

FAQ_ 关于 BlueNRG-1/2 OTA 的流程与优化

0 Keywords

OTA, 升级,

1 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 low 和 high 对调升，会导致升成砖头等问题。

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。

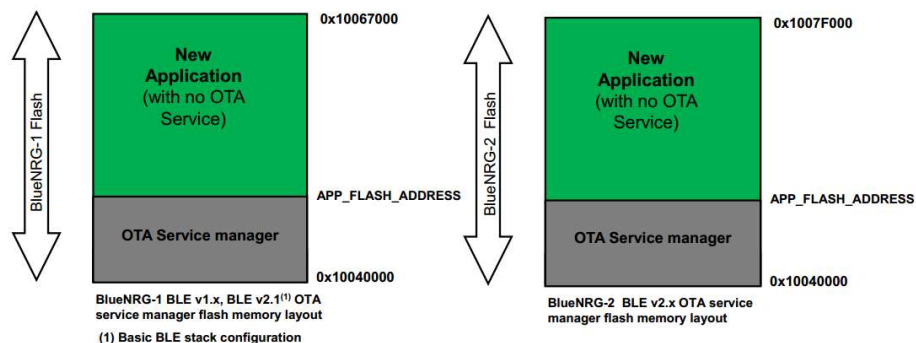
优化后固定协议栈仿方式大概可在 5s 左右完成升级过程，实际试手机和升级固件大小而定；非固定协议栈大概在 28s 左右完成升级。

1.1 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。

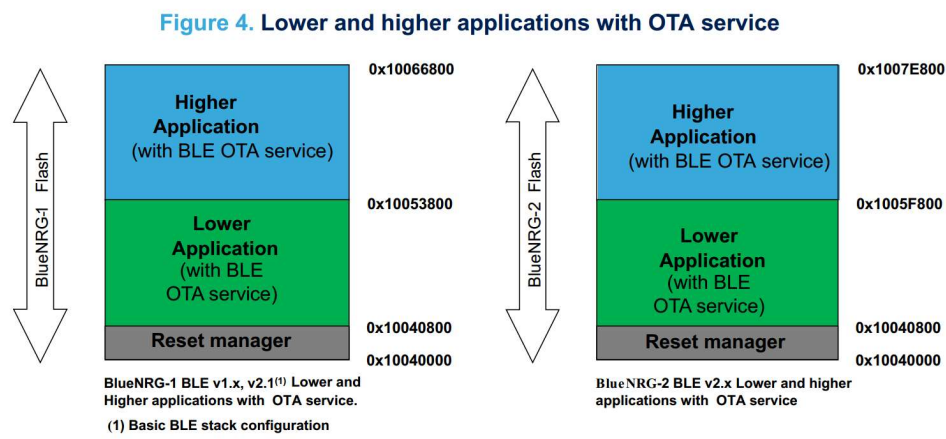
一种是 OTA Reset Manager (BootLoader) + APP(内包好 OTA 升级服务),

Figure 5. OTA service manager Flash memory layout



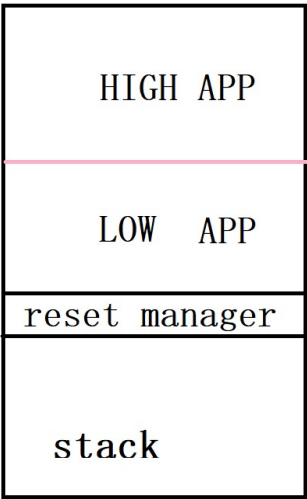
这种升级方式是将 OTA 服务放在 BootLoader 那。OTA Reset Manager 与 APP 都包含协议栈，协议栈可以被升级。当需要 OTA 升级时，需要跳转到执行 BootLoader 程序。

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。

还有一种是固定协议栈的方式



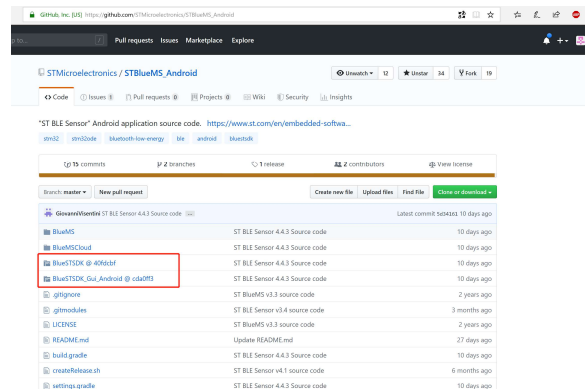
这种方式是将 OTA 服务放在 APP 那，但是区别与上一种是，这种协议栈是采用固定的方式。OTA 升级时不升级协议栈。

