

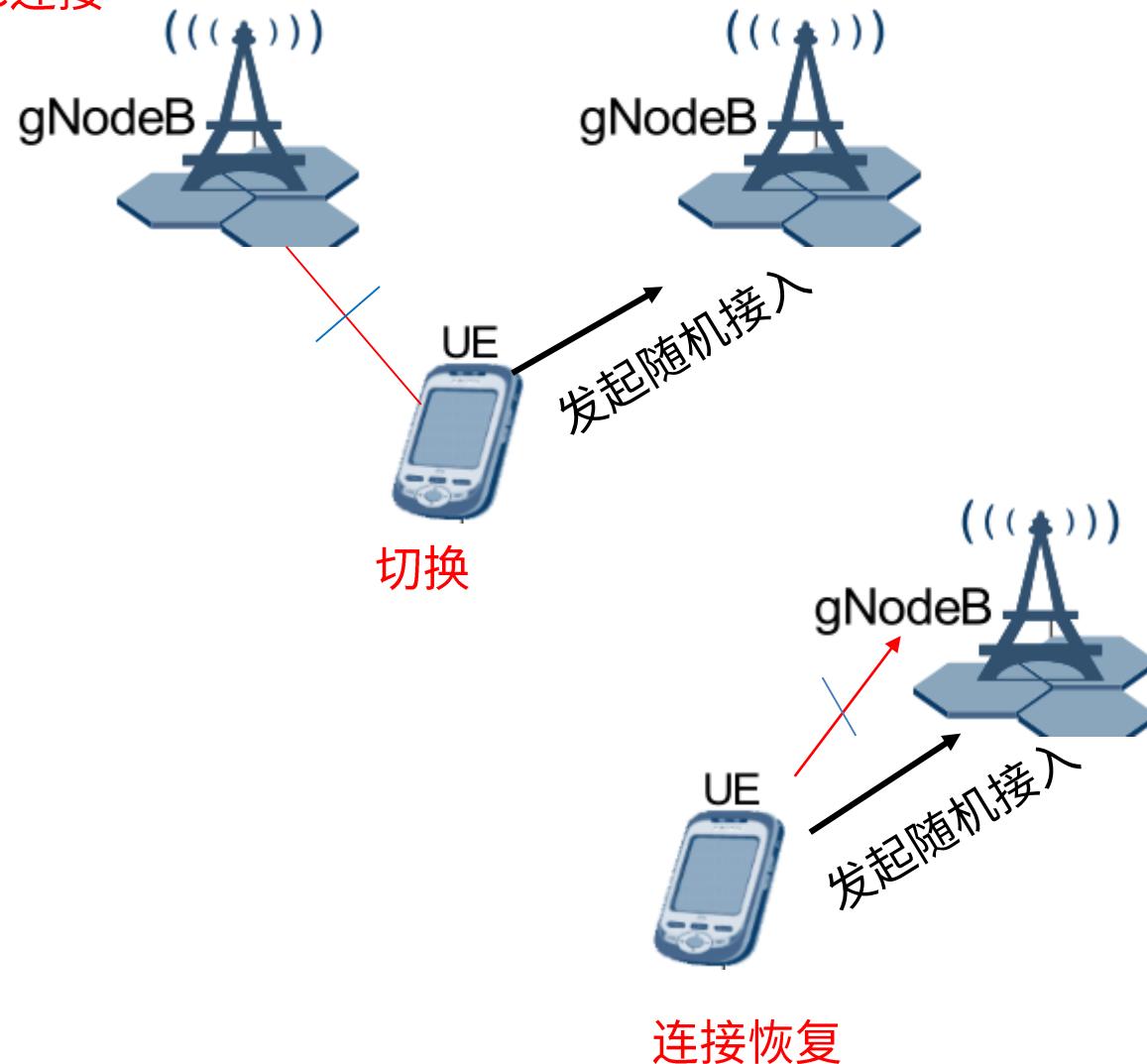
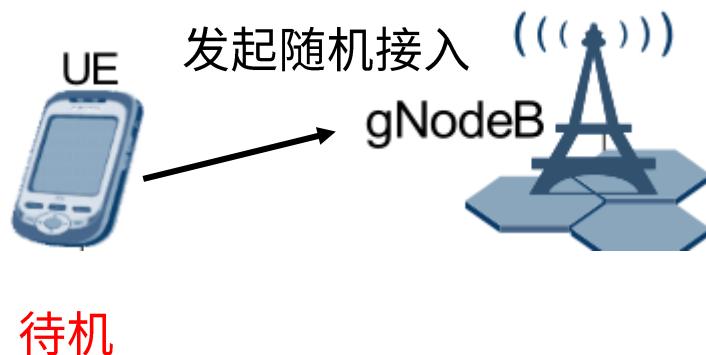
5G PRACH

讲师：捻叶成剑

再次简单介绍随机接入

随机接入主要用于建立或恢复UE与网络侧的RRC连接
(连接建立, 切换, 连接恢复)

关于随机接入, 后面有专门一次课完整讲解。



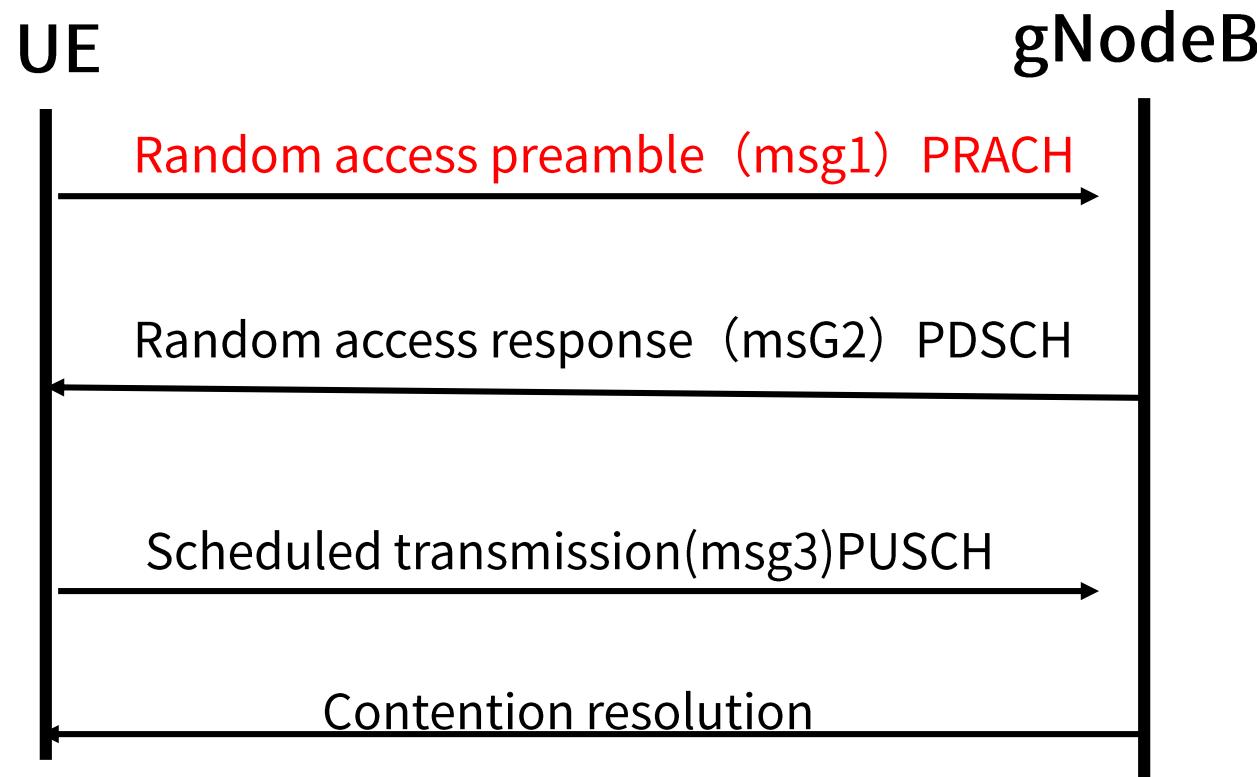
随机接入第一步

随机接入的第一步，就是在PRACH信道上，发送preamble（前导序列），这个信令也称为为消息1（MSG1）

这个步骤有如下两个目的：

触发基站给UE调度上下行空口资源

基站评估UE上行时间延时，从而给UE授予时间提前量（TA），另UE完成上行同步。



PRACH

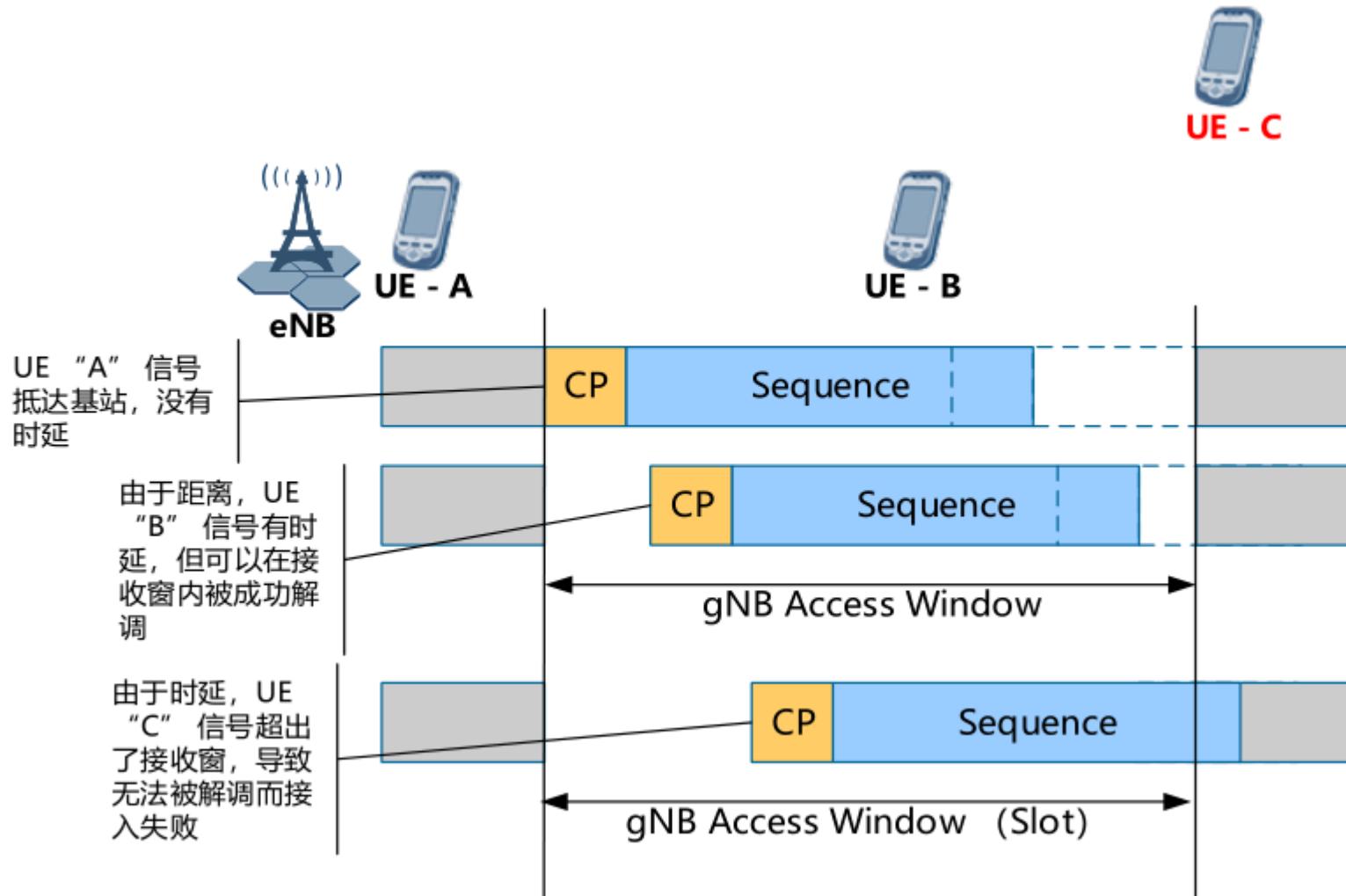
PRACH: Physical Random Access Channel 物理随机接入信道

作用：传输随机接入前导序列（preamble）

PRACH由CP（循环前缀）、前导序列（preamble）和保护间隔组成。



保护间隔有啥用？



Preamble序列的生成

Preamble序列

Preamble序列使用Zadoff-Chu根序列循环位移产生，总共有64个

5G里面支持两种长度的序列，分别是L=139和L=839，使用的长度由站点定义的preamble Format决定

为了搞清楚这句话，我们先看看“**序列**”这个词的意思

序列有很多种：

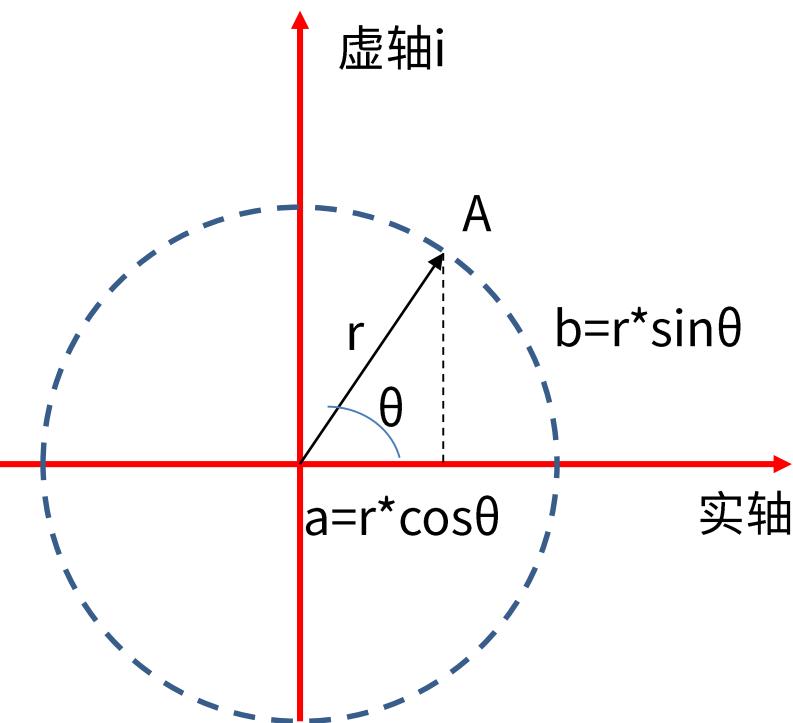
二进制序列：01011101111001010

整数序列：12345154225456475

复数序列：复数序列是平面坐标中的一个个的点组成的序列

欧拉复数序列

先看图



A的位置，用复数形式来表示，就是：

$$a + bi = r\cos\theta + ir\sin\theta = r(\cos\theta + i\sin\theta) = re^{i\theta}$$

