

GOKIT_OTA 教程

机智云

编制人	TerryLi	审核人		批准人	
产品名称		产品型号		文档编号	
会签日期			版本		

GizWits

修改记录:

修改时间	修改记录	修改人	版本	备注
20160510	初建	TerryLi	1.0.0	
20160530	Bootloader 区拷贝固件后增加 APP 区域新固件的 MD5 校验	TerryLi	1.1.0	
20160907	修改部分图示	TerryLi	1.2.0	

目录

GOKIT_OTA 教程.....	1
简述 STM32 启动.....	4
FLASH 区间划分.....	4
GOKIT_OTA 方案.....	6
开始 Bootloader.....	7
Bootloader 程序流程.....	7
Bootloader 编译设置.....	7
开始写 APP 关于固件升级部分.....	9
固件接收流程.....	9
编译器设置.....	10
运行日志.....	11
APP 接收云端固件数据.....	11
Bootloader 执行升级任务.....	12
云端操作.....	13
添加新固件.....	13
下载固件.....	14
Keil 生成 bin 文件.....	16
合并 hex.....	18

简述 STM32 启动

ARM7/ARM9 内核的控制器在复位后，CPU 会从存储空间的绝对地址 0x000000 取出第一条指令执行复位中断服务程序的方式启动，即固定了复位后的起始地址为 0x000000（PC = 0x000000）同时中断向量表的位置并不是固定的。然而，Cortex-M3 内核启动有 3 种情况：

- 1、通过 boot 引脚设置可以将中断向量表定位于 SRAM 区，即起始地址为 0x2000000，同时复位后 PC 指针位于 0x2000000 处；
- 2、通过 boot 引脚设置可以将中断向量表定位于 FLASH 区，即起始地址为 0x8000000，同时复位后 PC 指针位于 0x8000000 处；
- 3、通过 boot 引脚设置可以将中断向量表定位于内置 Bootloader 区；

Cortex-M3 内核规定，起始地址必须存放堆顶指针，而第二个地址则必须存放复位中断入口向量地址，这样在 Cortex-M3 内核复位后，会自动从起始地址的下一个 32 位空间取出复位中断入口向量，跳转执行复位中断服务程序。对比 ARM7/ARM9 内核，Cortex-M3 内核则是固定了中断向量表的位置而起始地址是可变化的。

总结一下 STM32 的启动文件和启动过程。首先对栈和堆的大小进行定义，并在代码区的起始处建立中断向量表，其第一个表项是栈顶地址，第二个表项是复位中断服务入口地址。然后在复位中断服务程序中跳转 C/C++ 标准实时库的 main 函数，完成用户堆栈等的初始化后，跳转.c 文件中的 main 函数开始执行 C 程序。假设 STM32 被设置为从内部 FLASH 启动（这也是最常见的一种情况），中断向量表起始地址为 0x8000000，则栈顶地址存放于 0x8000000 处，而复位中断服务入口地址存放于 0x8000004 处。当 STM32 遇到复位信号后，则从 0x8000004 处取出复位中断服务入口地址，继而执行复位中断服务程序，然后跳转 main 函数，最后进入 mian 函数。

