DRX

本节主要介绍处于 RRC_CONNECTED 态下的 UE 的 DRX 处理流程。结合 3GPP 协议，介绍了几个 timer 的作用，同时还简单介绍了载波聚合对 DRX 的影响。

1.1 DRX 介绍

基于包的数据流通常是突发性的，在一段时间内有数据传输，但在接下来的一段较长时间内没有数据传输。在没有数据传输的时候，可以通过停止接收 PDCCH（此时会停止 PDCCH 盲检）来降低功耗，从而提升电池使用时间。这就是 DRX（Discontinuous Reception，非连续接收）的由来。

DRX 的基本机制是为处于 RRC_CONNECTED 态的 UE 配置一个 DRX cycle。DRX cycle 由“*On Duration*”和“Opportunity for DRX”组成：在“On Duration”时间内，UE 监听并接收 PDCCH（激活期）；在“Opportunity for DRX”时间内，UE 不接收 PDCCH 以减少功耗（休眠期）。

从图 1 可以看出，在时域上，时间被划分成一个个连续的 DRX Cycle。

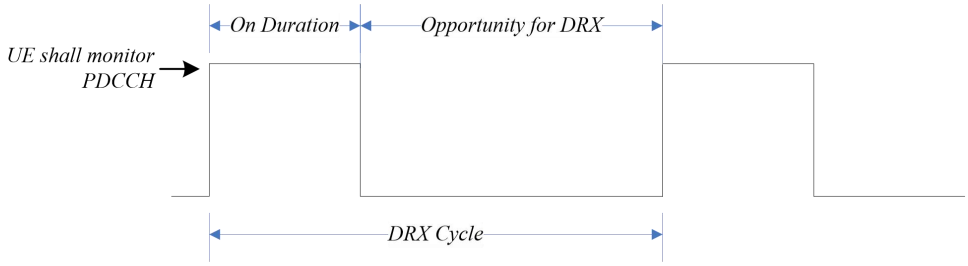

图 1：DRX cycle

注意：处于休眠期的 UE，只是不接收 PDCCH，但是可以接收来自其它物理信道的数据，如 PDSCH、ACK/NACK 等。例如：在 SPS 调度中，处于休眠期的 UE 可以接收周期性配置的下行子帧上发送的 PDSCH 数据。

eNodeB 通过 *DRX-Config* 来配置某个 UE 的 DRX 相关参数。

```
DRX-Config ::=                      CHOICE {
    release                             NULL,
```

```
    setup                                   SEQUENCE {
        onDurationTimer                         ENUMERATED {
                                                    psf1, psf2, psf3, psf4, psf5, psf6,
                                                    psf8, psf10, psf20, psf30, psf40,
                                                    psf50, psf60, psf80, psf100,
                                                    psf200},-------从一个DRX Cycle的起始处算起，
连续监听的PDCCH子帧数。
        drx-InactivityTimer                     ENUMERATED {
                                                    psf1, psf2, psf3, psf4, psf5, psf6,
                                                    psf8, psf10, psf20, psf30, psf40,
                                                    psf50, psf60, psf80, psf100,
                                                    psf200, psf300, psf500, psf750,
                                                    psf1280, psf1920, psf2560, psf0-v1020,
                                                    spare9, spare8, spare7, spare6,
                                                    spare5, spare4, spare3, spare2,
                                                    spare1},-------当UE成功解码一个指示初传的UL或
DL用户数据的PDCCH后，持续处于激活态的连续PDCCH子帧数。
        drx-RetransmissionTimer                 ENUMERATED {
                                                    psf1, psf2, psf4, psf6, psf8, psf16,
                                                    psf24, psf33}, -------从UE期待收到DL重传的子
帧（HARQ RTT之后）开始，连续监听的PDCCH子帧数。
        longDRX-CycleStartOffset            CHOICE {
            sf10                                INTEGER(0..9),
            sf20                                INTEGER(0..19),
            sf32                                INTEGER(0..31),
            sf40                                INTEGER(0..39),
            sf64                                INTEGER(0..63),
            sf80                                INTEGER(0..79),
            sf128                               INTEGER(0..127),
            sf160                               INTEGER(0..159),
            sf256                               INTEGER(0..255),
            sf320                               INTEGER(0..319),
            sf512                               INTEGER(0..511),
            sf640                               INTEGER(0..639),
            sf1024                              INTEGER(0..1023),
            sf1280                              INTEGER(0..1279),
            sf2048                              INTEGER(0..2047),
            sf2560                              INTEGER(0..2559)
        },-------指定了longDRX-Cycle和drxStartOffset。
        shortDRX                            SEQUENCE {
            shortDRX-Cycle                      ENUMERATED  {
                                                    sf2, sf5, sf8, sf10, sf16, sf20,
                                                    sf32, sf40, sf64, sf80, sf128, sf160,
                                                    sf256, sf320, sf512, sf640},-------指
定了short DRX Cycle持续的子帧数，即short DRX Cycle的大小。
            drxShortCycleTimer                  INTEGER (1..16)-------指定了UE在多长的时间
内，使用的是short DRX Cycle。该值为shortDRX-Cycle的倍数。
        }       OPTIONAL                                                    -- Need OR
    }
}
```

