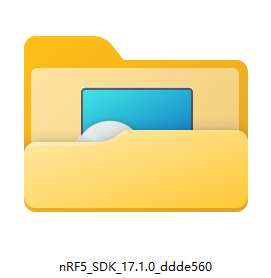
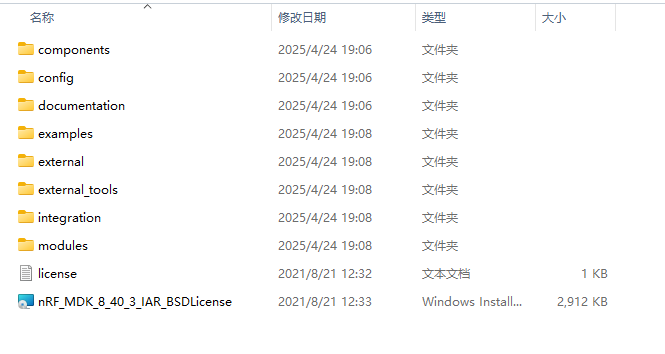
# SDK目录结构

下面是从官网下载的SDK解压后的目录



进入此目录之后是下面的目录

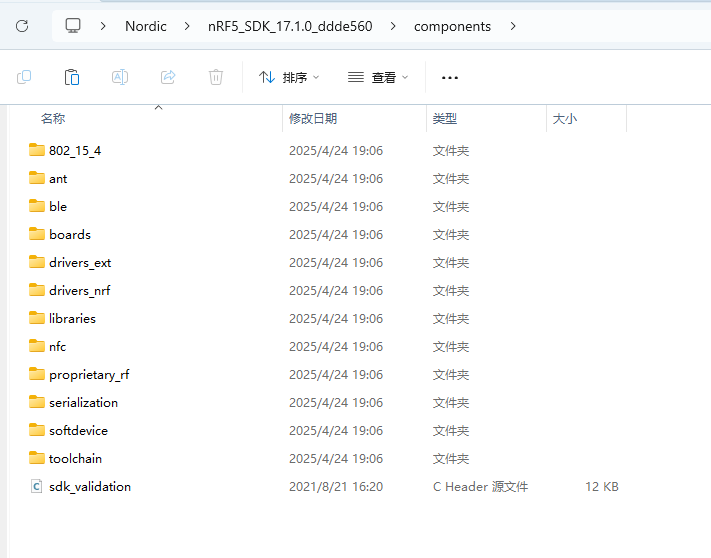


## SDK目录结构

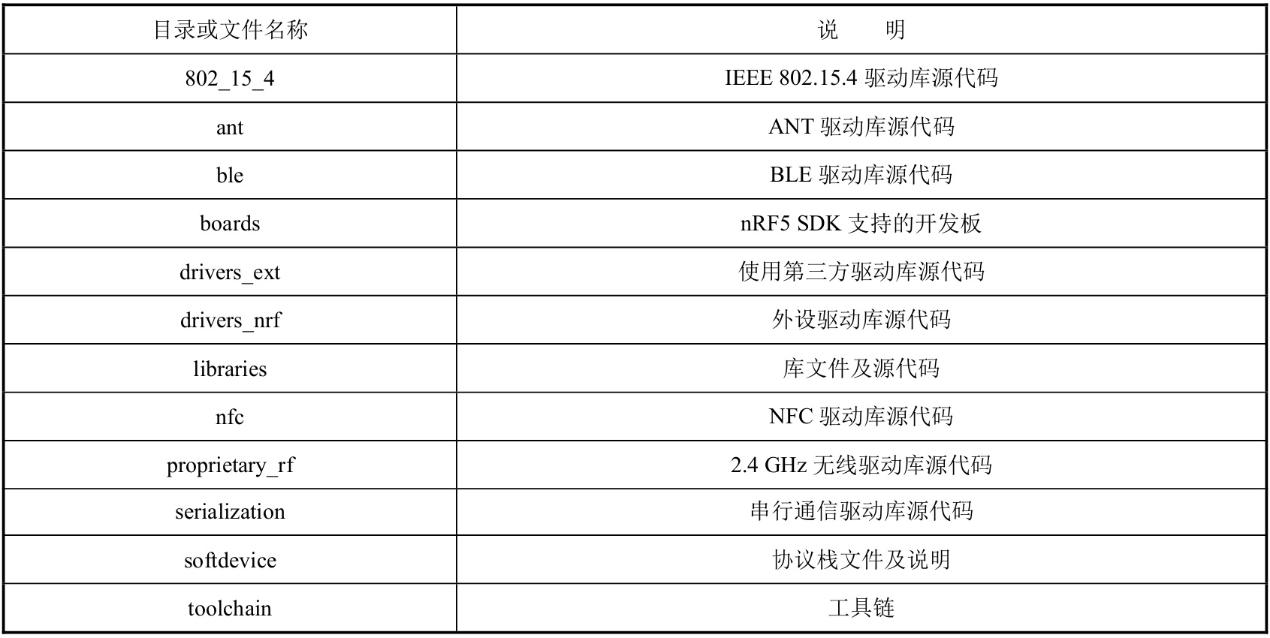
下表是对上面目录的说明



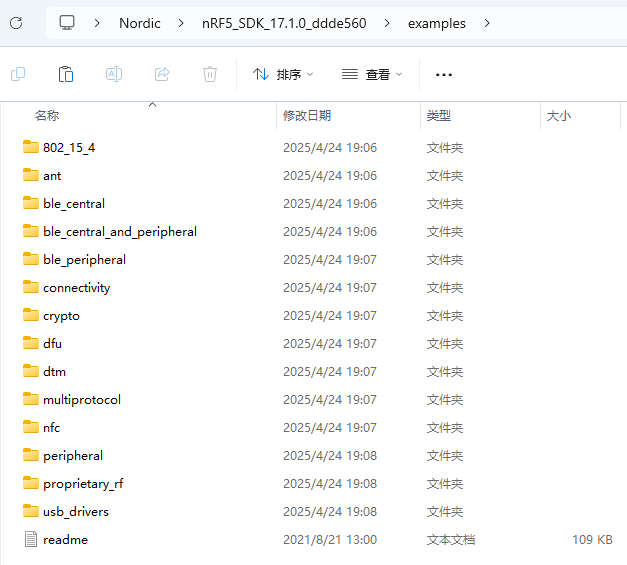
## components目录结构

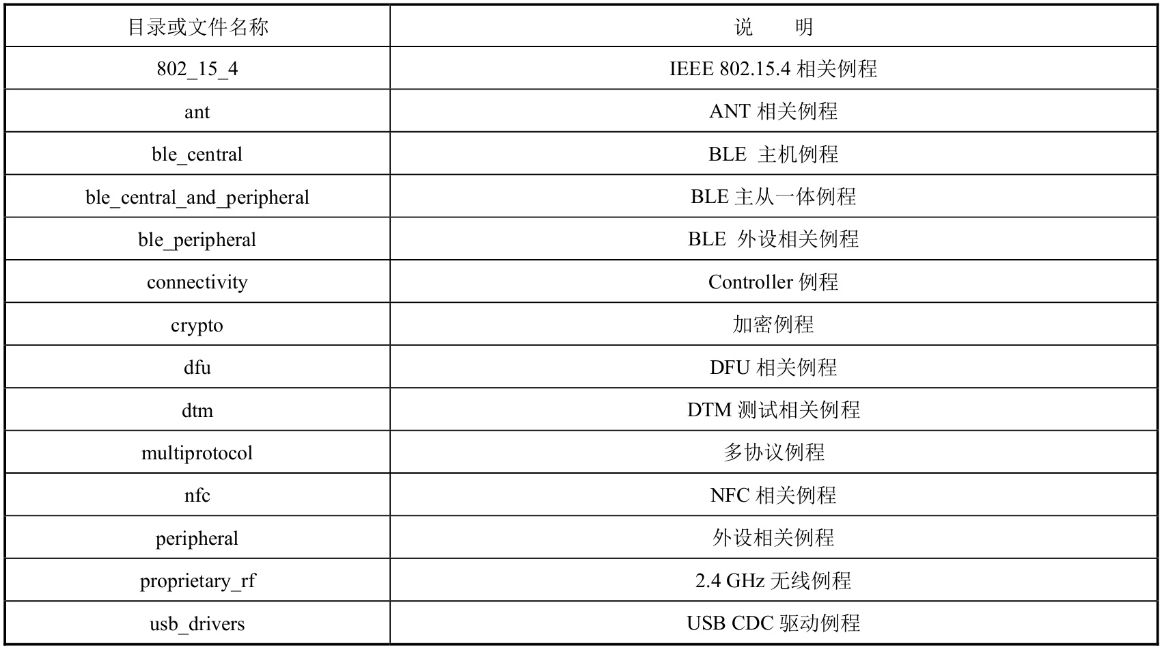


对components目录的说明

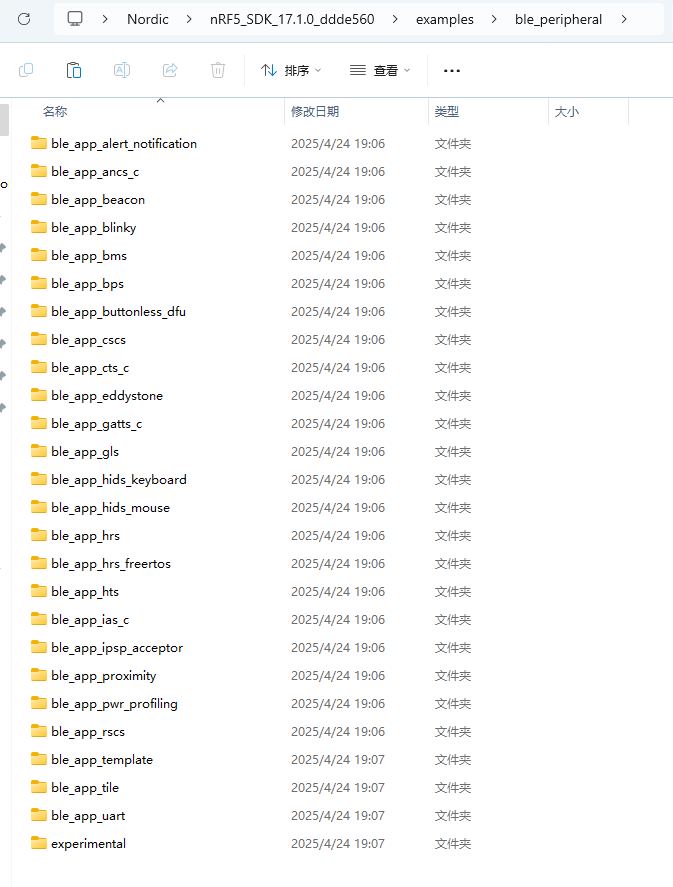


## examples目录结构





ble\_peripheral目录结构



目录说明



# 广播初始化

## 基础知识

广播数据包。

扫描回应数据包。

广播间隔。

### 广播数据包的类型

传统广播类型（Legacy Advertising PDU）

ADV\_IND：可连接、可扫描的非定向广播

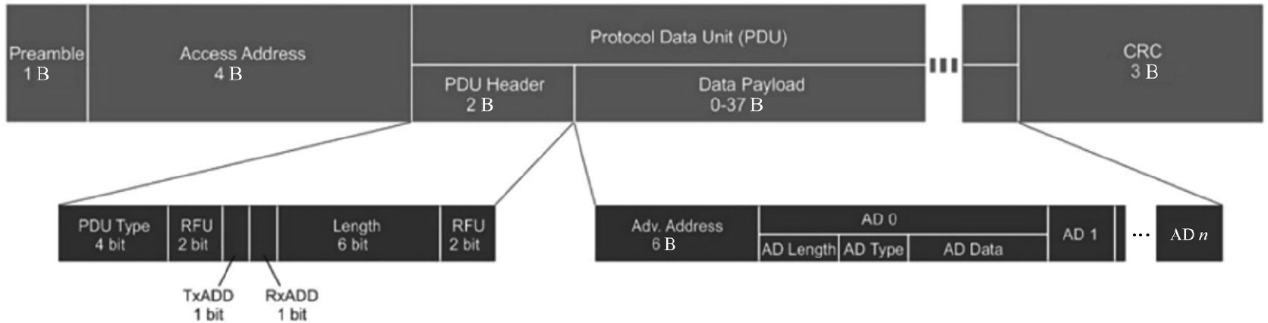
ADV\_DIRECT\_IND：可连接的定向广播

ADV\_NONCONN\_IND：不可连接、不可扫描的非定向广播

ADV\_SCAN\_IND：可扫描的非定向广播

扩展广播类型（Extended Advertising PDU）

### 广播数据包的格式



Preamble：1B先导码

PDU段内容：

PDU Type：PDU Type占4bit，表示广播数据包的类型

RFU：RFU占2bit，保留。

TxADD：TxADD占1bit，当TxADD为0时表示广播的地址是public类型的地址；当TxADD为1时表示广播的地址是random类型的地址。

RxADD：RxADD占1bit，当RxADD为0时表示对端地址的类型是public类型；当RxADD为1时表示对端地址的类型是random类型。RxADD在定向广播中使用，因为定向广播携带了对端地址，其他类型的广播不使用RxADD。

Length：Length占6bit，表示Data Payload数据载荷的长度，以字节为单位。

### Data Payload的广播内容

（1）Flags。

（2）Service UUID。

（3）Local Name。

广播名称，Data是表示广播名称的字符串，Local Name既可以是广播名称，也可以是广播名称的缩写。需要注意的是，缩写必须是全名的前若干个字符。

• Type=0x08：表示广播名称。

• Type=0x09：表示广播名称的缩写。

（4）TX Power Level。

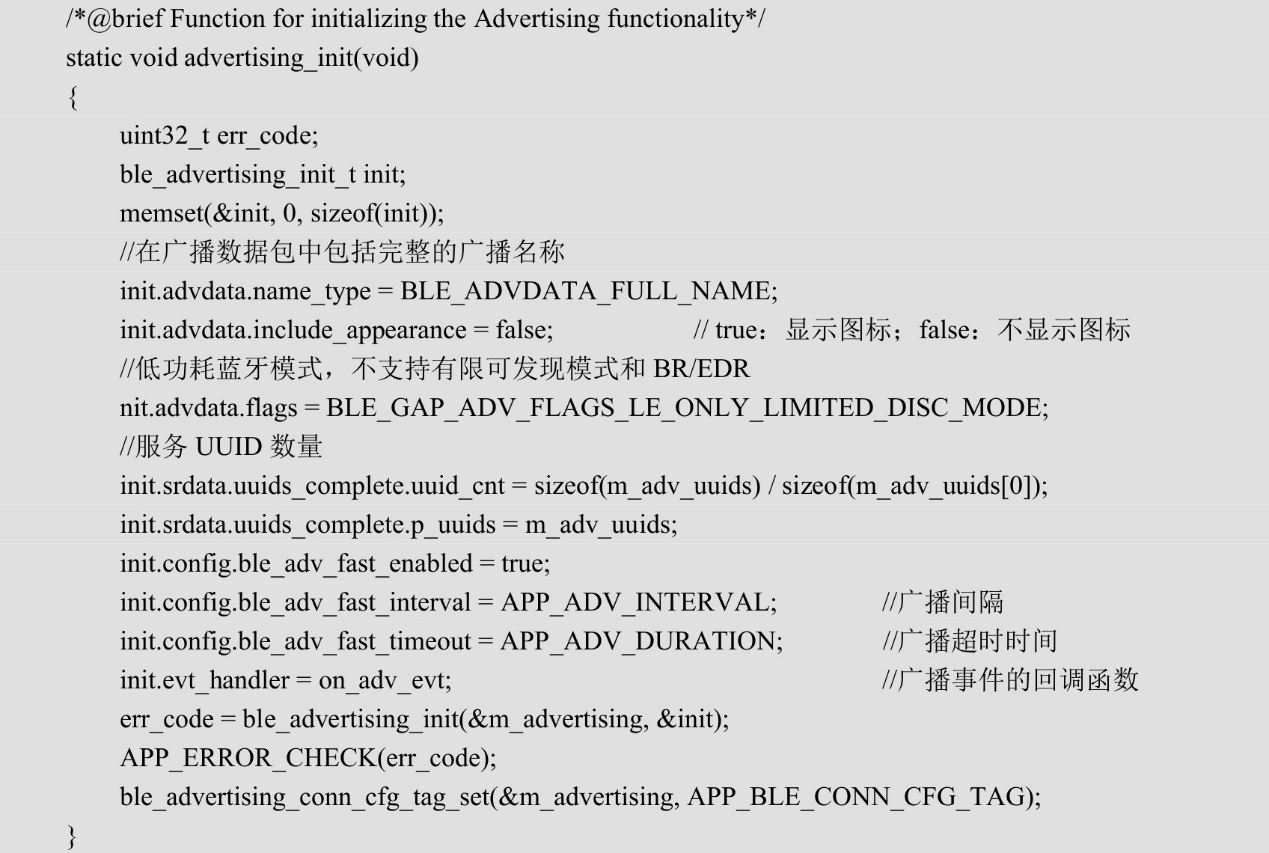
TX Power Level表示设备发送广播数据包的信号强度，当Type=0x0A时，Data是1B，表示发射功率为-127～+127dBm。根据发射功率和RSSI可以计算主机到从机的距离，这也是iBeacon室内定位的原理。

