

**产品型号**

**版本号V001R001**

**BLE协议学习整理**

**深圳市银河风云网络系统股份有限公司**

**Shenzhen GALAXYWIND Network Systems Co., Ltd.**

地址：深圳市高新技术产业园区北区新西路5号银河风云大厦

Address: GALAXYWIND building, No.5 Xinxi road, Shenzhen High-Tech Industry Park, NanShan district, China

邮编 P.C.: 518055

电话 Tel: +86-755-83400088

传真 FAX: +86-755-86139063

网站 Web: http://[www.galaxywind.com](http://www.galaxywind.com)

客服 Hotline: 4000009879

企业QQ: 4000009879

微信公众号 WeChat:



**版权声明**

版权所有©深圳市银河风云网络系统股份有限公司2015。深圳市银河风云网络系统股份有限公司（以下简称"银河风云"）对本资料进行版权声明，未经银河风云书面许可，任何单位及个人不得以任何方式或理由对本资料的任何内容进行复制、修改、抄录、传播。

**商标声明**

"银河风云"，"GALAXYWIND"等是深圳市风云实业有限公司的注册商标，本文档提及的其他商标、服务标志、注册商标及注册服务标志均为其所有者拥有。

**免责声明**

本资料会进行不定期更新，本资料中的所有陈述、信息和建议不构成任何明示或暗示的担保。

目录

[1 BLE简介 4](#_Toc503192247)

[2 BLE核心系统结构介绍 5](#_Toc503192248)

[3 BLE之物理层（PHY） 6](#_Toc503192249)

[3.1 频率和信道 6](#_Toc503192250)

[3.2 发送器特点 6](#_Toc503192251)

[3.3 发送功率 6](#_Toc503192252)

[3.3.1.1 发送功率等级 6](#_Toc503192253)

[3.3.1.2 调制方式 7](#_Toc503192254)

[3.3.1.3 速率和频偏 8](#_Toc503192255)

[3.4 接收器特点 8](#_Toc503192256)

[3.4.1 实际接收灵敏度水平 8](#_Toc503192257)

[4 BLE之链路层（Link Layer） 9](#_Toc503192258)

[4.1 链路层状态机 9](#_Toc503192259)

[4.2 传输顺序 11](#_Toc503192260)

[4.3 器件地址（Device Address） 12](#_Toc503192261)

[4.4 信道 13](#_Toc503192262)

[5 AIR INTERFACE PACKETS 15](#_Toc503192263)

[5.1 包格式（LE Uncoded PHYS） 15](#_Toc503192264)

[5.1.1 Preamble 16](#_Toc503192265)

[5.1.2 Access Address 16](#_Toc503192266)

[5.1.3 CRC 16](#_Toc503192267)

[5.2 PDU 16](#_Toc503192268)

[5.2.1 广播信道PDU格式 16](#_Toc503192269)

[5.2.2 Advertising channel PDU Header 16](#_Toc503192270)

[5.2.3 Advertising PDU类型及含义 19](#_Toc503192271)

[5.2.4 DATA信道PDU 21](#_Toc503192272)

[5.2.5 Data channel PDU Header 22](#_Toc503192273)

[6 Bit流处理流程 23](#_Toc503192274)

[6.1 错误检查（CRC） 24](#_Toc503192275)

[6.2 数据白化（DATA WHITENING） 25](#_Toc503192276)

[6.3 编码（Coding） 26](#_Toc503192277)

[7 AIR INTERFACE PROTOCOL 27](#_Toc503192278)

[7.1 广播 27](#_Toc503192279)

[7.1.1 广播信道选择 27](#_Toc503192280)

[7.1.2 广播间隔（Advertising Interval） 27](#_Toc503192281)

[7.2 BLE建立连接与连接保持 29](#_Toc503192282)

[7.2.1 创建连接 29](#_Toc503192283)

[7.2.2 连接保持 33](#_Toc503192284)

[7.2.3 连接参数更新规程 34](#_Toc503192285)

[7.2.4 连接参数的修改 35](#_Toc503192286)

[7.2.5 信道选择（Data Channel Index Selection） 37](#_Toc503192287)

[7.3 设备过滤机制 39](#_Toc503192288)

[7.3.1 白名单（White List） 39](#_Toc503192289)

[7.3.2 Advertising Filter Policy 39](#_Toc503192290)

[7.3.3 Scanner Filter Policy 39](#_Toc503192291)

[7.3.4 Initiator Filter Policy 39](#_Toc503192292)

[8 总结 41](#_Toc503192293)

前言

**概述**

本文档针对“XX产品”的“XX内容”进行了介绍。

**产品版本**

与本文档相对应的产品版本如下所示：

|  |  |
| --- | --- |
| **产品名称** | **产品版本** |
|  |  |

**阅读对象**

本文档主要适用于以下用户：

结构设计工程师【√】

硬件开发工程师【√】

软件开发工程师【√】

系统规划工程师【√】

系统维护工程师【√】

**约定**

**符号约定**

文档中可能出现的符号定义如下所示：

|  |  |
| --- | --- |
| **符号** | **说明** |
| 警告警告 | 表示有中度或低度潜在危险，如果不能避免，可能导致较严重结果。 |
| 注意注意 | 表示有潜在风险，如果忽视这些文本，可能导致不可预知的结果。 |
| 备注备注 | 表示对正文的强调和补充。 |

**通用格式约定**

|  |  |
| --- | --- |
| **格式** | **说明** |
| 黑体 | 中文：1 -6级标题，图表/表格标题采用黑体 |
| Arial | 英文：1 -6级标题，图表/表格标题采用Arial |
| 宋体 | 中文：正文采用宋体 |
| Times New Roman | 英文：正文采用Times New Roman |

**修改记录**

|  |  |  |  |
| --- | --- | --- | --- |
| **修订时间** | **修订人** | **审核人** | **修订版本** |
| 2018-1-8 | 孙世玮 |  | 草稿 |

# BLE简介

Bluetooth（蓝牙）协议包括两种技术：Basic Rate（简称BR）和Low Energy（简称LE）。这两种技术，都包括搜索（discovery）管理、连接（connection）管理等机制，但它们是不能互通的！

BR传输很快，但能量是守恒的，你想传的更快，代价就是消耗更多的能量。而有很多的应用场景，并不关心传输速率，反而非常关心功耗。这就是Bluetooth LE（称作蓝牙低功耗）产生的背景。

Bluetooth Low Energy（简称BLE）是一种低成本、低功耗低功耗(ULP)无线技术，工作在免许可的2.4GHz ISM射频频段。有BLE4.0, BLE4.1, BLE4.2, BLE 5.0等多个协议版本。

采用可变连接时间间隔，这个间隔根据具体应用可以设置为几毫秒到几秒不等。另外，因为BLE技术采用非常快速的连接方式，因此平时可以处于“非连接”状态（节省能源），此时链路两端相互间只是知晓对方，只有在必要时才开启链路，然后在尽可能短的时间内关闭链路。

BLE的共有两种芯片构成：单模芯片和双模芯片，其中单模芯片是一种只支持蓝牙低能耗技术的芯片。

BLE 将channel的个数减少为40个，并保留了不少于3个的固定channel，用于广播通信。为仅仅在剩下的37个data channel上跳频。正因为这种改变，原有的搜索/连接/配对等概念，在BLE上就不再存在了，取而代之的是Advertisor、Initiator等概念。

# BLE核心系统结构介绍

尽管每项部署都有具体要求（详见蓝牙规格），但蓝牙核心系统结构有许多统一要素。该系统包含射频收发器、基带和协议栈，支持设备连接和交换各类数据。

蓝牙设备交换根据蓝牙规格协议信号。核心系统协议包括射频 (RF) 协议、链路控制 (LC) 协议、链路管理器 (LM) 协议以及逻辑链路控制和适配协议 (L2CAP)，蓝牙规格详细定义了这些协议。

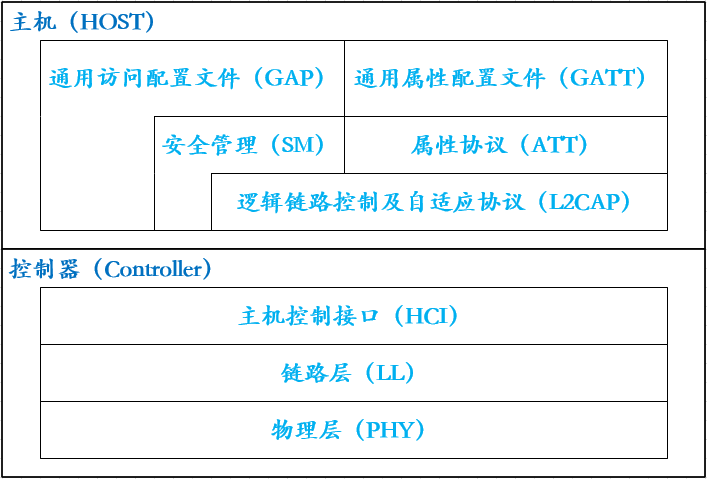
最低的三个系统层—射频、链路控制和链路管理器协议—通常被归属于称为蓝牙控制器的子系统。这是一种采用可选标准接口—主机控制器接口 (HCI)—的通用部署，支持与蓝牙系统的其他设备（即蓝牙主机）进行双向通信。

主控制器可能是以下配置之一，具体取决于用例：

* BR/EDR 控制器，包括射频、基带、链路管理器和可选 HCI
* LE 控制器，包括 LE PHY、链路层和可选 HCI
* BR/EDR 组合控制器和 LE 控制器，组合控制器共享一个蓝牙设备地址