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。

1.2 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。

1.3 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。

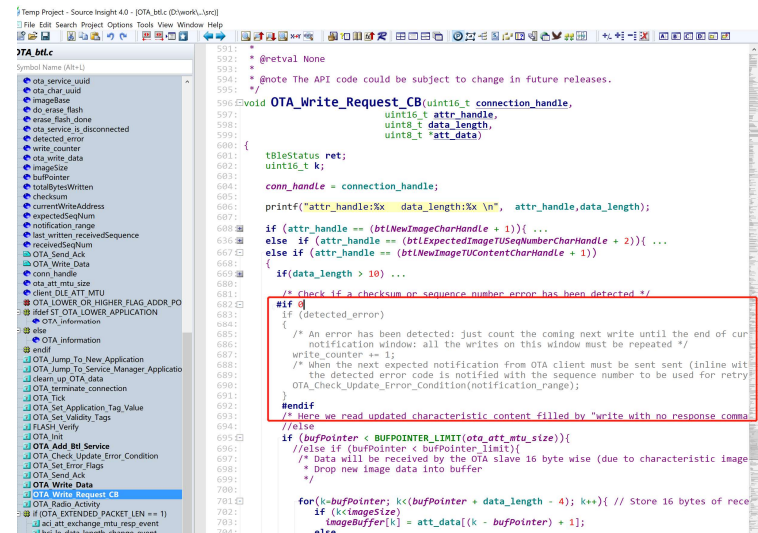


这里修改主要是因为 BlueNRG-1/2 OTA 设计时和 Android 端应用接口的差异。Android 或者 IOS 应用接口只要支持长包默认就是长包，并不会有一条单独的指令设置长包发送。

2. 修改支持长包后，会发现有部分手机的 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。

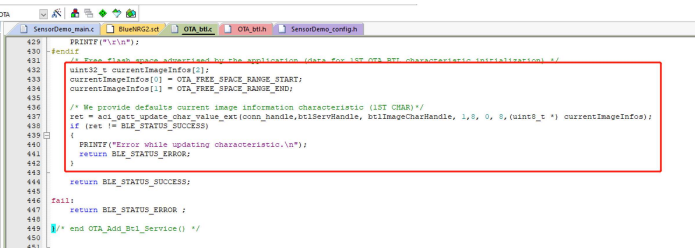
1.4 BlueNRG-1/2 OTA 的流程:

a. 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

b. OTA 升级服务和 Android APP 交互过程简单介绍

0. loadFw	Android app	device
1. MTU_REQUEST		aci_att_exchange_mtu_resp_event aci_l2cap_connection_parameter_update_req
	onMtuChange: 220	
2. READ_BLUENRG_SERVER_TYPE	addFeatureListener(onNewImageTUContentFeature); readFeature(mChunkData); onNewImageTUContentFeature.onUpdate	
3. RANGE_FLASH_MEM	addFeatureListener(onImageFeature); readFeature(mRangeMem); onImageFeature.onUpdate	
4. PARAM_FLASH_MEM	addFeatureListener(onNewImageFeature); writeParamMem(OTA_ACK_EVERY, cntExtended, \base_address, onWriteParamFlashMemDone)	OTA_Write_Request_CB(btlNewImageCharHandle + 1)
	onCharacteristicWrite onWriteParamFlashMemDone READ_PARAM_FLASH_MEM readFeature(mParamMem) onNewImageFeature.onUpdate	
5. START_ACK_NOTIFICATION	addFeatureListener(onAckNotification) enableNotification(mStartAckNotification)	OTA_Write_Request_CB(btlExpectedImageTUSeqNumberCharHandle + 2) aci_gatt_update_char_value_ext(btlExpectedImageTUSeqNumberCharHandle)
	onAckNotification.onUpdate	
6. FIRST_RECEIVED_NOTIFICATION	copy firmware data to ram	
7. WRITE_CHUNK_DATA	upload	OTA_Write_Request_CB(btlNewImageTUContentCharHandle)

8. WRITE_CHUNK_DATA	upload	OTA_Write_Request_CB(btlNewImageTUContentCharHandle)
9. finished		Restart

1.5 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 low bin 无法放在 high bin 的 Flash 位置上运行）。

为了解决这个问题，下面方法供参考：

- 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。
- 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）：