（5）Appearance。当Type=0x19时，Data表示设备的外观。

（6）厂商自定义数据。当Type=0xFF时，厂商自定义数据的前2B表示厂商ID，剩下的是厂商自己按照需求添加的，数据内容由厂商定义。

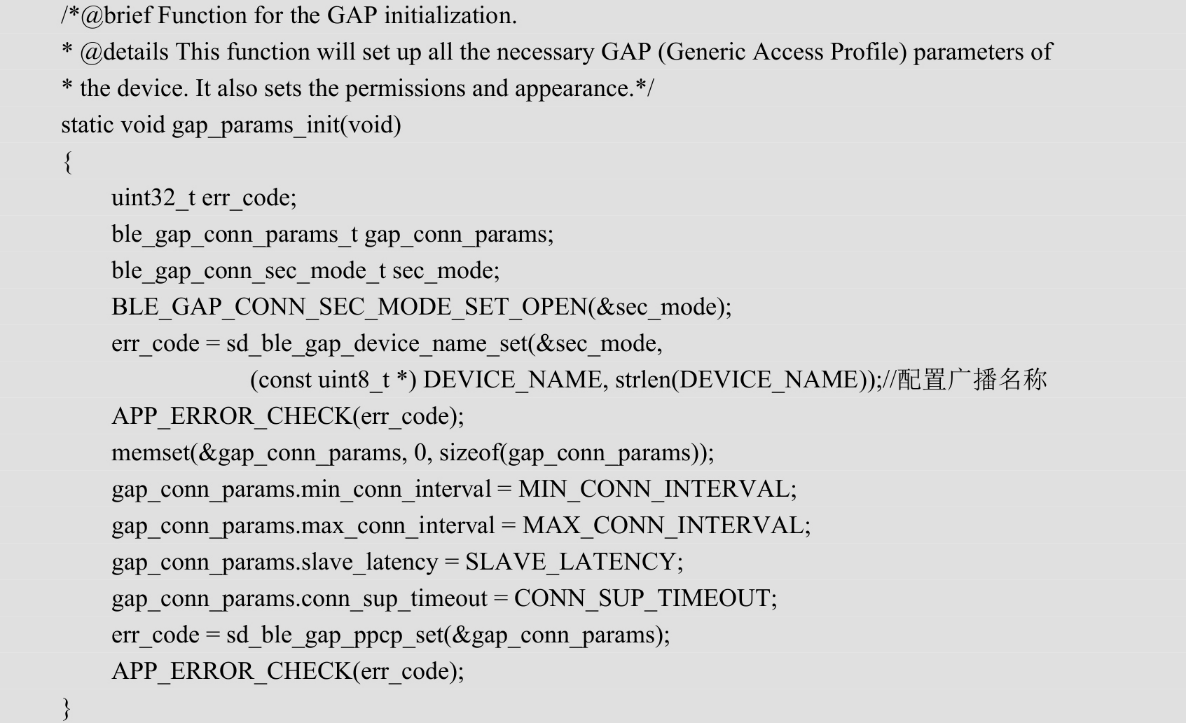
## 代码

### 广播初始化



### 修改蓝牙广播名称

开发者可以在SDK中通过函数gap\_params\_init(void)中的宏定义DEVICE\_NAME来修改广播名称，代码如下：

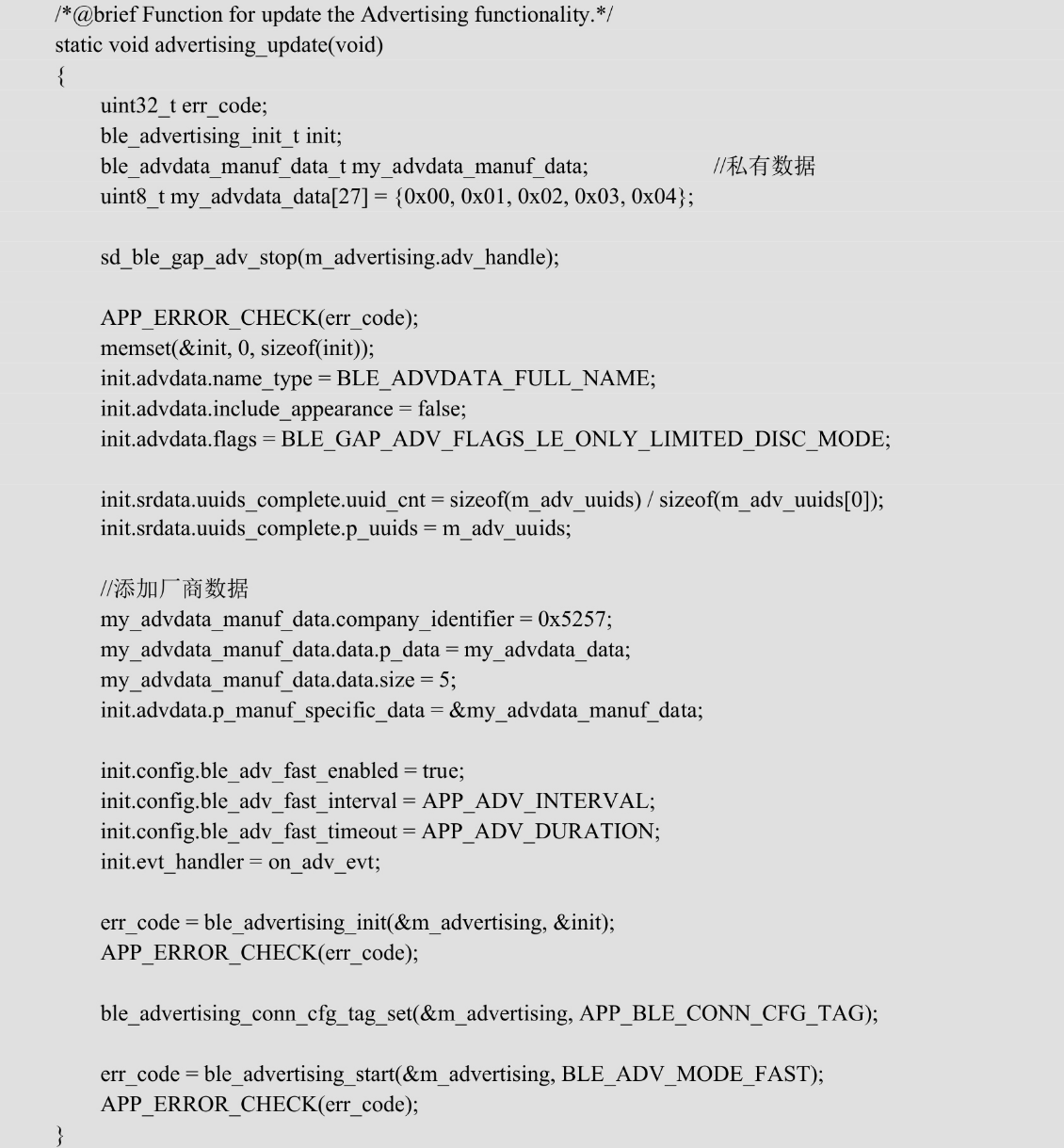




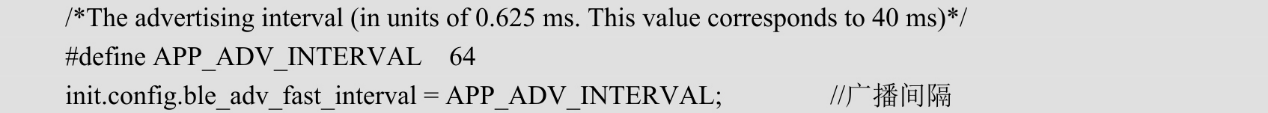
### 修改广播内容

添加自定义厂商数据

修改的函数是static void advertising\_init(void)

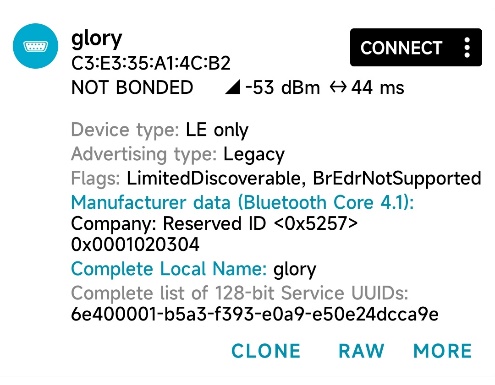


### 修改广播间隔



广播间隔的最小值为20ms，最大值为10.24s。在修改广播间隔时，能进行的最小修改幅度是0.625ms，如上述代码中的“64”，对应的广播间隔是64×0.625ms=40ms。

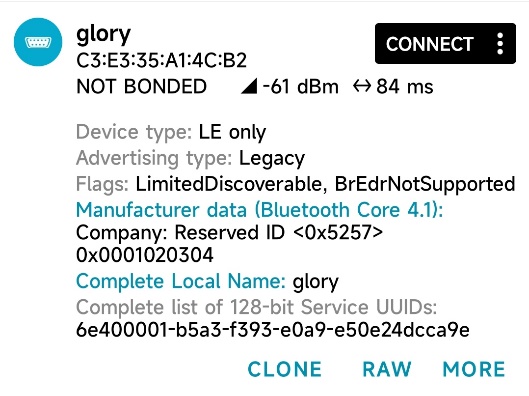
在调试中可以看到蓝牙的广播时间是44ms



修改为128



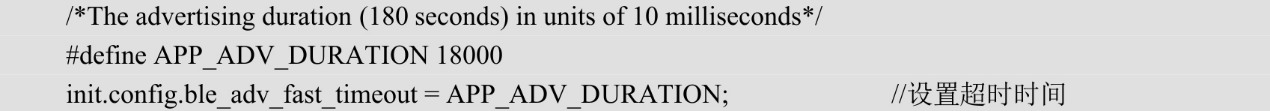
可以看到蓝牙广播时间为84ms



### 修改广播超时时间

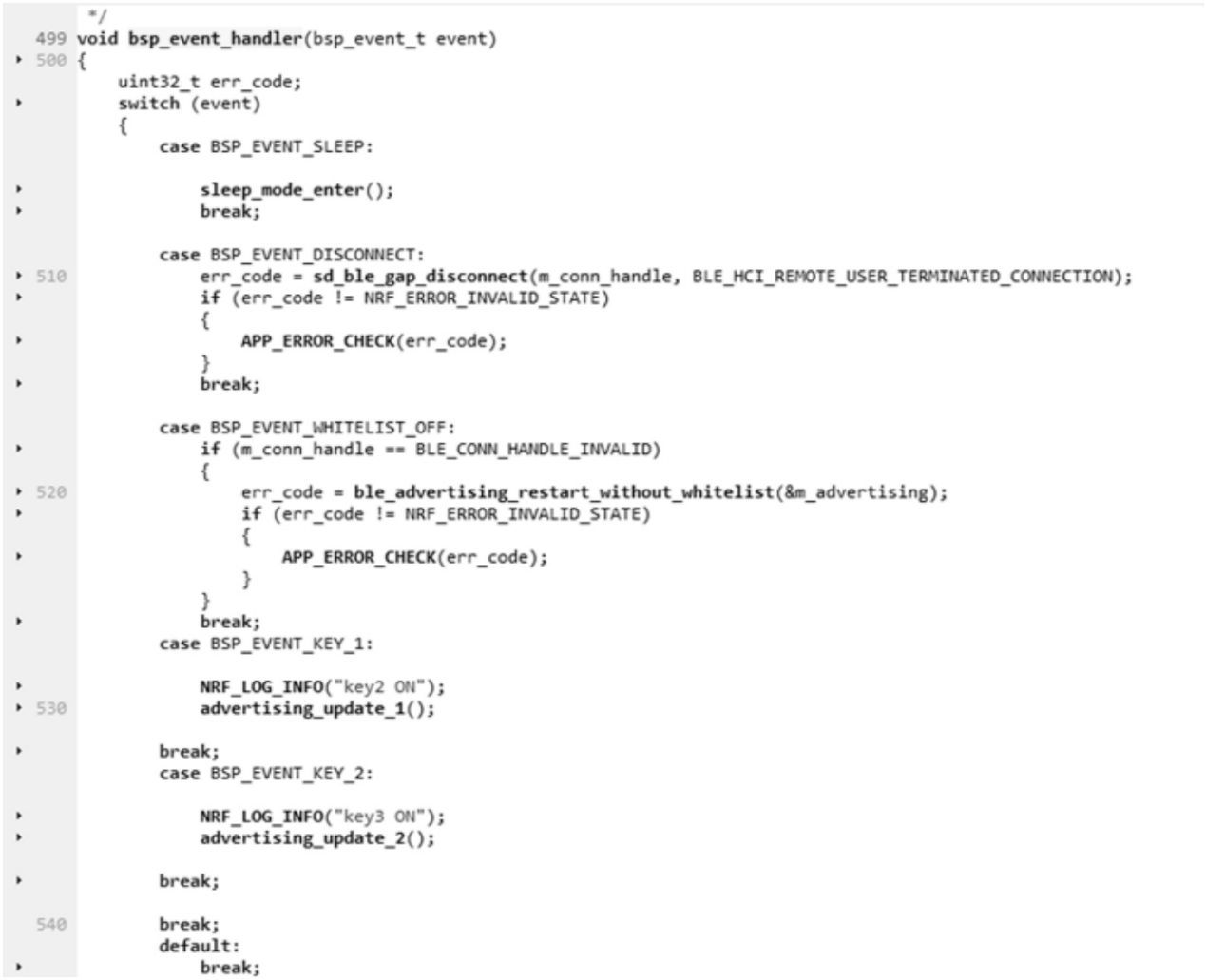
在SDK中，通过宏定义APP\_ADV\_DURATION可以修改广播超时时间。例如，将广播超时时间设置为180s，在180s后，就会产生广播超时事件，nRF52840 DK开发板可以根据需要选择重新广播或者进入休眠状态。代码如下：