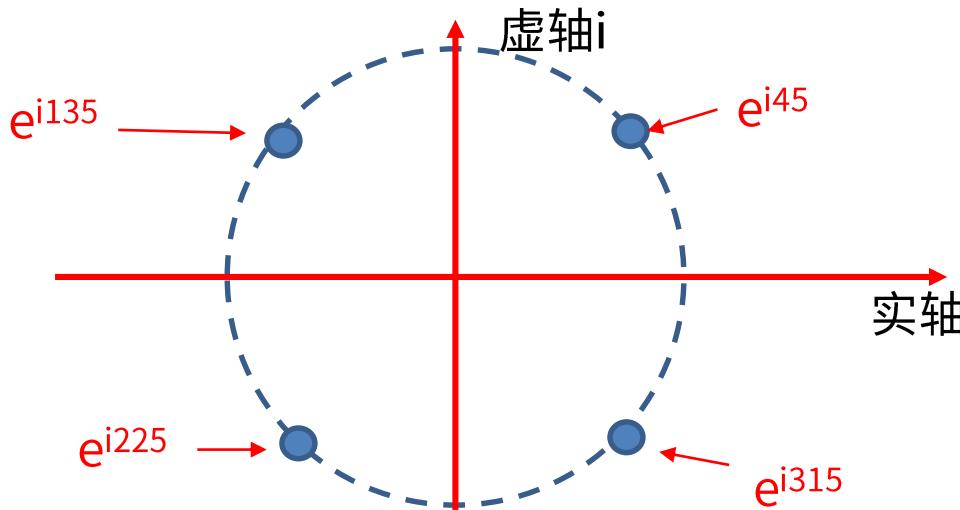
r : 模 θ : 幅角

$$\text{欧拉公式 } e^{i\theta} = \cos\theta + i\sin\theta$$

这个公式里，如果R不变，只变化θ，那么，从图形上看，就是得到了圆上的一系列点。

基于此，我们以欧拉公式，可以形成一个序列公式：欧拉复数序列= $re^{i\theta}$
比如，我现在取 $\theta=45^\circ, 135^\circ, 225^\circ, 315^\circ$, $r=1$

这样，我就得到了一个复数序列： e^{i45} 、 e^{i135} 、 e^{i225} 、 e^{i315}



ZC序列

Zadoff - Chu序列，顾名思义，就是Zadoff 和Chu两个人共同发现的序列

ZC序列本质上就是欧拉复数序列， ZC序列的生成公式

$$x_u(i) = e^{-j \frac{\pi u i (i+1)}{L_{RA}}}, i = 0, 1, \dots, L_{RA} - 1$$

欧拉复数序列 = $r e^{i\theta}$

$L_{RA}=139$ 或者 839

ZC序列就是把欧拉复数序列公式赋值， $r=1$,

$$\Theta = \frac{\pi u i (i+1)}{L_{RA}}, i = -j$$

其中， $-j$ 表达的意思是圆上的点取值是顺时针的。

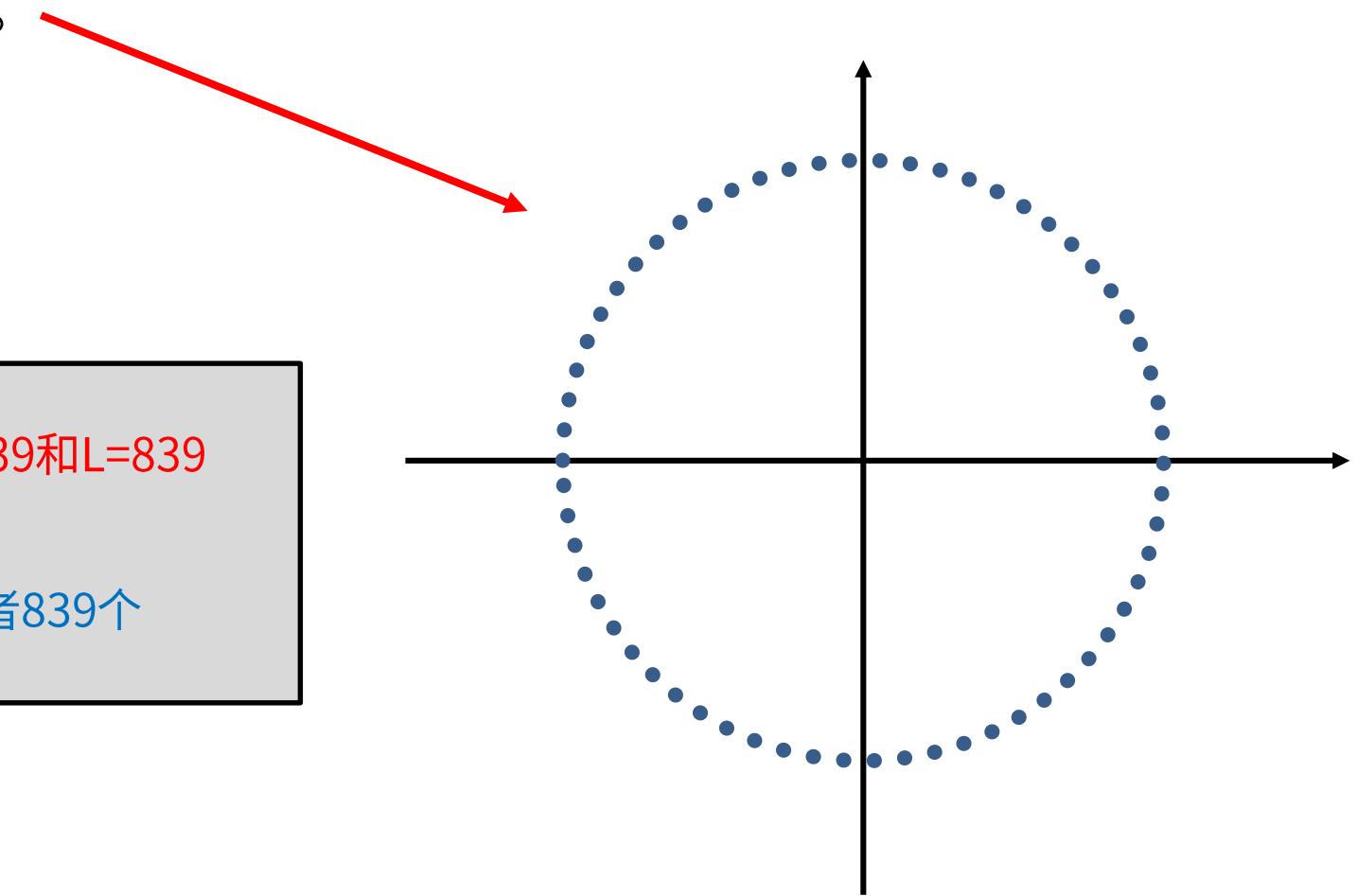
ZC序列

所以说，一个ZC序列，如果用数字来表示，就是一串 $e^{-j\theta}$ ，其中 θ 的数值不同

如果用图来表示，就是圆上的一堆点。

5G里面支持两种长度的序列，分别是 $L=139$ 和 $L=839$

这句话的意思就是，点的数量是139个或者839个



ZC序列

$$x_u(i) = e^{-j \frac{\pi u i (i+1)}{L_{\text{RA}}}}, i = 0, 1, \dots, L_{\text{RA}} - 1$$

$$L_{\text{RA}}=139 \text{或者} 839$$

这个公式里面， L_{RA} 和 i 都有，只有 u 是未知数了，因此，知道了 u ，就得到了一个ZC序列。

5G系统中定义了一个参数：根序列索引 i (PRACHRootSequenceIndex i)，每一个 i 对应了一个 u ，也就是说，当根序列索引 i 被定义的时候，我们就得到了 u 是多少，也就得到了一个根序列！

由于preamble是由ZC序列生成的，因此，ZC序列也叫做preamble的根序列

根序列索引*i*与*u*

$$L_{RA} = 839$$

PRACHRootSequenceIndex in SIB

$$x_u(i) = e^{-j\frac{\pi u i(i+1)}{L_{RA}}}, i = 0, 1, \dots, L_{RA} - 1$$



<i>i</i>	Sequence number <i>u</i> in increasing order of <i>i</i>																			
0 – 19	129	710	140	699	120	719	210	629	168	671	84	755	105	734	93	746	70	769	60	779
20 – 39	2	837	1	838	56	783	112	727	148	691	80	759	42	797	40	799	35	804	73	766
40 – 59	146	693	31	808	28	811	30	809	27	812	29	810	24	815	48	791	68	771	74	765
60 – 79	178	661	136	703	86	753	78	761	43	796	39	800	20	819	21	818	95	744	202	637
80 – 99	190	649	181	658	137	702	125	714	151	688	217	622	128	711	142	697	122	717	203	636
100 – 119	118	721	110	729	89	750	103	736	61	778	55	784	15	824	14	825	12	827	23	816
120 – 139	34	805	37	802	46	793	207	632	179	660	145	694	130	709	223	616	228	611	227	612
140 – 159	132	707	133	706	143	696	135	704	161	678	201	638	173	666	106	733	83	756	91	748
160 – 179	66	773	53	786	10	829	9	830	7	832	8	831	16	823	47	792	64	775	57	782
180 – 199	104	735	101	738	108	731	208	631	184	655	197	642	191	648	121	718	141	698	149	690
200 – 219	216	623	218	621	152	687	144	695	134	705	138	701	199	640	162	677	176	663	119	720
220 – 239	158	681	164	675	174	665	171	668	170	669	87	752	169	670	88	751	107	732	81	758
240 – 259	82	757	100	739	98	741	71	768	59	780	65	774	50	789	49	790	26	813	17	822
260 – 279	13	826	6	833	5	834	33	806	51	788	75	764	99	740	96	743	97	742	166	673
280 – 299	172	667	175	664	187	652	163	676	185	654	200	639	114	725	189	650	115	724	194	645
300 – 319	195	644	192	647	182	657	157	682	156	683	211	628	154	685	123	716	139	700	212	627
320 – 339	153	686	213	626	215	624	150	689	225	614	224	615	221	618	220	619	127	712	147	692

根序列索引与u

$$L_{\text{RA}} = 139$$

PRACHRootSequenceIndex in SIB

$$x_u(i) = e^{-j\frac{\pi u i (i+1)}{L_{RA}}}, \quad i = 0, 1, \dots, L_{RA} - 1$$

UE如何获取根序列索引

通过SIB1里面的RACH-configcommon中的PRACH-rootsequenceindex

```
RACH-ConfigCommon ::= SEQUENCE {
    rach-ConfigGeneric           RACH-ConfigGeneric,
    totalNumberOfRA-Preambles    INTEGER (1..63)          OPTIONAL, -- Need S
    ssb-perRACH-OccasionAndCB-PreamblesPerSSB CHOICE {
        oneEighth   ENUMERATED {n4,n8,n12,n16,n20,n24,n28,n32,n36,n40,n44,n48,n52,n56,n60,n64},
        oneFourth   ENUMERATED {n4,n8,n12,n16,n20,n24,n28,n32,n36,n40,n44,n48,n52,n56,n60,n64},
        oneHalf     ENUMERATED {n4,n8,n12,n16,n20,n24,n28,n32,n36,n40,n44,n48,n52,n56,n60,n64},
        one         ENUMERATED {n4,n8,n12,n16,n20,n24,n28,n32,n36,n40,n44,n48,n52,n56,n60,n64},
        two         ENUMERATED {n4,n8,n12,n16,n20,n24,n28,n32},
        four        INTEGER (1..16),
        eight       INTEGER (1..8),
        sixteen     INTEGER (1..4)
    } OPTIONAL, -- Need M

    groupBconfigured      SEQUENCE {
        ra-Msg3SizeGroupA   ENUMERATED {b56, b144, b208, b256, b282, b480, b640, b800, b1000,
                                         spare7, spare6, spare5, spare4, spare3, spare2, spare1},
        messagePowerOffsetGroupB ENUMERATED {minusinfinity, dB0, dB5, dB8, dB10, dB12,
                                              dB15, dB18},
        numberOfRA-PreamblesGroupA   INTEGER (1..64)
    } OPTIONAL, -- Need R

    ra-ContentionResolutionTimer ENUMERATED {sf8, sf16, sf24, sf32, sf40, sf48, sf56, sf64},
    rsrp-ThresholdSSB           RSRP-Range           OPTIONAL, -- Need R
    rsrp-ThresholdSSB-SUL       RSRP-Range           OPTIONAL, -- Need R
    prach-RootSequenceIndex     CHOICE {
        1839                 INTEGER (0..837),
        1139                 INTEGER (0..137)
    },
    msg1-SubcarriersSpacing     SubcarriersSpacing,
    restrictedSetConfig        ENUMERATED {unrestrictedset, restrictedsetTypeA, restrictedsetTypeB},
    msg3-transformPrecoding    ENUMERATED {enabled} OPTIONAL, -- Need R
```

Preamble的产生

Preamble序列使用Zadoff-Chu根序列**循环位移**产生

也就是说，preamble序列并不是直接使用ZC序列，而是以**ZC序列为基础**，然后进行**位移**而得到。

接下来，我们研究preamble到底如何通过位移ZC序列，而得到。

协议中定义了**循环位移**的公式：

$$x_{u,v}(n) = x_u((n + C_v) \bmod L_{\text{RA}})$$

公式解释：

- preamble序列 $x_{u,v}(n)$ 由 ZC 序列 x_u 通过循环位移生成。
- ZC 序列 x_u 的序号 n 偏差 C_v 位。
- 循环位移 C_v ：表示偏移量。
- 模运算 \bmod 表示计算两数相除的余数。
- 参数 n 的取值范围为 $0, 1, 2, 3 \dots$