蓝牙规格通过定义等效层之间交换的协议信息来实现系统之间的互操作性。它还通过定义蓝牙控制器和蓝牙主机之间的公用接口来实现独立蓝牙子系统之间的互操作性。

蓝牙系统结构如下图所示：



# BLE之物理层（PHY）

通过蓝牙通信信道控制 2.4Ghz 射频的传输/接收。BR/EDR 提供的信道较多但带宽较窄，而 LE 使用的信道较少但带宽较宽。

## 频率和信道

BLE工作在2.4G ISM 频段（2400-2483.5MHz），共分为40个信道，中心频率为2402 + k \* 2MHz ，其中k=0,1,……,39。

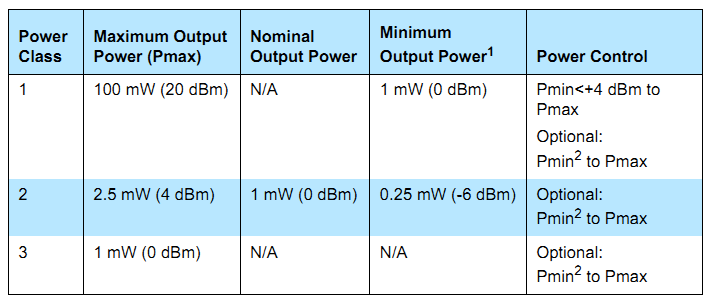
## 发送器特点

## 发送功率

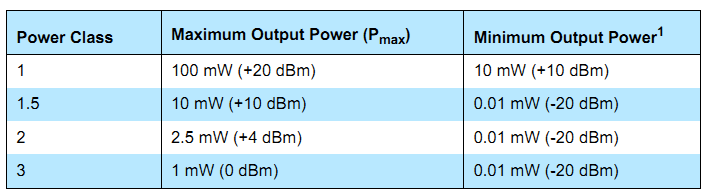
|  |  |  |
| --- | --- | --- |
| 协议版本 | 最小发送功率 | 最大发送功率 |
| V4.0 | 0.01mW(-20dBm) | 10mW(10dBm) |
| V4.1 | 0.01mW(-20dBm) | 10mW(10dBm) |
| V4.2 | 0.01mW(-20dBm) | 10mW(10dBm) |
| V5.0 | 0.01mW(-20dBm) | 100mW(20dBm) |

### 发送功率等级

V4.0、V4.1、V4.2中的定义：



V5.0中的定义：



### 调制方式

The modulation is Gaussian Frequency Shift Keying (GFSK) with a bandwidth-

bit period product BT=0.5. The modulation index shall be between 0.45 and

0.55. A binary one shall be represented by a positive frequency deviation, and

a binary zero shall be represented by a negative frequency deviation.

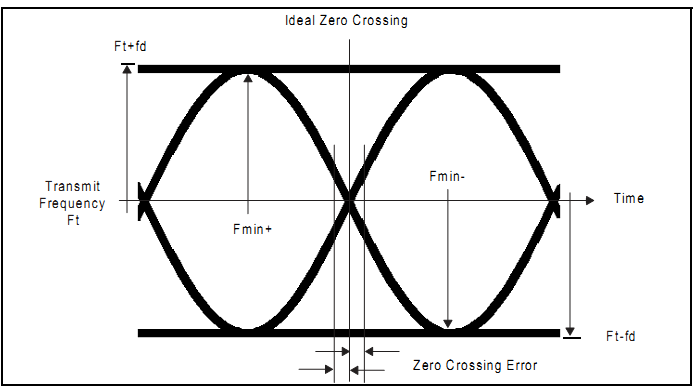


Figure 3.1: GFSK parameters definition

For each transmission the minimum frequency deviation,

Fmin = min { |Fmin+ | , Fmin- }

which corresponds to a 1010 sequence, shall be no smaller than ±80% of the

frequency deviation with respect to the transmit frequency, which corresponds

to a 00001111 sequence.

**The minimum frequency deviation shall never be less than 185 kHz when**

**transmitting at 1 megasymbol per second (Msym/s) symbol rate and never be**

**less than 370 kHz when transmitting at 2 Msym/s symbol rate. The symbol**

**timing accuracy shall be better than ±50 ppm**.

The zero crossing error is the time difference between the ideal symbol period

and the measured crossing time. This shall be less than ±1/8 of a symbol

period.

### 速率和频偏

|  |  |  |
| --- | --- | --- |
| 协议版本 | 速率 | 频偏 |
| V4.0 | 1Msym/s±1 ppm | never be less than 185kHz |
| V4.1 | 1Msym/s±1 ppm | never be less than 185kHz |
| V4.2 | 1Msym/s±1 ppm | never be less than 185kHz |
| V5.0 | 1Msym/s±1 ppm，  ,2Msym/s±1 ppm | never be less than 185kHz at 1 Msym/s  never be less than 370kHz at 2 Msym/s |

频率误差

中心频率误差不能超过±150 kHz，频率漂移要小于400 Hz/µs。

## 接收器特点

### 实际接收灵敏度水平

|  |  |
| --- | --- |
| 协议版本 | 频偏 |
| V4.0 | ≤﹣70dBm@1Msym/s, BER<0.1% |
| V4.1 | ≤﹣70dBm@1Msym/s, BER<0.1% |
| V4.2 | ≤﹣70dBm@1Msym/s  Payload length ≤37byte,BER<0.1%  Payload length ≥38byte and ≤63Byte,BER<0.064%  Payload length ≥64byte and ≤127Byte,BER<0.034%  Payload length ≥128Byte,BER<0.017% |
| V5.0 | LE Uncoded PHYs ≤﹣70dBm  LE Coded PHY with S=2 coding ≤﹣75dBm  LE Coded PHY with S=8 coding ≤﹣82dBm  Payload length ≤37byte,BER<0.1%  Payload length ≥38byte and ≤63Byte,BER<0.064%  Payload length ≥64byte and ≤127Byte,BER<0.034%  Payload length ≥128Byte,BER<0.017% |

# BLE之链路层（Link Layer）

BLE的Link层，应当是了解BLE需要首先熟悉的一部分，BLE的Controller部分主要都在围绕这一部分实现的。Link层定义数据包结构/信道、发现/连接程序以及发送/接收数据。

## 链路层状态机

Link层使用了状态机，即通过不同的事件的发生来切换BLE的不同状态，这样使整个蓝牙通信的实现显得更加清晰。链路层定义设备处于状态机中五种状态的一种，并以如下的状态机进行切换（设备在同一时刻只能处于这些状态的一种）：

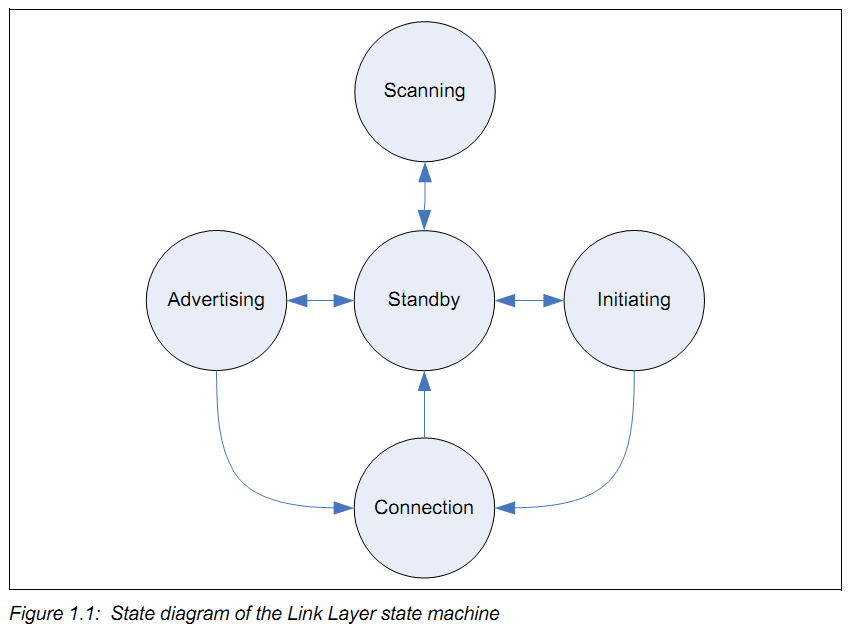
• Standby State 准备

• Advertising State 广播

• Scanning State 监听

• Initiating State 发起连接

• Connection State 已连接



（1）就绪态；

上电后，链路层进入并保持就绪态，直到接到主机的命令。从就绪态可进入广播态、扫描态或发起态，如图2所示。从其他任意状态也可以进入就绪态。就绪态是链路层状态机的中心状态。

（2）广播态；

处于广播态的链路层可以发送广播报文，也可以发送扫描响应，用以回应主动扫描设备。可被发现或者可被连接的设备需要处于广播态。想向一定区域内其他设备广播数据的设备也需要处于广播态。

广播态的设备停止广播后可进入就绪态。在收到发起者的连接请求之后，广播态的设备也可以进入连接态。

（3）扫描态；

扫描态可监听那些设备正在广播。扫描态有两个字状态：被动扫描和主动扫描。被动扫描进接收广播报文。主动扫描则发送扫描请求给广播态设备、并获取附加的扫描响应数据。扫描态的设备只能进入就绪态，转换条件就是停止扫描。

（4）发起态；

为了发起连接，链路层需要处于发起态，如下图所示。处于发起态的发起者，侦听自己想要连接的设备，如果收到了来自该设备的广播报文，链路层会向其发起连接请求并进入连接态，并假设广播者也进入了连接态。如果发起者不在试图发起连接，也可以进入就绪态。

（5）连接状态；

从广播态或发起态都可进入连接态，如下图所示。连接态有两种子状态：主或从，也可以说是两种身份。主设备只能从发起态进入连接态，而从设备只能从广播态进入连接态。从设备不断向主设备进行广播，主设备则发起连接，这样双方都进入了连接态。进入连接态后，主设备必须定期向从设备发送报文，从设备只能通过回复这些报文才能发送自己的报文。

## 传输顺序

**此部分等最终抓包分析后再确认。**

bit序：LSB send first。（CRC除外）

byte序：least significantoctet send first。（CRC和MIC除外）

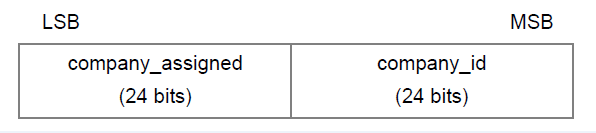
CRC传输顺序：most significantoctet send first，MSB send first。详见3.4。