单位是10ms



### 广播内容





### 持续性广播

通过修改广播的发现模式可以将nRF52840 DK开发板设置为一直处于持续性广播的状态，广播的发现模式有以下两种：

• BLE\_GAP\_ADV\_FLAGS\_LE\_ONLY\_LIMITED\_DISC\_MODE：有限可发现模式，不支持BR/EDR。

• BLE\_GAP\_ADV\_FLAGS\_LE\_ONLY\_GENERAL\_DISC\_MODE：一般可发现模式，不支持BR/EDR。

有限可发现模式和一般可发现模式的主要区别是：

① 有限可发现模式有时间的限制，一般维持的时间比较短；一般可发现模式没有时间的限制。

② 有限可发现模式的广播间隔比一般可发现模式的广播间隔小。

③ 从时间的限制来看，有限可发现模式对连接的迫切性和目的性比一般可发现模式高。

在advertising\_init()函数中将APP\_ADV\_DURATION广播超时时间设置为0，表示不会超时。实现持续性广播的方法如下：首先将



修改为：

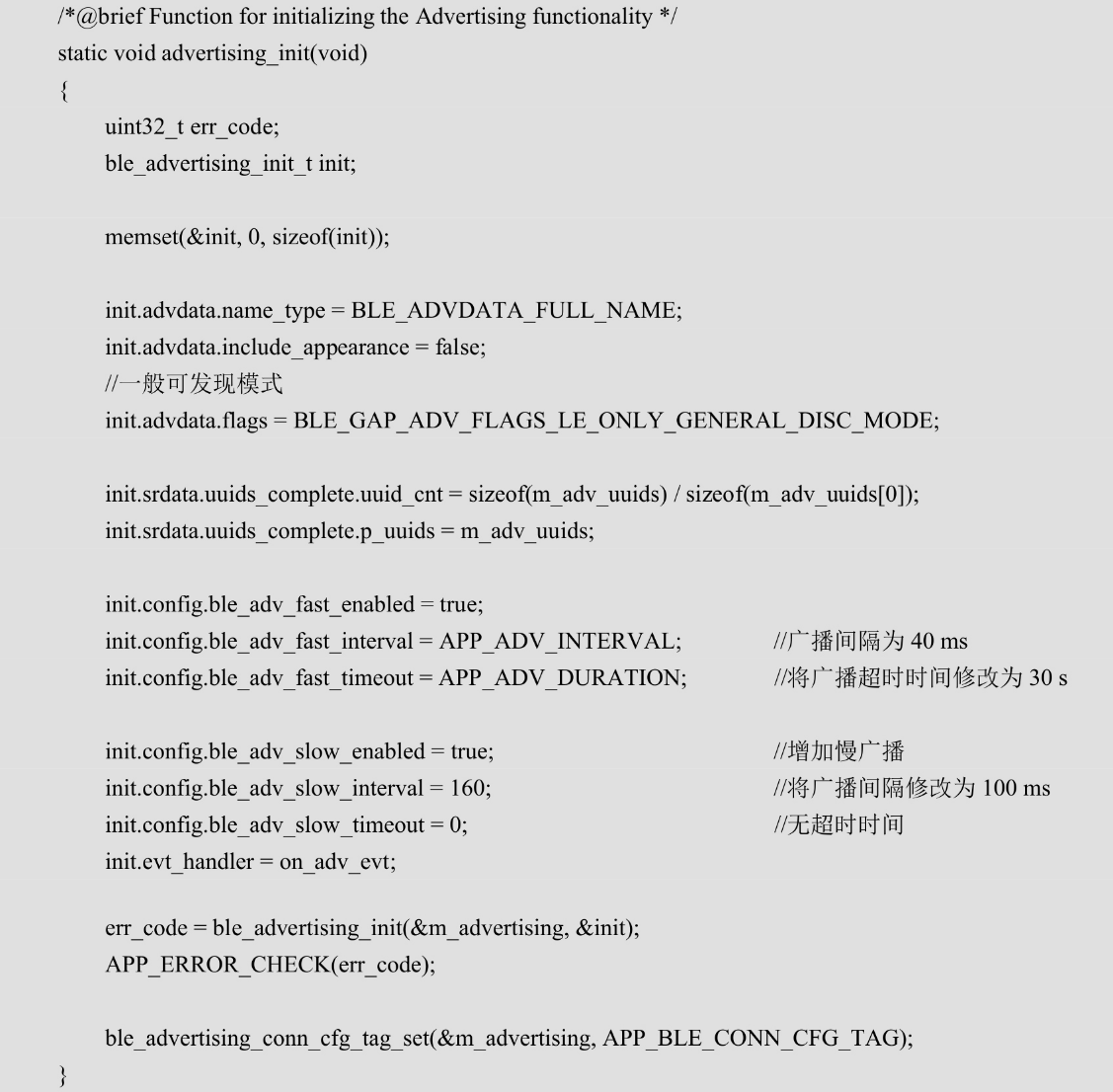


然后将广播超时时间修改为0，代码如下：

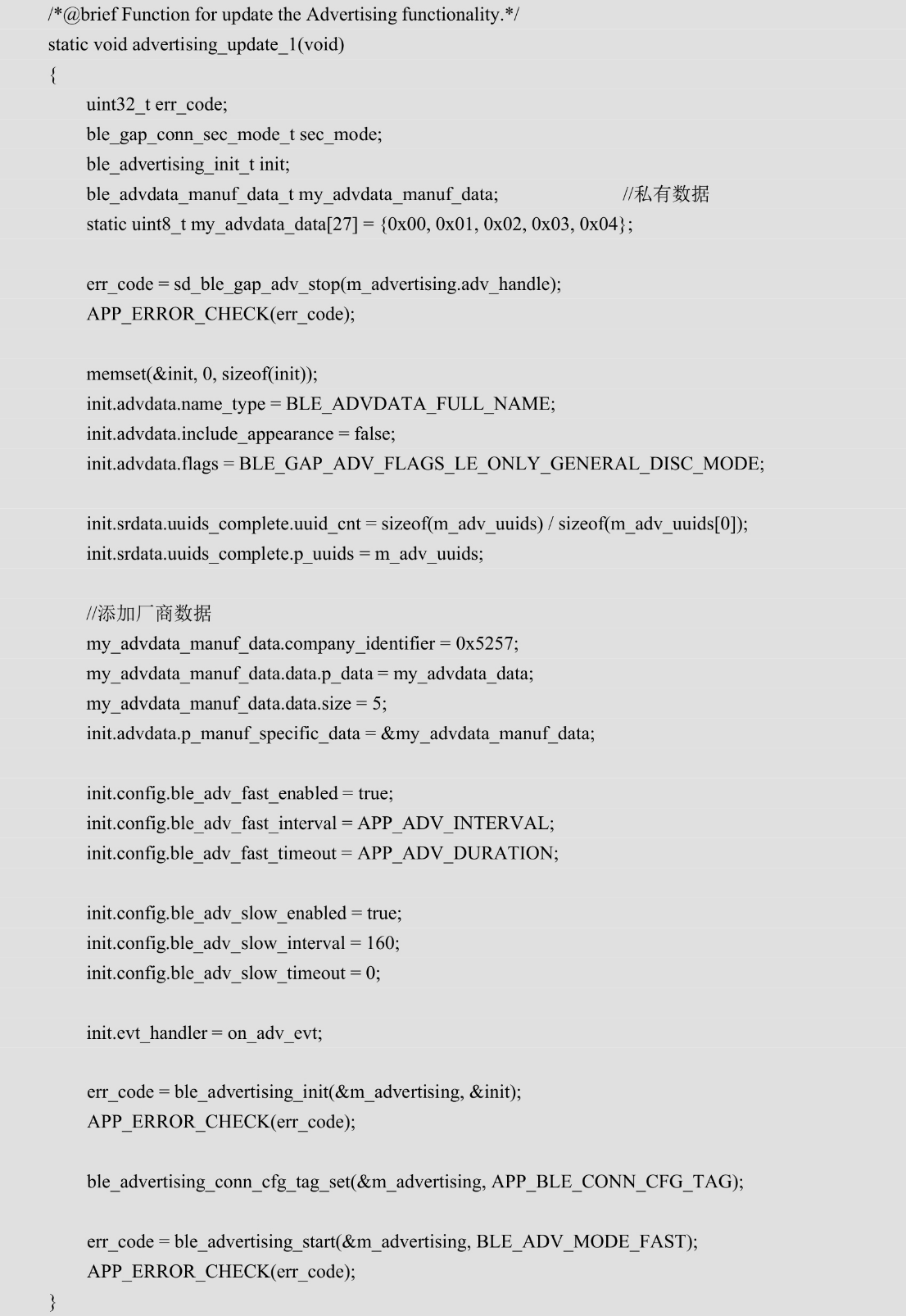


## 快广播转慢广播代码

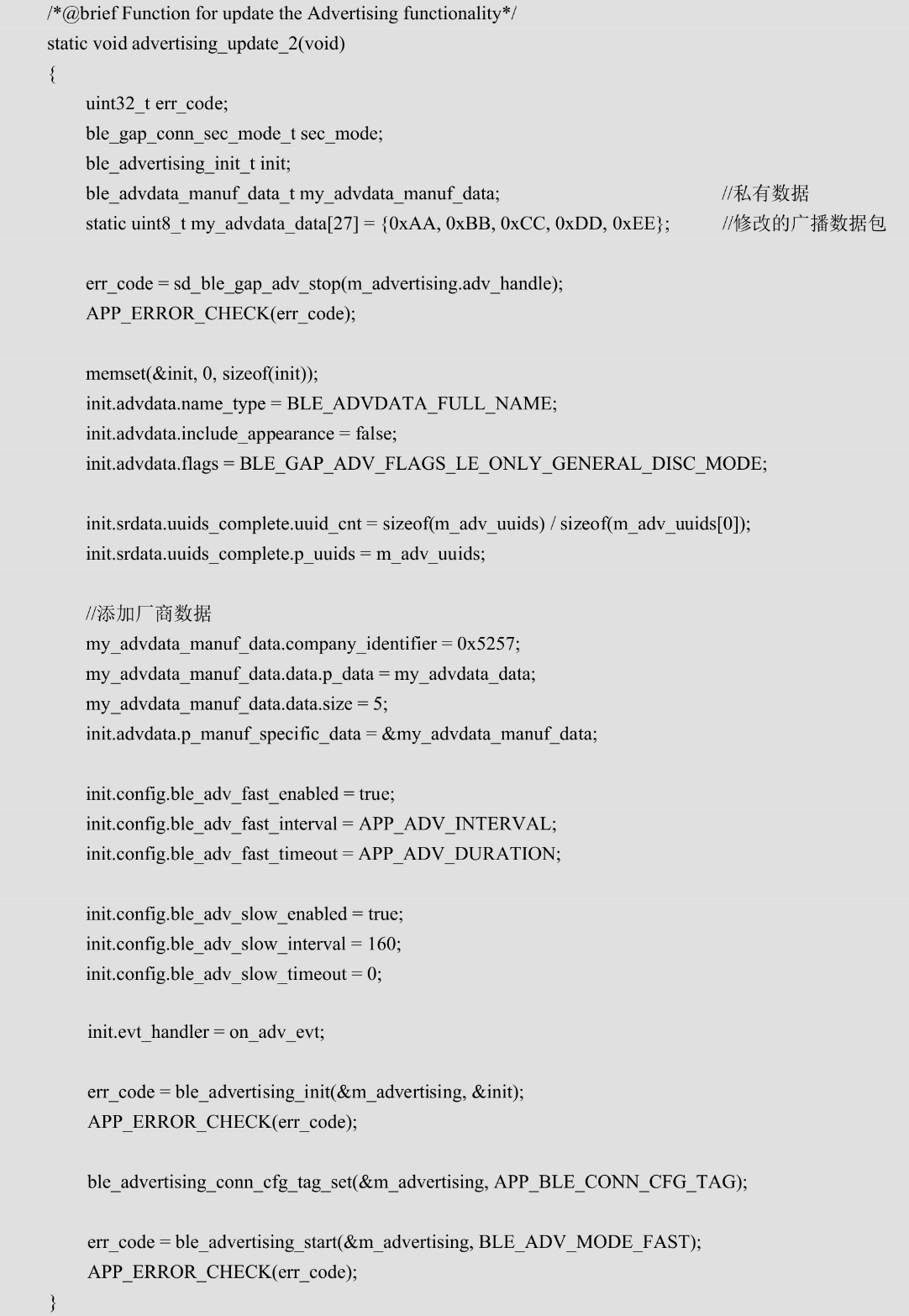
### 初始化



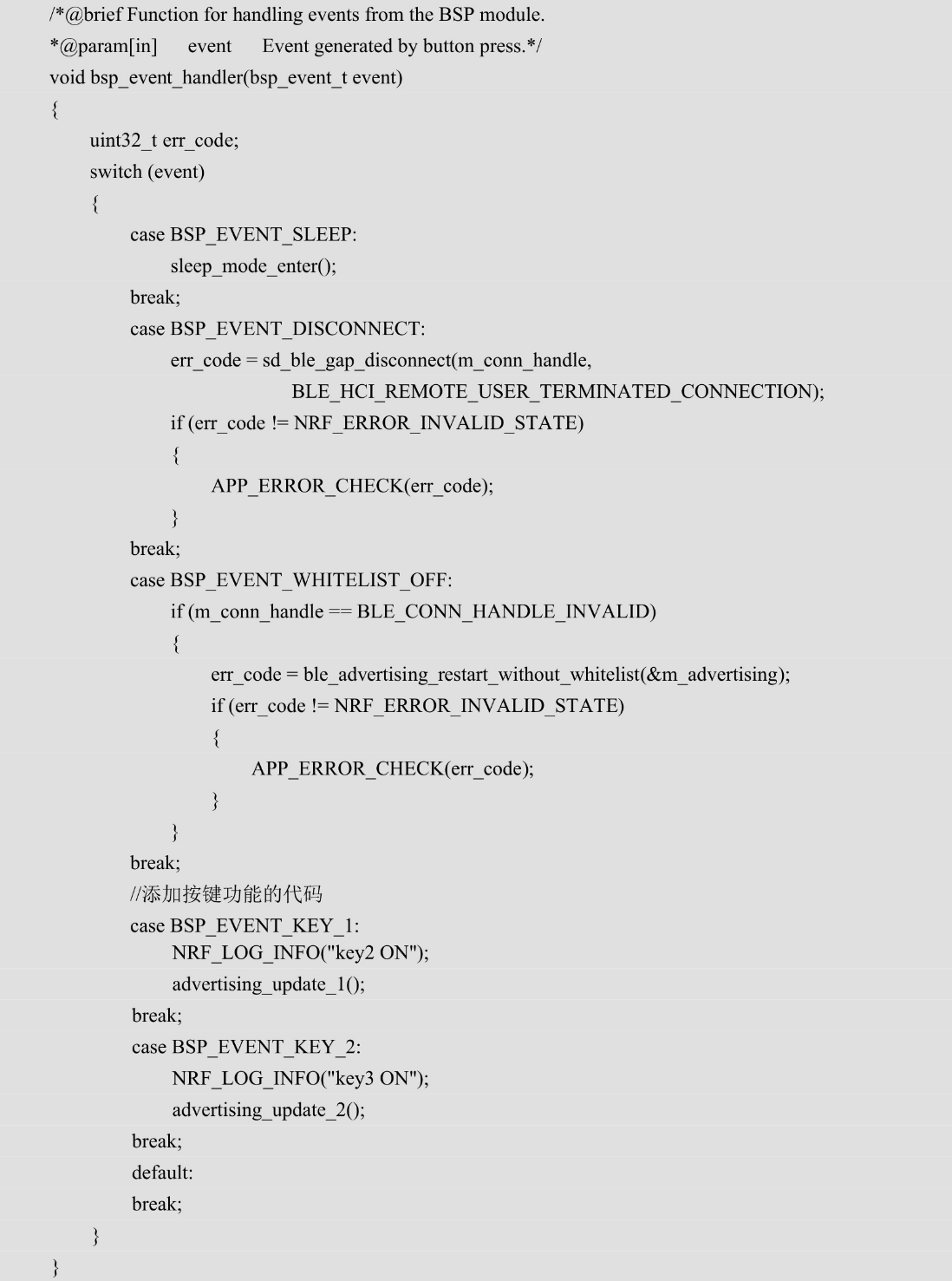
### 按键修改函数1



### 添加按键修改函数2

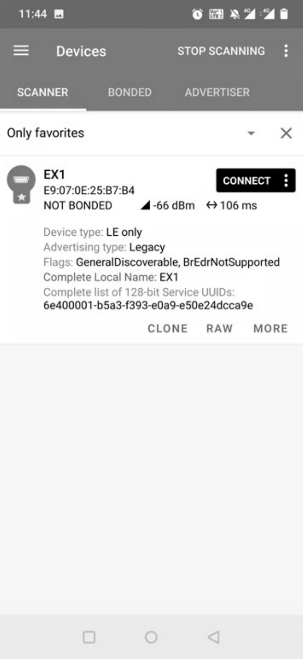
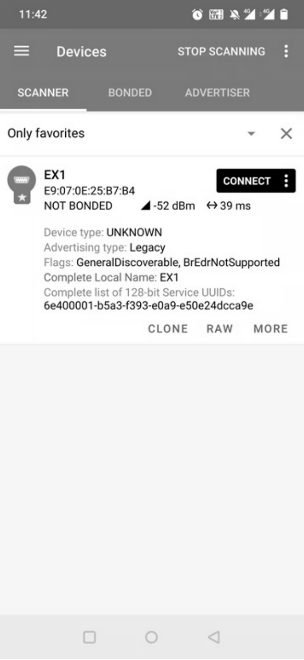


