**FAQ**

****

FAQ\_ 关于BlueNRG-1/2 OTA的流程与优化

# **Keywords**

OTA, 升级,

# **Q&A**

Q:

客户经常问：BlueNRG-1/2的升级怎么这么慢？能不快一点，别的厂家的芯片升级都很快，BlueNRG-1/2为什么这么慢？ BlueNRG-1/2 OTA的APP有源码吗？OTA升级如果固件low和固件high对调升，会导致升成砖头如何避免这个问题？等问题。

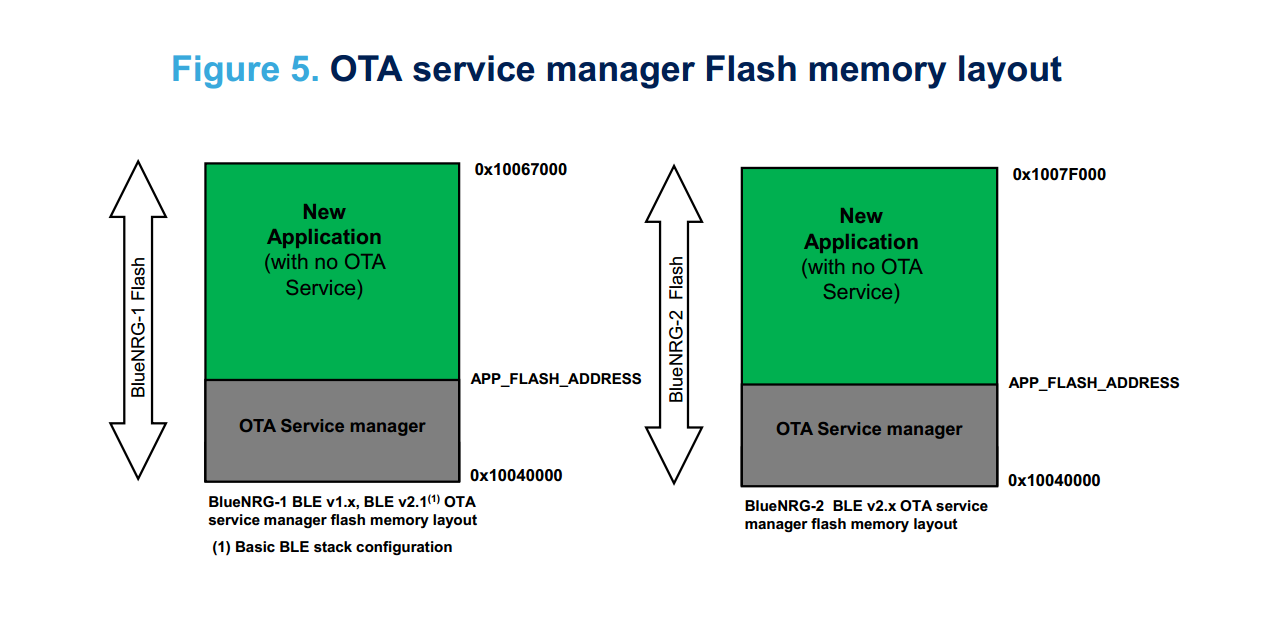
A: 针对这些问题，我一一做解答。长文预警，请选择需要的部分阅读。如果需要测试可以直接到文章末尾，替换相关文件，在代码中使能宏定义OTA\_EXTENDED\_PACKET\_LEN =1，安装相关的附件apk.然后然后编译下载测试。

优化后固定协议栈的方式大概可在5s左右完成升级过程，实际测试与手机和升级固件大小而定；非固定协议栈大概在28s左右完成升级。

## BlUeNRG-1/2的默认支持的升级方式：

BlueNRG-1/2 默认支持三种升级方式。

一种是 OTA Reset Manager（BootLoader） + APP(内包好OTA升级服务)， （备注: APP既是下图中的application的缩写）



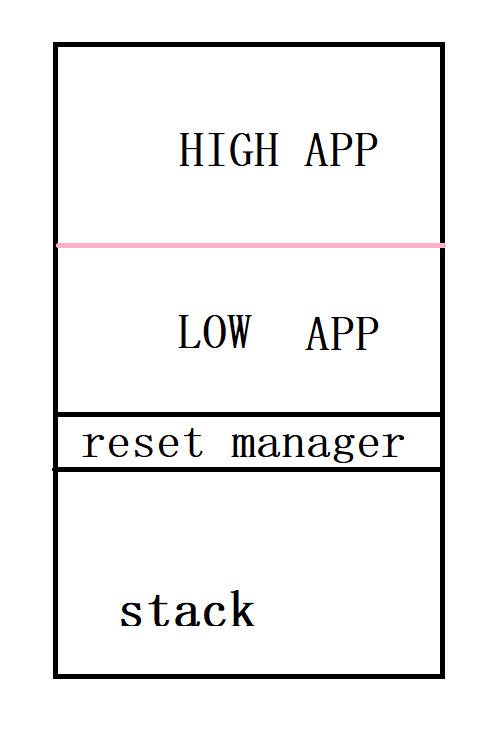
这种升级方式是将OTA服务放在BootLoader那。OTA Reset Manager与APP都包含协议栈，协议栈可以被升级。当需要OTA升级时，需要跳转到执行BootLoader程序。

另一种是OTA service manager（BootLoader）+APP(内不包含OTA服务)



这种升级方式是将OTA服务放在APP那。OTA Reset Manager与APP都包含协议栈，协议栈可以被升级。应用使用两块区域有备份。注意Reset manager会提前擦除好相应的数据块，如果使用了片内的Flash存储数据时需要注意这点，如果没有提前预留数据区域的存储位置,默认升级后，复位后，Flash相应的区域会被擦除。