公式的意思是说，preamble的序列号，与ZC序列的序号偏差 Cv 位

\bmod 是计算两数相除的余数

循环位移Cv举个例子

假设，ZC序列是 $e^0, e^1, e^2, e^3, e^4, e^5$ ，那么 $L_{RA}=6$

$$x_{u,v}(n) = x_u((n + C_v) \bmod L_{RA})$$

也就是说， $x_u(0) = e^0, x_u(1) = e^1, x_u(2) = e^2 \dots$

假设 Cv=2

那么基于这个位移，计算的preamble序列如下：

$$n=0, x_{u,v}(0)=x_u((0+2)\bmod 6)=x_u(2)=e^2$$

$$n=1, x_{u,v}(1)=x_u((1+2)\bmod 6)=x_u(3)=e^3$$

$$n=2, x_{u,v}(2)=x_u((2+2)\bmod 6)=x_u(4)=e^4$$

$$n=3, x_{u,v}(3)=x_u((3+2)\bmod 6)=x_u(5)=e^5$$

$$n=4, x_{u,v}(4)=x_u((4+2)\bmod 6)=x_u(0)=e^0$$

$$n=5, x_{u,v}(5)=x_u((5+2)\bmod 6)=x_u(1)=e^1$$

既preamble序列为： $e^2, e^3, e^4, e^5, e^0, e^1$

因此，一个Cv就可以得到一个preamble序列

循环位移参数C_v

vN_{CS}	$v = 0, 1, \dots, \lfloor L_{RA}/N_{CS} \rfloor - 1, N_{CS} \neq 0$	for unrestricted sets	非限制集
0	$N_{CS} = 0$	for unrestricted sets	
$\bar{d}_{start} \lfloor v/n_{shift}^{RA} \rfloor + (v \bmod n_{shift}^{RA})N_{CS}$	$v = 0, 1, \dots, w - 1$	for restricted sets type A and B	限制集
$\bar{\bar{d}}_{start} + (v - w)N_{CS}$	$v = w, \dots, w + \bar{n}_{shift}^{RA} - 1$	for restricted sets type B	
$\bar{\bar{\bar{d}}}_{start} + (v - w - \bar{\bar{n}}_{shift}^{RA})N_{CS}$	$v = w + \bar{n}_{shift}^{RA}, \dots, w + \bar{\bar{n}}_{shift}^{RA} + \bar{\bar{\bar{n}}}_{shift}^{RA} - 1$	for restricted sets type B	

$$w = n_{shift}^{RA}n_{group}^{RA} + \bar{n}_{shift}^{RA}$$

使用**非限制集与限制集**，主要取决于UE的移动速度，UE的高速运动，会产生多普勒频移，进而会影响PRACH的检测。

非限制集：当用户静止或者低速运动场景（**目前现网主流的参数配置**）因为要使用位移，因此 $C_v = vN_{CS}$

限制集：在用户高速移动场景下，针对不同根序列，限制某些循环移位，来规避频移，保证接入成功率。

限制集又分为**类型A**和**类型B**，区别是**支持的最大频移不同**，类型A：<1SCS，类型B：1SCS到2SCS之间。换句话说，**A支持高速移动场景，而B支持超高速场景**。

限制集与非限制集的信令设置

SIB1里面的RACH-ConfigCommon 中，有restrictedSetConfig字段：
unrestricted, restricted type A, restricted type B

```
RACH-ConfigCommon ::= SEQUENCE {
    rach-ConfigGeneric          RACH-ConfigGeneric,
    totalNumberOfRA-Preambles   INTEGER (1..63)           OPTIONAL, -- Need S
    ssb-perRACH-OccasionAndCB-PreamblesPerSSB CHOICE {
        oneEighth    ENUMERATED {n4,n8,n12,n16,n20,n24,n28,n32,n36,n40,n44,n48,n52,n56,n60,n64},
        oneFourth   ENUMERATED {n4,n8,n12,n16,n20,n24,n28,n32,n36,n40,n44,n48,n52,n56,n60,n64},
        oneHalf     ENUMERATED {n4,n8,n12,n16,n20,n24,n28,n32,n36,n40,n44,n48,n52,n56,n60,n64},
        one         ENUMERATED {n4,n8,n12,n16,n20,n24,n28,n32,n36,n40,n44,n48,n52,n56,n60,n64},
        two         ENUMERATED {n4,n8,n12,n16,n20,n24,n28,n32},
        four        INTEGER (1..16),
        eight       INTEGER (1..8),
        sixteen    INTEGER (1..4)
    } OPTIONAL, -- Need M

    groupBconfigured      SEQUENCE {
        ra-Msg3SizeGroupA   ENUMERATED {b56, b144, b208, b256, b282, b480, b640, b800, b1000,
                                         spare7, spare6, spare5, spare4, spare3, spare2, spare1},
        messagePowerOffsetGroupB ENUMERATED {minusinfinity, dB0, dB5, dB8, dB10, dB12,
                                              dB15, dB18},
        numberofRA-PreamblesGroupA  INTEGER (1..64)
    } OPTIONAL, -- Need R

    ra-ContentionResolutionTimer ENUMERATED {sf8, sf16, sf24, sf32, sf40, sf48, sf56, sf64},
    rsrp-ThresholdSSB           RSRP-Range           OPTIONAL, -- Need R
    rsrp-ThresholdSSB-SUL       RSRP-Range           OPTIONAL, -- Need R
    prach-RootSequenceIndex     CHOICE {
        1839                INTEGER (0..837),
        1139                INTEGER (0..137)
    },
    msg1-SubcarrierSpacing     SubcarrierSpacing,
    restrictedSetConfig        ENUMERATED {unrestrictedSet, restrictedSetTypeA, restrictedSetTypeB},
    msg3-transformPrecoding    ENUMERATED {enabled} OPTIONAL, -- Need R
    ...
}
```

N_{CS}

C_v=vN_{CS}公式当中的N_{CS}, 取决于zeroCorrelationZoneConfig参数设置以及限制集A,B或者非限制集

```
RACH-ConfigGeneric ::= SEQUENCE {  
    prach-ConfigurationIndex      INTEGER (0..255),  
    msg1-FDM                      ENUMERATED {one, two, four, eight},  
    msg1-FrequencyStart           INTEGER (0..maxNrofPhysicalResourceBlocks-1),  
    zeroCorrelationZoneConfig     INTEGER(0..15),  
    preambleReceivedTargetPower   INTEGER (-200..-74),  
    preambleTransMax              ENUMERATED {n3,n4,n5,n6,n7,n8,n10,n20,n50,n100,n200},  
    powerRampingStep              ENUMERATED {dB0, dB2, dB4, dB6},  
    ra-Responsewindow             ENUMERATED {s11, s12, s14, s18, s110, s120, s140, s180}  
}
```

Table 6.3.3.1-5: N_{CS} for preamble formats with Δf^{RA} = 1.25 kHz.

zeroCorrelationZoneConfig	N _{CS} value		
	Unrestricted set	Restricted set type A	Restricted set type B
0	0	15	15
1	13	18	18
2	15	22	22
3	18	26	26
4	22	32	32
5	26	38	38
6	32	46	46
7	38	55	55
8	46	68	68
9	59	82	82
10	76	100	100
11	93	128	118
12	119	158	137
13	167	202	-
14	279	237	-
15	419	-	-

我们知道5G正常的子载波间隔最低是15KHz,
而这里的子载波叫做RACH子载波 (RA-SCS)

N_{CS} -----RA子载波间隔5KHZ

Table 6.3.3.1-6: N_{CS} for preamble formats with $\Delta f^{RA} = 5 \text{ kHz}$.

$zeroCorrelationZoneConfig$	N_{CS} value		
	Unrestricted set	Restricted set type A	Restricted set type B
0	0	36	36
1	13	57	57
2	26	72	60
3	33	81	63
4	38	89	65
5	41	94	68
6	49	103	71
7	55	112	77
8	64	121	81
9	76	132	85
10	93	137	97
11	119	152	109
12	139	173	122
13	209	195	137
14	279	216	-
15	419	237	-

N_{CS} ----RA子载波间 $15 \cdot 2^\mu$ KHZ

Table 6.3.3.1-7: N_{CS} for preamble formats with $\Delta f^{RA} = 15 \cdot 2^\mu$ kHz where $\mu \in \{0,1,2,3\}$.

<u>zeroCorrelationZoneConfig</u>	N_{CS} value↓ for unrestricted set
0	0
1	2
2	4
3	6
4	8
5	10
6	12
7	13
8	15
9	17
10	19
11	23
12	27
13	34
14	46
15	69

Preamble生成方法 (非限制集)

ZC根序列 $x_u(i) = e^{-j\frac{\pi u i (i+1)}{L_{RA}}}, i = 0, 1, \dots, L_{RA} - 1$ PRACHRootSequenceIndex i --索引出 u

PRACHRootSequenceIndex in SIB

i	Sequence number u in increasing order of i																			
0 - 19	129	710	140	699	120	719	210	629	168	671	84	755	105	734	93	746	70	769	60	779
20 - 39	2	837	1	838	56	783	112	727	148	691	80	759	42	797	40	799	35	804	73	766
40 - 59	146	693	31	808	28	811	30	809	27	812	29	810	24	815	48	791	68	771	74	765

第一步：一个根序列索引就得到一个ZC根序列

$$x_{u,v}(n) = x_u((n + C_v) \bmod L_{RA})$$

第二步：一个根序列+位移得出preamble序列

循环位移参数 C_v $C_v = v N_{CS}$ $v = 0, 1, \dots, \lfloor L_{RA} / N_{CS} \rfloor - 1, N_{CS} \neq 0$

N_{CS} 取决于 zeroCorrelationZoneConfig

Table 6.3.3.1-5: N_{CS} for preamble formats with $\Delta f^{RA} = 1.25$ kHz.

zeroCorrelationZoneConfig	N_{CS} value		
	Unrestricted set	Restricted set type A	Restricted set type B
0	0	15	15
1	13	18	18
2	15	22	22
3	18	26	26

一个 C_v 得到一个一个 preamble 序列，总共可以生成 L_{RA} / N_{CS} 个 preamble。

Preamble生成方法举例

对于一个根序列，根据循环位移后生成的preamble码个数为 L_{RA}/N_{cs} ,如果小于64时，则逻辑根序列索引号+1后，继续通过循环位移生成preamble，直至满足64个preamble码为止！

以 $L_{RA} = 839$ 为例子

根序列索引 PRACHRootSequenceIndex $i=20$, 查表得知, $U=2$

PRACHRootSequenceIndex in SIB

$$x_u(i) = e^{-j \frac{\pi u i (i+1)}{L_{RA}}}, i = 0, 1, \dots, L_{RA} - 1$$

i	Sequence number u in increasing order of i																			
0 - 19	129	710	140	699	120	719	210	629	168	671	84	755	105	734	93	746	70	769	60	779
20 - 39	21	837	1	838	56	783	112	727	148	691	80	759	42	797	40	799	35	804	73	766
40 - 59	146	693	31	808	28	811	30	809	27	812	29	810	24	815	48	791	68	771	74	765

Preamble生成方法举例

假设：非限制集，zeroCorrelationZoneConfig=6, RA-SCS=1.25KHZ

Table 6.3.3.1-5: N_{CS} for preamble formats with $\Delta f^{RA} = 1.25 \text{ kHz}$.

zeroCorrelationZoneConfig	Unrestricted set	N_{CS} value	Restricted set type A	Restricted set type B
0	0	15	15	15
1	13	18	18	18
2	15	22	22	22
3	18	26	26	26
4	22	32	32	32
5	26	38	38	38
6	32	46	46	46
7	38	55	55	55
8	46	68	68	68
9	59	82	82	82
10	76	100	100	100
11	93	128	118	118
12	119	158	137	137
13	167	202	-	-
14	279	237	-	-
15	419	-	-	-

查表得到 $N_{CS}=32$

$$L_{RA}/N_{CS}=839/32 \approx 26$$

$$v = 0, 1, \dots, \lfloor L_{RA}/N_{CS} \rfloor - 1, N_{CS} \neq 0$$

$$v=0,1,2,\dots,25$$

$$C_v=vN_{CS}$$

$$C_v=0, 32, 64, \dots, 800$$

由于 C_v 只有 26 个，也就是说只能生成 26 个 preamble 序列，因此，需要根序列号 +1，继续计算生成 preamble 序列
序列索引 PRACHRootSequenceIndex $i=21$ ，查表得知， $U=837$ ，然后继续生成 26 个 preamble，
再用下一个根序列继续，直至生成 64 个 preamble 序列为止！

Preamble format

Preamble format

Preamble序列使用Zadoff-Chu根序列循环位移产生，总共有64个

5G里面支持两种长度的序列，分别是 $L=139$ 和 $L=839$ ，使用的长度由站点定义的preamble Format决定

接下来，我们第6页的这句话，只剩最后一个问题，就是preamble format（前导格式）

5G 前导格式分为两大类，第一类格式就是 $L=839$ 的长格式，第二类格式就是 $L=139$ 的短格式。

前导格式（长格式）

5G支持4种长度为839的preamble序列格式，分别为format0/1/2/3.支持子载波间隔为1.25KHZ和5KHZ。仅用于FR1频段。支持非限制集，限制集A和限制集B。