### 按键处理函数

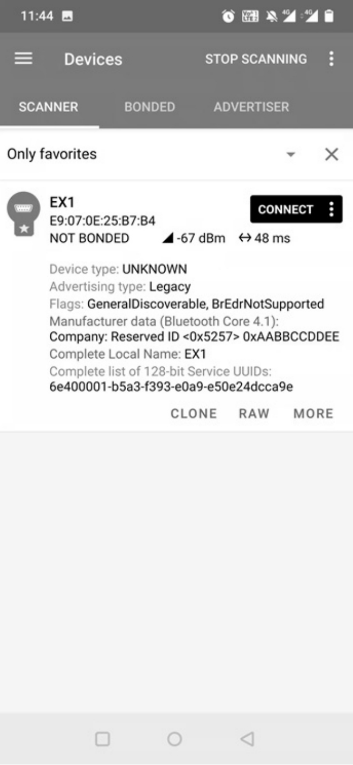
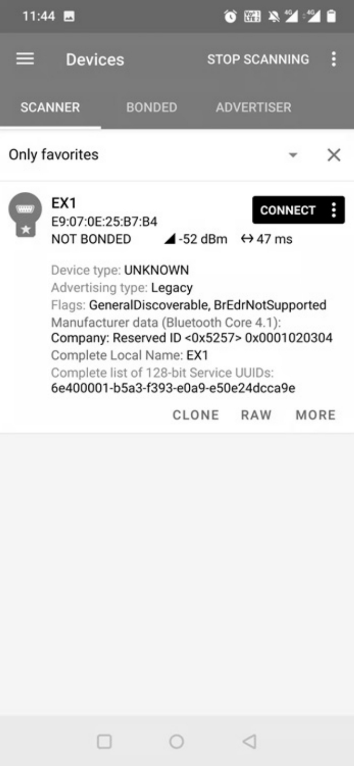


### 现象

在启动广播后进入快广播，一段时间后进入慢广播



按下按键后，可以修改广播的内容



# 双向通信

## 属性协议（Attribute Protocol）

属性协议（ATT协议）定义了两种角色：服务器（Server）和客户端（Client）。

服务器（Server）：提供数据的蓝牙设备。

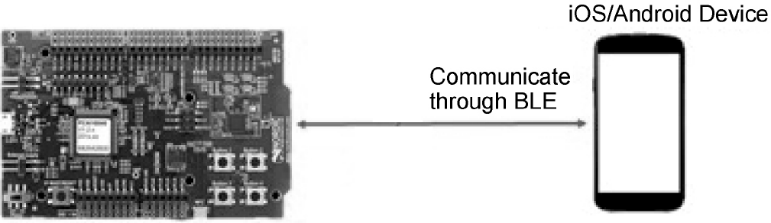
客户端（Client）：需要数据的蓝牙设备。

属性协议就是用于在服务器和客户端之间进行通信的协议。服务器保存了一个类似“属性数据库”的东西，其中包含了一系列的属性及其特性。

客户端可以通过属性协议从服务器获取这些属性。

## 例程实现过程

通过手机实现与单片机之间的通信

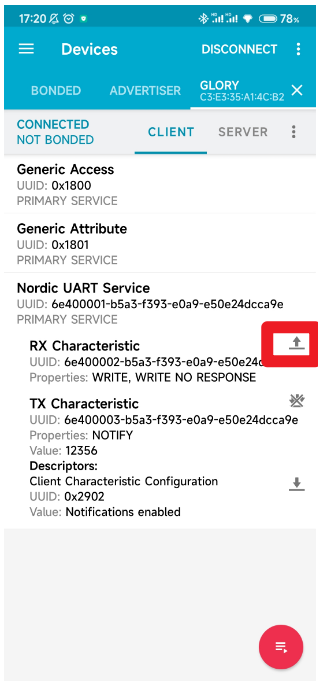
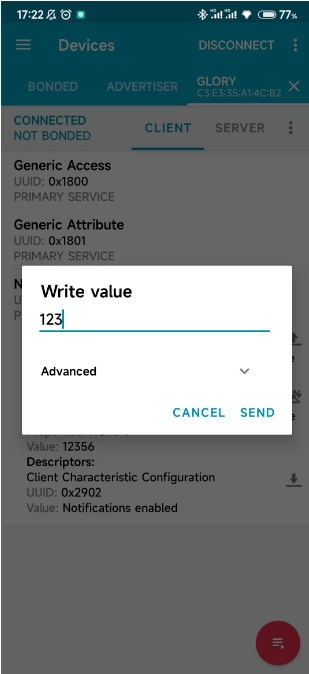
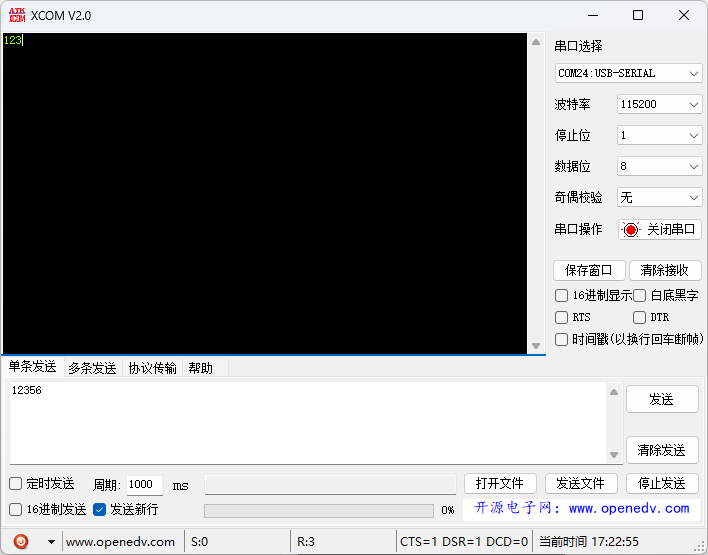


使用例程ble\_app\_uart

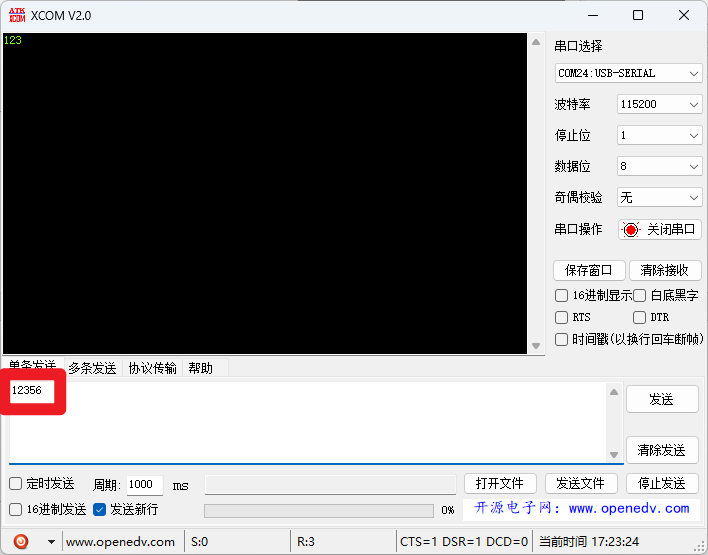
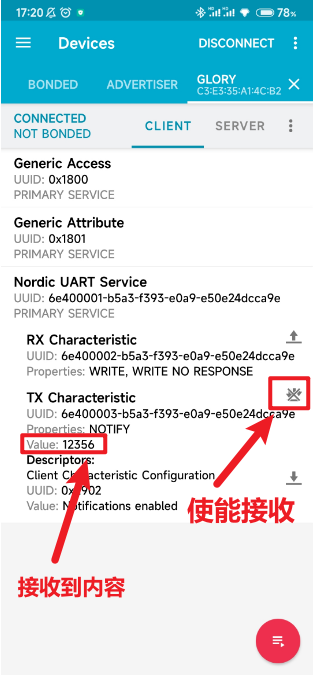
使用串口调试助手连接对应的串口，调整串口的参数，波特率为115200