还有一种是固定协议栈的方式



这种方式是将OTA服务放在APP那，但是区别于上一种是，这种协议栈是采用固定的方式。OTA升级时不升级协议栈。

从三种方式来说，采用固定协议栈的方式理论上是最快的。固定协议栈的修改方式见其他资料。（固定协议栈的资料请在工程目录下找到工程：

BlueNRG-1\_2 DK 3.1.0\Project\BLE\_Examples\BLE\_SensorDemo\_Static\_Stack

BlueNRG-1\_2 DK 3.1.0\Project\BLE\_Examples\BLE\_Static\_Stack

工程目录下带相关使用说明，更详细的使用说明，请参考SDK安装目录下index.html

:C:\Program Files (x86)\STMicroelectronics\BlueNRG-1\_2 DK 3.1.0\Docs\index.html

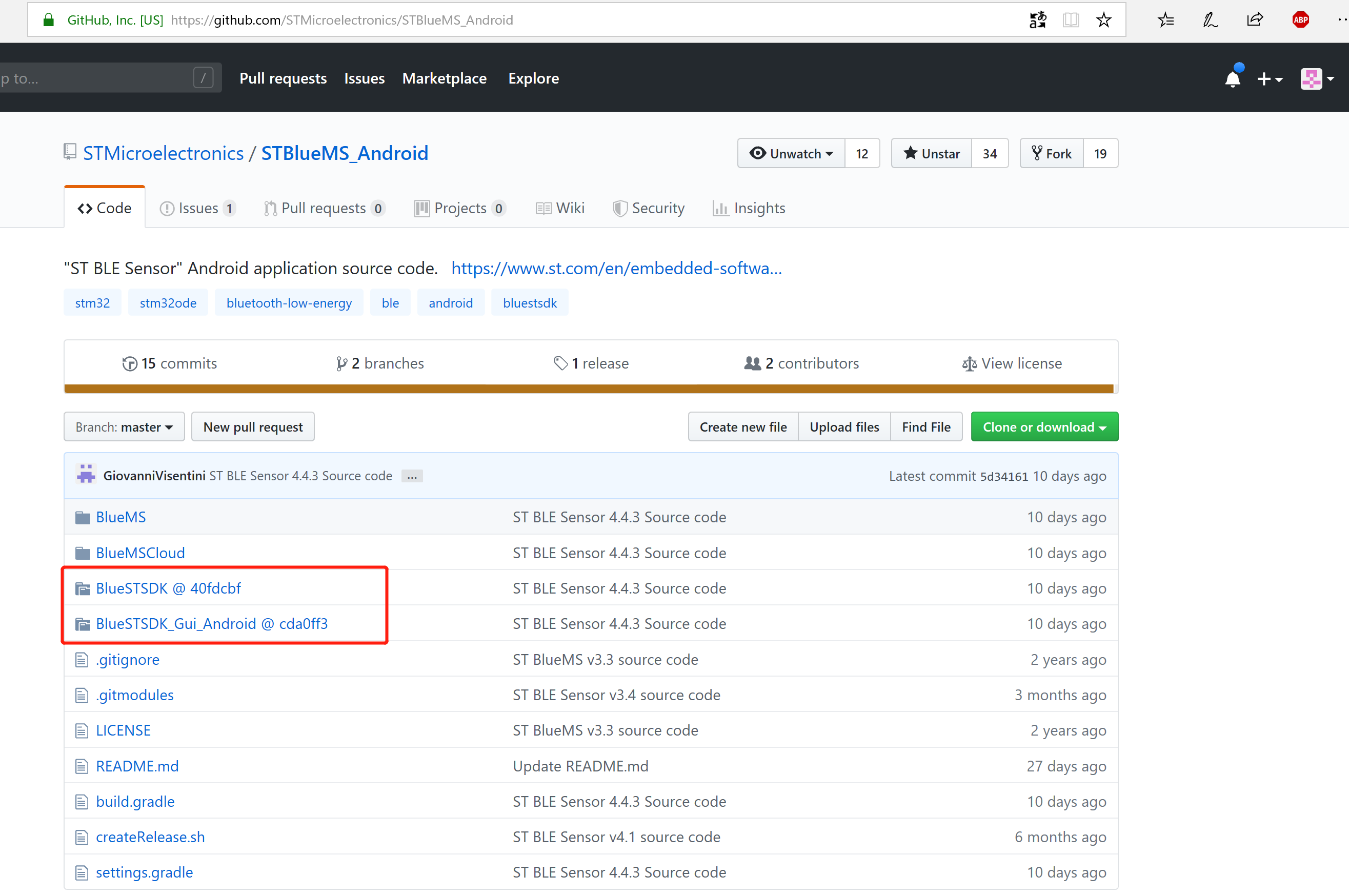
双击打开index.html后选择：

[BlueNRG-BLE Stack Library Static Stack documentation](file:///C:\Program%20Files%20(x86)\STMicroelectronics\BlueNRG-1_2%20DK%203.1.0\Docs\BLE_Static_Stack\BLE_Static_Stack.html)