MIC：软件处理，芯片不关心。

## 器件地址（Device Address）

包含公共器件地址（Public Device Address）和随机器件地址Random Device Address），随机器件地址（又包含静态地址（Static Address）和私有地址（Private Address）

Public Device Addrss：公有设备地址是设备所特有的并且是不可改变的。类似网络设备的MAC地址，它的长度为48位。由两部分组成：



Ramdom Device Address：随机设备地址（私有设备地址），它也是48位。组成如下所示：



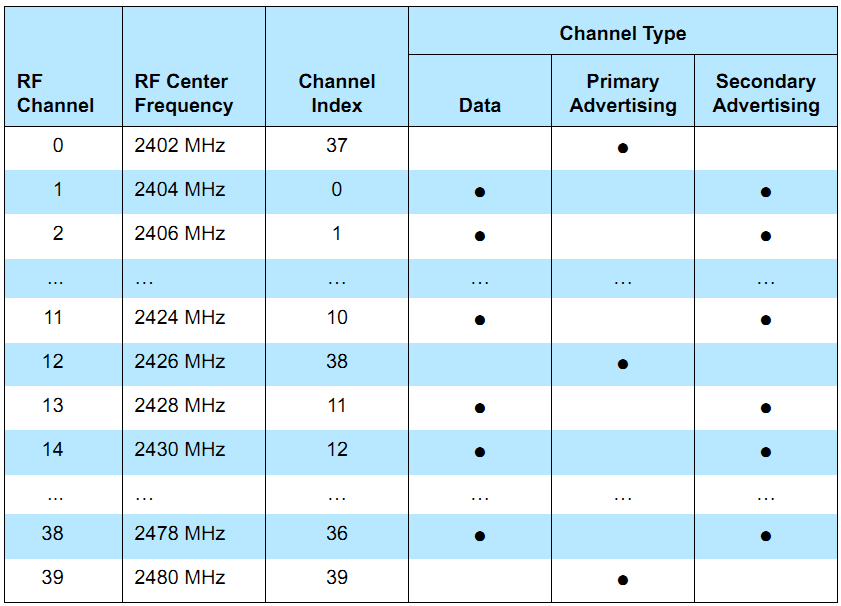
## 信道

总共40个物理信道。

蓝牙4.0/4.1/4.2的广播通道，比较简单、直接，预留3个（37（chanel 0）,38（chanel 12）,39（channel 39））如下表所示，用于传输Advertising Event。可传输的数据长度为6～37 octets（加上了协议开销）。剩下的37个信道为数据信道。

而蓝牙5.0，抽象出primary advertising channel和secondary advertising channel的概念。primary advertising channel就是蓝牙4.2及以前的、预留出的、用于传输Advertising Event。

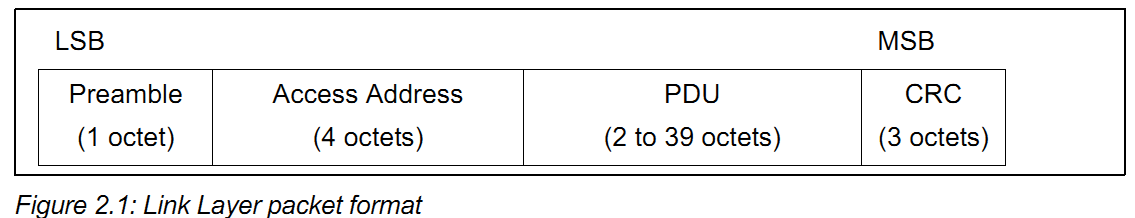
而secondary advertising channel，则直接复用了剩余的37个data channel，用于传输扩展的Advertising Event（称作Extended Advertising Event）。此时可传输的数据长度为0 ～ 255 octets。



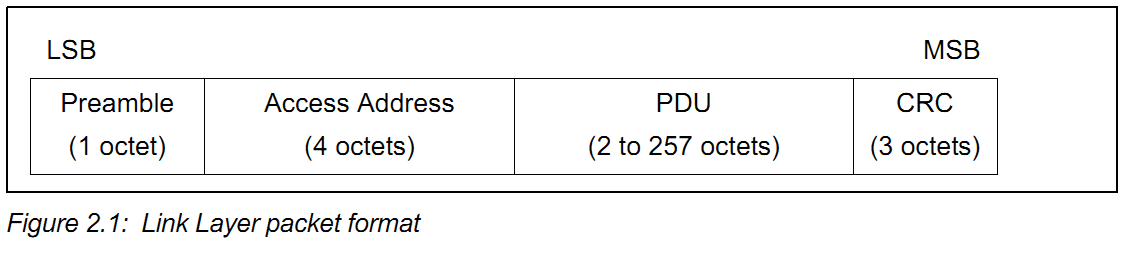
# AIR INTERFACE PACKETS

## 包格式（LE Uncoded PHYS）

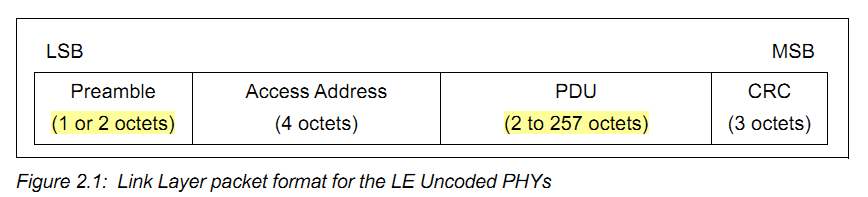
V4.0/v4.1



V4.2



V5.0



### Preamble

发送bit序： 0,1,0,1,0,1,0,1（如果Access Address LSB为0）

1,0,1,0,1,0,1,0（如果Access Address LSB为1）

**蓝牙4.0/4.1/4.2的preamble为1个字节，而蓝牙5.0的preamble为2个字节**

### Access Address

用来做报文过滤，广告信道Access Address固定为**0x8E89BED6**，数据信道Access Address随机分配。芯片支持一个Access Address可配置就够了。

广告信道的Access Address的LSB为0，所以前导发送顺序为：0,1,0,1,0,1,0,1。

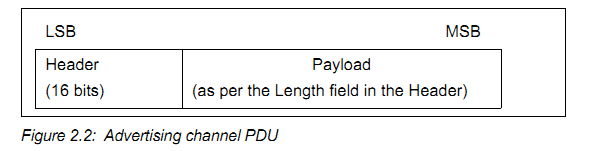
### CRC

24位CRC，用于对报文的正确性进行校验，**校验范围为报文中的PDU部分。**

## PDU

PDU分为广播信道PDU和DATA信道PDU

### 广播信道PDU格式



包含2字节头部和payload；

**蓝牙4.0/4.1/4.2：**

所有广播信道的报文PDU长度最大37字节。

**蓝牙5.0：**

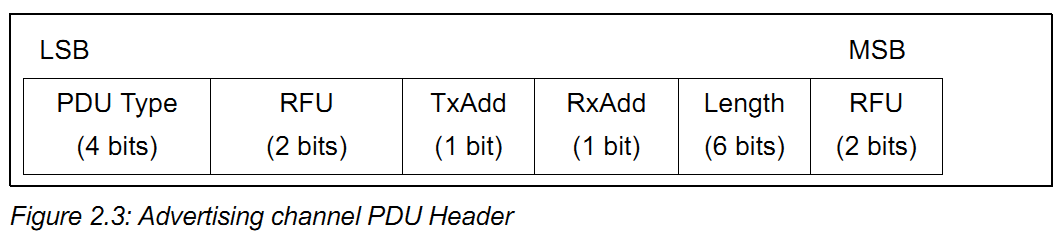
蓝牙5.0原有的用于传输广播数据的PDU（ADV\_IND、ADV\_DIRECT\_IND、ADV\_NONCONN\_IND以及ADV\_SCAN\_IND，称作legacy PDUs）的基础上，增加了扩展的PDU（ADV\_EXT\_IND、AUX\_ADV\_IND、AUX\_SYNC\_IND以及AUX\_CHAIN\_IND，称作extended advertising PDUs）。

对应的Advertising Event也分为Legacy Advertising Event和Extenteded Advertising Event。

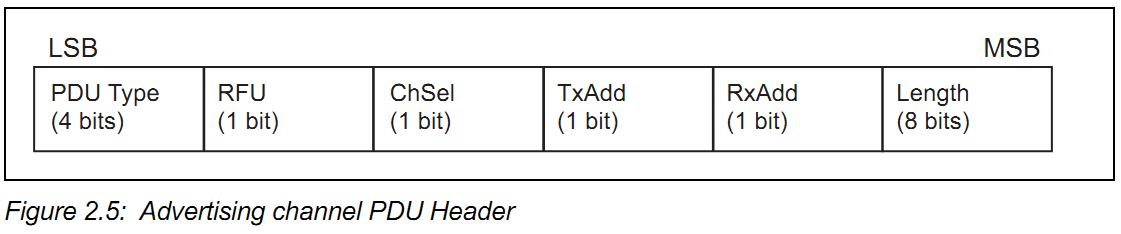
Legacy Advertising Event可传输的数据长度还是为**6～37 octets**，而Extenteded Advertising Event可传输的数据长度为**0～255 octets**。