手机端发送数据，串口通信软件会接收到对应的数据

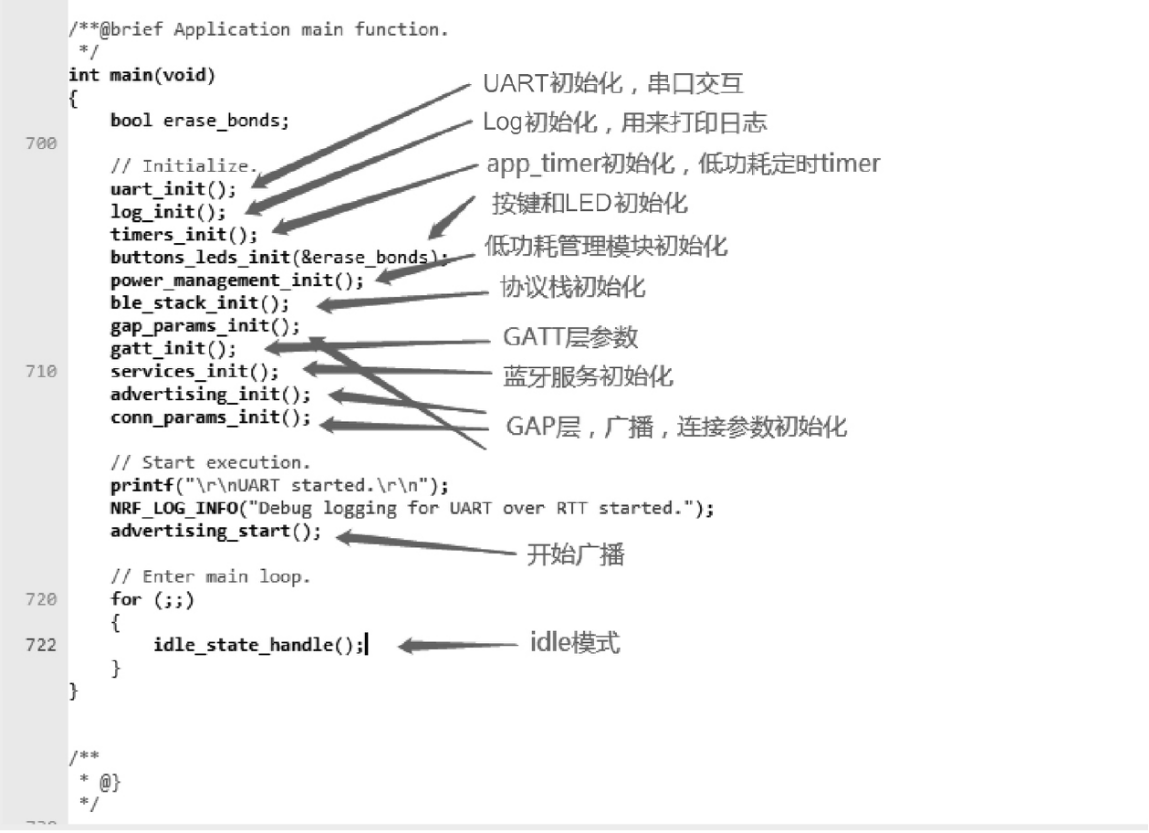
  

串口通信软件发送数据

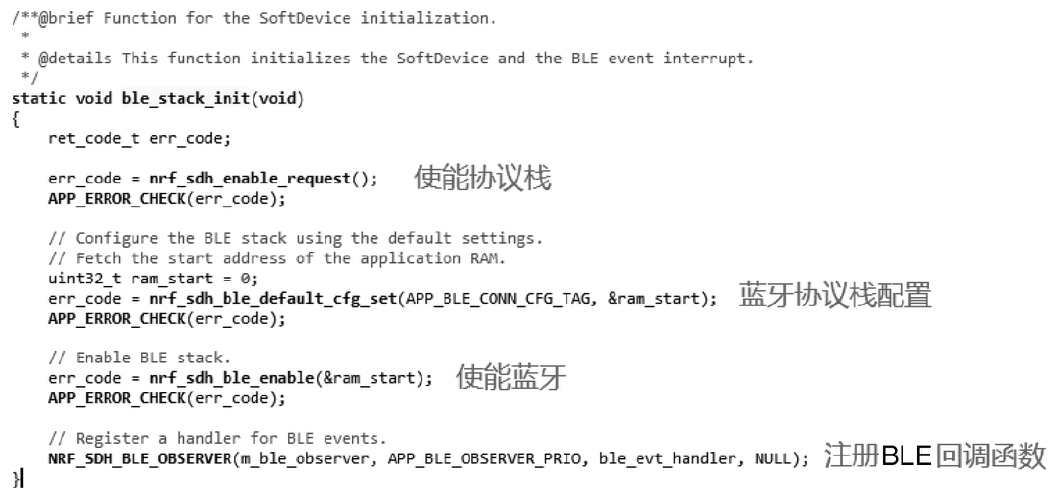
 

## 代码解析

### main函数

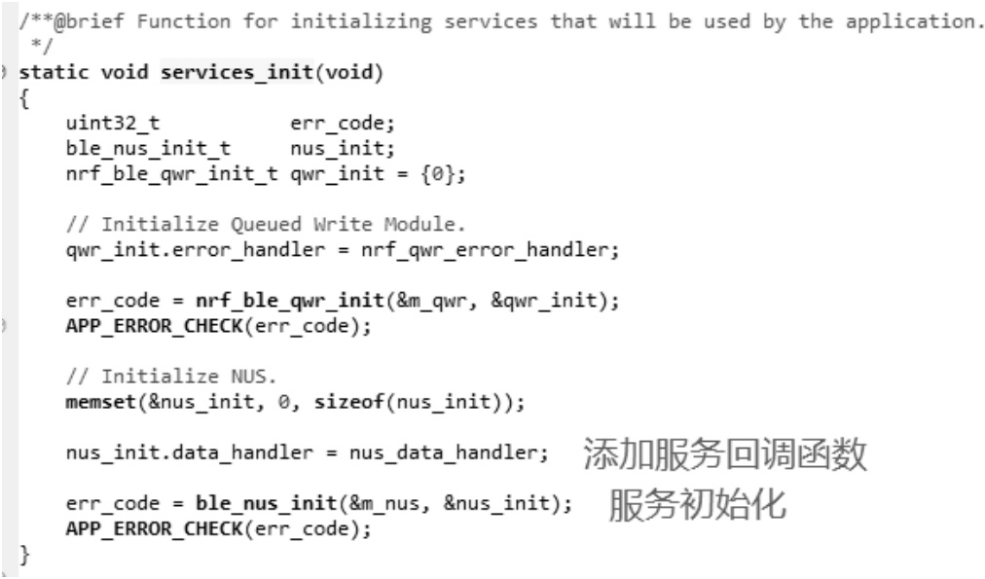


### 协议栈初始化函数

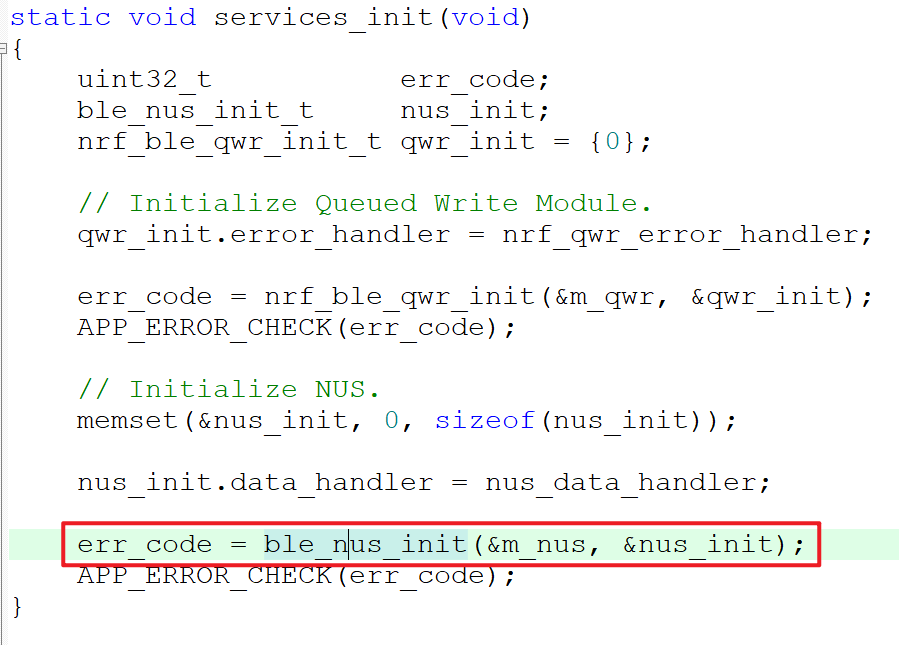


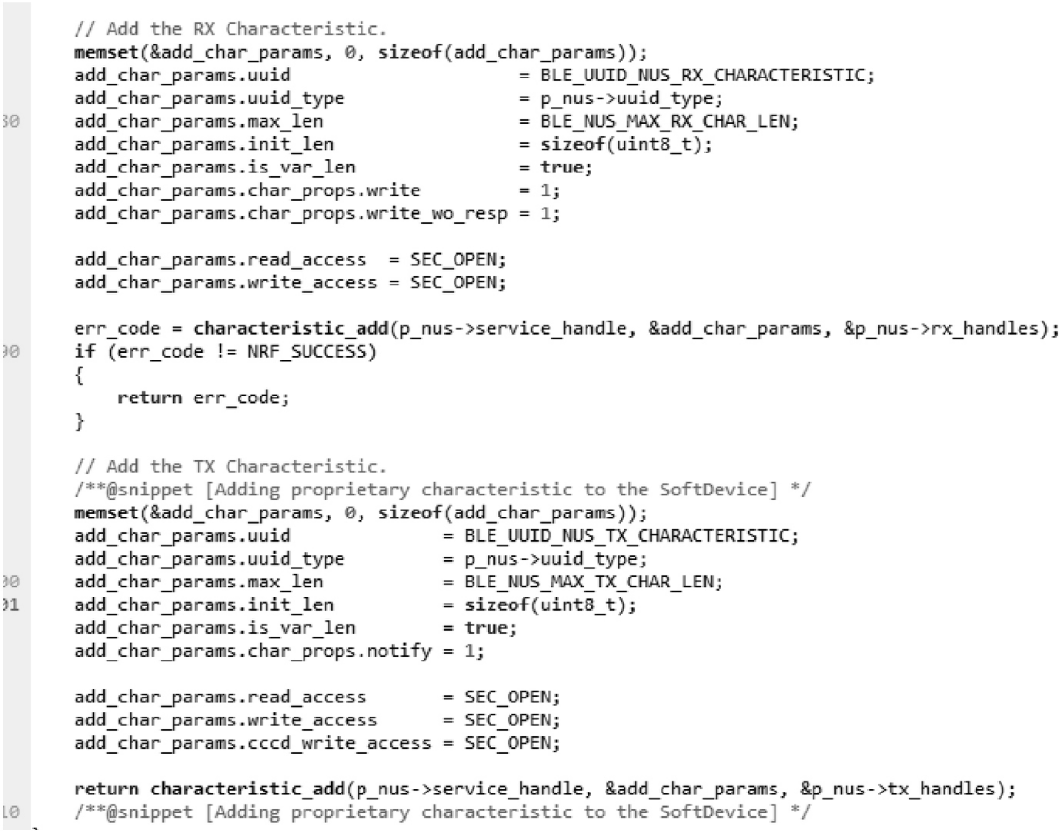
### 服务的初始化

该函数可以添加低功耗蓝牙5.x服务特性

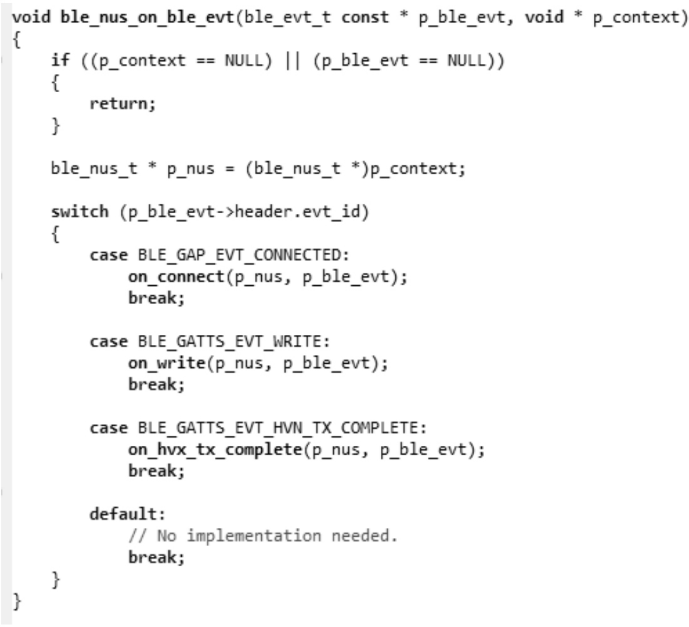


添加TX Characteristic和RX Characteristic的代码如图

下面内容位于此函数中



当事件被触发时，协议栈会将事件通知给应用层，这时就需要处理事件的回调函数，相关代码如图



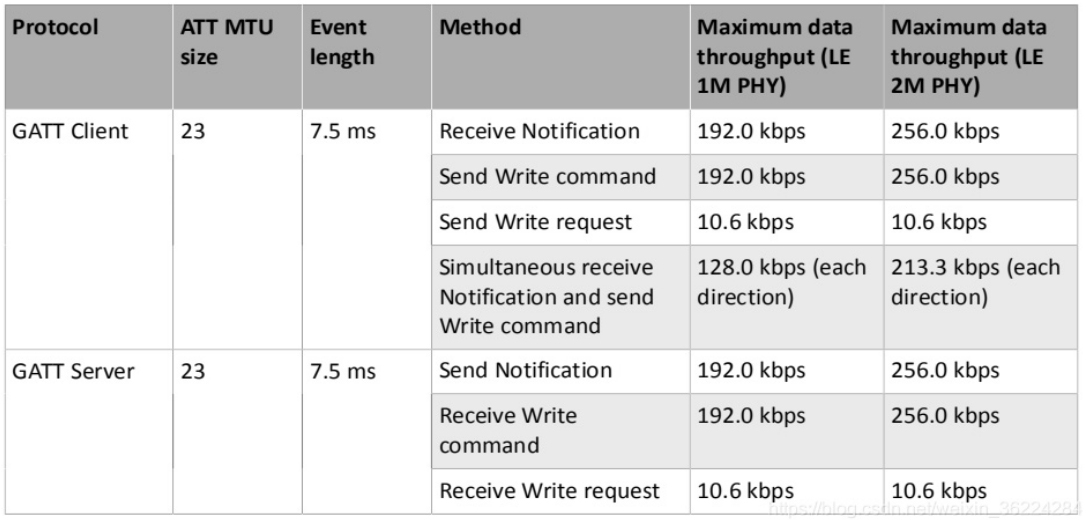
图中的函数都是SDK中的库函数，开发者无须修改，只需要知道协议栈是如何把事件通知到应用层的，调用了哪些回调函数即可。当添加TX Characteristic和RXCharacteristic后，协议栈就会通过结构体指针通知应用层，通过函数on\_write()可以查看具体的实现。

# 低功耗蓝牙5.x的传输速率

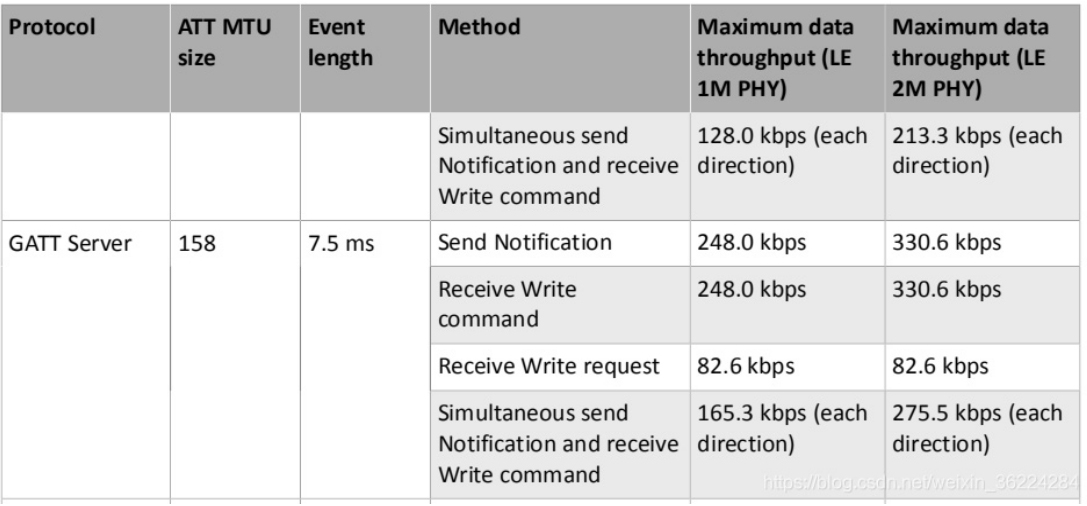
Nordic官方公布的传输速率理论值如下：

在ATT\_MTU=23，并且其他特性都不打开的情况下，即BLE 4.0的标准特性

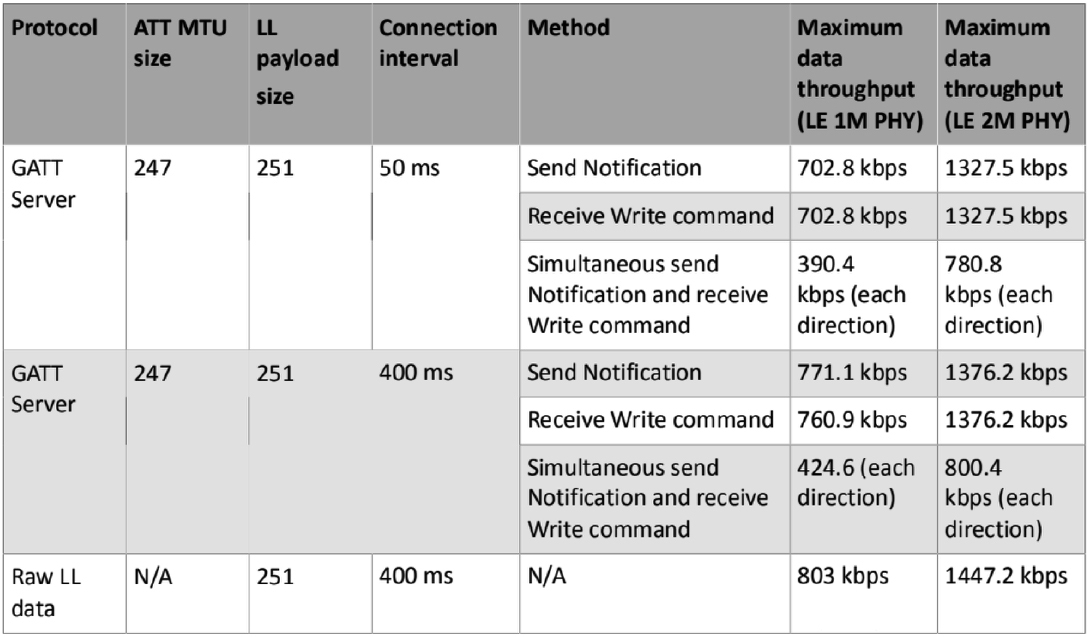
在LE 1M PHY选项下，nRF52芯片的传输速率为192kbps，提高了53kbps左右；在LE 2M PHY选项下，传输速率为256kbps。



在ATT\_MTU=158，并且其他特性都不打开的情况下，传输速率如图，在LE 2M PHY选项下，传输速率为330.6kbps。



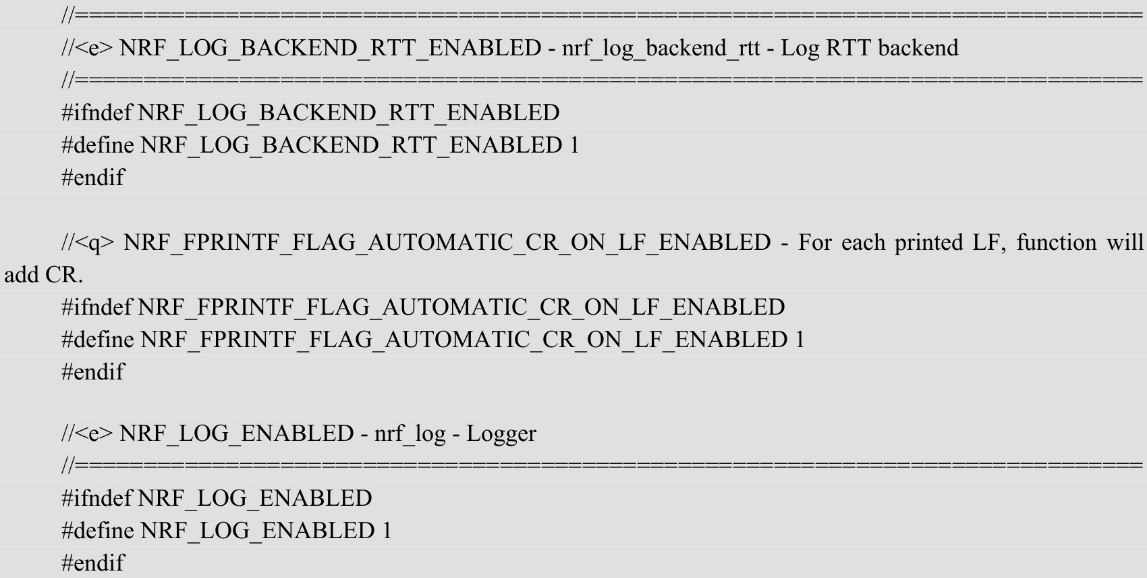
在ATT\_MTU=247、DLE=251、CLE=conn interval情况下，传输速率如图，在打开CLE的情况下，传输速率最高可达1376.2kbps，这是低功耗蓝牙5.x目前所能达到的最大传输速率。