DRX cycle的选择需要考虑电池节约与延迟之间的平衡。从一个方面讲，长DRX周期有益于延长UE的电池使用时间；例如网页浏览过程中，当用户正在阅读已经下载好的网页时，UE持续接收下行数据是对资源的浪费。从另一个方面讲，当有新的数据传输时，一个更短的DRX周期有益于更快的响应，例如用户请求另一个网页或者进行VoIP通话时。为了满足上述需求，每个UE可以配置两个DRX cycle：*shortDRX-Cycle*和*longDRX-Cycle*。如果UE配置了*shortDRX-Cycle*，则*longDRX-Cycle*应该配置为*shortDRX-Cycle*的倍数。但在任一时刻，UE只能使用其中一种配置。

drxStartOffset指定DRX cycle的起始子帧，***longDRX-Cycle***指定了一个long DRX cycle占多少个子帧（即连续的“子帧数”），这两个参数都是由***longDRX-CycleStartOffset***字段确定的。***onDurationTimer***指定了从DRX cycle的起始子帧算起，需要监听PDCCH的连续“PDCCH子帧数”。

对于DRX，需要注意“连续的子帧数”与“连续的PDCCH子帧数”的区别。FDD中，PDCCH子帧可以是任意子帧；但在TDD中，PDCCH子帧只包含下行子帧和包含DwPTS的子帧，这是因为只有下行子帧才有可能传输PDCCH。

DRX中定义了多个定时器（timer），有些指定的是“连续的子帧数”，而另一些指定的是“连续的PDCCH子帧数”。在TDD中，如果某个定时器指定的是“连续的PDCCH子帧数”，则上行子帧是不统计在该定时器的持续时间中的，此时该定时器实际持续的“子帧数”可能大于其指定的“PDCCH子帧数”。（见图3）

在大多数情况下，当一个UE在某个子帧被调度并接收或发送数据后，很可能在接下来的几个子帧内继续被调度，如果要等到下一个DRX cycle再来接收或发送这些数据将会带来额外的延迟。为了降低这类延迟，UE在被调度后，会持续处于激活期，即会在配置的激活期内持续监听PDCCH。其实现机制是：每当UE被调度以初传数据时，就会启动（或重启）一个定时器*drx-InactivityTimer*，UE将一直处于激活态直到该定时器超时。***drx-InactivityTimer***指定了当UE成功解码一个指示初传的UL或DL用户数据的PDCCH后，持续处于激活态的连续PDCCH子帧数。即当UE有初传数据被调度时，该定时器就启动或重启一次。**注意：**（1）这里是初传而不是重传，即指示重传的PDCCH并不会重启该定时器；（2）周期性的SPS子帧上发送的PDSCH虽然是初传，但并没有伴随着传输PDCCH，因此该PDSCH并不会重启该定时器；（3）*drx-InactivityTimer*指定的是连续的“PDCCH子帧数（下行子帧）”，而不是连续的“子帧数”。

HARQ重传并不关心DRX cycle，配置了DRX的UE与没有配置DRX时使用相同的方式来接收/发送HARQ反馈和重传。上行使用同步方式，前一次传输与重传之间有固定的timing关系。下行使用异步方式，前一次传输与重传之间没有固定的timing关系，因此LTE定义了一个时间窗（HARQ RTT Timer），允许UE从前一次下行传输算起，并持续该时间窗之后，才开始监听下行的重传。