## BlueNRG-1/2 OTA的APP的源码：

BlueNRG-1/2 本身提供使用BlueNRG-1/2 板子加GUI工具进行OTA，也提供APP升级方式。

<https://github.com/STMicroelectronics/STBlueMS_Android>



需要注意的是红框部分需要单独分别下载到。

## BlueNRG-1/2 OTA的APP的源码：

如果优化升级速度。方式有很多种。最常见的是打开扩展包的方式。修改MTU。

打开方式使能 全局宏OTA\_EXTENDED\_PACKET\_LEN=1

其他修改步骤

1. 在OTA\_blt.c中拷贝这段代码到上面（详细代码可以参考本文档中最后的附件文件）。



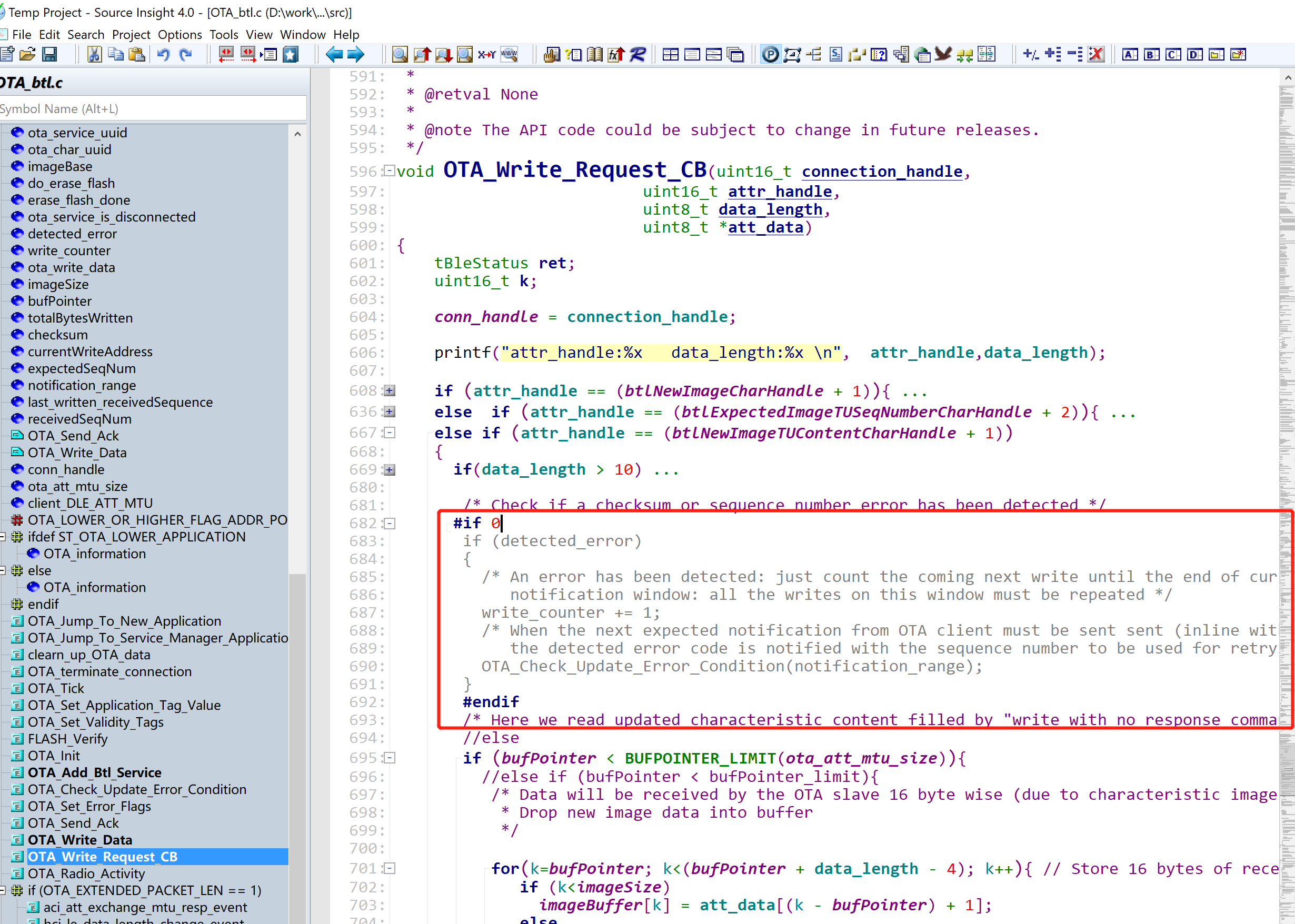
这里修改主要是因为BlueNRG-1/2 OTA设计时和Android端应用接口的差异。Android 或者IOS应用接口只要支持长包默认就是长包，并不会有一条单独的指令设置长包发送。

1. 修改支持长包后，会发现有部分手机的的OTA升级兼容性并不是那么好。

用华为某些老款的手机测试结合空中抓包发现，第一个长包数据包容易丢失。

当数据丢失实测和APP端结合的重发机制有bug。

建议修改代码:



这里将检测到错误发送注释，主要是因为APP端本身有一个超时机制，如果超过一定时间没有收到设备端的回复就会重发。当收到错误反馈也会重置发送，在这点上如果同时起作用会导致无法同步上。

## BlueNRG-1/2 OTA的流程：

1. OTA服务特征值简单介绍。

OTA server characteristic

Btl image characteristic （device—>mobile phone app）

---特征值记录了可供的升级位置的起始地址和结束地址



Btl new image characteristic (（device<—>mobile phone app）)

---特征值记录了OTA版本信息， 以及APP将要升级的信息\

Free Image base = 0x1005F800 ; Image size = 0x00013730, numPages = 39

Btl new image content characteristic (（device<-----mobile phone app）)