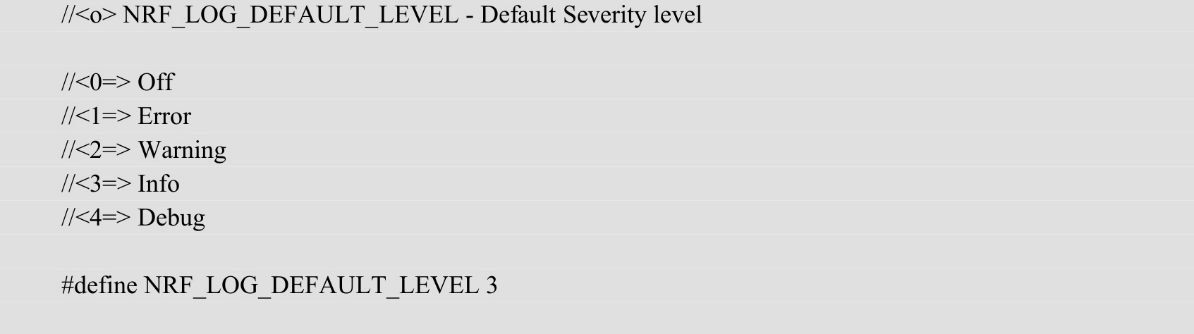
# 开启RTT打印Log

在开发过程中，往往需要查看系统的运行情况，以便进行调试，这时可通过RTT打印Log。开启Log的方法如下。

在sdk\_config.h中修改以下代码：

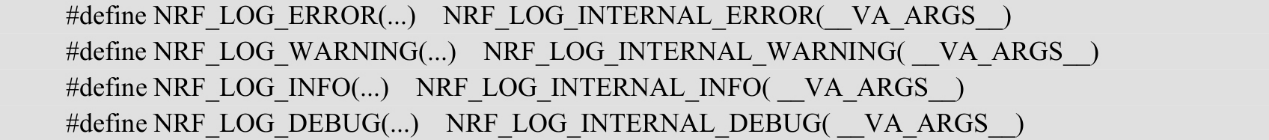


RTT打印Log有4个等级（1～4），相关的宏如下所示：



上述的宏可以看到，0表示关闭Log，1表示只打印Error，2表示打印Error和Warning，3表示打印Error、Warning和Info，4表示打印所有的Log，等级最高。

上述4个等级对应的API函数如下：



# 低功耗蓝牙5.x标准服务

## 配置文件

低功耗蓝牙5.x的配置文件。低功耗蓝牙5.x协议定义了一些标准的配置文件（Profile），配置文件描述了在某个应用场景中低功耗蓝牙设备应具有什么功能、执行什么工作。每个配置文件中都会包含一个或多个服务（Service），每个服务都代表从机的一种能力。

## 服务（Service）

有多个服务，如电池电量服务、系统信息服务等。

如果某个服务是一个低功耗蓝牙5.x协议定义的标准服务，则可将该服务看成配置文件。

## 特性（Characteristic）

在低功耗蓝牙5.x中，数据是通过特性来封装的，而一个或多个特性组成了一个服务。服务可以看成一个独立的单元或者一个基本的低功耗蓝牙应用。主机和从机之间的通信是通过特性来实现的，可以将特性看成一个标签，主机或者从机可以通过这个标签来获取或者写入想要的内容。

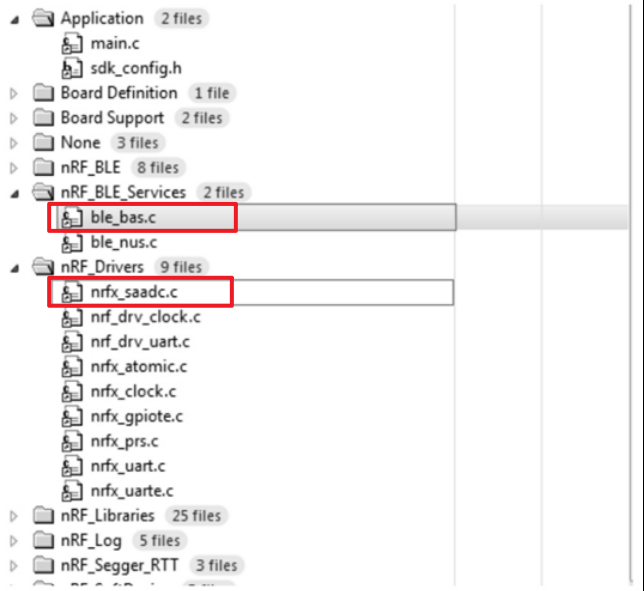
## UUID

UUID即统一识别码，每一个服务和特性都需要一个唯一的UUID来标识。

## 例程实验

### 添加库文件

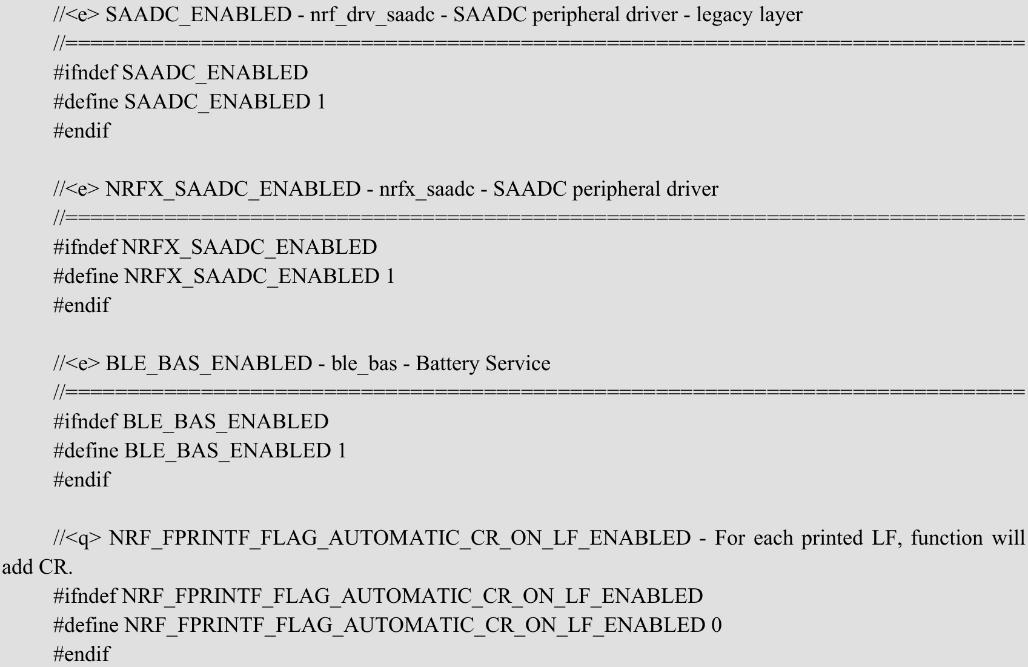
本实验使用Nordic提供的电池电量服务的库文件ble\_bas.c来发送电池电量数据，使用库文件nrfx\_saadc.c来驱动ADC采集电池电量数据。



### 修改sdk\_config.h中相应的宏

打开文件sdk\_config.h，修改相应的宏，以便使能SAADC、使能BAS电池电量服务、开启RTT打印Log，代码如下：

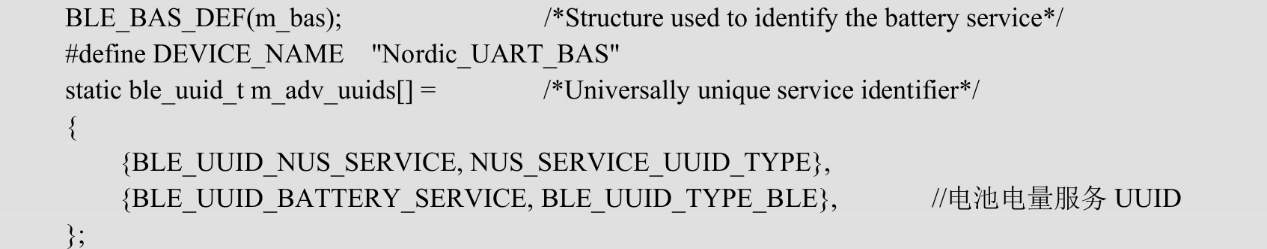
默认其实是打开的



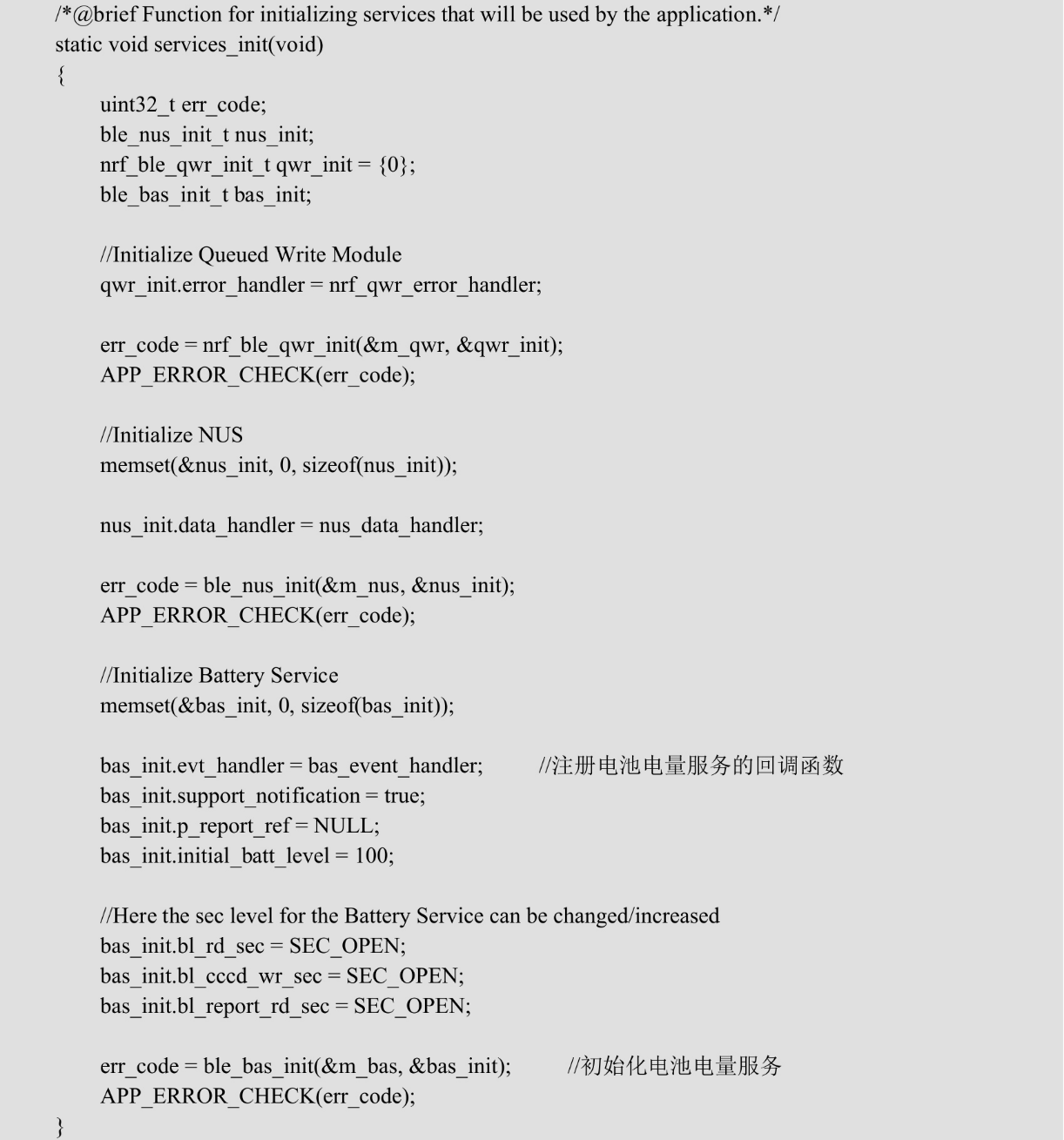
### 初始化电池电量服务及代码

电池电量服务是低功耗蓝牙5.x协议定义的标准服务，无须开发者构建该服务的函数。

Nordic提供了电池电量服务的库文件，只需要在main.c中添加“#include "ble\_bas.h"”即可引用相关的函数。下面的代码声明了电池电量服务实例m\_bas，将广播名称修改为“Nordic\_UART\_BAS”，并修改了广播中的UUID，添加了电池电量服务UUID（0x180F）。



在服务初始化函数中添加电池电量服务的初始化，代码如下：



### 添加ADC功能模块

实现VDD电压检测。

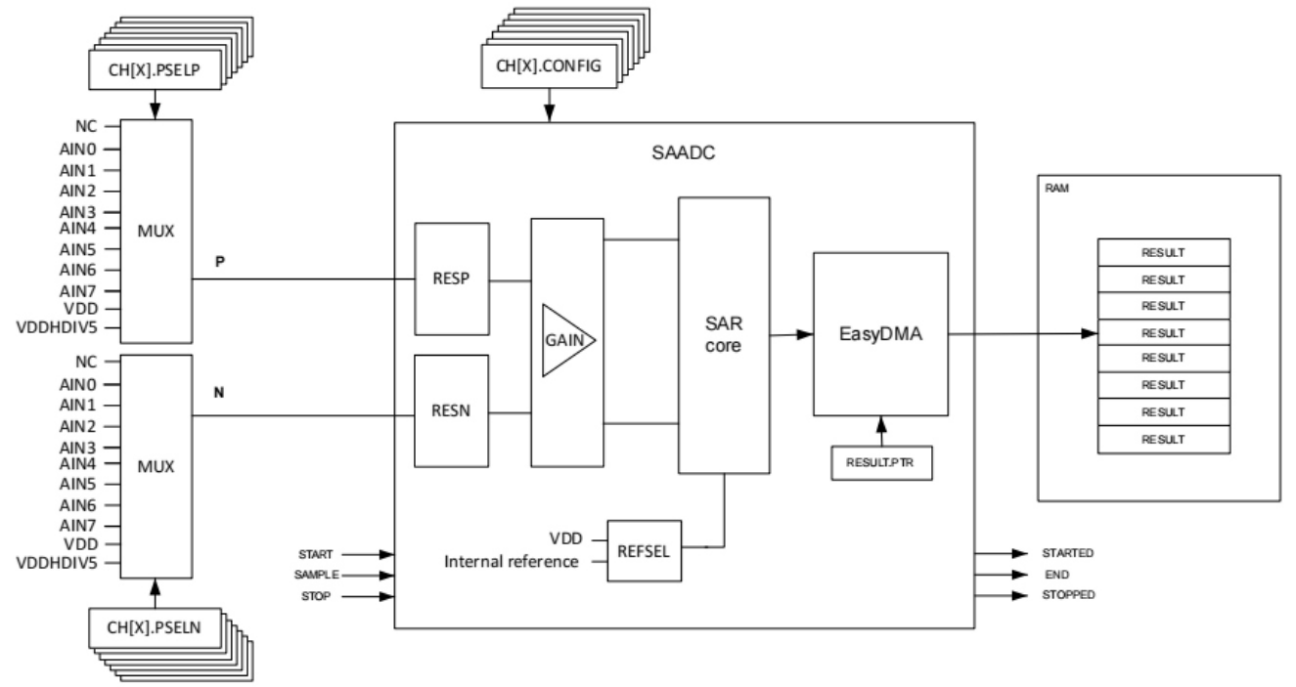
低功耗蓝牙设备通常采用电池供电，因此监测设备的电池电量变化是低功耗设备正常工作的重要环节，可以确保设备的正常工作并在电池电量下降到一定范围时及时预警。