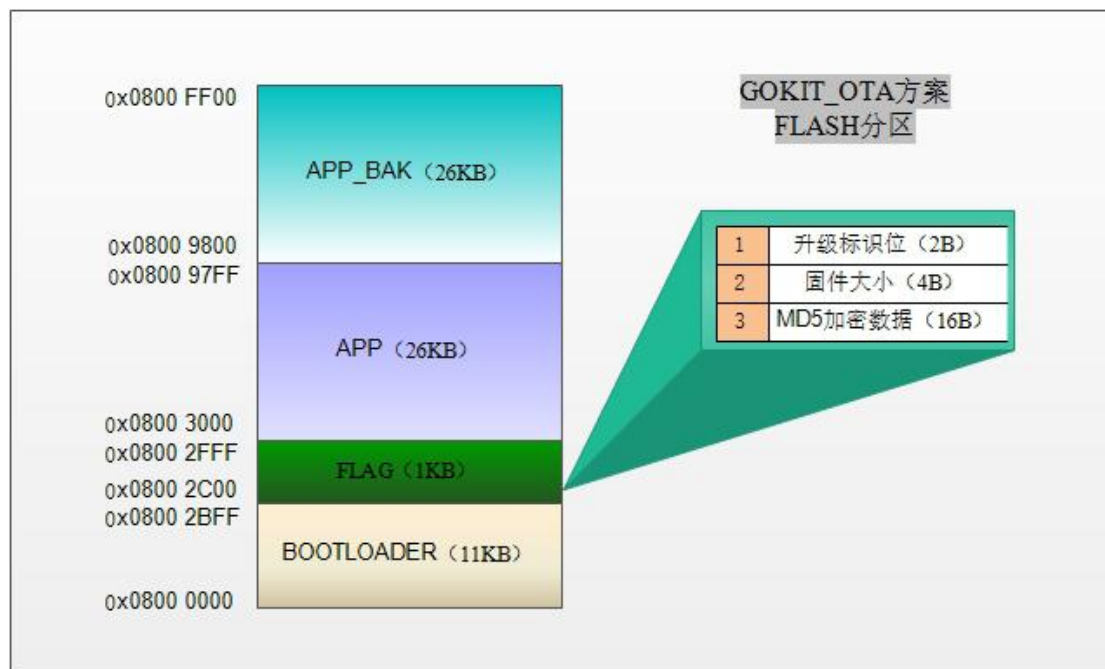
分析下 GOKIT_OTA 需求，我们将建立两个工程，分别是 Bootloader 还有 APP，我们将 Bootloader 下载到 FLASH 空间 0x8000000 地址处，那么 STM32 启动后会首先执行我们的 Bootloader 程序，然后就可以按照我们意愿实现 OTA 了。

FLASH 区间划分

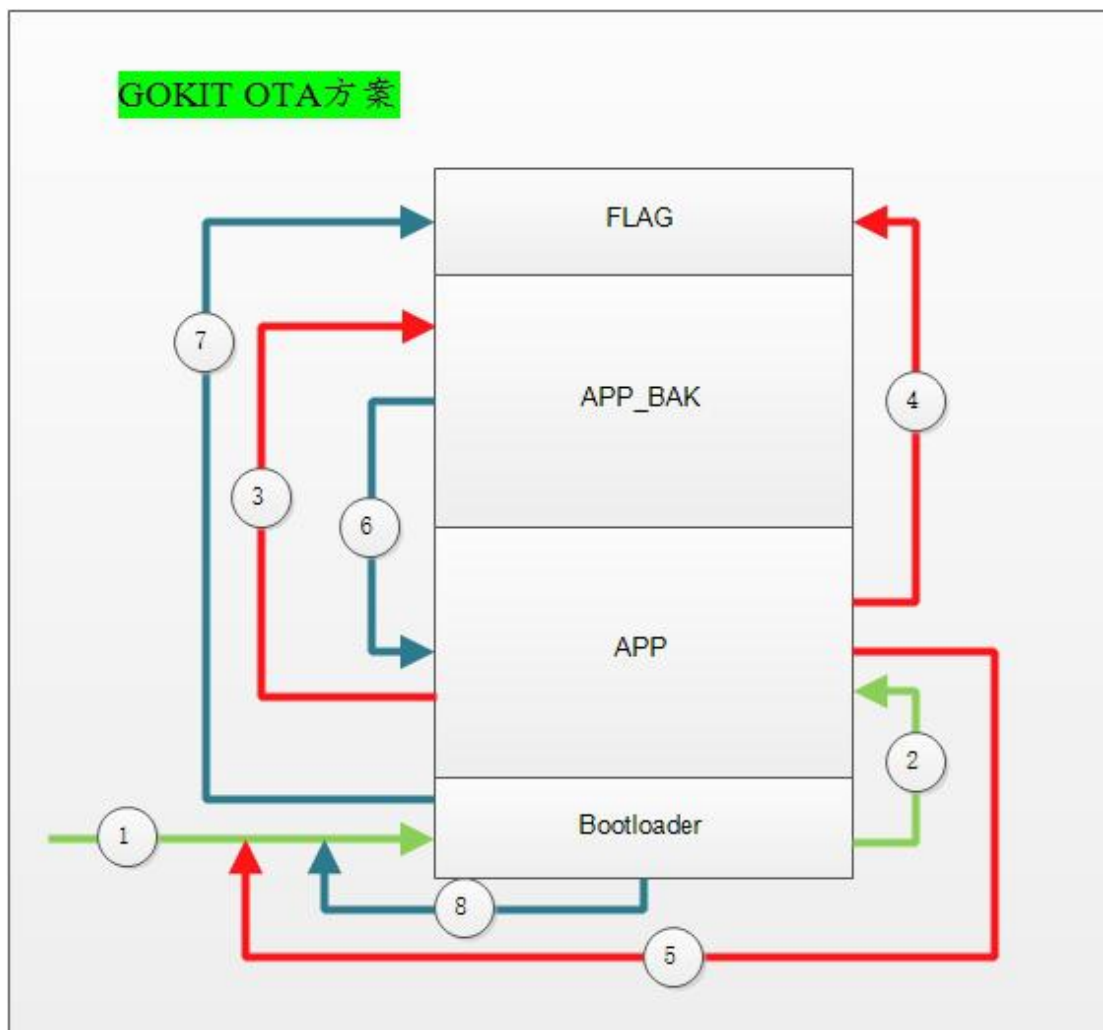
根据需求，我们将 STM32F103C8T6 这个芯片（GOKIT2 代）Flash 空间划分出 4 个区域：Bootloader、FLAG、APP、APBBAK。四个区间作用描述如下：