### Advertising channel PDU Header

v4.0/v4.1/v4.2



v5.0



PDU Type，指示PDU的类型，具体可参考后面的介绍。

RFU，reserved for future use。

TxAdd、RxAdd，由具体的PDU Type决定其意义。

Length，PDU的长度。

PDU Type: 广播报文类型

蓝牙v4.0/v4.1/v4.2有7中类型

ADV\_IND：通用广播指示

ADV\_DIRECT\_IND：定向连接指示

ADV\_NONCONN\_IND：不可连接指示

ADV\_SCAN\_IND：可扫描指示

SCAN\_REQ：主动扫描请求

SCAN\_RSP：主动扫描响应

CONNECT\_REQ：连接请求

蓝牙V5.0新增中广播报文类型

AUX\_SCAN\_REQ

ADV\_EXT\_IND、

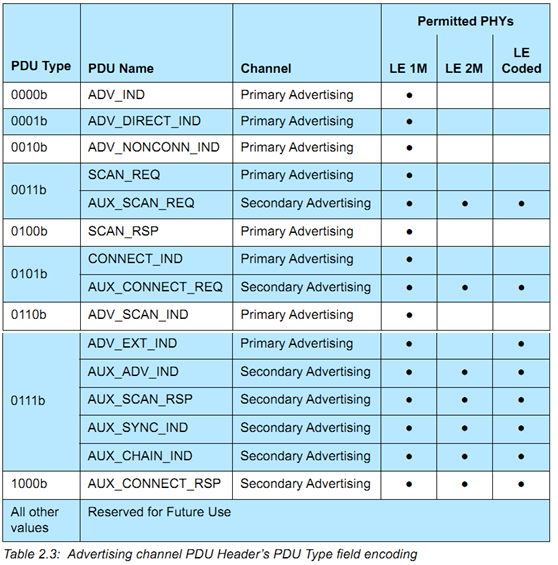
AUX\_ADV\_IND、

AUX\_SCAN\_RSP

AUX\_SYNC\_IND、

AUX\_CHAIN\_IND

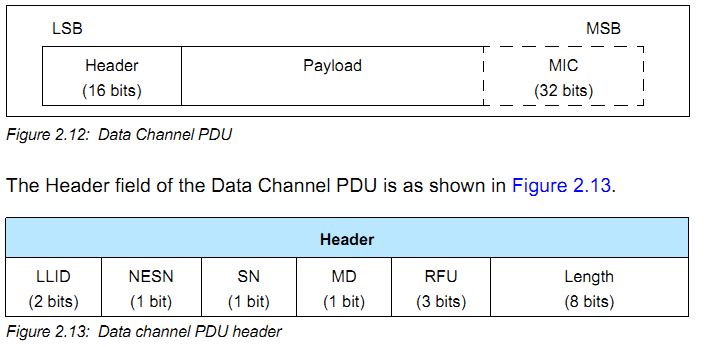
AUX\_CONNECT\_RSP



**Advertising PDU类型及含义**

|  |  |  |  |
| --- | --- | --- | --- |
| 状**态** | **PDU类型** | **PDU格式** | **说明** |
| Advertising | ADV\_IND | AdvA(6 octets)  AdvData(0~31 octets) | connectable undirected advertising event，用于常规的广播，可携带不超过31bytes的广播数据，可被连接，可被扫描：  AdvA，6bytes的广播者地址，并由PDU Header的TxAdd bit决定地址的类型（0 public，1 random）；  AdvData，广播数据。 |
| ADV\_DIRECT\_IND | AdvA(6 octets)  InitA(6 octets) | connectable directed advertising event，专门用于点对点连接，且已经知道双方的蓝牙地址，不可携带广播数据，可被指定的设备连接，不可被扫描：  AdvA，6bytes的广播者地址，并由PDU Header的TxAdd bit决定地址的类型（0 public，1 random）；  InitA，6bytes的接收者（也是连接发起者）地址，并由PDU Header的RxAdd bit决定地址的类型（0 public，1 random）。 |
| ADV\_NONCONN\_IND | AdvA(6 octets)  AdvData(0~31 octets) | 和ADV\_IND类似，但不可以被连接，不可以被扫描。 |
| ADV\_SCAN\_IND | AdvA(6 octets)  AdvData(0~31 octets) | 和ADV\_IND类似，但不可以被连接，可以被扫描。 |
| Scanning | SCAN\_REQ | ScanA(6 octets)  AdvA(6 octets) | 当接收到ADV\_IND或者ADV\_SCAN\_IND类型的广播数据的时候，可以通过该PDU，请求广播者广播更多的信息：  ScanA，6bytes的本机地址，并由PDU Header的TxAdd bit决定地址的类型（0 public，1 random）；  AdvA，6bytes的广播者地址，并由PDU Header的RxAdd bit决定地址的类型（0 public，1 random）。 |
| SCAN\_RSP | AdvA(6 octets)  ScanRspData(0~31 octets) | 广播者收到SCAN\_REQ请求后，通过该PDU响应，把更多的数据传送给接受者。  AdvA，6bytes的广播者地址，并由PDU Header的TxAdd bit决定地址的类型（0 public，1 random）；  ScanRspData，scan的应答数据。 |
| Initiating | CONNECT\_REQ | InitA (6 octets)  AdvA (6 octets)  LLData (22 octets) | 当接收到ADV\_IND或者ADV\_DIRECT\_IND类型的广播数据的时候，可以通过该PDU，请求和对方建立连接：  InitA，6bytes的本机地址，并由PDU Header的TxAdd bit决定地址的类型（0 public，1 random）；  AdvA，6bytes的广播者地址，并由PDU Header的RxAdd bit决定地址的类型（0 public，1 random）；  LLData，BLE连接有关的参数信息，具体请参考后续文章的介绍。 |

### DATA信道PDU



包含2字节头部和payload；所有DATA channel PDUs 且 Payload部分长度不为0的报文都需要加密。

加密报文增加了一个32位MIC在payload后，芯片可不关心，由软件处理。只有payload和MIC会加密，header不加密。

**注意：只有DATA信道PDU才用到MIC。**

**MIC：Payload和header部分（NESN、SN、MD取0）的认证码，算法是AES-128bit中的计数器（counter）模式。**

**另外,报文的加密和认证采用的AES-128的CCM模式。**

**TI的蓝牙芯片都有AES-128bit协处理器，建议我们的SOC也由硬件实现AES-128bit算法。**

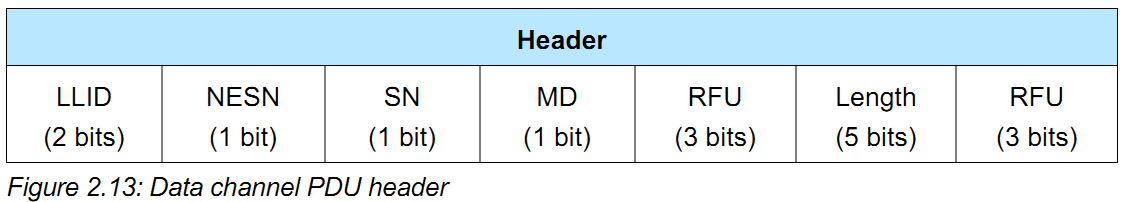
TI蓝牙芯片中对AES core的描述：

****

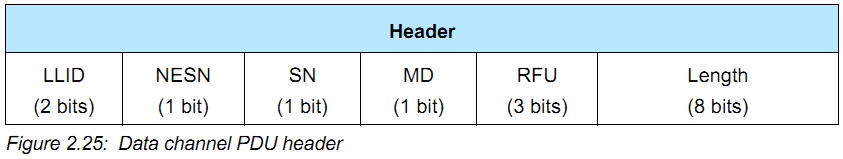
**AES-128算法定义：**[**http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf**](http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf)

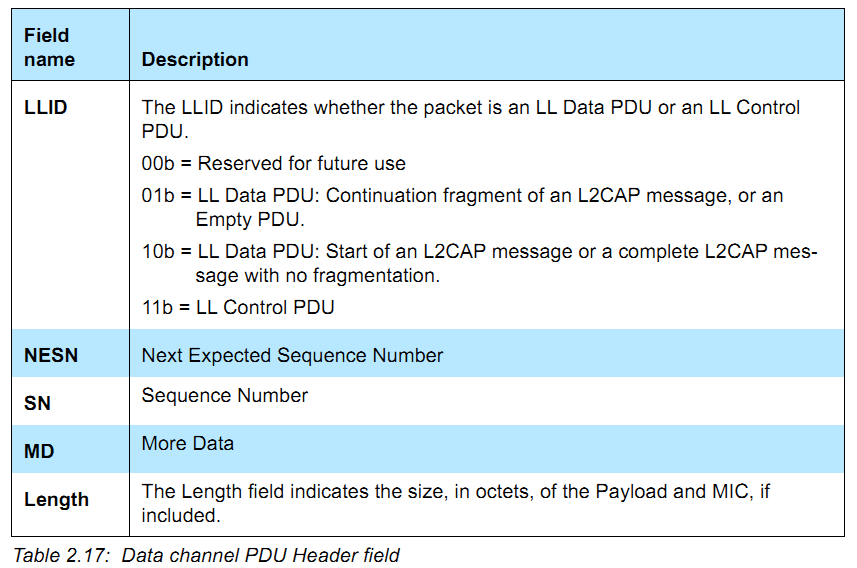
### Data channel PDU Header

v4.0/v4.1



v4.2及以上

****



**LLID（逻辑链路标识符）**：

LLID用来判断数据报文属于下列哪种类型：

链路层控制报文（11）——用来管理连接

高层报文开始（10）——也可用于一个完整报文

高层报文延续（01）

**SN（序列号）：**

在数据包中，用一个比特来表示序列号；在发送的第一个数据包中将该位设置为0，接下来当设备发送新的数据包时，该值在1和0之间交替。这使得接收装置可以根据序列号判断接收的数据包的性质，如果序列号和之前的一样，则为重传报文，如果序列号和之前的不同，则为新报文。

**NESN（下一个预期序列号）**

数据包的确认需要用到另一个比特，即下一个预期序列号（NESN）。NESN的发送方用其通知对方自己期望接收的数据包的序列号。如果设备成功接收序列号为0的报文，在其确认报文中，应将NESN设为1，否则序列号为0的数据包将被重传。