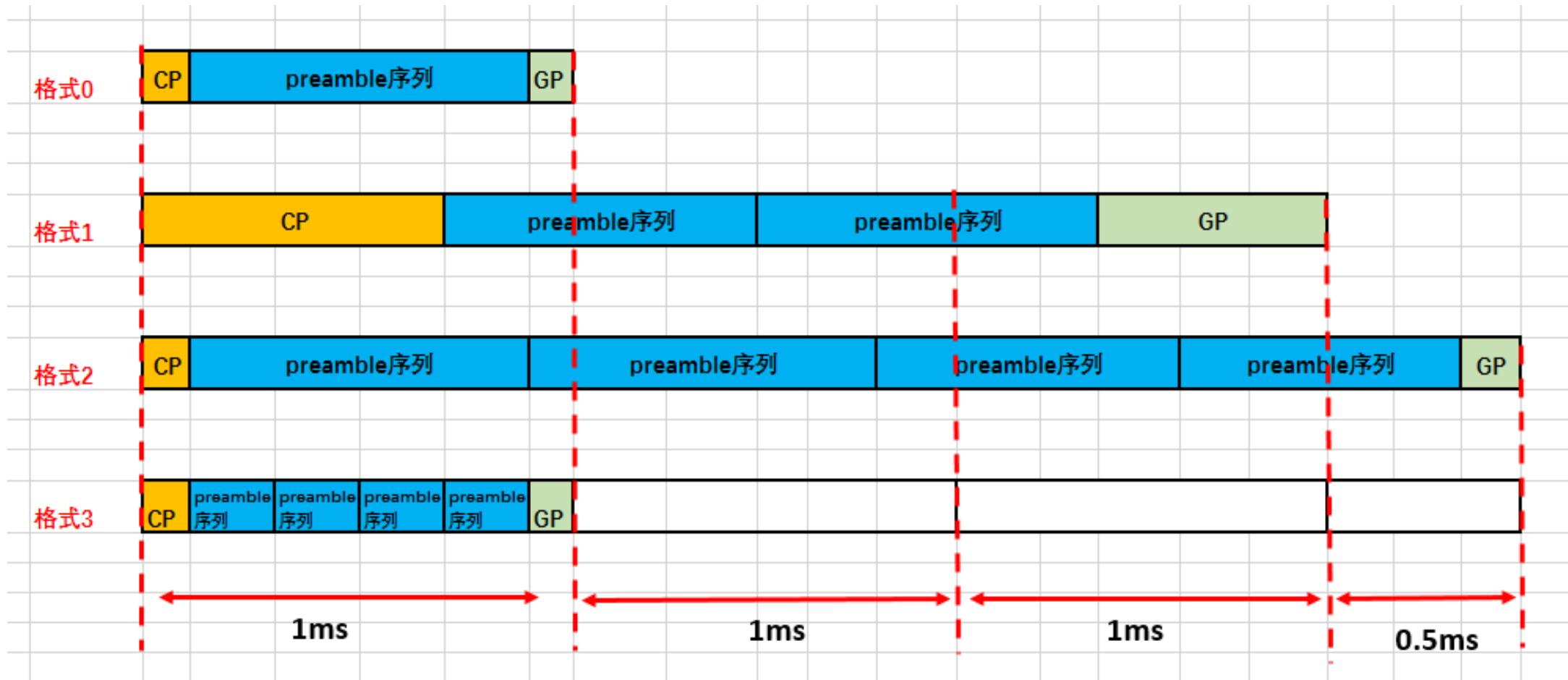
长格式的RACH子载波间隔直接和Format格式相对应，无需另外配置。

Format	序列长度	子载波间隔	时域总长	占用带宽	最大小区半径	典型场景
0	839	1.25 kHz	1.0 ms	1.08 MHz	14.5 km	常规半径
1	839	1.25 kHz	3.0 ms	1.08 MHz	100.1 km	超远覆盖
2	839	1.25 kHz	3.5 ms	1.08 MHz	21.9 km	增强覆盖
3	839	5.0 kHz	1.0 ms	4.32 MHz	14.5 km	超高速

其中，长格式里面的format3，非常适合高铁优化场景。

前导格式 (长格式)

Preamble format 0/1/2/3 所包含的 preamble 个数分别为 1,2,4,4. 并且是同一个序列的重复



前导格式（短格式）

5G NR支持9种长度为139的preamble格式，支持RACH子载波间隔（RA-SCS）为15,30,60,120KHZ
(FR1时支持15KHZ和30KHZ, FR2时支持60KHZ和120KHZ) , 短格式仅支持非限制集。

现网主要用短格式C2

Format	序列长度	子载波间隔	时域总长	占用带宽	最大小区半径	典型场景
A1	139	$15 \cdot 2^{\mu} (\mu=0/1/2/3)$ 3)	$0.14/2^{\mu}$ ms	$2.16 \cdot 2^{\mu}$ MHz	$0.937/2^{\mu}$ km	small cell
A2	139	$15 \cdot 2^{\mu}$	$0.29/2^{\mu}$ ms	$2.16 \cdot 2^{\mu}$ MHz	$2.109/2^{\mu}$ km	Normal cell
A3	139	$15 \cdot 2^{\mu}$	$0.43/2^{\mu}$ ms	$2.16 \cdot 2^{\mu}$ MHz	$3.515/2^{\mu}$ km	Normal cell
B1	139	$15 \cdot 2^{\mu}$	$0.14/2^{\mu}$ ms	$2.16 \cdot 2^{\mu}$ MHz	$0.585/2^{\mu}$ km	small cell
B2	139	$15 \cdot 2^{\mu}$	$0.29/2^{\mu}$ ms	$2.16 \cdot 2^{\mu}$ MHz	$1.054/2^{\mu}$ km	Normal cell
B3	139	$15 \cdot 2^{\mu}$	$0.43/2^{\mu}$ ms	$2.16 \cdot 2^{\mu}$ MHz	$1.757/2^{\mu}$ km	Normal cell
B4	139	$15 \cdot 2^{\mu}$	$0.86/2^{\mu}$ ms	$2.16 \cdot 2^{\mu}$ MHz	$3.867/2^{\mu}$ km	Normal cell
C0	139	$15 \cdot 2^{\mu}$	$0.14/2^{\mu}$ ms	$2.16 \cdot 2^{\mu}$ MHz	$5.351/2^{\mu}$ km	Normal Cell
C2	139	$15 \cdot 2^{\mu}$	$0.43/2^{\mu}$ ms	$2.16 \cdot 2^{\mu}$ MHz	$9.297/2^{\mu}$ km	Normal Cell

这里面，B2 和B3不能单独使用，，只能与A2和A3组合使用，组合成A2/B2、A3/B3，另外，B1可以单独使用，也可以与A1组合成A1/B1使用

短格式的RACH子载波间隔通过信令配置SIB1--RACH-ConfigCommon --msg1-SubcarrierSpacing

RA-SCS (短格式)

SIB1--RACH-ConfigCommon --msg1-SubcarrierSpacing

```
RACH-ConfigCommon ::= SEQUENCE {
    rach-ConfigGeneric           RACH-ConfigGeneric,
    totalNumberOfRA-Preambles    INTEGER (1..63)           OPTIONAL,   -- Need S
    ssb-perRACH-OccasionAndCB-PreamblesPerSSB CHOICE {
        oneEighth    ENUMERATED {n4,n8,n12,n16,n20,n24,n28,n32,n36,n40,n44,n48,n52,n56,n60,n64},
        oneFourth    ENUMERATED {n4,n8,n12,n16,n20,n24,n28,n32,n36,n40,n44,n48,n52,n56,n60,n64},
        oneHalf      ENUMERATED {n4,n8,n12,n16,n20,n24,n28,n32,n36,n40,n44,n48,n52,n56,n60,n64},
        one         ENUMERATED {n4,n8,n12,n16,n20,n24,n28,n32,n36,n40,n44,n48,n52,n56,n60,n64},
        two          ENUMERATED {n4,n8,n12,n16,n20,n24,n28,n32},
        four         INTEGER (1..16),
        eight        INTEGER (1..8),
        sixteen      INTEGER (1..4)
    } OPTIONAL,   -- Need M

    groupBconfigured     SEQUENCE {
        ra-Msg3SizeGroupA   ENUMERATED {b56, b144, b208, b256, b282, b480, b640, b800, b1000,
                                         spare7, spare6, spare5, spare4, spare3, spare2, spare1},
        messagePowerOffsetGroupB ENUMERATED {minusinfinity, dB0, dB5, dB8, dB10, dB12,
                                              dB15, dB18},
        numberofRA-PreamblesGroupA  INTEGER (1..64)
    } OPTIONAL,   -- Need R

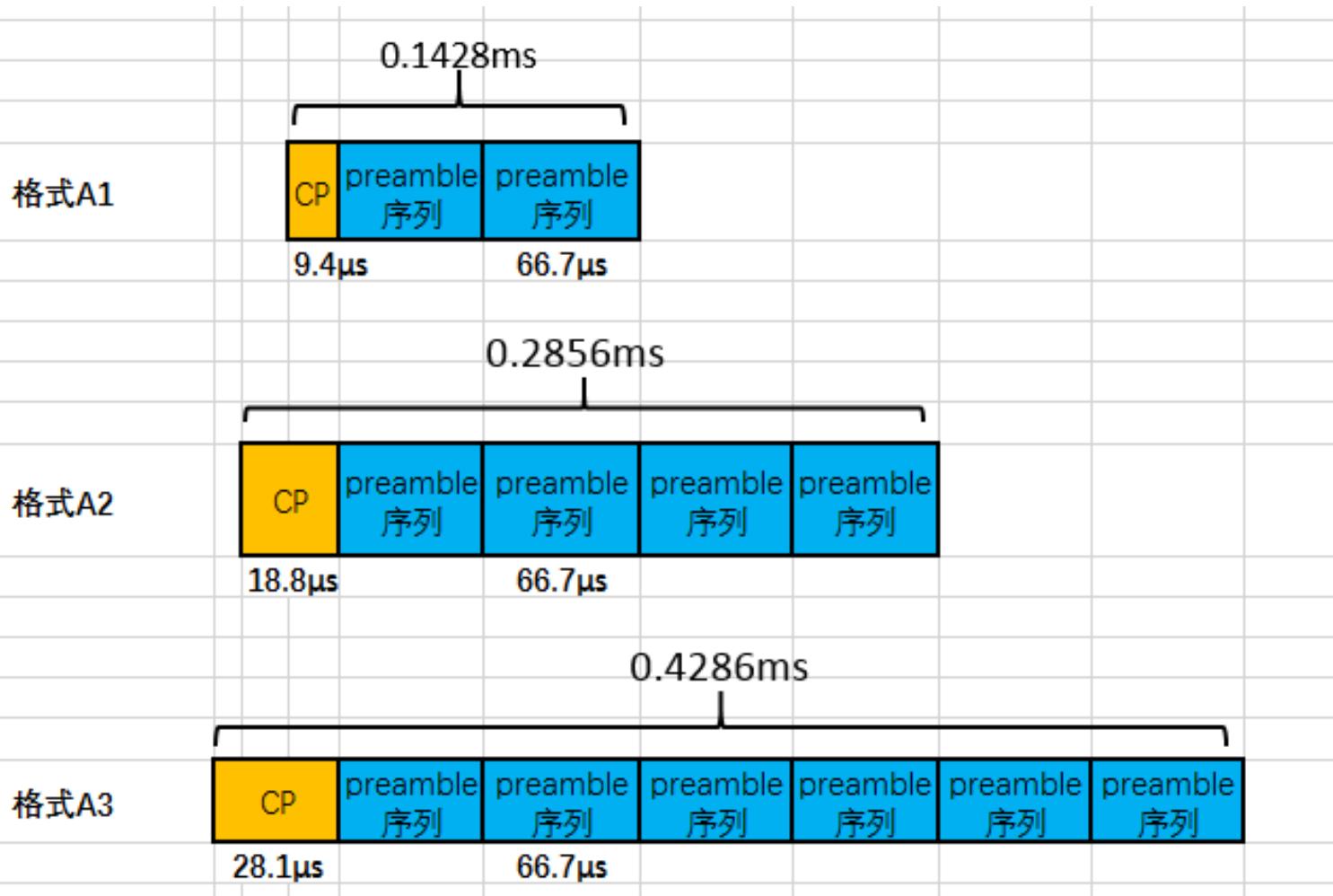
    ra-ContentionResolutionTimer ENUMERATED {sf8, sf16, sf24, sf32, sf40, sf48, sf56, sf64},
    rsrp-ThresholdSSB           RSRP-Range           OPTIONAL,   -- Need R
    rsrp-ThresholdSSB-SUL       RSRP-Range           OPTIONAL,   -- Need R
    prach-RootSequenceIndex     CHOICE {
        1839                INTEGER (0..837),
        1139                INTEGER (0..137)
    },
    msg1-SubcarrierSpacing     SubcarrierSpacing,
    restrictedSetConfig        ENUMERATED {unrestrictedSet, restrictedSetTypeA, restrictedSetTypeB},
}
```

随机接入子载波间隔 (RA-SCS)

前导格式 (短格式A)

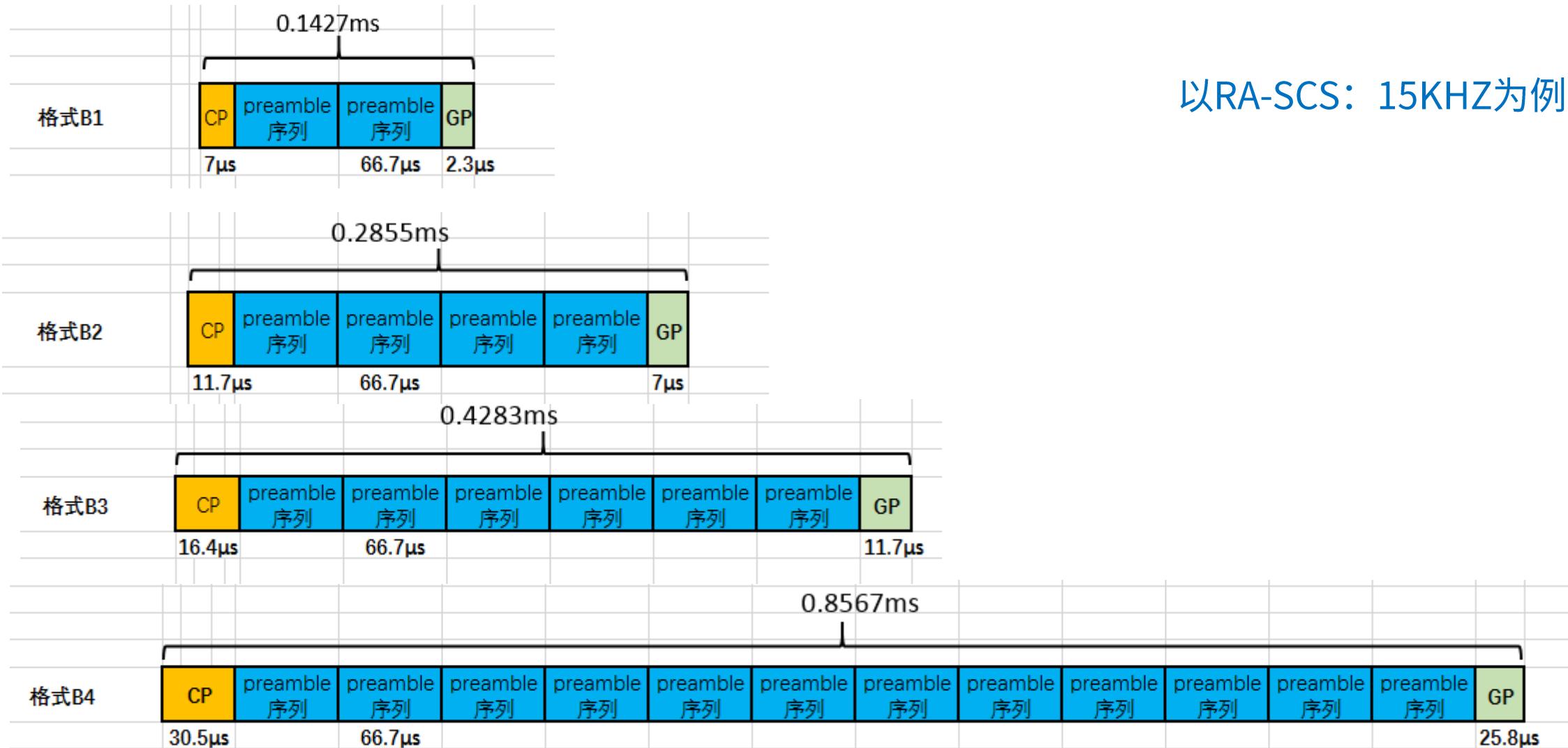
A1, A2, A3分别含有2,4,6个preamble序列 (并且是同一个序列的重复)