- Bootloader: 存储 Bootloader 固件，MCU 上电后首先运行该固件。
- FLAG: 存储有关升级的相关标志位，Bootloader 和 APP 都需要操作该区域。
- APP: 存储用户程序固件。
- APBBAK: 临时存储云端下发的新固件，升级固件的一个过渡存储区。

Gokit 分区方案如下图所示：



GOKIT_OTA 方案



正常启动流程：

1、上电进入 Bootloader 区域运行, 检测 FLAG 区域标识选择是否需要升级, 若无升级任务, 则运行 2

2、跳转到 APP 区域运行应用程序

有升级任务执行流程

3、APP 区域运行应用程序时接收到模组升级命令, 接收固件分片数据, 写数据到 APP_BAK 区域, 接收完成, 执行 4

4、写入 FLAG 区域升级标识, 并写入 MD5 加密数据, 执行 5

5、MCU 重启, 开始执行 1

1、检测到 FLAG 区域有升级任务, 读出 APP_BAK 区域数据, 验证固件有效性, 若固件有效, 执行 6

6、读出 APP_BAK 区域数据, 写入 APP 区域, 检验新固件 MD5, 若校验成功则执行 7, 若校验失败, 则 MCU 重启

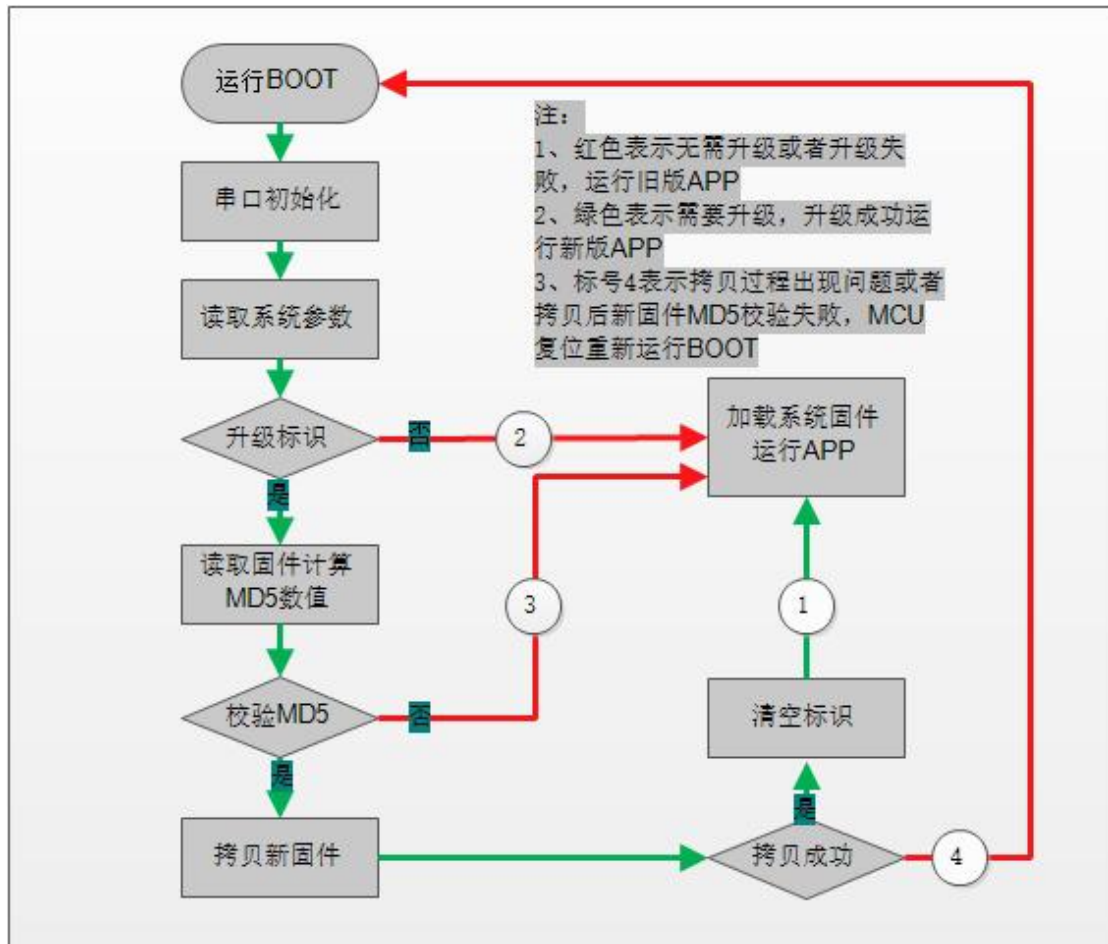
7、擦除 FLAG 区域有效升级标志, 执行 8

8、MCU 重启, 进入 Bootloader 区域, 未检测到升级任务, 执行 2

开始 Bootloader

Bootloader 程序流程

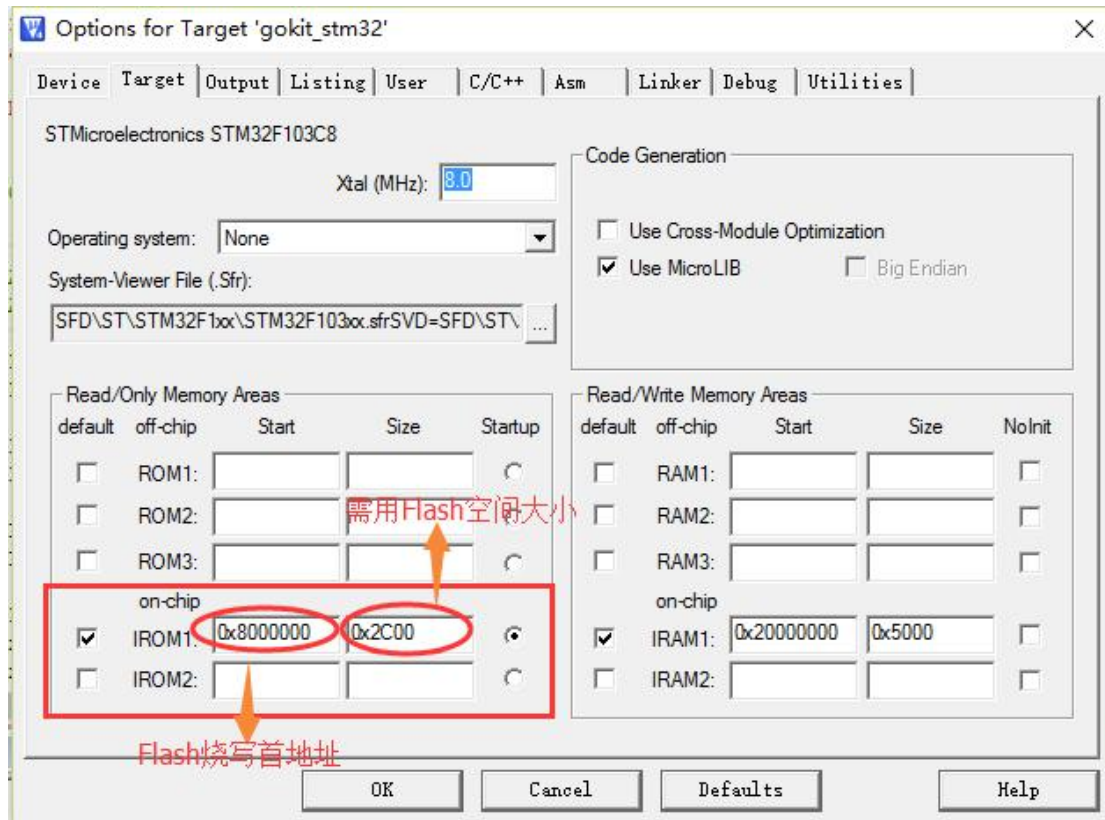
Bootloader 的主要职能是在有升级任务的时候将 APPBAK 里面的固件拷贝到 APP 区域。当然，这期间需要做很多的工作，比如升级失败的容错等等。具体的流程可以参考图示。**需要注意的是，在校验 MD5 正确后开始搬运固件数据期间，MCU 出现故障（包括突然断电），MCU 应发生复位操作（FLAG 区域数据未破坏），复位后重新开始执行 Bootloader，从而避免 MCU 刷成板砖。**