---特征值用来传输firmware的数据， 16-byte \* N

Btl expected image sequence number characteristic

---特征值用来设备端告知APP 希望接收固件的序列号。 即第几个数据包

1. OTA升级服务和Android APP交互过程简单介绍



## BlueNRG-1/2 OTA升级防止升级成砖头的措施：

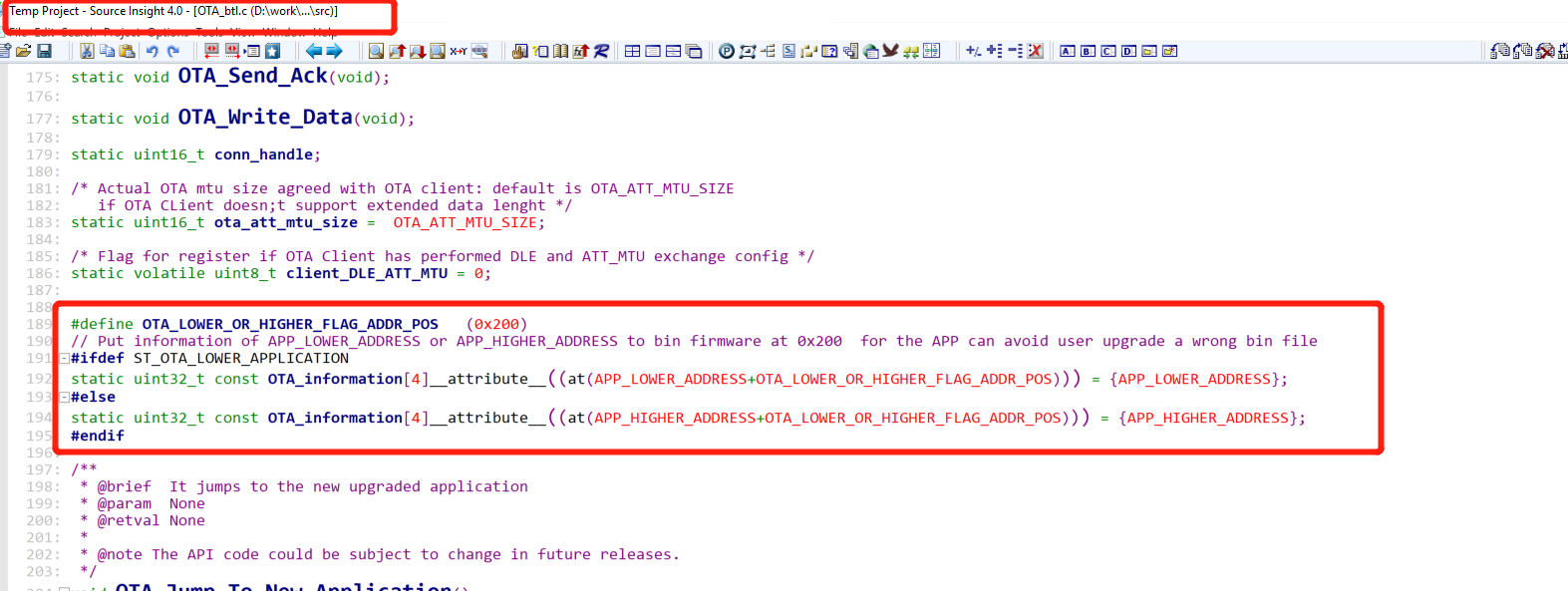
由于BlueNRG-1/2体系结构在编译代码的时候，产生的bin有部分是绝对寻址的方式，在low APP 和high APP对调后烧录是没有办法跑起来的。即当OTA选择错误的固件，升级后设备就变成砖头，无法通过再次OTA升级（即lower bin无法放在higher bin的Flash位置上运行）。

为了解决容易选择错误文件导致设备变砖头的问题，下面方法供参考：

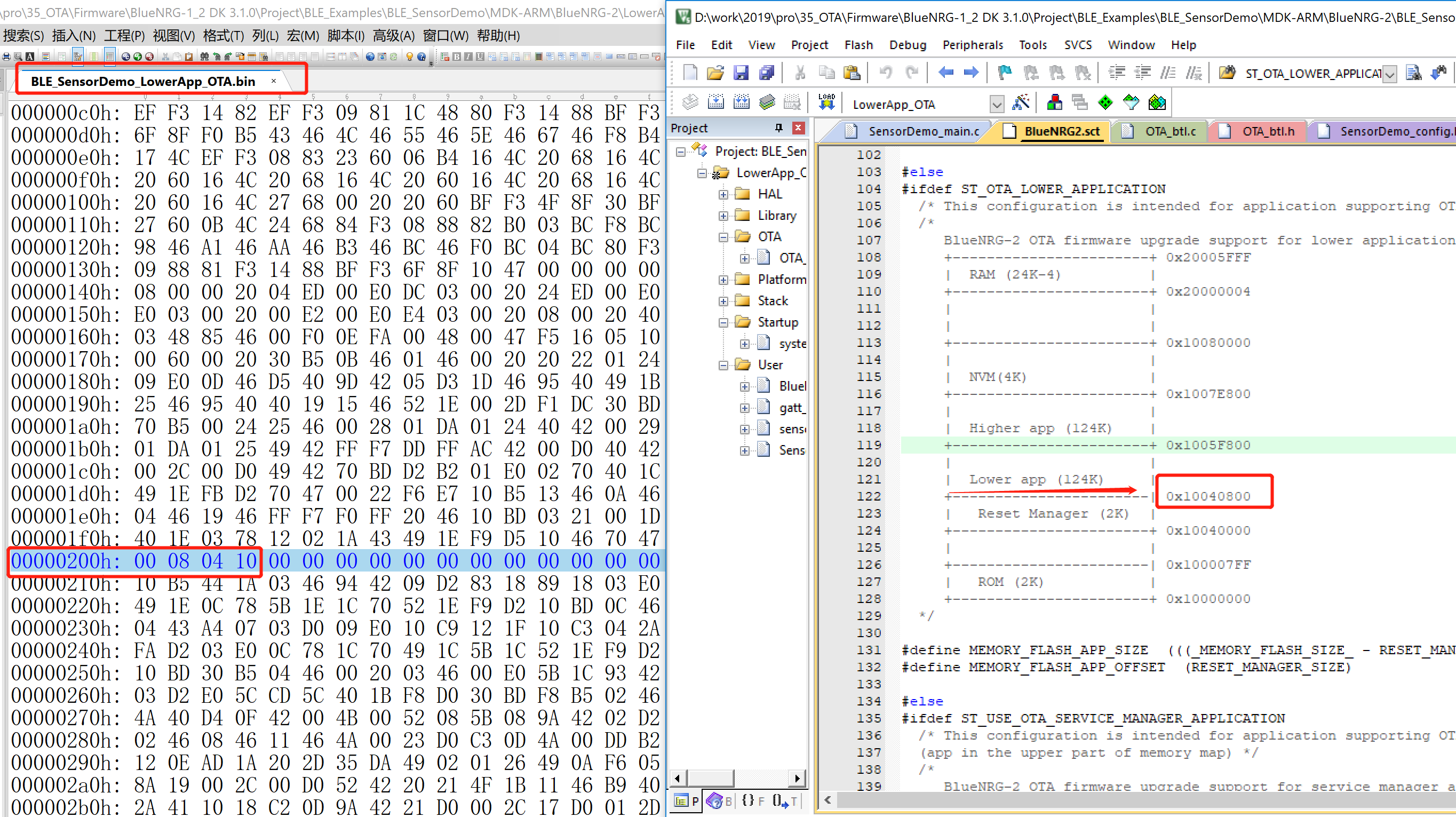
1. 这个方法原理上是增加检测机制检测代码当前选择的bin文件是否是和当前运行的bin文件同属于low 或者同属于high。如果是则是则不允许升级。
2. 固件端代码修改处：增加一个常量数据防止在编译出来的bin文件的0x200这个位置（**这里所示代码是在keil环境验证过，如果IAR环境编译修饰符可能略微有些变化**）