为了允许UE在HARQ RTT期间内休眠，每个DL HARQ process定义了一个“HARQ RTT(Round Trip Time) timer”。当某个下行HARQ process的TB解码失败时，UE可以假定至少在“HARQ RTT”子帧后才会

有重传，因此当HARQ RTT timer正在运行时，UE没必要监听PDCCH。当HARQ RTT timer超时，且对应HARQ process接收到的数据没有被成功解码时，UE会为该HARQ process启动一个*drx-RetransmissionTimer*。当该timer运行时，UE会监听用于HARQ重传的PDCCH。*drx-RetransmissionTimer*的长度与eNodeB调度器的灵活度要求相关。如果是要达到最优的电池消耗，就要求eNodeB在HARQ RTT timer超时之后，立即调度HARQ重传，这就也要求eNodeB为此预留无线资源，此时*drx-RetransmissionTimer*也就可以配得短些。***drx-RetransmissionTimer***指定了从UE期待收到DL重传的子帧（HARQ RTT之后）开始，连续监听的“PDCCH子帧数”。注意：这里针对的是“下行重传”，而不是“上行重传”。

对FDD而言，HARQ RTT Timer的大小固定为8个子帧。对TDD而言，HARQ RTT Timer的大小为k + 4个子帧，其中k值为下行传输与对应HARQ反馈之间的时间间隔（k值见36.213的Table 10.1.3.1-1）。

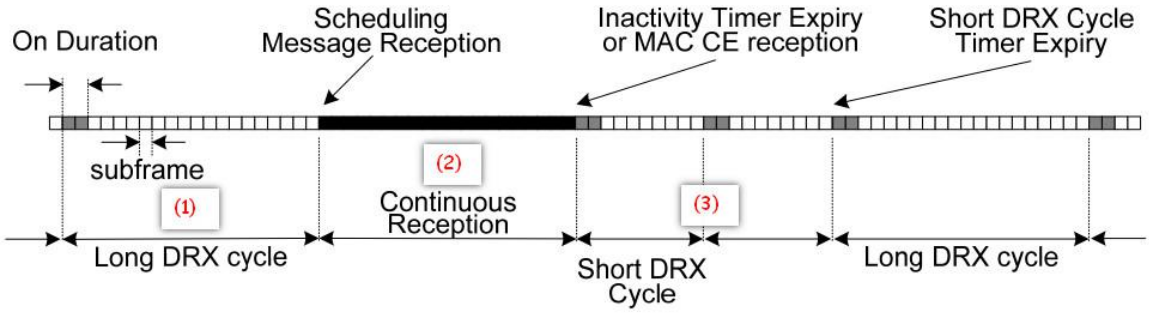

图 2：DRX 流程

当UE在“On Duration”期间收到一个调度消息（指示初传的PDCCH）时，UE会启动一个“*drx-InactivityTimer*”并在该timer运行期间的每一个下行子帧监听PDCCH。当“*drx-InactivityTimer*”运行期间收到一个调度信息（指示初传的PDCCH）时，UE会重启该timer。（对应图2中标红为(2)的部分）

当“*drx-InactivityTimer*”超时或收到DRX Command MAC control element时：1）如果UE没有配置short DRX cycle，则直接使用long DRX cycle；2）如果UE配置了short DRX cycle，UE会使用short DRX cycle并启动（或重启）“***drxShortCycleTimer***”，当“*drxShortCycleTimer*”超时，UE使用long DRX cycle。（对应图2中标红为(3)的部分）

如果UE当前使用short DRX cycle，且[(SFN * 10) + subframe number] modulo (shortDRX - Cycle) = (drxStartOffset) modulo (shortDRX-Cycle)；或者UE当前使用long DRX cycle，且[(SFN * 10) + subframe number] modulo (longDRX-Cycle) = drxStartOffset，则启动“*onDurationTimer*”。（对应图2中标红为(1)的部分）

注：*drxShortCycleTimer*启动后，只说明当前使用short DRX cycle，但此时未必启动了DRX short cycle。DRX short cycle是与*onDurationTimer*同时启动的。类似的，long DRX cycle也是与*onDurationTimer*同时启动的。