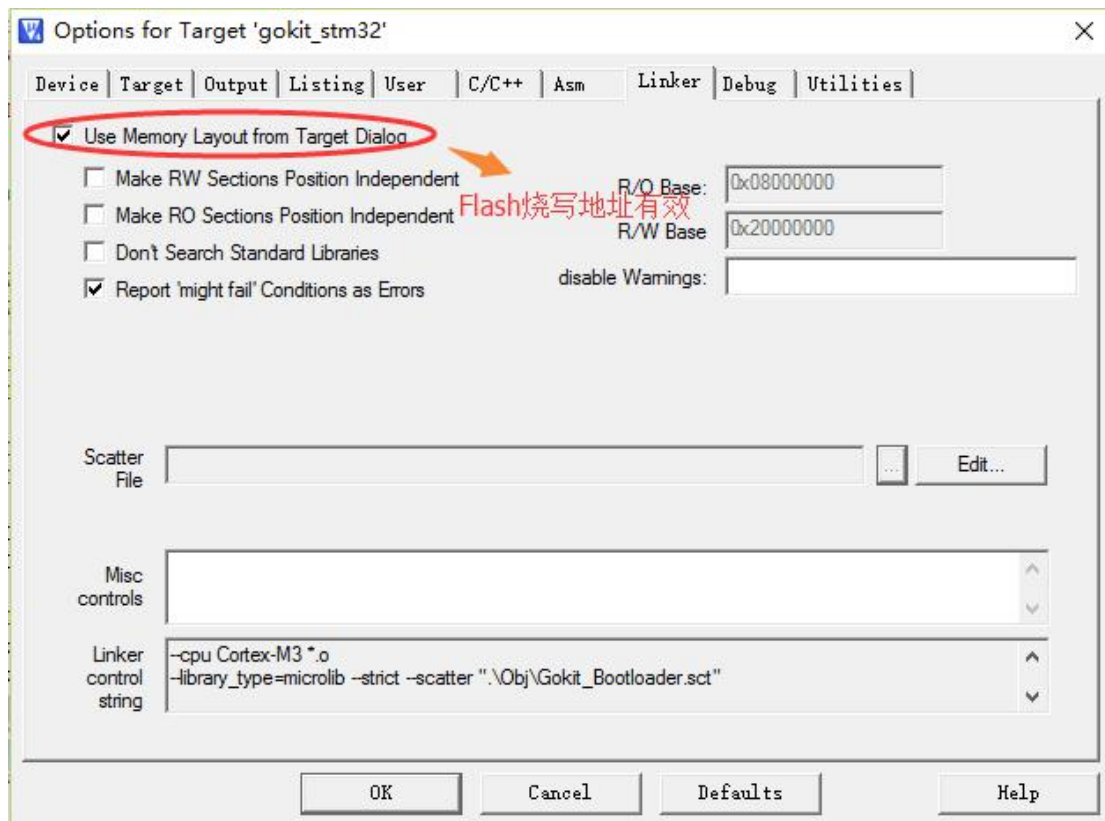
Bootloader 编译设置

按照 Bootloader 流程编写好代码，需要我们对 KEIL 工程做相应配置，需要注意的是编译的 Bootloader 固件大小不超过最大可允许的 11KB。