层的数据包得大小’----MAC 数据包的大小也就是输入了。

注：如果是在‘特殊子帧’，由于下行没有占满整个 TTI，所以 PRB 数目的计算不能满算，必须有个缩减  $N_{\text{PRB}} = \max \left\{ \left\lfloor N'_{\text{PRB}} \times 0.75 \right\rfloor, 1 \right\}$ ，这也就是 36213 的 7.1.7 中所描述的。

- 从前面‘DCI 格式’的介绍来看，对于 RA-RNTI,P-RNTI,SI-RNTI，TC-RNTI 的处理要特殊一些。

#### 4.4.8.4 流程



- **速率匹配：**下行 PDSCH 此时速率匹配的时候和 RVidx 有关了，由 DCI 里面的 Redundancy version 决定。
- **加扰：**

$$c_{init} = n_{RNTI} \cdot 2^{14} + q \cdot 2^{13} + \lfloor n_s/2 \rfloor \cdot 2^9 + N_{ID}^{cell}$$
- **调制：**由 DCI 里面的 MCS 决定

**Table 6.3.2-1: Modulation schemes**

Physical channel	Modulation schemes
PDSCH	QPSK, 16QAM, 64QAM
PMCH	QPSK, 16QAM, 64QAM

- 层映射是通过配置决定多少层的。预编码最后映射的天线 port。如果没有针对 ue 特定的参考信号，就跟 PBCH 一样的 port 发送；否则在 port5 上发送。

#### 4.4.8.5 时频位置

- 分配的 PRB 上，按照 RE 进行排列，先频域后时域。

4.4.9 下行信道的总结

信道名称	CRC	Seg+CRC	信道编码	速率匹配	串联	加扰	调制	层映射	预编码	资源映射
PBCH	CRC16		convolutional	1920bit		$c_{init} = N_{ID}^{cell}$	QPSK	层数和天线数一样	发射分集	单位是RE
PCFICH			固定编码格式， 参考36212的Table 5.3.4	32bit		$c_{init} = (\lfloor n_s/2 \rfloor + 1) \cdot (2N_{ID}^{cell} + 1) \cdot 2^9 + N_{ID}^{cell}$	QPSK	层数和天线数一样	发射分集	单位是REG，4个REG
PHICH			固定编码格式， 参考36212的Table 5.3.5-1	3bit		$c_{init} = (\lfloor n_s/2 \rfloor + 1) \cdot (2N_{ID}^{cell} + 1) \cdot 2^9 + N_{ID}^{cell}$	BPSK	层数和天线数一样	发射分集	单位是REG，3个REG
PDCCH	CRC16		convolutional	1,2,4,8 CCE		$c_{init} = \lfloor n_s/2 \rfloor 2^9 + N_{ID}^{cell}$	QPSK	层数和天线数一样	发射分集	单位是CCE
PDSCH	CRC24A	有	turbo	分配的RB有效的RE*Qm	有	$c_{init} = n_{RNTI} \cdot 2^{14} + q \cdot 2^{13} + \lfloor n_s/2 \rfloor \cdot 2^9 + N_{ID}^{cell}$	QPSK/16QAM/64QAM DCI中的MCS指定	和TM模式有关	和TM模式有关	RE