当UE配置了DRX cycle时，UE处于激活期的时间包括：

- *onDurationTimer*、或*InactivityTimer*、或*drx-RetransmissionTimer*、或*mac-ContentionResolutionTimer*正在运行时；
- UE已经在PUCCH上发送了SR，且该SR当前处于pending态；
- UE的HARQ buffer存在数据，并等待HARQ重传的UL grant时；
- UE成功接收了用于响应非UE选择的preamble的RAR，却没有收到指示初传（使用C-RNTI）的PDCCH时。

DRX是UE级别的特性，而不是基于每个无线承载来配置的。

当UE配置了DRX时， UE只能在“激活期”的时间内发送周期性CQI。eNodeB在使用RRC来配置周期性CQI上报时，可以进一步地限制UE只能在“on-duration”的时间内发送CQI。

图3结合36.213的5.7节总结了关于各种DRX相关的timer启动和停止的触发条件。

Timer	Start（Restart）	Stop	“子帧数”或“PDCCH子帧数”
onDurationTimer	UE当前使用short DRX cycle，且[(SFN * 10) + subframe number] modulo (shortDRX-Cycle) = (drxStartOffset) modulo (shortDRX-Cycle)；或者UE当前使用long DRX cycle，且[(SFN * 10) + subframe number] modulo (longDRX-Cycle) = drxStartOffset	（1）收到DRX Command MAC control element；（2）timer超时	PDCCH子帧数
drx-InactivityTimer	收到用于调度new transmission的PDCCH（DL和UL的均可）	（1）收到DRX Command MAC control element；（2）timer超时	PDCCH子帧数
drx-RetransmissionTimer	HARQ RTT Timer超时且对应HARQ process的buffer中的数据没有成功解码	（1）收到指示下行传输的PDCCH；（2）当前子帧是周期性配置的下行SPS子帧（此时没有伴随着传输PDCCH）；（3）timer超时	PDCCH子帧数
drxShortCycleTimer	当配置了Short DRX cycle时，如果*drx-InactivityTimer*超时，或收到DRX Command MAC control element，则启动或重启	Timer超时，此时开始使用Long DRX cycle	子帧数

	drxShortCycleTimer，并开始使用Short DRX cycle		
HARQ RTT timer	（1）UE收到一个指示下行传输的PDCCH；（2）当前子帧是周期性配置的下行SPS子帧（此时没有伴随着传输PDCCH）	timer超时	子帧数

图 3：与 DRX 相关 timer 的启动和停止

注：*longDRX-Cycle*和*shortDRX-Cycle*指定的是连续的“子帧数”。

除了HARQ RTT timer和drx-RetransmissionTimer是每个DL HARQ process都有一个外，其它的timer是每个UE只有一个。

从图3可以看出，当任一timer启动时，不会影响其它timer的运行。也即，UE处于激活态的最短时间为*onDurationTimer*指定的时间，而最长时间是不定的。

需要说明的是，对于eNodeB的调度器而言，需要知道UE何时处于激活期，何时处于休眠期，以便只在激活期调度该UE。

1.2 载波聚合（CA）对 DRX 的影响

如果配置了一个或多个 SCell，则所有的 serving cells 使用相同的 DRX 操作：

- 对于所有的 DL 载波单元（component carrier）而言，PDCCH 监测的激活时间是相同的；
- 当 UE 处于休眠期时，所有的载波单元都不接收数据；
- 当 UE 被激活时，所有 activated 的载波单元都将被激活以接收数据。

虽然 DRX 降低了 UE 的功耗，但 *CA* 可能进一步提高功耗，因此，LTE 提供了载波单元的 activation/deactivation 机制。（详见我的博客中关于 *CA* 的介绍）

关于 RRC_IDLE 态下的 DRX，请参见参考资料中的[7]，这篇文章介绍得相当详细。

【参考资料】

[1] 36.321的5.7节、6.1.3.3节和7.7节
[2] 36.300的12章
[3] 《LTE - The UMTS Long Term Evolution, 2nd Edition》的4.4.2.5节
[4] 《4G LTE/LTE-Advanced for Mobile Broadband》的13.2.6节
[5] 36.321的*DRX-Config*
[6] 36.300的12章　　DRX in RRC_CONNECTED
[7] 《Discontinuous Reception (DRX) in RRC_IDLE: Part 1》和《Discontinuous Reception (DRX) in RRC_IDLE: Part 2》

作者：温金辉