以RA-SCS: 15KHZ为例



前导格式 (短格式B)

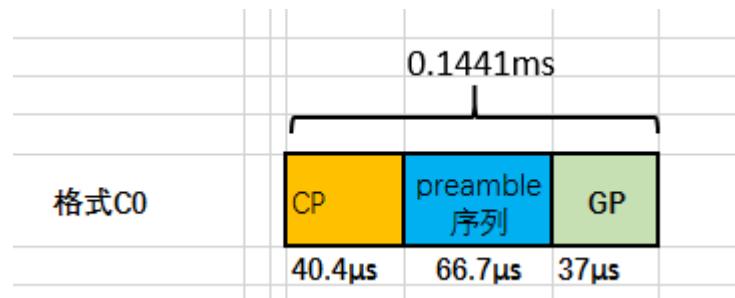
B1, B2, B3, B4分别含有2,4,6,12个preamble序列 (并且是同一个序列的重复)



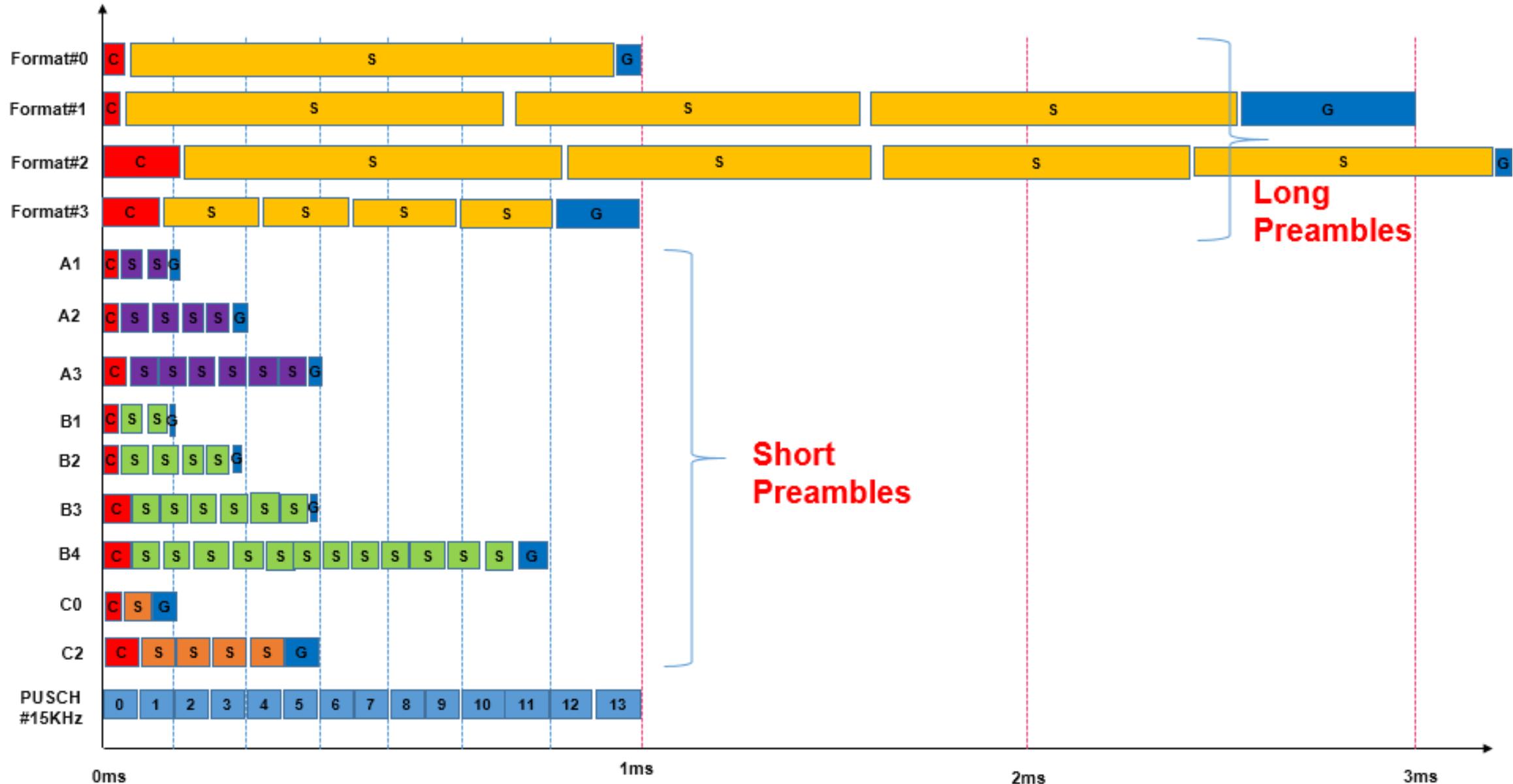
前导格式 (短格式C)

C0, C2分别含有1,4个preamble序列 (并且是同一个序列的重复)

以RA-SCS: 15KHZ为例



preamble所有格式全景图



UE如何知道使用哪种preamble格式?

在SIB1中RACH-ConfigGeneric (通用配置) ->prach-ConfigurationIndex (配置列表) 指示

```
RACH-ConfigGeneric ::= SEQUENCE {  
    prach-ConfigurationIndex      INTEGER (0..255),  
    msg1-FDM                      ENUMERATED {one, two, four, eight},  
    msg1-FrequencyStart            INTEGER (0..maxNrofPhysicalResourceBlocks-1),  
    zeroCorrelationZoneConfig      INTEGER(0..15),  
    preambleReceivedTargetPower   INTEGER (-200..-74),  
    preambleTransMax              ENUMERATED {n3,n4,n5,n6,n7,n8,n10,n20,n50,n100,n200},  
    powerRampingStep              ENUMERATED {dB0, dB2, dB4, dB6},  
    ra-Responsewindow              ENUMERATED {s11, s12, s14, s18, s110, s120, s140, s180}  
}
```

取值范围: 0-255

根据小区不同的频域和双工模式，38.211的第6.3.3节中给出了prach-ConfigurationIndex数值所对应的表格。

- 小区频段为FR1，FDD模式(对称频谱)/SUL，查表38.211 6.3.3.22;
- 小区频段为FR1，TDD模式(非对称频谱)，查表38.211 6.3.3.23;
- 小区频段为FR2，TDD模式(非对称频谱)，查表38.211 6.3.3.24;

现网主要是FR1的TDD模式，因此，我们看一下6.3.3.23这个表格

prach-ConfigurationIndex所对应的表格

Table 6.3.3.2-3: Random access configurations for FR1 and unpaired spectrum

PRACH Configuration Index	Preamble format	$n_{SFN} \bmod x = y$		Subframe number	Starting symbol	Number of PRACH slots within a subframe	$N_t^{\text{RA},\text{slot}}$, number of time-domain PRACH occasions within a PRACH slot	$N_{\text{dur}}^{\text{RA}}$, PRACH duration
0	0	16	1	9	0	-	-	0
1	0	8	1	9	0	-	-	0
2	0	4	1	9	0	-	-	0
3	0	2	0	9	0	-	-	0
4	0	2	1	9	0	-	-	0
5	0	2	0	4	0	-	-	0
6	0	2	1	4	0	-	-	0
7	0	1	0	9	0	-	-	0
8	0	1	0	8	0	-	-	0
9	0	1	0	7	0	-	-	0
10	0	1	0	6	0	-	-	0
11	0	1	0	5	0	-	-	0
12	0	1	0	4	0	-	-	0

只要知道prach-ConfigurationIndex参数的数值，查表就能知道系统配置的是哪种preamble的格式

prach-ConfigurationIndex所对应的表格

PRACH Configuration Index	Preamble format	$n_{SFN} \bmod x = y$	Subframe number	Starting symbol	Number of PRACH slots within a subframe	$N_t^{\text{RA},\text{slot}}$, number of time-domain PRACH occasions within a PRACH slot	$N_{\text{dur}}^{\text{RA}}$, PRACH duration
26	0	1	0	1,4,6,9	0	-	0
27	0	1	0	1,3,5,7,9	0	-	0
28	1	16	1	7	0	-	0
29	1	8	1	7	0	-	0
30	1	4	1	7	0	-	0
31	1	2	0	7	0	-	0
32	1	2	1	7	0	-	0
33	1	1	0	7	0	-	0
34	2	16	1	6	0	-	0
35	2	8	1	6	0	-	0
36	2	4	1	6	0	-	0
37	2	2	0	6	7	-	0
38	2	2	1	6	7	-	0
39	2	1	0	6	7	-	0
40	3	16	1	9	0	-	0
41	3	8	1	9	0	-	0
42	3	2	0	9	0	-	0

prach-ConfigurationIndex所对应的表格

PRACH Configuration Index	Preamble format	$n_{SFN} \bmod x = y \leftarrow$		Subframe number	Starting symbol	Number of PRACH slots within a subframe	$N_t^{\text{RA},\text{slot}}, \downarrow$ number of time-domain PRACH occasions within a PRACH slot	$N_{\text{dur}}^{\text{RA}}, \downarrow$ PRACH duration
209	C2	1	0	1,3,5,7,9	2	1	2	6
210	C2	1	0	0,1,2,3,4,5,6,7,8,9	8	1	1	6
211	A1/B1	2	1	9	2	1	6	2
212	A1/B1	2	1	4,9	8	1	3	2
213	A1/B1	2	1	7,9	8	1	3	2
214	A1/B1	2	1	7,9	2	1	6	2
215	A1/B1	2	1	4,9	2	2	6	2
216	A1/B1	2	1	8,9	2	2	6	2
217	A1/B1	1	0	9	2	2	6	2
218	A1/B1	1	0	9	8	1	3	2
219	A1/B1	1	0	9	2	1	6	2
220	A1/B1	1	0	9	2	1	6	2
251	A3/B3	1	0	4,9	0	1	2	6
252	A3/B3	1	0	7,9	2	1	2	6
253	A3/B3	1	0	3,4,8,9	0	2	2	6
254	A3/B3	1	0	1,3,5,7,9	0	1	2	6
255	A3/B3	1	0	0,1,2,3,4,5,6,7,8,9	2	1	2	6

PRACH的时域资源分布

PRACH的时域资源分布

PRACH根据参数prach-ConfigurationIndex来配置时域资源。

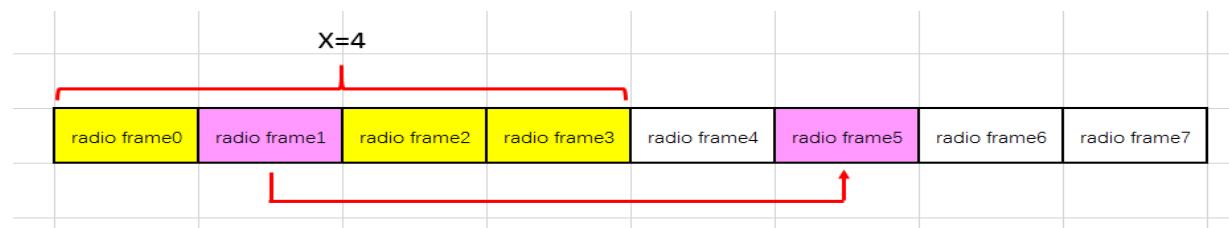
PRACH Configuration Index	Preamble format	$n_{SFN} \bmod x = y$	Subframe number	Starting symbol	Number of PRACH slots within a subframe	$N_t^{\text{RA,slot}}$, number of time-domain PRACH occasions within a PRACH slot	$N_{\text{dur}}^{\text{RA}}$, PRACH duration
0	0	16	1	9	0	-	0
1	0	8	1	9	0	-	0
2	0	4	1	9	0	-	0
3	0	2	0	9	0	-	0

根序列号 前导格式



PRACH
发送周期

X=4就是每隔4个无线帧 (40ms)，才会发送preamble，而x=2就是每隔2个无线帧 (20ms)，才会发送preamble。当X越小，UE接入越及时，X越大，接入越不及时。



PRACH的时域资源分布

PRACH根据参数prach-ConfigurationIndex来配置时域资源。

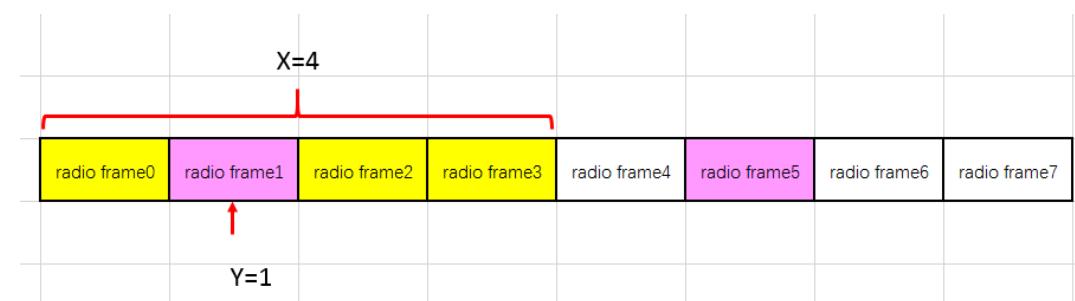
PRACH Configuration Index	Preamble format	$n_{SFN} \bmod x = y$	Subframe number	Starting symbol	Number of PRACH slots within a subframe	$N_t^{\text{RA,slot}}$, number of time-domain PRACH occasions within a PRACH slot	$N_{\text{dur}}^{\text{RA}}$, PRACH duration
0	0	16	1	9	0	-	0
1	0	8	1	9	0	-	0
2	0	4	1	9	0	-	0
3	0	2	0	9	0	-	0