**MD（更多数据）**

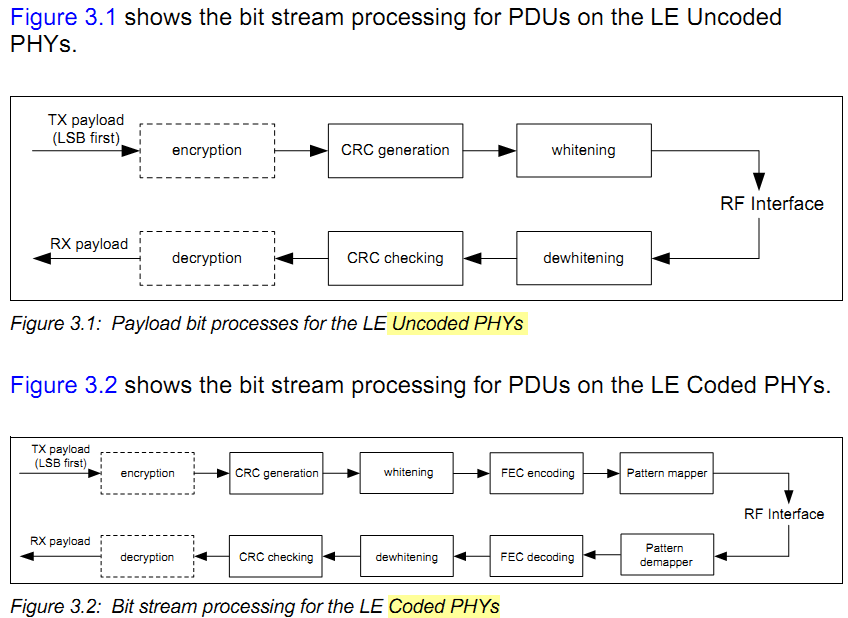
更多数据位用来通知对端设备自己还有其他数据准备发送。如果收到MD=1的数据包，应该在当前连接事件中继续与对端设备通信。这样一来，只要还有数据要发送，连接事件就会自动扩展；一旦不再有数据发送，连接事件会迅速关闭。

**补充说明**

SN和NESN总是处于互锁状态，确保了可靠数据送达。 可以通过NESN进行流量控制，设备一旦没有足够的缓冲区空间来处理消息，可以不更新NESN。这将迫使对端设备重新发送当前消息，从而把对缓存的要求从接收设备转移到发送设备。

如果由于一些位的错误导致CRC校验失败，数据包接收出错，这种情况一旦在同一连接事件里发生两次，设备将立刻停止当前事件，并在下次连接事件中重新同步和尝试传输。

# Bit流处理流程



## 错误检查（CRC）

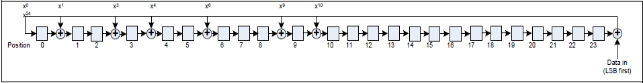
24位CRC，多项式：x24 + x10 + x9 + x6 + x4 + x3 + x + 1

初始值：广告信道是0x555555，其余信道可根据协商设置，统一做成可设置即可。

计算范围：PDU。

处理顺序：按照bit发送顺序。

LFSR：

初始值LSB放在POS0，MSB放在POS23。

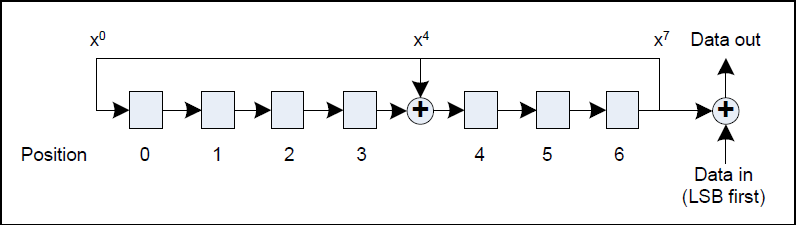
CRC发送顺序：从POS23到POS0。

## 数据白化（DATA WHITENING）

**处理范围：PDU和CRC。**

多项式：x7 + x4 + 1。

LFSR：



白化/解白化前需放入初始值，方法：

POS0=1，

POS1 ~ POS6：channel index，MSB in POS1，LSB in POS6。

例如：channel index=23（0x27）注意是下表右边的才是channel index。

Position 0 = 1

Position 1 = 0

Position 2 = 1

Position 3 = 0

Position 4 = 1

Position 5 = 1

Position 6 = 1

## 编码（Coding）

**蓝牙V5.0中的coded phy用到了编码处理。**

Coding **only applies** to the LE Coded PHY.

Coding consists of two processes. Data is first coded by the Forward Error Correction (FEC) convolutional encoder as defined in Section 3.3.1 and then spread by the pattern mapper as defined in Section 3.3.2.

3.3.1 Forward Error Correction Encoder

The convolutional FEC encoder uses a non-systematic, non-recursive rate ½ code with constraint length K=4. The generator polynomials are:

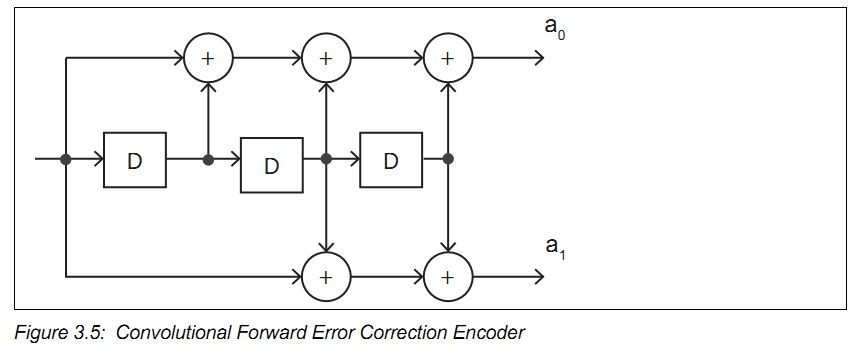
G0(x) = 1 + x + x2 + x3

G1(x) = 1 + x2 + x3

The bit coming from generator polynomial G0 (a0) is transmitted first; the bit coming from generator polynomial G1 (a1) is transmitted second.

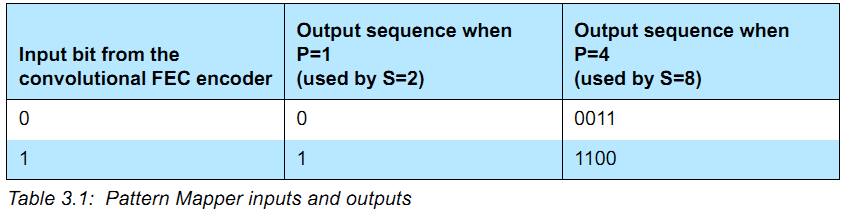
The initial state of the convolutional FEC encoder is set to all zeros. An input sequence of three consecutive zeros always brings the convolutional FEC encoder back to its original state. This sequence is known as the termination sequence.

Figure 3.5 illustrates operation of the convolutional FEC encoder. Squares represent bit storage operations and circles represent mod 2 binary additions.



3.3.2 Pattern Mapper

The pattern mapper converts each bit from the convolutional FEC encoder into P symbols, where the value of P depends on the coding scheme in use, according to Table 3.1 (the entries in the table are in transmission order):



# AIR INTERFACE PROTOCOL& LINK LAYER CONTROL

BLE的两种组网方式：

和传统蓝牙类似的基于连接的星型网络（由1个master和多个slave组成）；

无链接的网状网络（由多个advertiser和多个scanner组成）。

使用场景一：数据量比较少、发送不频繁、对时延不是很敏感：

例如一个传感器节点（如温度传感器），需要定时（如10s）向处理中心发送传感器数据（如温度）。针对这种场景，BLE的link layer采取一种比较懒的通信方式-广播通信。

使用场景二：数据量较大、发送频率较高、对时延较敏感：

BLE的Link Layer会从剩余的37个Physical Channel中，选取一个，为这种场景里面的通信双方建立单独的通道（data channel）。这就是连接（connection）的过程。同时，为了增加容量，增大抗干扰能力，连接不会长期使用一个固定的Physical Channel，而是在多个Channel（如37个）之间随机但有规律的切换，这就是BLE的跳频（Hopping）技术。

## 广播

低功耗蓝牙设备通过广播信道发现其他设备，一个设备进行广播，而另一个设备进行扫描。

广播相关的参数大致有以下几种：

1.Advertising interval

2.Advertising\_Type

3.Own\_Address\_Type

4.Direct\_Address\_Type

5.Direct\_Address

6.Advertising\_Channel\_Map

7.Advertising\_Filter\_Policy

8.Advertising Data

9.ScanReponse Data

### 广播信道选择

**对于蓝牙4.0/4.1/4.2，**BLE可以使用40个Physical Channel中的3个作为广播通信的物理信道，综合各种因素（抗干扰等），最终选取了如下三个：

|  |  |  |
| --- | --- | --- |
| RF Channel | RF Center Frequency | Advertising Channel Index |
| 0 | 2402MHz | 37 |
| 12 | 2426MHz | 38 |
| 39 | 2480MHz | 39 |

与此同时，Link Layer允许Host在这这三个物理信道中，任意选取一个或者多个，用于广播。Link Layer将相同的广播数据，在每一个被中的Channel中，发送一次。

### 广播间隔（Advertising Interval）

BLE有3个广播信道，37个数据信道。设备每次广播时，会在3个广播信道上发送相同的报文。这些报文被称为一个广播事件。除了定向报文以外，其他广播事件均可以选择“20ms ~ 10.28s”不等的间隔。通常，一个广播中的设备会每一秒广播一次。两个相邻广播事件之间的时间称为广播间隔。

但是，设备周期性的发送广播会有一个问题：由于设备间的时钟会不同程度的漂移，两个设备可能在很长一段时间同时广播而造成干扰。为防止这一情况的发生，除定向广播之外的其他广播类型，发送时间均会被扰动。BLE的链路层是如何解决不同BLE节点的冲突问题呢？

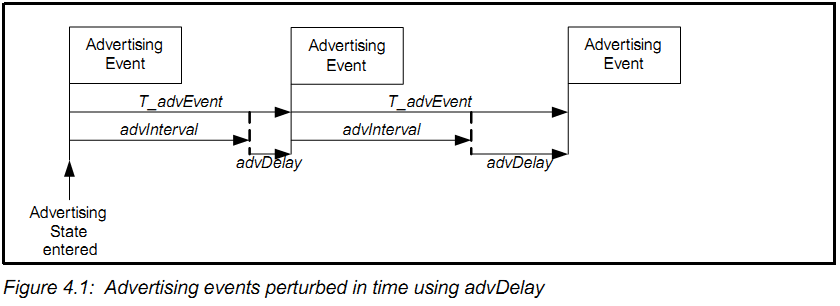
在节点进入advertising state时，它会广播advertising events，协议中规定两个相邻的广播事件的之间的时间间隔（T\_advEvent）为：