**IAR的绝对寻址可以写成类似如下（使用一个@+地址符号绝对寻址）：**

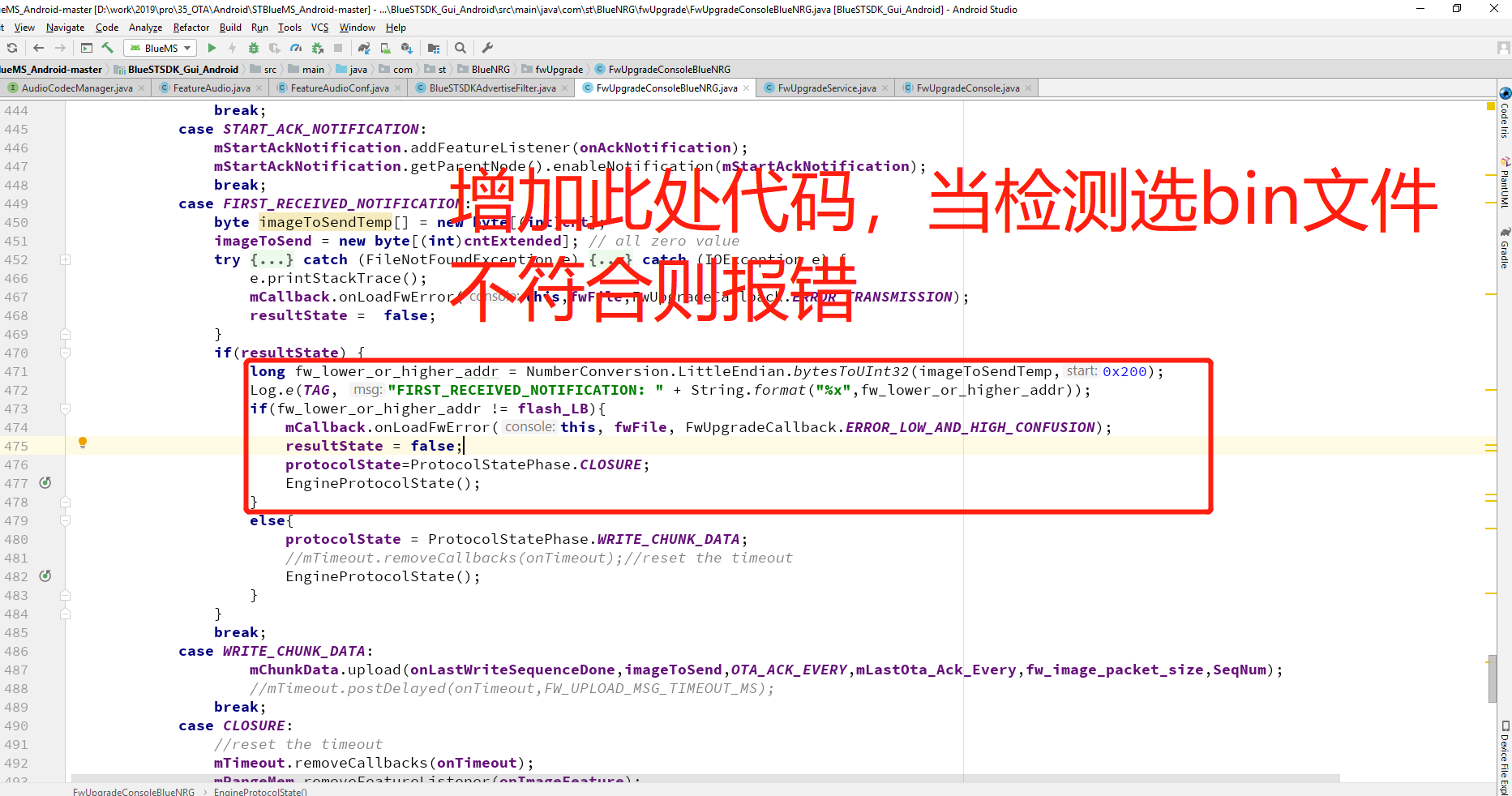
**static uint32\_t const OTA\_information[4] @ (((APP\_LOWER\_ADDRESS+OTA\_LOWER\_OR\_HIGHER\_FLAG\_ADDR\_POS))) = {APP\_LOWER\_ADDRESS};**