根序列号 前导格式

↑
↑
PRACH
发送周期

在PRACH配置周期内，
第几个帧有PRACH时隙
0是第一个帧，1是第二个帧

比如X=4, y=1



PRACH的时域资源分布

PRACH根据参数prach-ConfigurationIndex来配置时域资源。

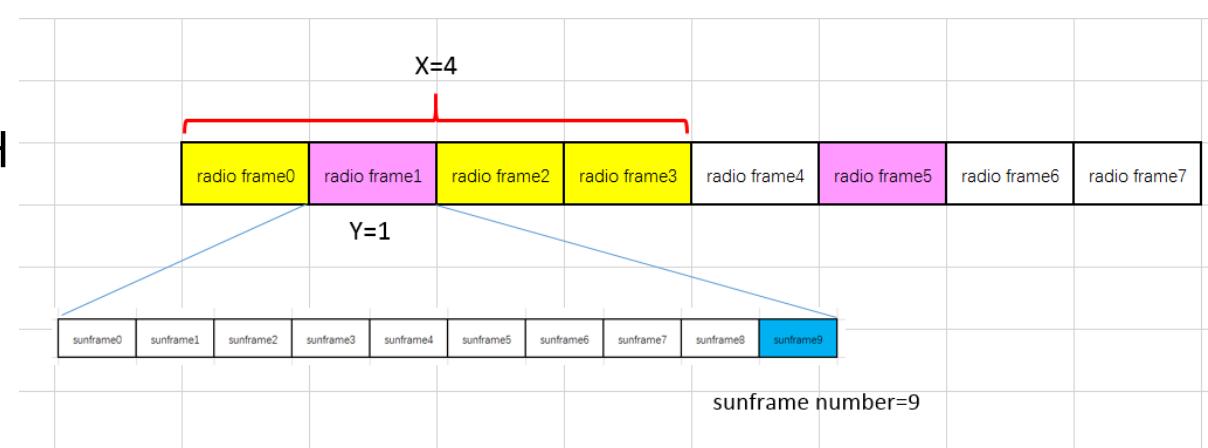
PRACH Configuration Index	Preamble format	$n_{SFN} \bmod x = y$	Subframe number	Starting symbol	Number of PRACH slots within a subframe	$N_t^{\text{RA,slot}}$, number of time-domain PRACH occasions within a PRACH slot	$N_{\text{dur}}^{\text{RA}}$, PRACH duration
0	0	16	1	9	0	-	0
1	0	8	1	9	0	-	0
2	0	4	1	9	0	-	0
3	0	2	0	9	0	-	0

根序列号 前导格式

↑
PRACH 配置周期

含有PRACH 的子帧号

在PRACH配置周期内，
第几个帧有PRACH时隙
0是第一个帧，1是第二个帧



PRACH开始符号与持续符号

PRACH Configuration Index	Preamble format	$n_{SFN} \bmod x = y \Leftrightarrow$ $x \Leftrightarrow$ $y \Leftrightarrow$	Subframe number	Starting symbol	Number of PRACH slots within a subframe	$N_t^{\text{RA},\text{slot}}, \downarrow$ number of time-domain PRACH occasions within a PRACH slot	$N_{\text{dur}}^{\text{RA}}, \downarrow$ PRACH duration
0	0	16	1	9	0	-	0
1	0	8	1	9	0	-	0
2	0	4	1	9	0	-	0
3	0	2	0	9	0	-	0

PRACH持续符号

PRACH开始符号

$$l = l_0 + n_t^{\text{RA}} N_{\text{dur}}^{\text{RA}} + 14n_{\text{slot}}^{\text{RA}}$$
 其中 $n_{\text{slot}}^{\text{RA}}$ 的取值范围是 0 到 $N_t^{\text{RA},\text{slot}} - 1$

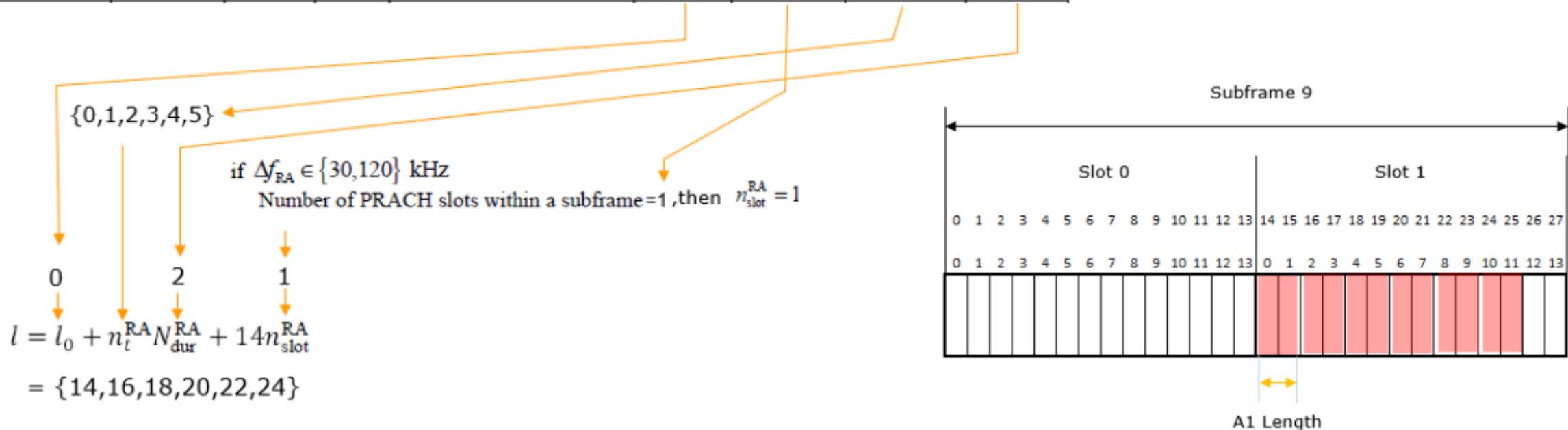
if $\Delta f_{\text{RA}} \in \{1.25, 5, 15, 60\}$ kHz, then $n_{\text{slot}}^{\text{RA}} = 0$
 if $\Delta f_{\text{RA}} \in \{30, 120\}$ kHz And
 Number of PRACH slots within a subframe = 1
 Number of PRACH slots within a 60 kHz slot = 1
 ,then $n_{\text{slot}}^{\text{RA}} = 1$
 otherwise, $n_{\text{slot}}^{\text{RA}} \in \{0, 1\}$

举个例子

TDD FR1 prach-ConfigurationIndex = 79, RA-SCS = 30Khz

< 38.211 v15.3.0-Table 6.3.3.2-3: Random access configurations for FR1 and unpaired spectrum >

PRACH Configuration Index	Preamble format	$n_{SFN} \bmod x = y$	Subframe number	Starting symbol	Number of PRACH slots within a subframe	$N_t^{\text{RA,slot}}$, number of time-domain PRACH occasions within a PRACH slot	$N_{\text{dur}}^{\text{RA}}$, PRACH duration
79	A1	1 0	9	0	1	6	2



举个例子

TDD FR1 RachConfig = 4, SCS = 15Khz

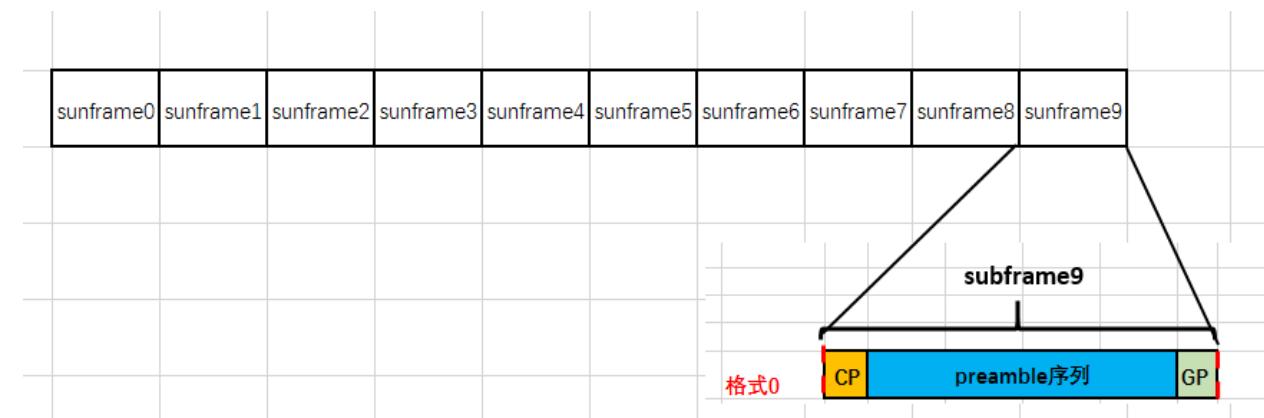
< 38.211 v15.3.0-[Table 6.3.3.2-3](#): Random access configurations for FR1 and unpaired spectrum >

PRACH Configuration Index	Preamble format	$n_{SFN} \bmod x = y$	Subframe number	Starting symbol	Number of PRACH slots within a subframe	$N_t^{\text{RA},\text{slot}}$, number of time-domain PRACH occasions within a PRACH slot	$N_{\text{dur}}^{\text{RA}}$, PRACH duration
4	0	2	1	9	0	-	0

if $\Delta f_{\text{RA}} \in \{1.25, 5, 15, 60\}$ kHz, then $n_{\text{slot}}^{\text{RA}} = 0$

$$l = l_0 + n_t^{\text{RA}} N_{\text{dur}}^{\text{RA}} + 14n_{\text{slot}}^{\text{RA}}$$

$$= 0$$



PRACH的频域资源分布

PRACH频域占用的RB数目

下表决定了PRACH的频域资源的大小

例如，PRACH和PUSCH的子载波间隔分别为1.25KHZ和15KHZ，则，PRACH对应6个PUSCH RB

Table 6.3.3.2-1: Supported combinations of Δf^{RA} and Δf , and the corresponding value of \bar{k} .

L_{RA}	Δf^{RA} for PRACH	Δf for PUSCH	$N_{\text{RB}}^{\text{RA}}$, allocation expressed in number of RBs for PUSCH	\bar{k}
839	1.25	15	6	7
839	1.25	30	3	1
839	1.25	60	2	133
839	5	15	24	12
839	5	30	12	10
839	5	60	6	7
139	15	15	12	2
139	15	30	6	2
139	15	60	3	2
139	30	15	24	2
139	30	30	12	2
139	30	60	6	2
139	60	60	12	2
139	60	120	6	2
139	120	60	24	2
139	120	120	12	2

PRACH频域位置

PRACH的频域资源主要由两个参数 “msg1-FrequencyStart” 和 “msg1-FDM” 决定。

msg1-FrequencyStart: PRACH频域资源在BWP中相对于PRB0的频域偏置

msg1-FDM: 决定了频域复用PRACH的资源个数，**取值为{1,2,4,8}**

这两个参数在信令**SIB1--- RACH-ConfigGeneric**中

```
RACH-ConfigGeneric ::=  
    prach-ConfigurationIndex  
    msg1-FDM  
    msg1-FrequencyStart  
    zeroCorrelationZoneConfig  
    preambleReceivedTargetPower  
    preambleTransMax  
    powerRampingStep  
    ra-ResponseWindow  
}
```

```
SEQUENCE {  
    INTEGER (0..255),  
    ENUMERATED {one, two, four, eight},  
    INTEGER (0..maxNrofPhysicalResourceBlocks-1),  
    INTEGER(0..15),  
    INTEGER (-200..-74),  
    ENUMERATED {n3,n4,n5,n6,n7,n8,n10,n20,n50,n100,n200},  
    ENUMERATED {dB0, dB2, dB4, dB6},  
    ENUMERATED {s11, s12, s14, s18, s110, s120, s140, s180}
```

msg1-FrequencyStart ----举个例子

假设 msg1-FrequencyStart=1

RACH子载波间隔: msg1-SubcarrierSpacing为30KHZ,
现网SCS为30KHZ

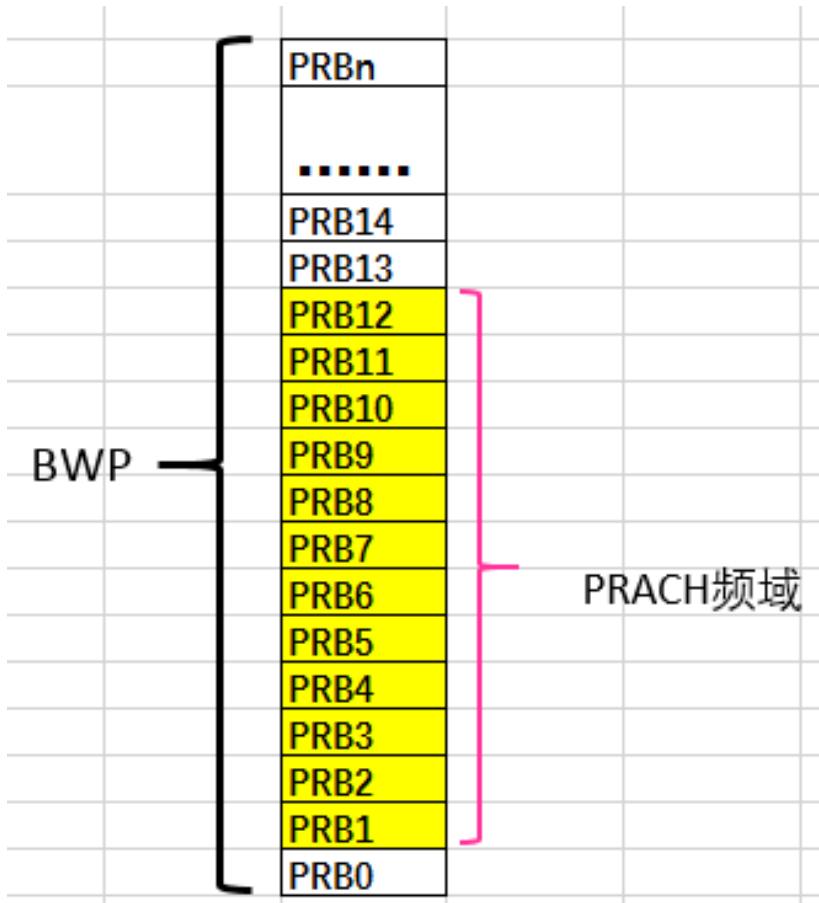
Table 6.3.3.2-1: Supported combinations of Δf^{RA} and Δf , and the corresponding value of \bar{k} .