T\_advEvent = advInterval +advDelay

其中，advInterval 必须是“0.625ms”的整数倍，范围是“20ms ~ 10.24s”之间。对于可扫描非定向广播（Scannable Undirected）和不可连接非定向广播（Non-connectable Undirected）这两种广播类型，该值最好不小于100ms，即（160个0.625ms）；对于可连接非定向广播（Connectable Undirected）和可连接定向广播（Low Duty Cycle Connectable Directe）两种Advertising Event，该值不能小于20ms（建立连接嘛，要快点）。

advDelay是Link Layer（链接层）分配的一个伪随机数，它的范围为“0 ~ 10ms”。

广播包的截图如下：



从上面的公式可以看出，如果advInterval取值较大，将会降低出现冲突的概率，也降低了节点的功耗，但是，同时也会加大连接建立的时延。反之，如果advInterval取值较小，将会增加冲突概率和节点功耗，好处就是降低连接建立时延。随机数advDelay可以在一定程度上降低advInterval相同的两个节点所发送的广播包的冲突概率。

蓝牙4.0将Scannable Undirected和Non-connectable Undirected两种Advertising Event的advInterval的最小值限制为100ms，这就限制了BLE广播的最高速率（2.48kbps，参考[4]）。而蓝牙5.0不再区别对待，将最小值统一限制为20ms，从理论上讲，最高的广播速率就可以提高5倍（12.4kbps）。

High Duty Cycle Connectable Directed Event则是一个比较狂暴的家伙，其Advertising周期不受上面的参数控制，可以小到3.75ms。不过呢，不过呢，BLE协议也同时规定：Link Layer必须在**1.28s**内退出这种狂暴状态。

### 总结

有关广播通信的PDU类型，总结如下：

1）如果只需要定时传输一些简单的数据（如某一个温度节点的温度信息），后续不需要建立连接，则可以使用ADV\_NONCONN\_IND。广播者只需要周期性的广播该类型的PDU即可，接收者按照自己的策略扫描、接收，二者不需要任何额外的数据交互。

2）如果除了广播数据之外，还有一些额外的数据需要传输，由于种种原因，如广播数据的长度限制、私密要求等，可以使用ADV\_SCAN\_IND。广播者在周期性广播的同时，会监听SCAN\_REQ请求。接收者在接收到广播数据之后，可以通过SCAN\_REQ PDU，请求更多的数据。

3）如果后续需要建立点对点的连接，则可使用ADV\_IND。广播者在周期性广播的同时，会监听CONNECT\_REQ请求。接收者在接收到广播数据之后，可以通过CONNECT\_REQ PDU，请求建立连接。

4）通过ADV\_IND/CONNECT\_REQ的组合建立连接，花费的时间比较长。如果双方不关心广播数据，而只是想快速建立连接，恰好如果连接发起者又知道对方（广播者）的蓝牙地址（如通过扫码的方式获取），则可以通过ADV\_DIRECT\_IND/CONNECT\_REQ的方式。

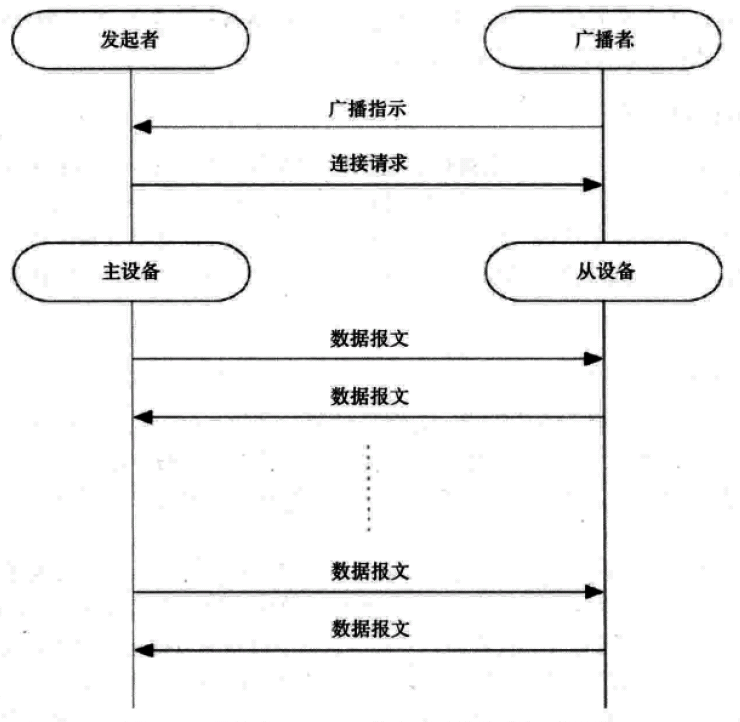
BLE协议对广播通信的期望，是非常明确的----不在乎速率、只在乎功耗。一般的广播通信（不以连接为目的），最高速率也就是31byte / 100ms = 2.48kbps。如果再算上可扫描的那段数据，也就是double，4.96kbps。

注3：对于连接来说，如果事先不知道连接发起者的设备地址，则最快的连接速度可能是20ms。如果事先知道地址，使用High Duty Cycle Connectable Directed Event的话，则可能在3.75ms内建立连接。由此可以看出，BLE的连接建立时间，比传统蓝牙少了很多，这也是BLE设备之间不需要保持连接的原因。

## BLE建立连接与连接保持

### 创建连接

应付比广播更为复杂的数据传输，或者要在设备之间实现可靠的数据传输，这些都依赖于连接。连接使用数据信道，采用自适应跳频增强鲁棒性，同时使用了非常低的占空比，尽可能地降低功率消耗。设备创建连接的过程如下图所示。



1、设备首先广播可连接广播事件，其他设备收到之后即可发起连接。在此过程中，广播者发送的事件类型要么是通用广播事件，要么是定向广播事件。

2、发起者收到正确的广播报文时，将向广播者发送一个连接请求，其中包括了连接开始时需要的所有信息。包含以下几项

i.连接中使用的接入地址

ii.CRC初始值

iii.发送窗口大小

iv.发送窗口偏移

v.连接间隔

vi.从设备延迟

vii.监控超时

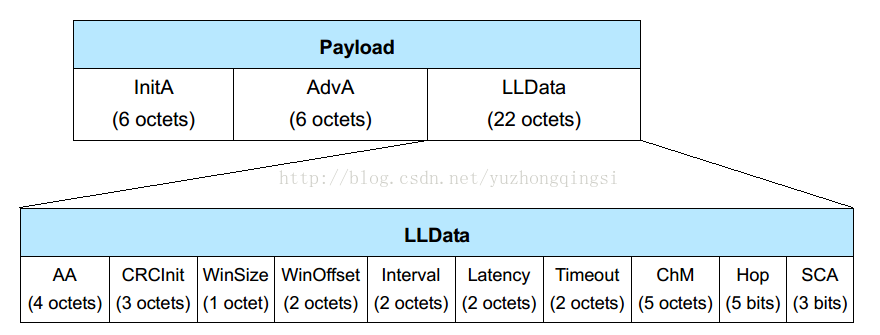
viii.自适应跳频信道图

ix.跳频算法增量

x.休眠时钟精度

3、一旦收到或发出连接请求报文，设备即建立了连接，数据交换随之开始

CONNECT\_REQ事件包格式如下：



AA：接入地址（Access Address），

接入地址总是由主设备来提供。地址通过随机生成，但是需要遵循一些规则。如果主设备有多个从设备，它会为每个从设备选择不同的随机接入地址。地址的随机性确保了在不同的主从设备之间发生的碰撞概率会很低。随机性也增强了隐私，扫描者无法得知是哪两个设备正在通信。

CRCInit：CRC初始值，

CRC初始值是另一个由主设备提供的随机数。随机的意义在于，如果在同一个区域有两个主设备正在和不同从设备通信，那么使用相同的接入地址的概率将会很低。如果确实发生了这种情况，从设备会从错误的主设备收到干扰数据包。因此，为每个从设备设置随机的CRC初始值，主从设备既有相同的接入地址又有相同的CRC初始值的概率就会变得微乎其微。

WinSize（发送窗口大小）：transmitWindowSize =WinSize \* 1.25 ms

WinOffset（发送窗口偏移）：transmitWindowOffset =WinOffset \* 1.25 ms.

WinSize和WinOffset一起用于定义发送窗口，当连接请求数据包发送完成之后，存在一个1.25ms的强制时延。紧接着是一个等待时间（发送窗口偏移值，由连接请求中的发送窗口偏移来定义，必须为1.25ms的整数倍（包括0倍））和一个发送侦听时间（发送窗口大小值）。

也就是说，在扫描者发送连接请求后，扫描者变成了主设备。然后等一个1.25ms的强制时延，之后再等发送窗口偏移值（0或者若干个1.25ms），这个时间留给广播者接收连接请求并进入从设备，也让主设备有时间准备要发送的数据报文。从发送窗口开始，从设备打开接收器，等待来自主设备的数据报文。如果直到发送窗口结束都没有收到数据报文，从设备终止侦听，并会在一个连接的间隔后再次尝试。

对于主设备而言，发送了连接请求只是表明连接已经created，并没有established。同理，从设备收到了连接请求，也是只表明连接created，并没有established。只有收到了数据报文，才能证明连接established。

Interval（连接间隔）：connInterval（连接间隔） = Interval \* 1.25 ms

Latency （从设备延迟）：connSlaveLatency （从设备延迟）=Latency

Timeout （监控超时）：connSupervisionTimeout （监控超时）= Timeout \* 10 ms（默认2S）

在一个连接当中，主设备会在每个连接事件里向从设备发送数据包。一个连接事件是指主设备和从设备之间相互发送数据包的过程。连接事件的进行始终位于一个频率，每个数据包会在上个数据包发完之后等待150us再发送。

连接间隔决定了主设备和从设备的交互间隔，它是指两个连续的连接事件开始处的时间距离，可以是7.5ms~4s内的任意值，但必须为1.25ms的整数倍。

从设备延迟是连接间隔的倍数，代表从设备在必须侦听之前可以忽略掉多少个连接事件。从设备延迟必须短于监控超时时间。

连接事件被一个个连接间隔分开。从主设备发送数据包开始，每个连接事件可以持续进行，直到主设备或从设备停止响应。在连接事件之外，主从设备之间不发送任何数据包。

chM：自适应跳频信道图

hop：跳频增量

自适应跳频信道图是数据信道的位掩码，用来标记信道的好坏。由于共有37个数据信道，信道图的长度于是设为37位。如果某一位被设置为1，表明该信道良好，可用于数据通信；如果该位为0，表明该信道很糟，不可用于数据通信。