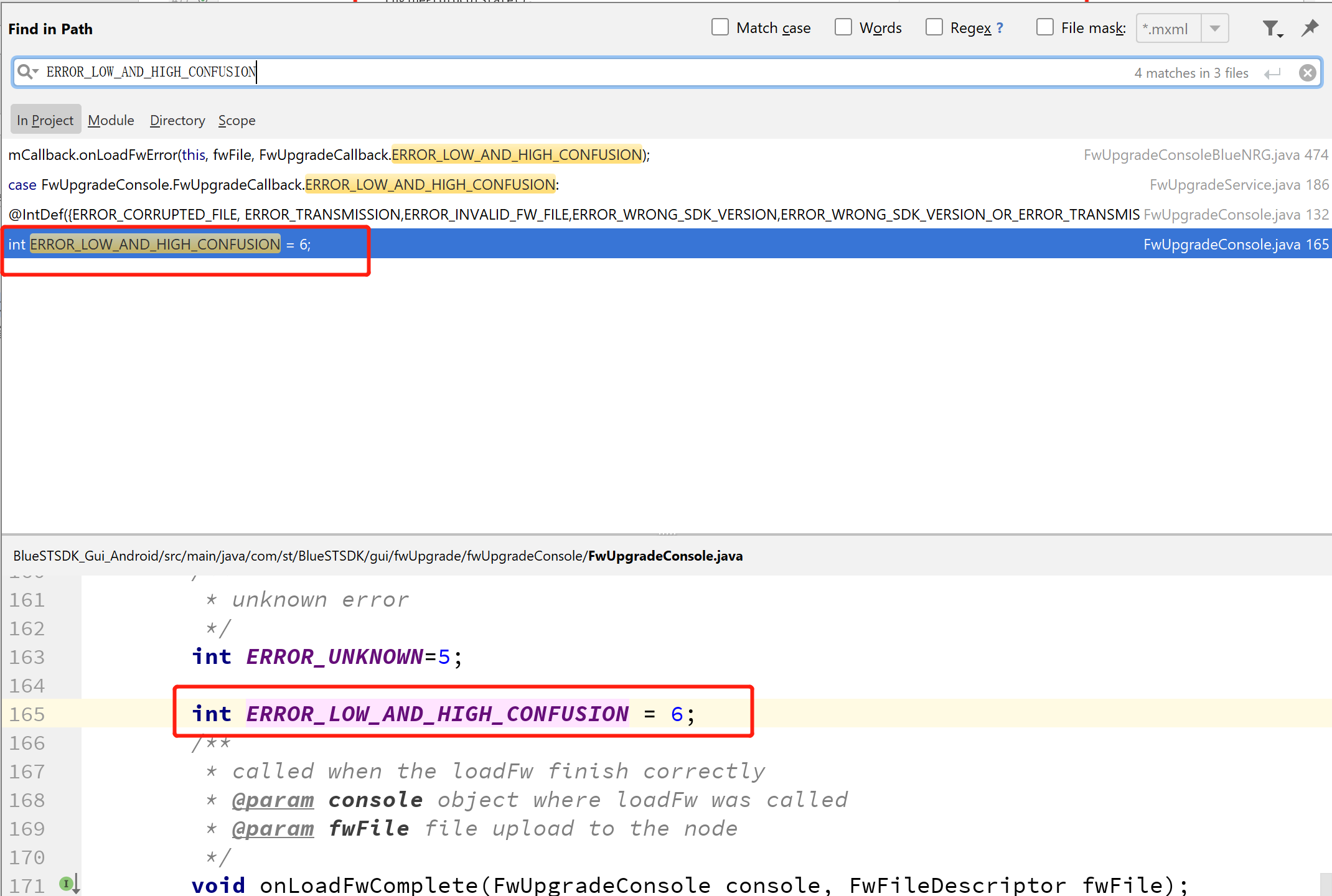
编译的bin文件内容如下图所示

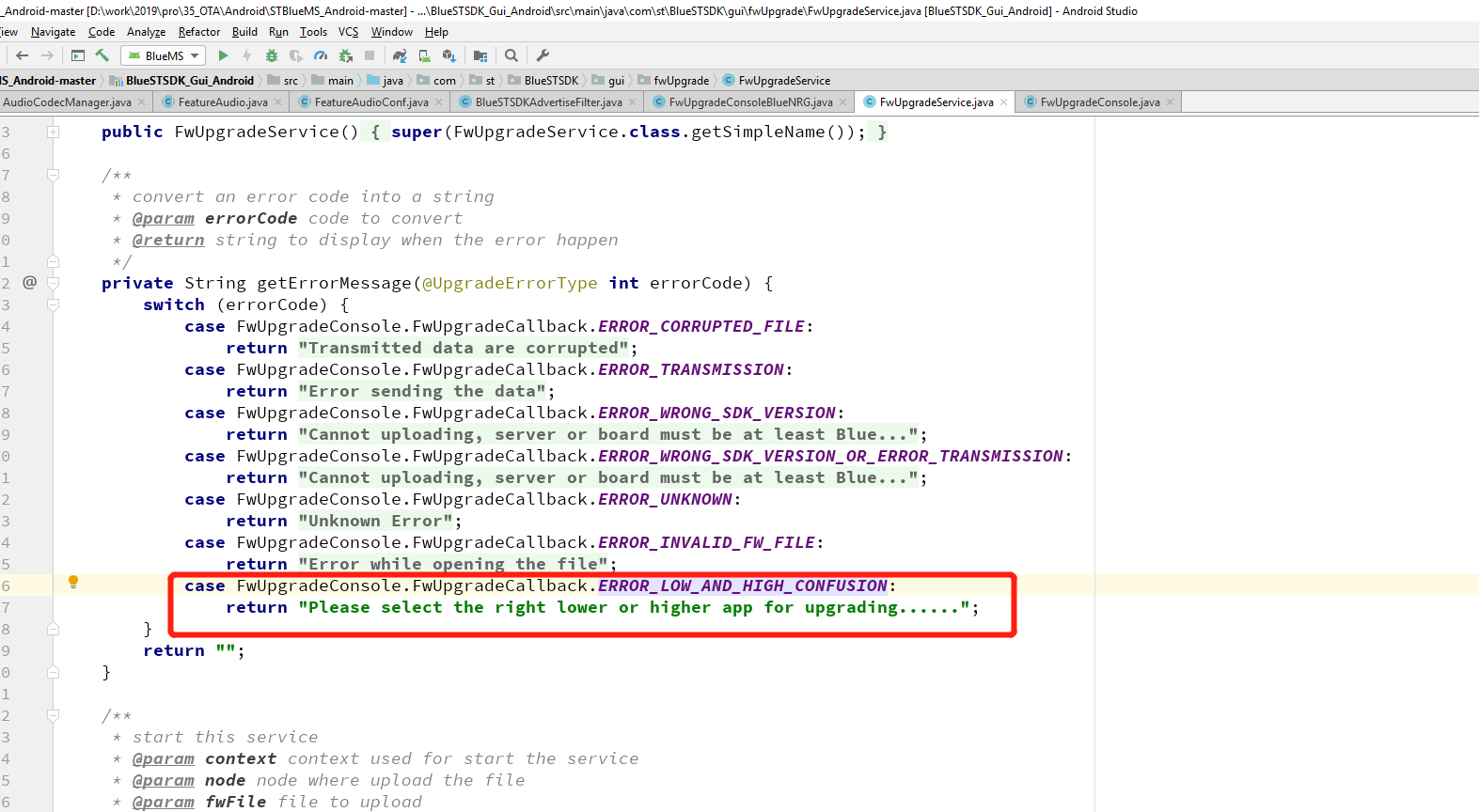


可以代码的作用就是实现通过编译当前固件的地址信息到bin文件里面。



增加错误提醒类型

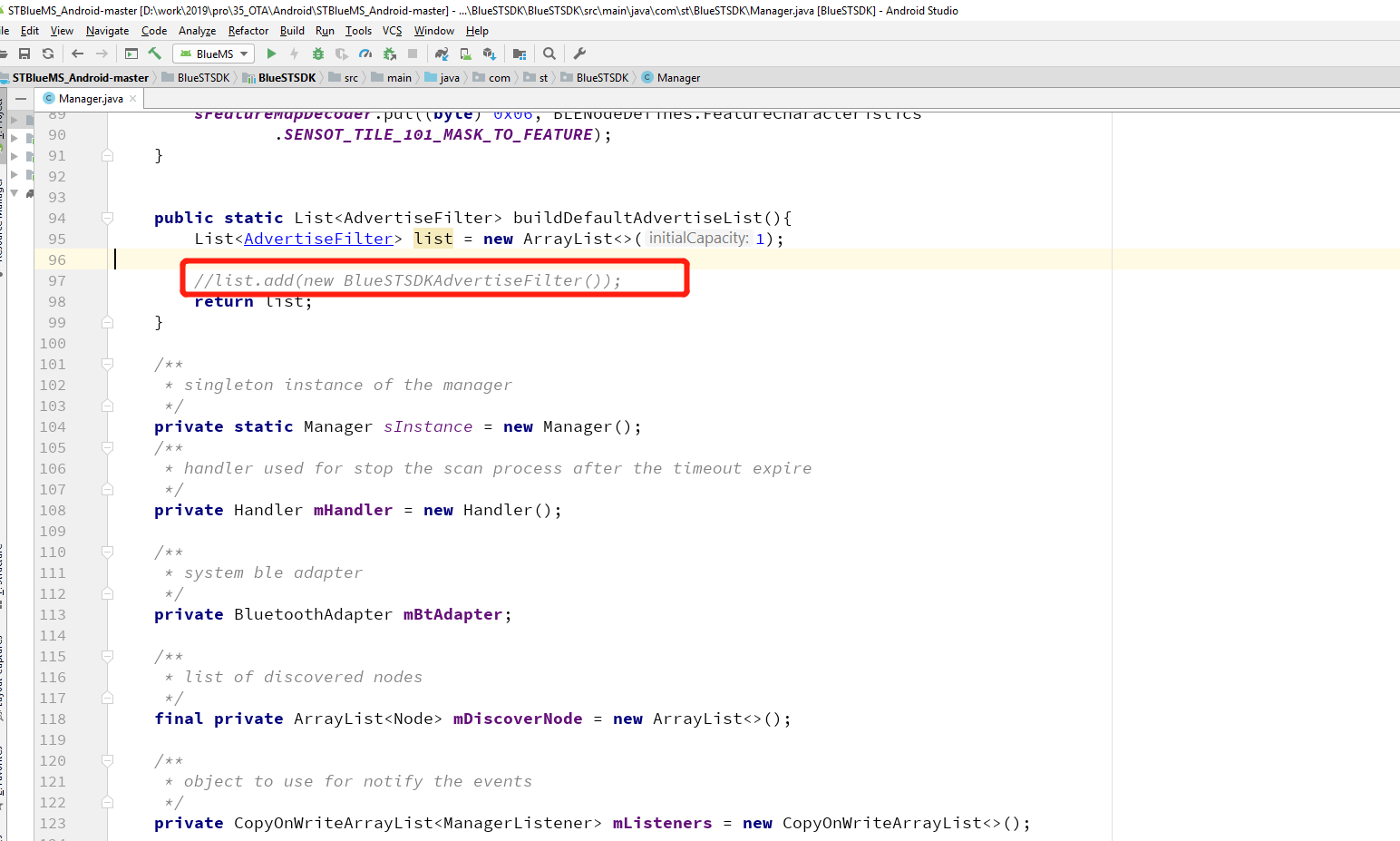




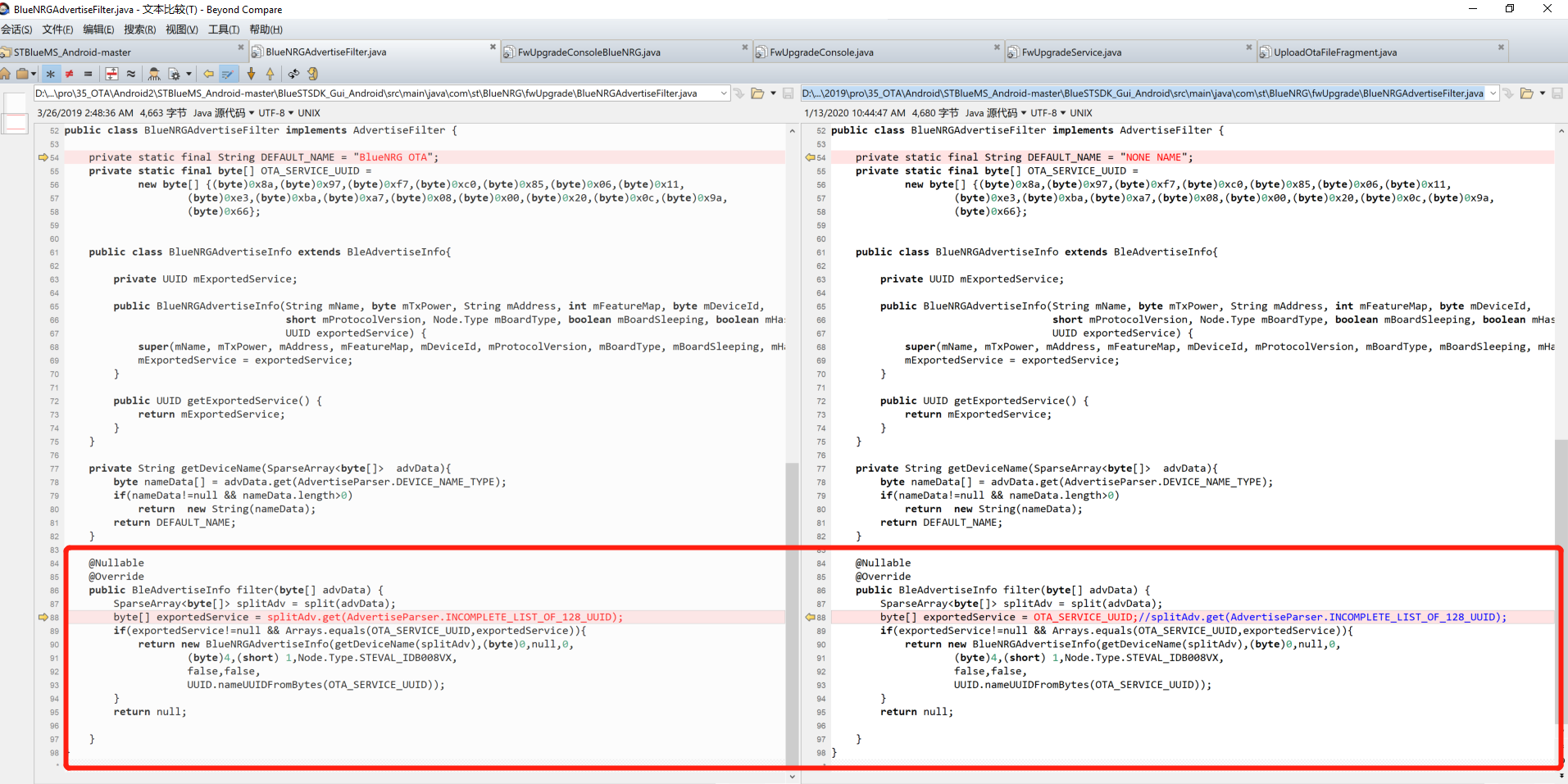
Android 端超时时间设置，建议可以改成

**private static final int *FW\_UPLOAD\_MSG\_TIMEOUT\_MS*** = 2000; *//8 msec instead of 7.5*

Android端程序默认按照广播数据识别不同的板子，如果设备端广播数据改变，有可能导致无法显示到扫描到的设备列表，因为Android端代码默认有个过滤机制，将广播数据不符合默认格式的设备给过滤了。用户可以根据应用需求自行更改，如果不设置过滤，可以屏蔽此处的代码（这里主要是过滤不同型号的ST的板子）。



具体这里则是针对BlueNRG具体应用数据的过滤。修改右侧，则不检测广播数据是否包含相关的UUID。



## BlueNRG-1/2 OTA修改的源码和修改后的apk如下:





如果使用此文档有问题，[可以发送邮件到 lucien.kuang@st.com](mailto:可以发送邮件到%20%20lucien.kuang@st.com)

# **Revision history**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Date** | **Author** | **Type** | **Device** | **Version** | **Changes** |
| 01-13-2020 | Lucien KUANG | IOS Android  Firmware | BlueNRG-1/2 | 1.3 | Initial release |