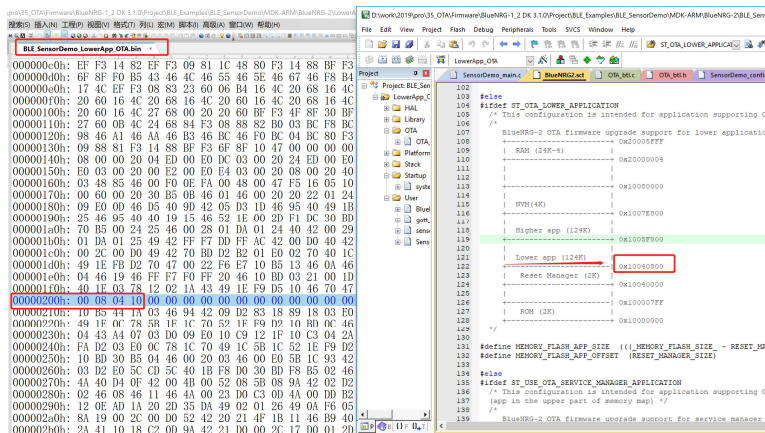
```
static uint32_t const OTA_information[4] @ (((APP_LOWER_ADDRESS+OTA_LOWER_OR_HIGHER_FLAG_ADDR_POS))) = {APP_LOWER_ADDRESS};
```

```

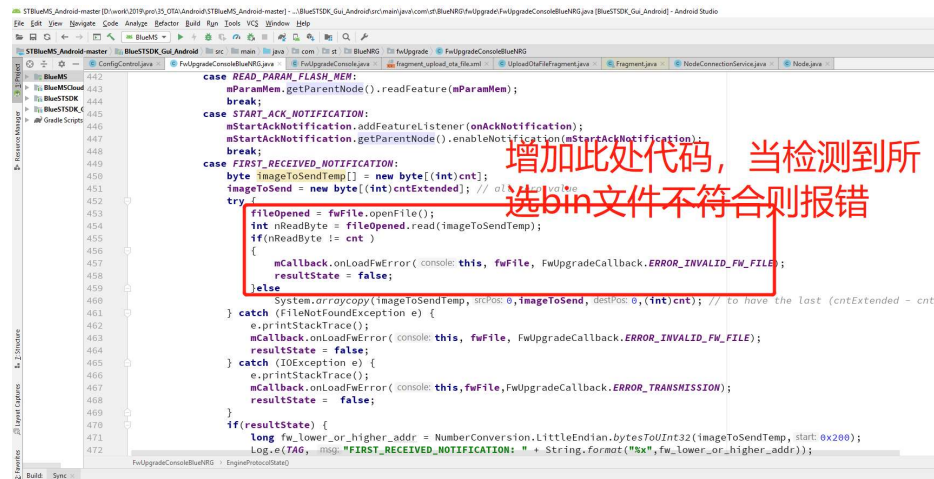
177: static void OTA_Send_Ack(void);
178:
179: static void OTA_Write_Data(void);
180:
181: static uint16_t com_handle;
182:
183: /* Actual OTA mtu size agreed with OTA client: default is OTA_ATT_MTU_SIZE
184:    if OTA client doesn't support extended data length */
185: static uint16_t ota_att_mtu_size = OTA_ATT_MTU_SIZE;
186:
187: /* Flag for register if OTA client has performed BLE and ATT_MTU exchange config */
188: static volatile uint8_t client_ble_att_mtu = 0;
189:
190:
191: #define OTA_LOWER_OR_HIGHER_FLAG_ADDR_POS (0x200)
192: // Put information of APP_LOWER_ADDRESS or APP_HIGHER_ADDRESS to bin firmware at 0x200 for the APP can avoid user upgrade a wrong bin file
193: #ifdef ST_OTA_LOWER_APPLICATION
194: static uint32_t const OTA_information[4] __attribute__((at(APP_LOWER_ADDRESS+OTA_LOWER_OR_HIGHER_FLAG_ADDR_POS))) = {APP_LOWER_ADDRESS};
195: #else
196: static uint32_t const OTA_information[4] __attribute__((at(APP_HIGHER_ADDRESS+OTA_LOWER_OR_HIGHER_FLAG_ADDR_POS))) = {APP_HIGHER_ADDRESS};
197: #endif
198:
199:
200: /**
201:  * @brief It jumps to the new upgraded application
202:  * @param None
203:  * @retval None
204:  */
205:
206: /* Note The API code could be subject to change in future releases.
207:  *
208:  * .. OTA Jump To New Application ..

```

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。



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`

1.6 BlueNRG-1/2 OTA 修改的源码和修改后的 apk 如下:



BlueMS-debug_v
0_8.apk

2 Revision history

Date	Author	Type	Device	Version	Changes
03-07-2019	Lucien KUANG	IOS Android Firmware	BlueNRG-1/2	1.1	Initial release