监测电池电量变化需要使用到ADC外设，即芯片内部的SAADC模块。

使用SAADC模块，需要在main.c中添加#include "nrf\_drv\_saadc.h"，引入SAADC库文件。

首先初始化SAADC模块，然后配置通道相应的参数和缓冲。

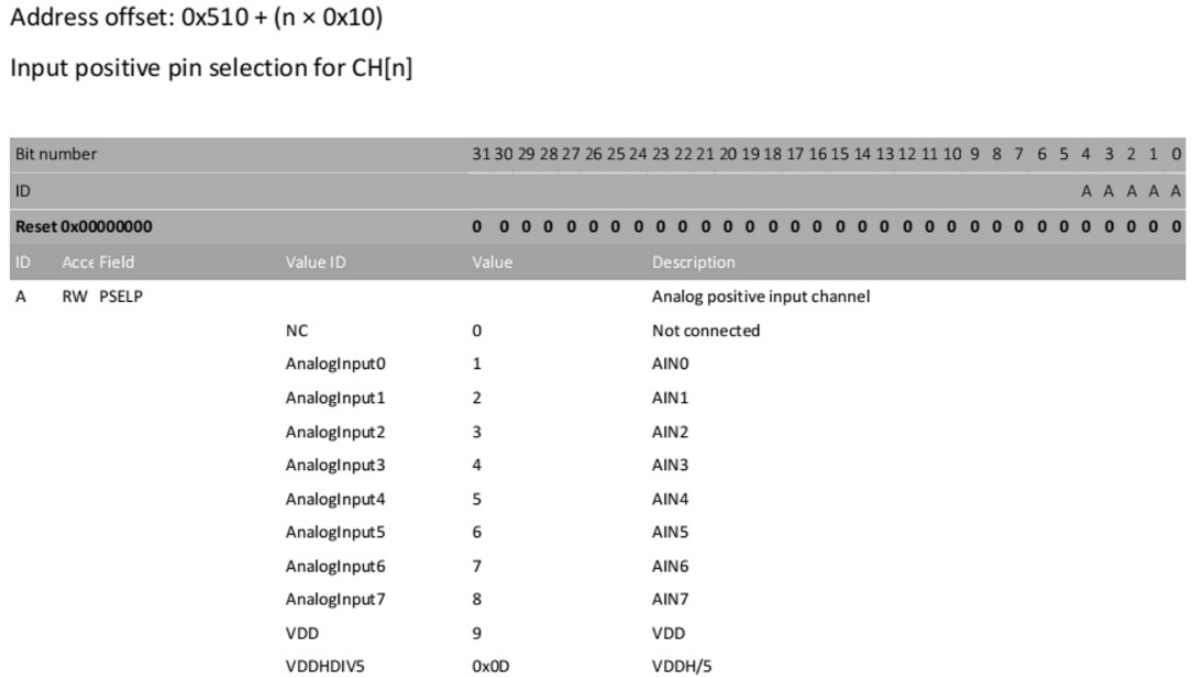
为了实验方便，可将ADC输入引脚配置为VDD引脚，相当于ADC输入引脚直接与VDD引脚相连，直接测量VDD引脚的电压。ADC输入引脚可以从AIN0～AIN7引脚选择，内部框图和相关硬件如图所示。



ADC外设总共有8个独立测量通道，最多可配置8组单端采样输入或4组差分采样输入。

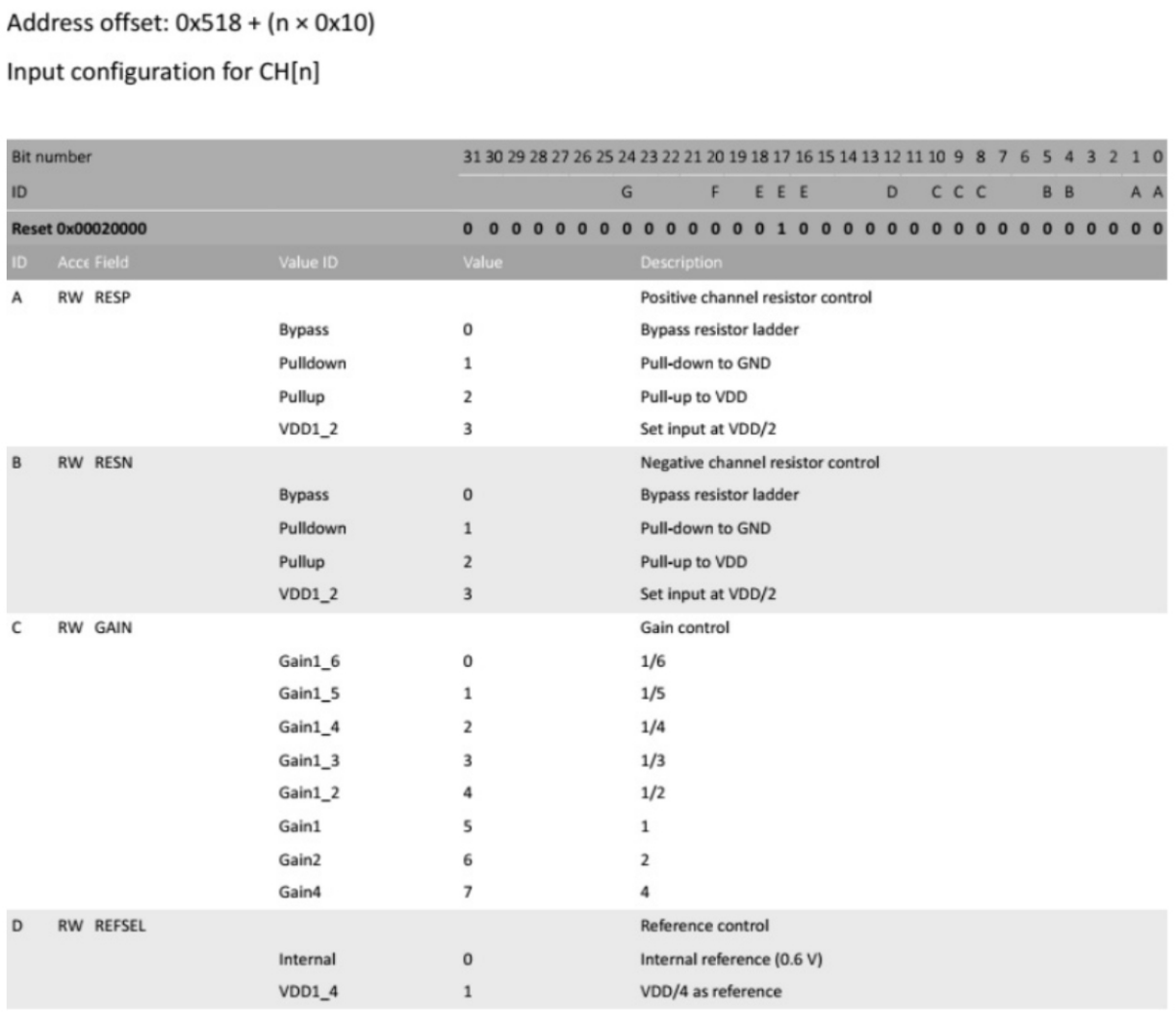
通过寄存器CH[n].PSELP（n=0～7）和CH[n].PSELN（n=0～7）可以配置ADC的采样输入，并将采样输入映射到相应的物理引脚。

本实验选择VDD引脚作为ADC采样输入，当采用电池供电时，采样的VDD电压即电池电压，这样做还可以节省一个I/O引脚。



参考电压可选择设置为VDD引脚电压的1/4或内部基准电压（0.6V），电压增益可配置为1/6到4（共8挡），由CH[n].CONFIG（n=0～7）寄存器进行设置

注意：如果选择VDD引脚电压的1/4作为参考电压，VDD引脚电压容易受到外部影响而产生波动，则会对测量结果有影响；芯片内部基准电压的精度高、稳定性好、温漂小，建议优先选择内部基准电压（0.6V）作为参考电压。