L_{RA}	Δf^{RA} for PRACH	Δf for PUSCH	$N_{\text{RB}}^{\text{RA}}$, allocation expressed in number of RBs for PUSCH	\bar{k}
839	1.25	15	6	7
839	1.25	30	3	1
839	1.25	60	2	133
839	5	15	24	12
839	5	30	12	10
839	5	60	6	7
139	15	15	12	2
139	15	30	6	2
139	15	60	3	2
139	30	15	24	2
139	30	30	12	2
139	30	60	6	2
139	60	60	12	2
139	60	120	6	2
139	120	60	24	2
139	120	120	12	2

查表可知：
PRACH频域占12个RB，
从BWP的PRB1开始。

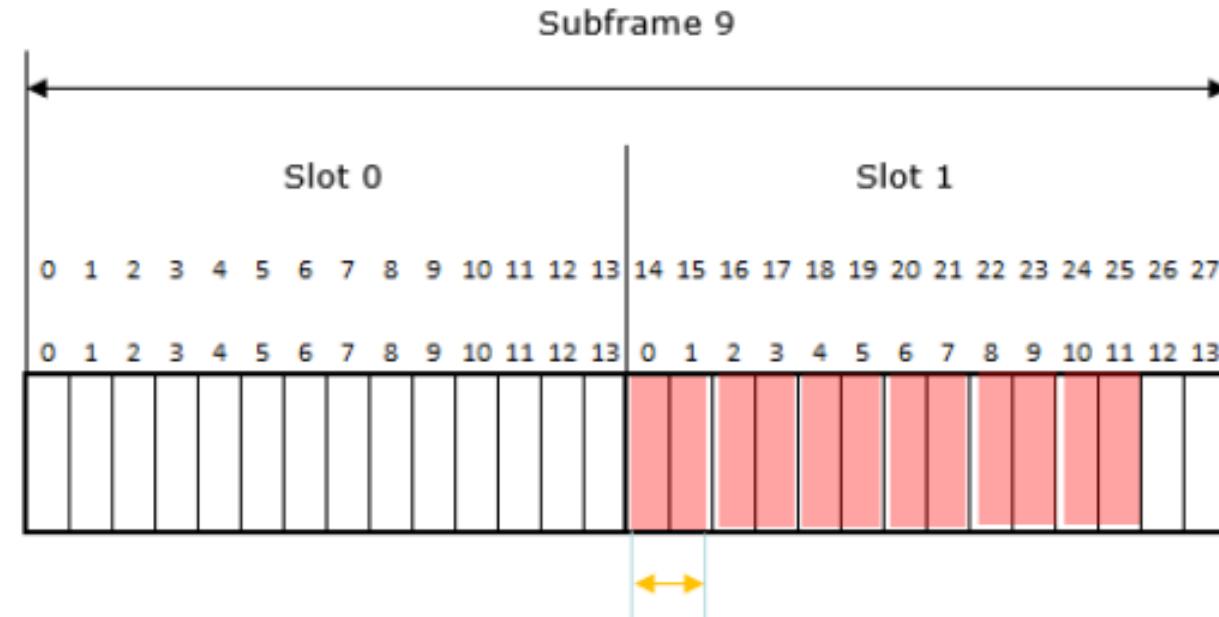
msg1-FrequencyStart ----举个例子

假设再加上时域 **prach-ConfigurationIndex=79**



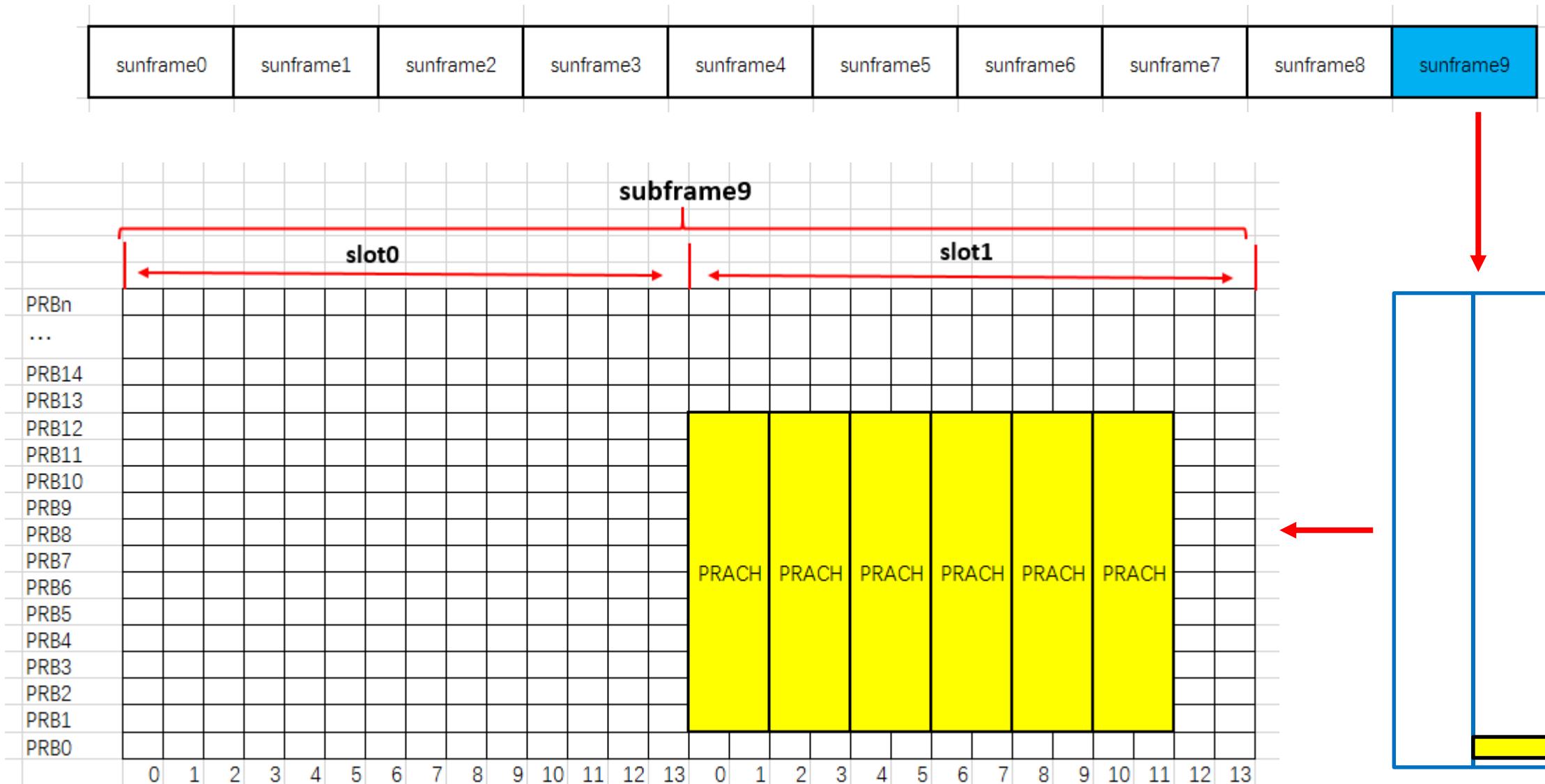
频域结果

(前面讲过的例子)



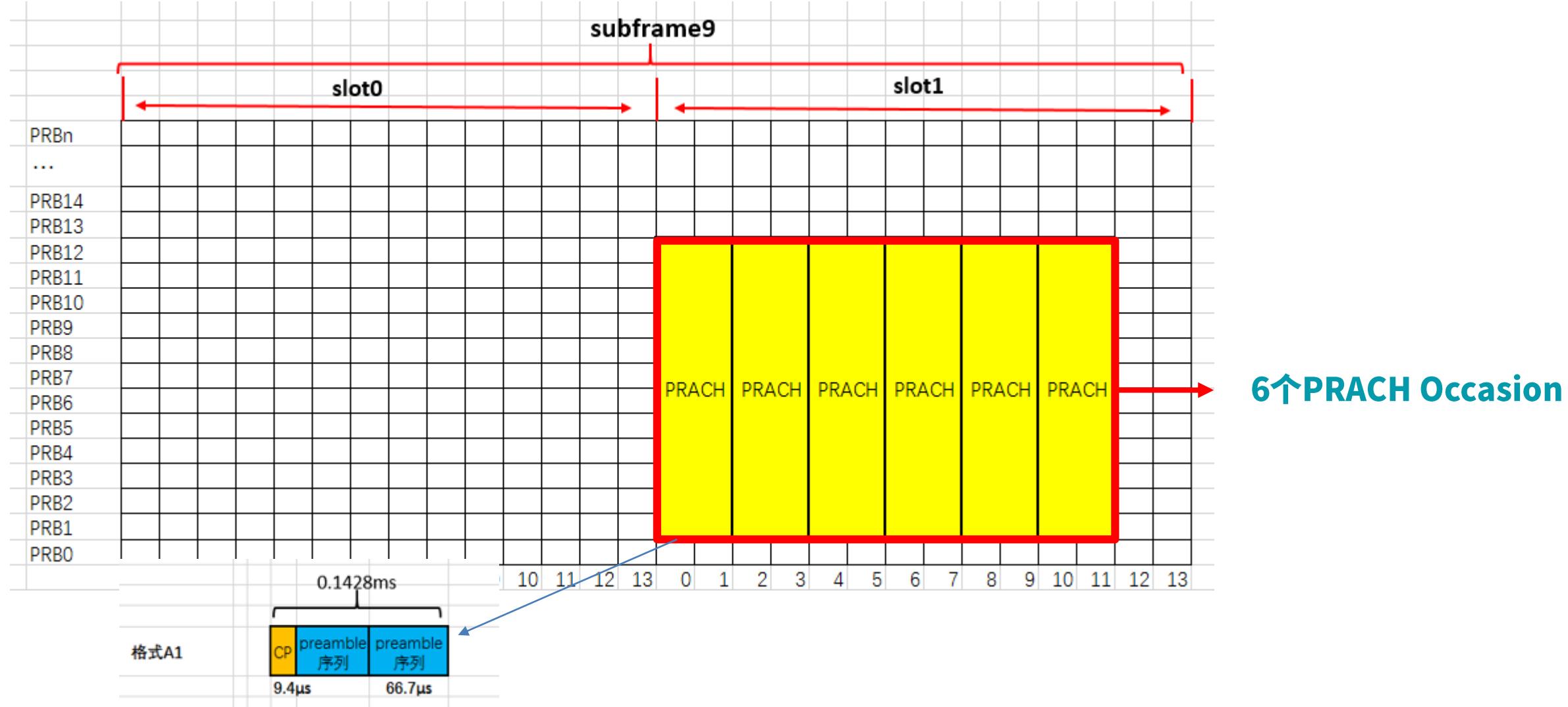
时域结果

该例子中PRACH在整个帧中的位置



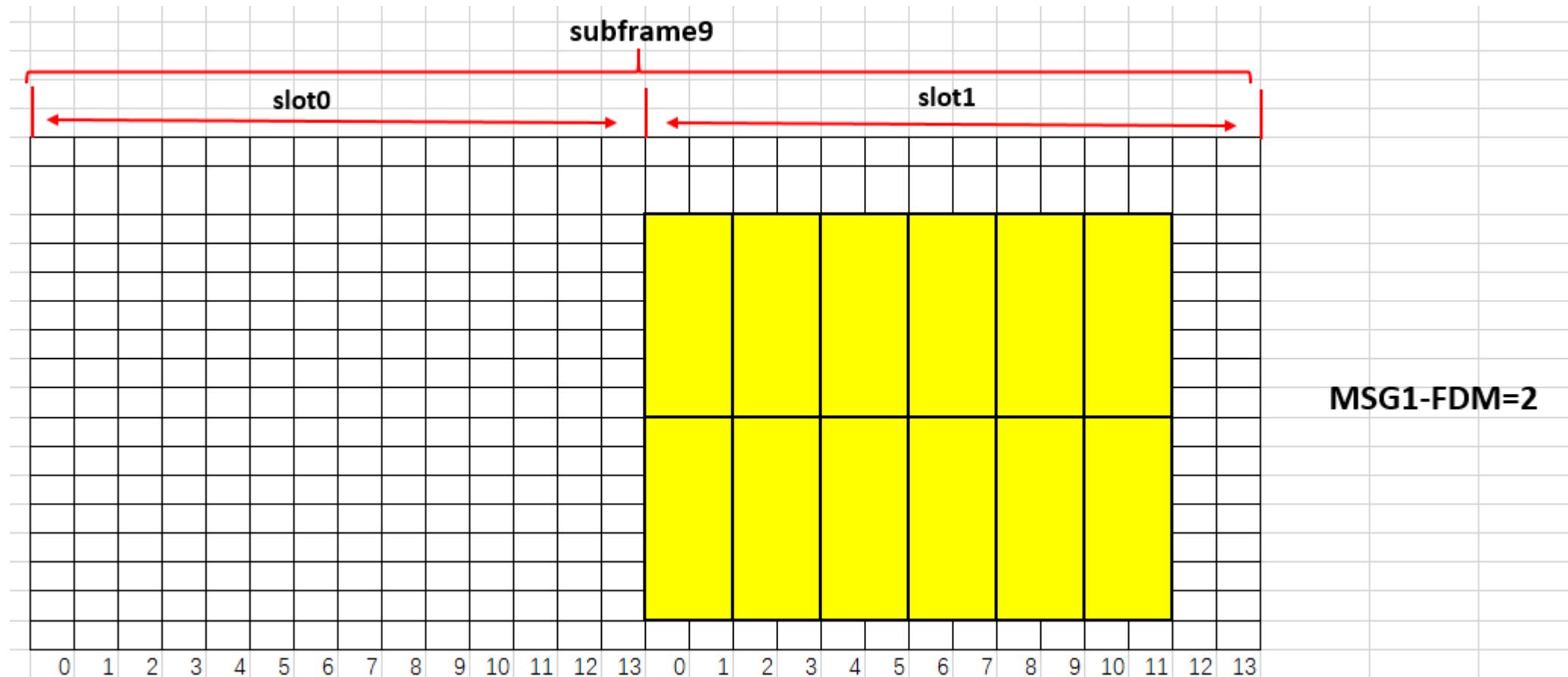
PRACH Occasion (时频资源)