跳频算法的跳数值是在5 ~ 16之间的一个随机数。跳数值被用在自适应重映射之前的跳频算法中。显然，跳数值不能为0，否则频率永远不会改变。也不能为太小的值，因为大多数干扰会持续若干兆带宽，使用非常小的hop不能快速的将传输带离干扰源；同理，太大也不可以，取模后频率差距也会过小。

SCA：休眠时钟精度；

主设备发往从设备的信息中还包括睡眠时间精度值。该值定义了时钟能够保证的精度范围。如果时钟由晶振提供，晶振有一个根据温度变化的精度范围，比如在室温下为 20ppm，在 0℃ 或 85℃ 时为 50ppm 。如此一来，设备可以声明其时钟精度达到 50ppm 。

时钟精度可以帮助从设备消除连接事件的不稳定性。如果从设备与主设备的不同步时间达到 1s，而这两个设备的定时精度均为 500ppm，那么，将合并的不确定性 1000ppm 乘以上述时间，就得到一个 1ms 的不确定窗口。也就是说，从设备必须提早 1ms醒来，并在这额外的 1ms 内保持侦听，以仿主设备和从设备的时钟在不同的方向发生了最大漂移。

更精确的时钟可以降低功耗。还是以使用晶振的设备为例，如果两个设备的时钟精度分别为 50ppm 和 150ppm ，二者合并后的精度只有 200ppm 。一秒钟之后，从设备只需要提前 200μs 醒来并保持侦听。如果设备被唤醒的频率不高，他们的可用工作时间将 5 倍于两个使用 500ppm 晶振的设备。因此，如果对降低功耗有特别的要求，建议在设备中使用高精度的晶振。

### 信道选择（Data Channel Index Selection）

BLE链路层信道选择的算法流程如下：

(1) 首先，Master node在收到Slave node的广播消息之后，会发送CONNECT\_REQPDU，其中，携带channel map用于通知slave node哪些信道可以用于数据连接，哪些信道不可以用于数据连接；

(2) 节点根据如下公式首先计算unmappedChannel：

**unmappedChannel =(lastUnmappedChannel + hopIncrement) mod 37**

其中，如果本次发送的event是数据连接的第一个connection event，上式中的lastUnmappedChannel为0；

hopIncrement的值在CONNECT\_REQ PDU中指定，它是一个跳信道间隔值，影响两次数据传输信道的间隔。

当一个连接event结束时，unmappedChannel值被赋予lastUnmappedChannel。

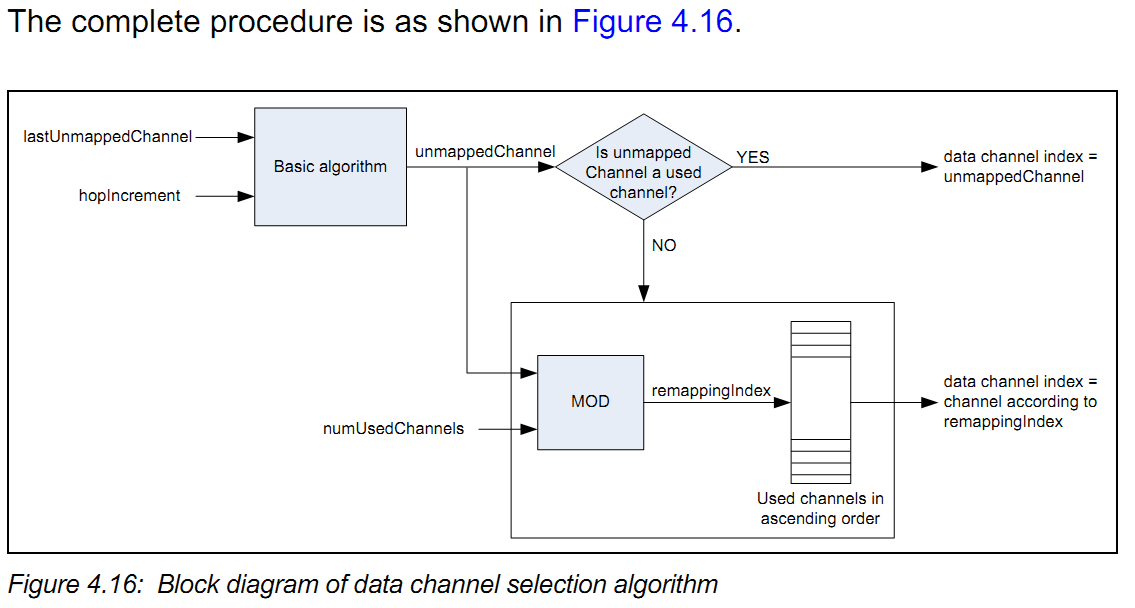
(3) 得到unmappedChannel值之后，将该值与channelmap中可以使用的信道列表进行对比，如果在可以使用的数据信道列表中，找到了unmappedChannel，则使用该unmappedChannel作为本次连接event的信道。

(4) 如果unmappedChannel存在于不可用信道列表中，那么，需要将unmappedChannel重新映射到可用信道列表。使用如下公式：

r**emappingIndex =unmappedChannel mod numUsedChannels**

其中，numUsedChannels表示channel map中可用信道的数量。

算法的流程图如下图所示：



下面以一个例子解释信道选择算法：

(1) 假设hopIncrement = 2，channel map中可用信道列表包含：24,25,26,27;

(2) 某次连接的第一个连接event所选择的信道应该计算为：

unmappedChannel = (0 + 2)mod 37 = 2；

在可用信道列表中(24,25,26,27)没有2，那么需要重映射;

remappingIndex = 2 mod 3 = 2；

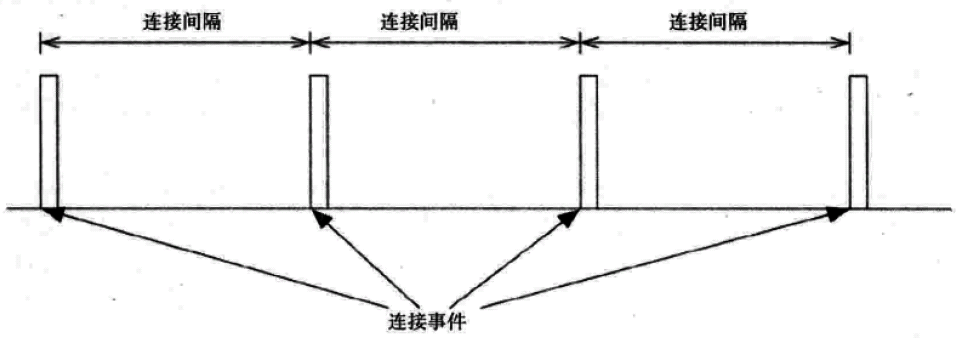
那么，第一次选定的信道应该为25.

### 连接保持

在一个连接当中，主设备会在每个连接事件里向从设备发送数据包。一个连接事件是指主设备和从设备之间相互发送数据包的过程。连接事件的进行始终位于一个频率，每个数据包会在上个数据包发完之后等待 150μs 再发送。

连接间隔决定了主设备与从设备的交互间隔；它是指两个连续的连接事件开始处的时间距离，可以是7.5ms ~ 4s内的任意值，但必须为 1.25ms 的整数倍。要确定从设备与主设备的实际交互间隔，需要用到从设备延迟这一参数，代表从设备在必须侦听之前可以忽略多少个连接事件。

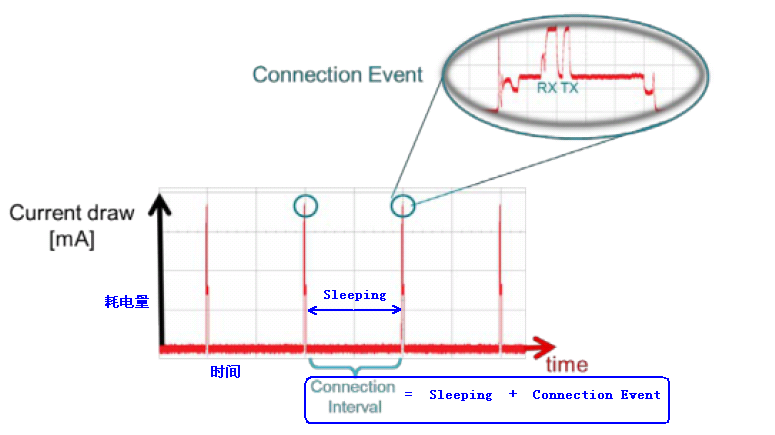
如下图所示，连接事件被一个个的连接间隔分开。从主设备发送数据包开始，每个连接事件可以持续进行，直至主设备或从设备停止响应。在连接事件之外，主从设备之间不发送任何数据包。



举个例子，如果连接间隔为 100ms，从设备延迟是 9，那么从设备可以忽略 9 个链接事件，但不得不侦听第 10 个连接事件。换言之，从设备必须每秒侦听一次，而此时监控超时的最小值应为 1010ms。反过来，另一个极端的例子是，如果监控超时使用了 32s 的最大值，对于间隔为 100ms 的链路，从设备延时必须小于等于 319。

虽然如此，如果将从设备延迟设为可行的最大值，在监控超时发生前从设备只能获得唯一一次侦听主设备的机会，这可不是一个好主意。因此，建议至少给从设备留出 6 次侦听的机会。在前面的例子中，如果连接间隔为 100ms ，从设备延迟为 9，那么监控超时应该至少为 6s，这样一来，链路在最终断开前从设备至少会有 6 次侦听的机会。