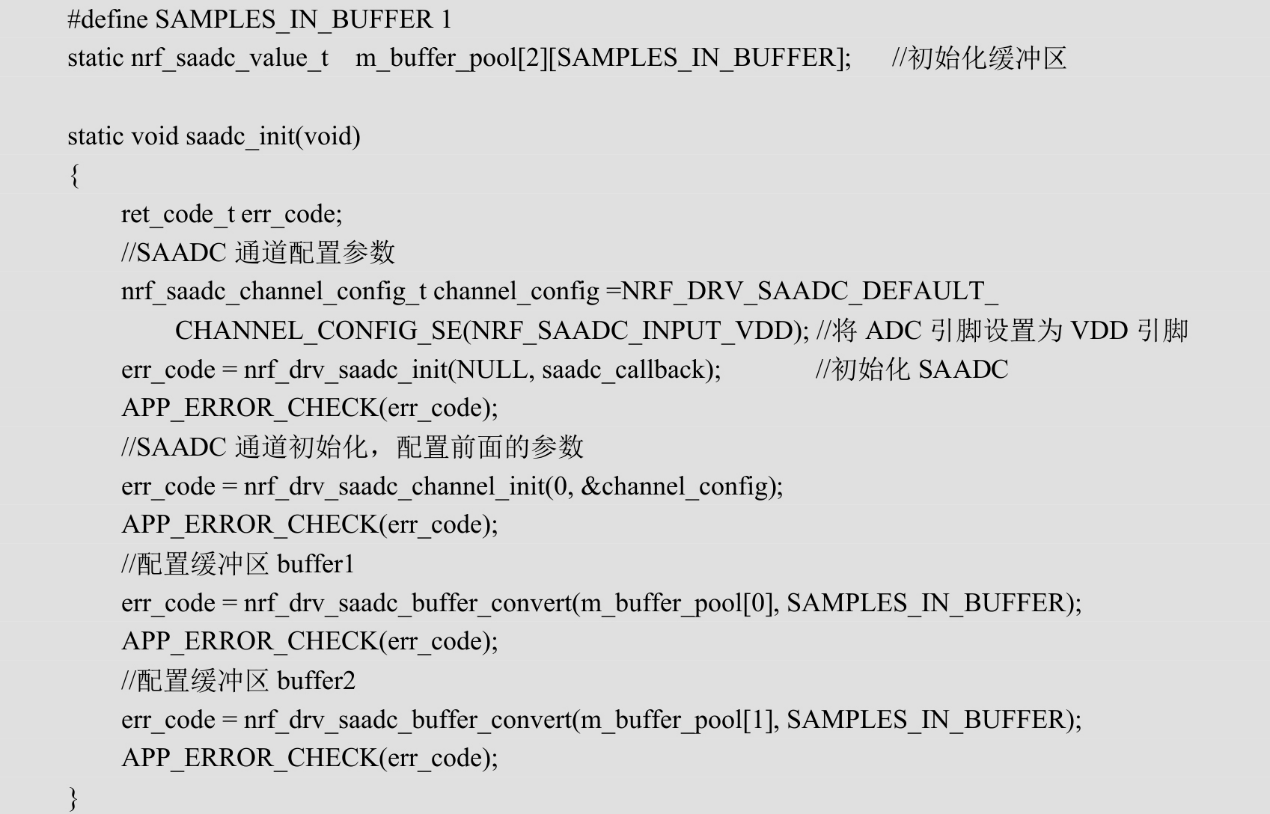


参考电压和电压增益可以根据SAADC的待测电压有效输入范围来进行相应设置，公式为：

ADC输入引脚的电压=(±0.6V或±VDD引脚电压的1/4)/电压增益

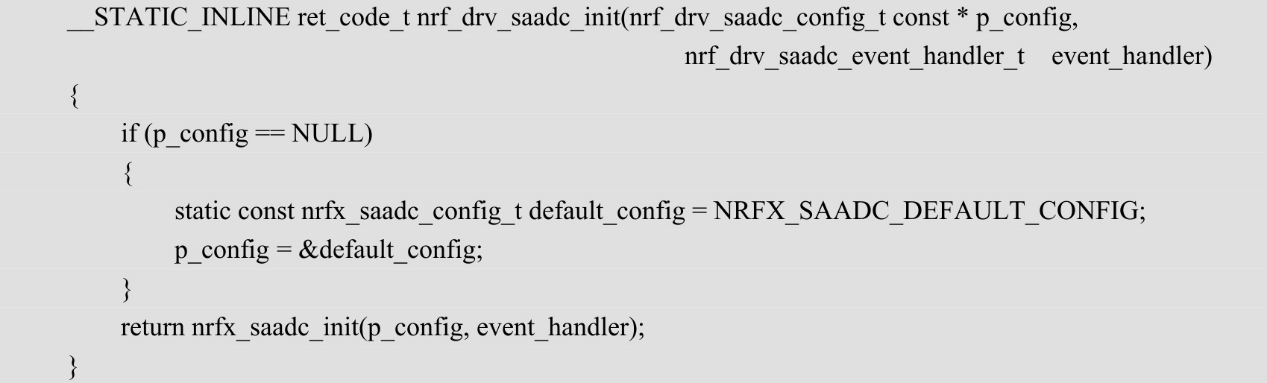
### 初始化ADC功能代码

#### 初始化SAADC

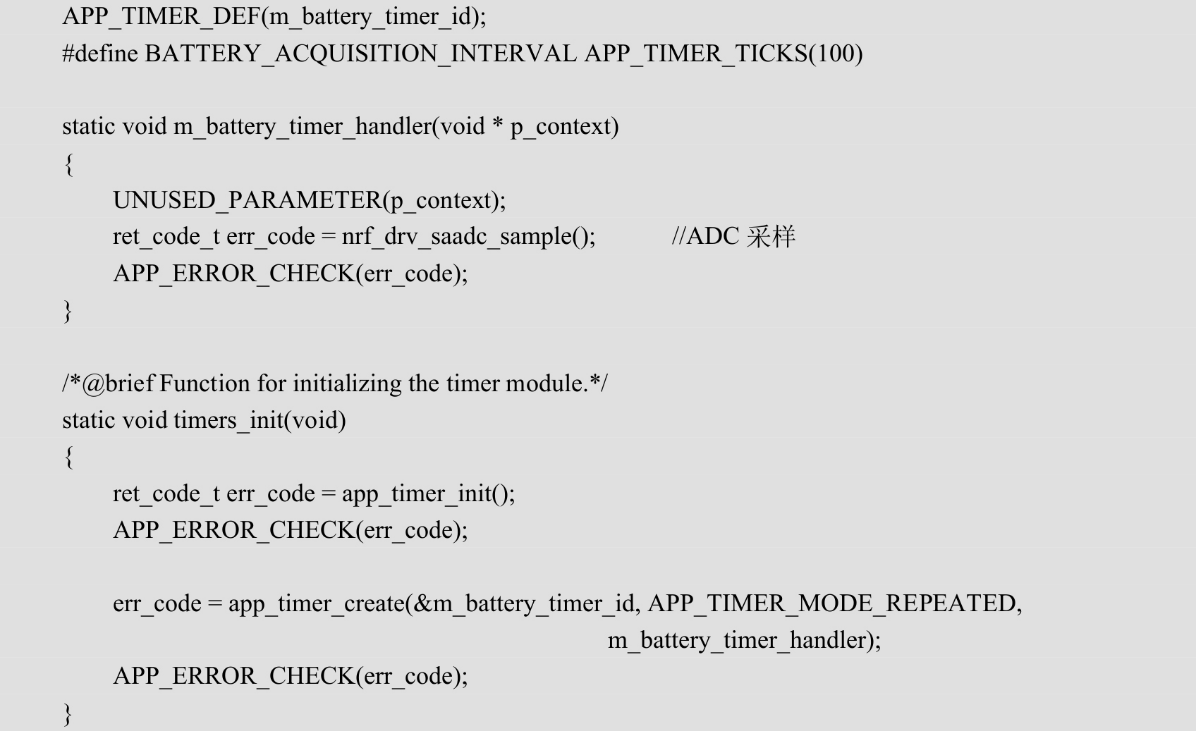


#### 回调函数

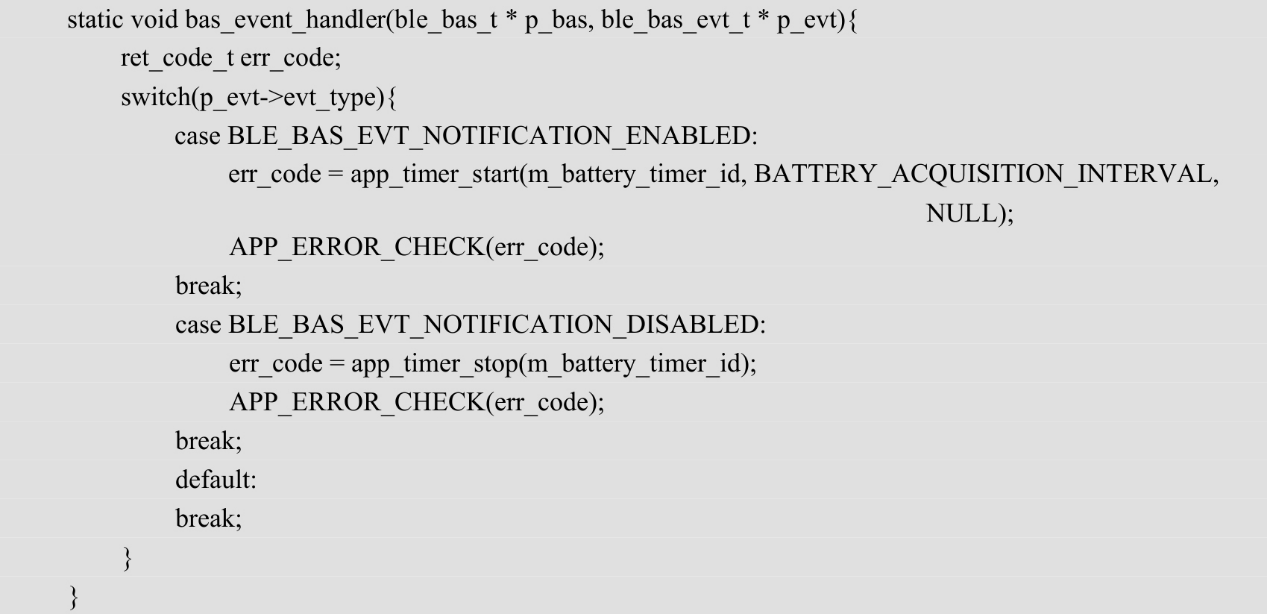
上述代码在调用函数nrf\_drv\_saadc\_init()时，该函数的第1个参数是NULL，表示使用默认的配置参数NRFX\_SAADC\_DEFAULT\_CONFIG；该函数的第2个参数注册了回调函数saadc\_callback()，在该回调函数中可对数据进行处理。



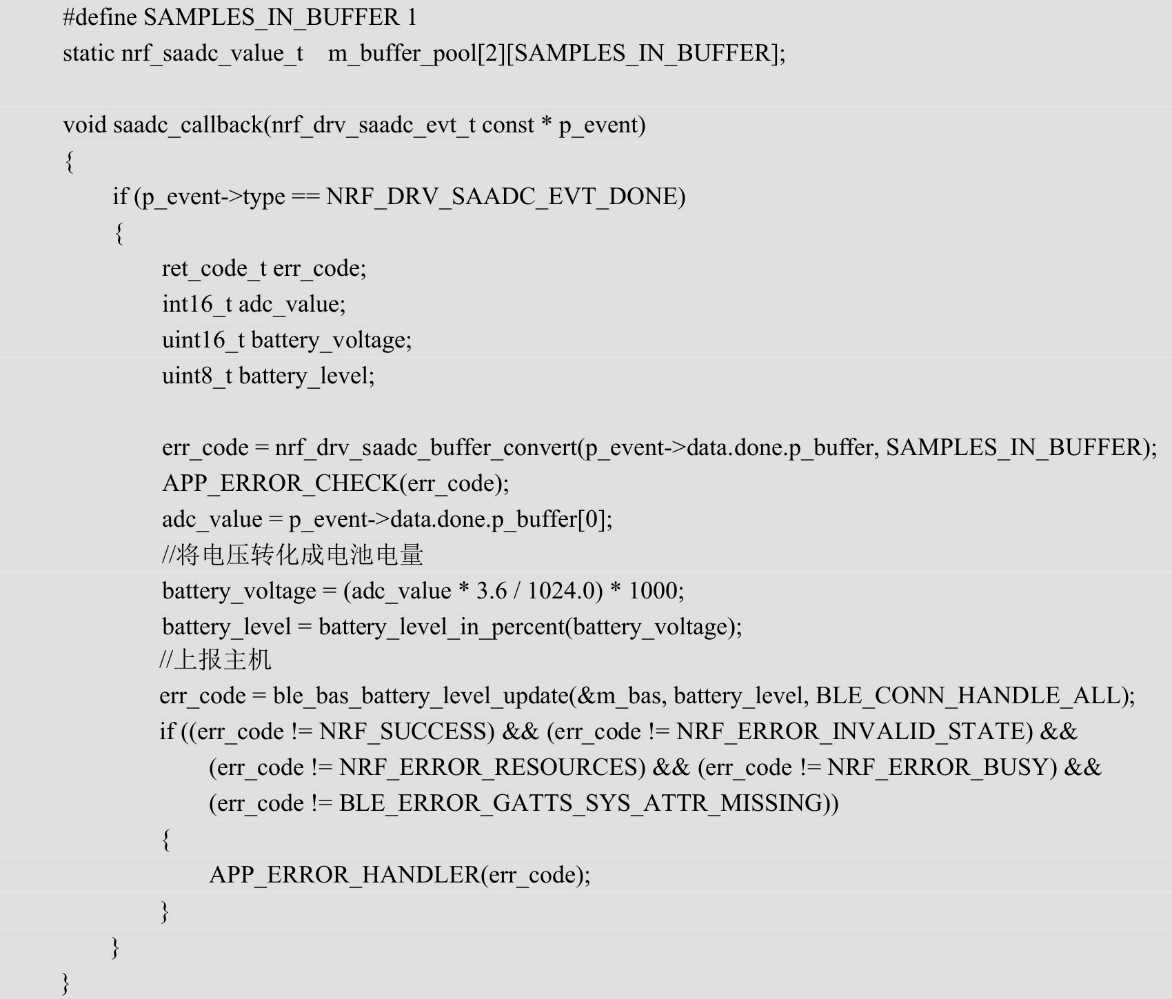
#### 添加定时器

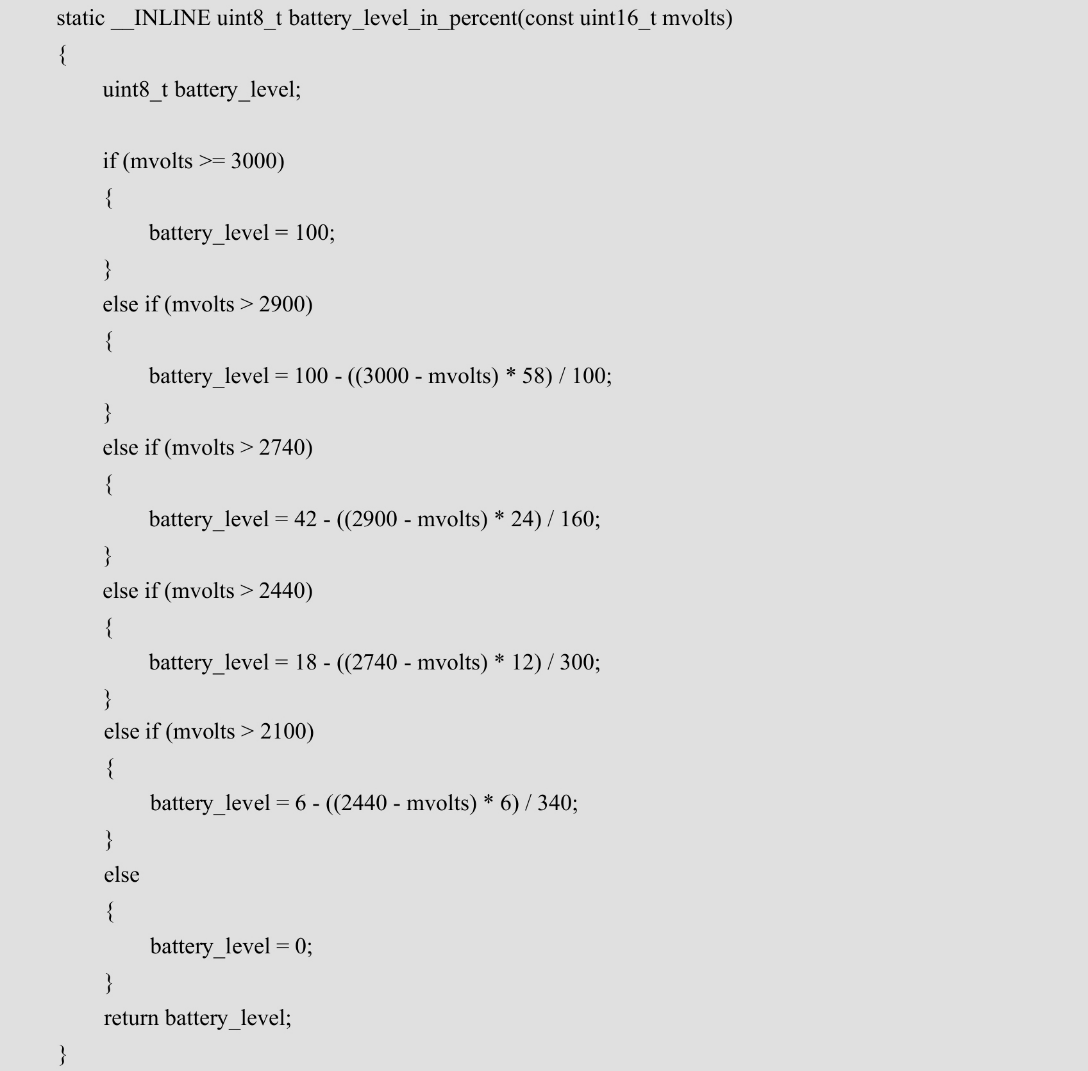


#### 电池电量服务回调函数

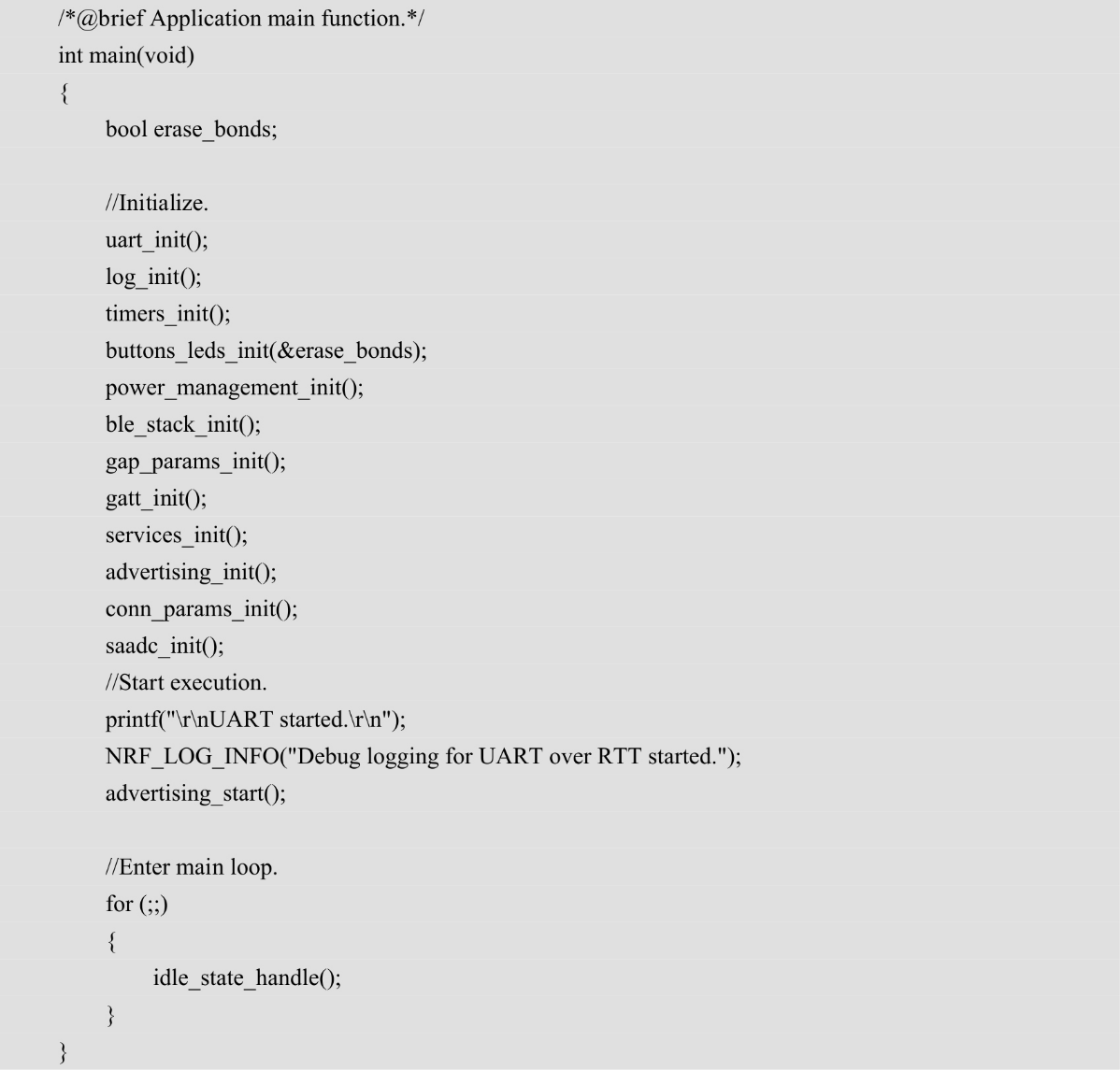


#### 获取采样结果并上报电池电量





#### main函数的编写



# 私有服务