5G定义了PRACH Occasion的概念，意思是PRACH的时频资源，也可以理解为机会



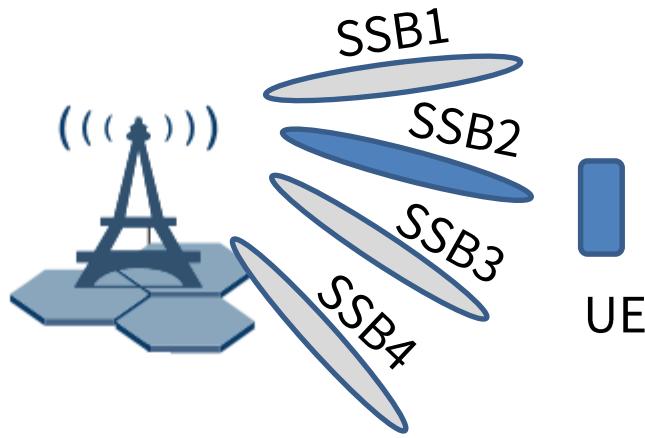
msg1-FDM

msg1-FDM：决定了频域PRACH occasion 的个数，取值为{1,2,4,8}



SSB与preamble的映射

3GPP协议中将preamble index与SSB index进行了关联映射，这样基站可以基站通过检测到的preamble index就能反过来推理出哪个SSB是对当前这个终端的最佳下波束。

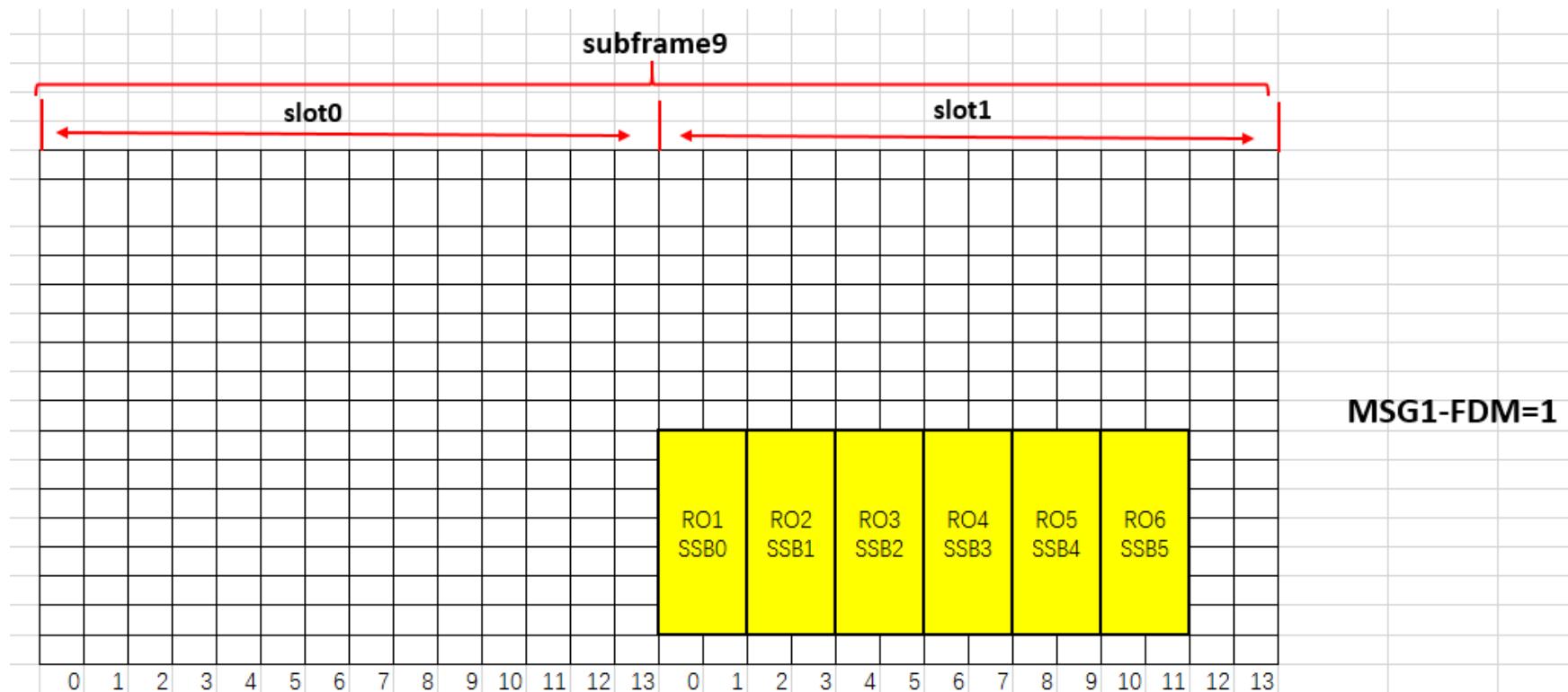


1. 基站发送多个SSB
2. UE检测SSB2最强，并驻留SSB2
3. UE要发起接入，在SSB2的放向发送preamble (SSB2与preamble关联)
4. 基站通过解码preamble，确定了SSB2是对于这个UE最佳发射方向
5. 后续给终端的发射都在SSB2的方向

SSB与preamble的映射

Preamble在PRACH Occasion当中传输，因此，SSB与preamble的关联映射，实际上就牵扯到PRACH occasion

SSB到PRACH occasion 的映射关系，支持一对一，一对多，多对一三种情况

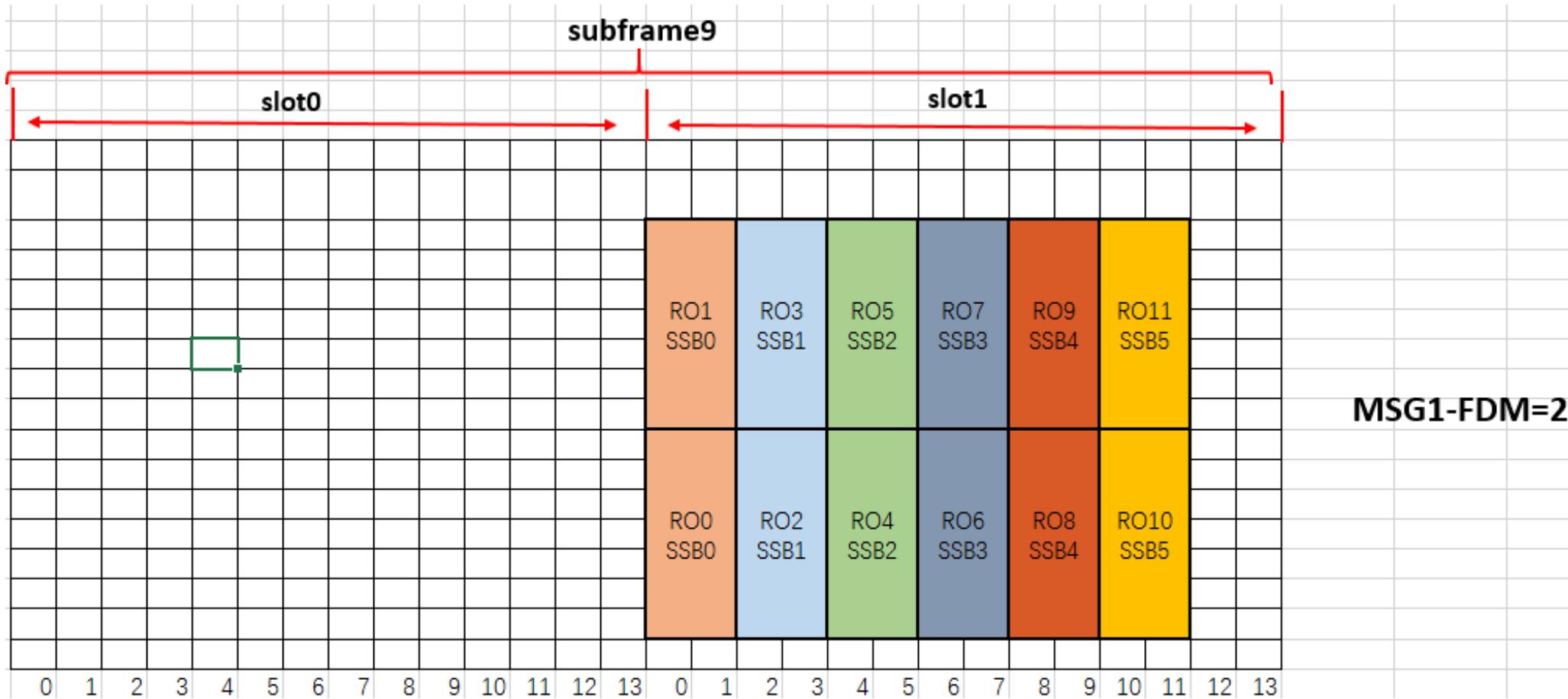


一对一

RO位置就区分了SSB

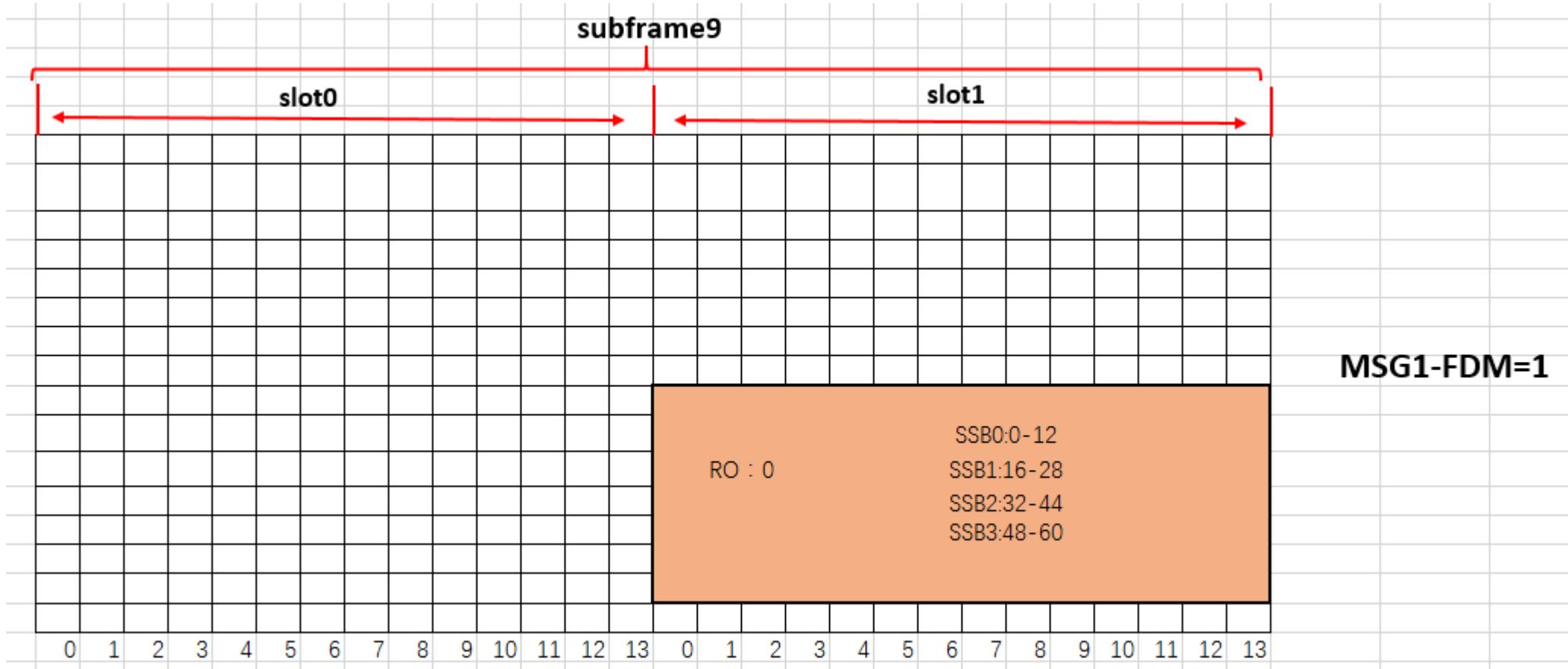
一对多

RO位置就区分了SSB



多对一

Preamble index区分SSB



SSB映射参数

一对一，一对多，多对一这三种情况映射关系，由以下信令设定

SIB1当中的RACH-ConfigCommon: ssb-perRACH-OccasionAndCB-PreamblesPerSSB 决定

```
RACH-ConfigCommon ::= SEQUENCE {
    rach-ConfigGeneric          RACH-ConfigGeneric,
    totalNumberOfRA-Preambles   INTEGER (1..63)   OPTIONAL, -- Need S
    ssb-perRACH-OccasionAndCB-PreamblesPerSSB CHOICE {
        oneEighth    ENUMERATED {n4,n8,n12,n16,n20,n24,n28,n32,n36,n40,n44,n48,n52,n56,n60,n64},
        oneFourth   ENUMERATED {n4,n8,n12,n16,n20,n24,n28,n32,n36,n40,n44,n48,n52,n56,n60,n64},
        oneHalf     ENUMERATED {n4,n8,n12,n16,n20,n24,n28,n32,n36,n40,n44,n48,n52,n56,n60,n64},
        one         ENUMERATED {n4,n8,n12,n16,n20,n24,n28,n32,n36,n40,n44,n48,n52,n56,n60,n64},
        two         ENUMERATED {n4,n8,n12,n16,n20,n24,n28,n32},
        four        INTEGER (1..16),
        eight       INTEGER (1..8),
        sixteen     INTEGER (1..4)
    }                           OPTIONAL, - Need M
}
```

ssb-perRACH-Occasion = 1/8、1/4、1/2分别表示一个SSB映射8、4、2个PRACH occasion

ssb-perRACH-Occasion = 1，表示SSB与PRACH occasion一对一映射

ssb-perRACH-Occasion = 2,4,8,16分别表示2,4,8,16个SSB映射到1个PRACH occasion

感谢观看