主设备和从设备建立连接之后，所有的数据通信都是在连接事件（Connection Events）中进行的。

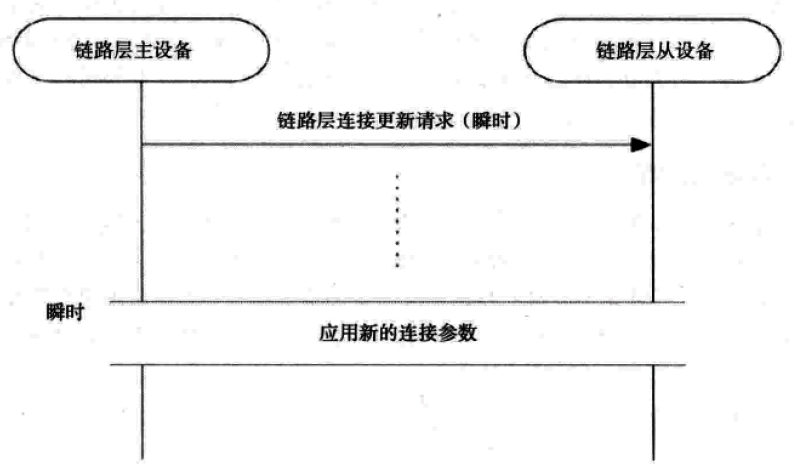


尖刺的波就是连接事件（Connection events），剩下的Sleeping是睡眠时间，设备在建立连接之后的大多数时间都是处于Sleeping。

每个连接事件（Connection events）中，都需要由Master发起包，再由Slave回复。

### 连接参数更新规程

连接建立时，主设备通过链接请求数据包发送连接参数。当连接活跃了一段时间，连接参数也许不再适用于当前使用的服务。出于提高效率的目的，连接参数需要进行更新。较之首先断开连接、接着更换新参数重新连接，还有一种在链路中更新参数更为简单的途径，如下图所示：



为此，主设备向从设备发送连接更新请求，即LL\_CONNECTION\_UPDATE\_REQ，当中携带了新的参数。这些参数不必进行协商，从设备或者接受和使用它们，或者断开链路。连接更新请求中包含了早先创建连接时用过的一部分参数，还有一个称为瞬时（instant）的新参数：

1.传输窗口大小

2.传输窗口偏移量

3.连接间隔

4.从设备延迟

5.监控超时

6.瞬时

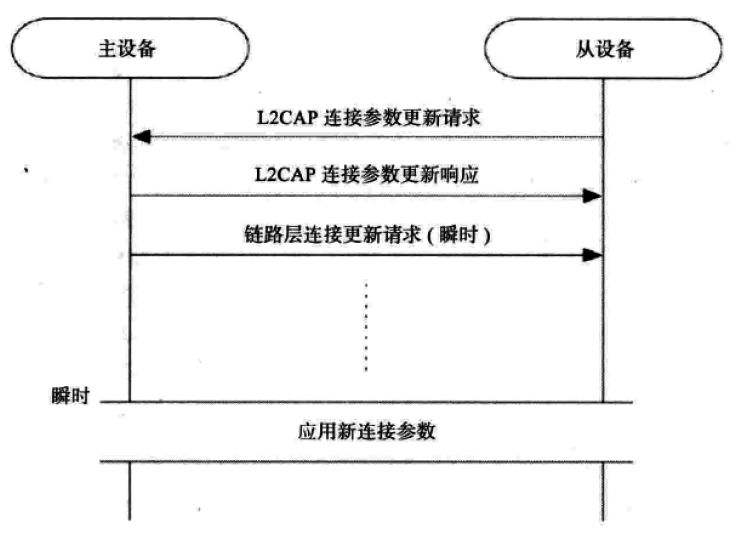
瞬时参数决定了连接更新的开始时刻。发送消息时，主设备为连接更新选定一个未来的时间点，并且放在消息中。接到消息后，从设备会记住这个未来的时刻，届时再切换至新的连接参数。这有助于解决无线系统里的一个最大问题----报文重传。只要数据包的重传次数足够，并最终在瞬时之前传输成功，上述过程执行起来就不会有问题。但是，如果该数据包届时没能完成传输，链路就有可能丢失。

由于低功耗蓝牙没有时钟，要决定瞬时时刻只有依靠计算连接事件的个数。因此，每一个连接事件都会被计数，链路上的第一个连接事件，也就是在连接请求之后的位于首个传输窗口里的连接事件记为 0。因此，瞬时实际上是一个连接事件的计数器，相应的连接事件到来时就使用新的参数。为了让从设备收到数据包，主设备必须为其提供足够的机会。不过从设备延迟是多少，都应该至少保证 6 次数据发送机会。也就是说，如果从设备延迟为 500ms，那么瞬时通常被设定在 3s 之后的某个未来时刻。

瞬时到来时，从设备开始侦听发送窗口，就好像连接建立的过程那样。主设备能够调整从设备的计时，总体而言不超过 1.25ms。不过，由于主设备可能还是一个经典蓝牙设备，上述调整使其得以协调低功耗蓝牙从设备，从而更好地完成调度。一旦该过程结束，新的连接间隔、监控超时、从设备延迟值将投入使用。

### 连接参数的修改

“连接参数更新请求”命令可以让从设备更新链路层的连接参数，如下图所示。这些参数包括连接间隔（从设备希望主设备允许从设备发送数据包的频率）、从设备延迟（从设备能够忽略主设备的连接事件的最大值）以及监控超时。



在连接中，如果从设备希望修改当前的连接参数则可以使用该命令。比方说，如果连接事件的间隔有可能太快了，导致过多的电量浪费。这在从设备时延很大时没有问题，但如果不是这样，从设备将会频繁的侦听链路。这在一些情况下是必要的，例如设备间首次绑定、互发多个数据包、探索服务和设备特性等。但在很多其他情况下，尽可能地减少从设备必须侦听连接事件的数量对提高电池寿命至关重要。

连接参数更新请求命令仅用于从设备向主设备发送，这是由于主设备随时都能启动链路层连接参数更新控制（Connection Parameter Update Control）规程。如果该命令由主设备发送，从设备会将其视为一个错误，并返回带有“命令不理解”原因代码的“命令拒绝”命令。

从设备可以在任何时候发送该命令；收到该信息的主设备如果可以修改连接参数，则将返回“连接参数更新响应”（Connection Parameter Update Response），其中的结果代码设为“接受（accepted）”。随后，主设备将会启动链路层连接参数更新控制规程。

当然，如果主设备不同意从设备的请求参数，它可以发送结果代码为“拒绝（rejected）”的连接参数更新响应命令以拒绝请求。此时从设备有两个选择：要么接受主设备希望的正在使用的连接参数，要么终止连接。终止连接的做法咋看起来可能让人觉得很激进，但是，假如使用当前的参数从设备将会在一周内耗尽电量，而使用请求的参数则可以持续数年，很明显，合理的选择只有一个。

修改连接参数时，如果要减少主设备拒绝从设备请求的可能性，可以在请求里设置一个可接受的参数范围。经过精心设计的从设备会乐意接受很宽的参数范围。由于主设备可能正忙于实时会话音频连接或者高质量语音连接等任务，它可以接受一定范围内的连接间隔参数。设备可接受的间隔参数会根据当前任务的不同而不同，可能有别于上一次设备连接时的参数。

要提高主设备接受连接参数的机率，还有个方法是从设备提供一个合理的从设备延迟。主设备可以选择最合适的连接事件间隔，从设备则使用最佳功耗的从设备延迟参数。

# 设备过滤机制

从前面的描述可知，BLE的广播功能，除了速率上面不给力之外，还是比较爽的。但有一个问题，需要引起我们的重视：

如果周围有很多的BLE设备在广播，对Scanner来说，它的Controller会扫描到很多广播数据，如果这些数据都上报给Host（甚至用户）的话，估计Host（或者用户）会疯掉。于是，基于有了白名单（White List）机制的设备过滤机制。

### 白名单（White List）

每一个BLE的Controller，可以保存一个设备列表，通过该列表，可以实现设备过滤的功能。这个列表就称作白名单（White List），保存了一些BLE设备地址。

白名单的大小由Controller自行觉得，并在reset的时候为空，后续可以由Host通过HCI接口配置。基于白名单，Link Layer可实现多种设备过滤的策略，包括：

Advertising Filter Policy；

Scanner Filter Policy；

Initiator Filter Policy。

具体可参考下面的描述。

#### Advertising Filter Policy

Advertising Filter Policy定义了Advertiser（处于Advertising状态）的Link Layer怎么处理SCAN\_REQ（扫描请求）和CONNECT\_REQ（连接请求），包括如下策略（Host可以根据实际情况配置，同一时刻只能配置一种）：

Link Layer只接受位于白名单中的设备的扫描和连接请求（最严格）；

Link Layer可以接受任何设备的扫描和连接请求（最不严格，Controller reset后的默认状态）；

Link Layer可以接受任何设备的扫描请求，但只接受位于白名单中的设备的连接请求；

Link Layer可以接受任何设备的连接请求，但只接受位于白名单中的设备的扫描请求。

#### Scanner Filter Policy

Scanner Filter Policy定一个Scanner（处于Scanning状态）的Link Layer怎么处理广播数据，包括如下策略：

Link Layer只处理位于白名单中的设备的广播数据，并且忽略没有包括自身地址的connectable Directed advertising packet；

Link Layer处理所有设备的广播数据，并且忽略没有包括自身地址的connectable Directed advertising packet（Controller reset后的默认状态）。

另外，如果设备支持“Extended Scanner Filter”策略，则可以同时支持如下的策略：

Link Layer只处理位于白名单中的设备的广播数据，并且不能忽略InitA地址为“resolvable private address”的connectable Directed advertising packet；

Link Layer处理所有设备的广播数据，并且不能忽略InitA地址为“resolvable private address”的connectable Directed advertising packet。

注4：有关resolvable private address，我们会在其它文章中介绍。

#### Initiator Filter Policy

Initiator Filter Policy定一个Initiator （处于Initiating状态）的Link Layer怎么处理可连接的广播数据，包括如下策略：

Link Layer只处理位于白名单中的设备发送的可连接的广播包，并在收到的时候发起连接请求；

忽略白名单，Link  Layer处理由Host指定的设备所发送的可连接的广播包，并在收到的时候发起连接请求。