Keil 编译器需要设置如下：

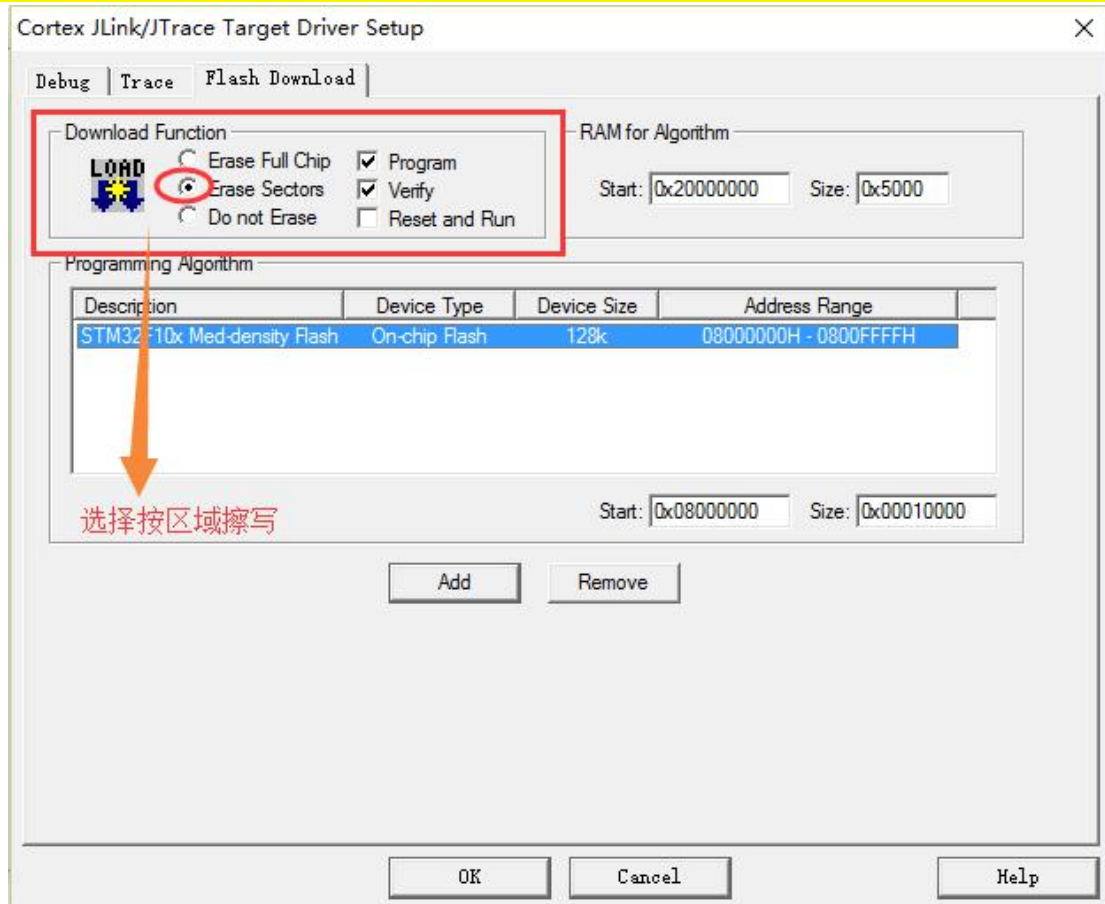
1、按照 FLASH 分区方案，设置 FLASH 固件下载地址，如下图所示：



2、Flash 烧写地址设置生效



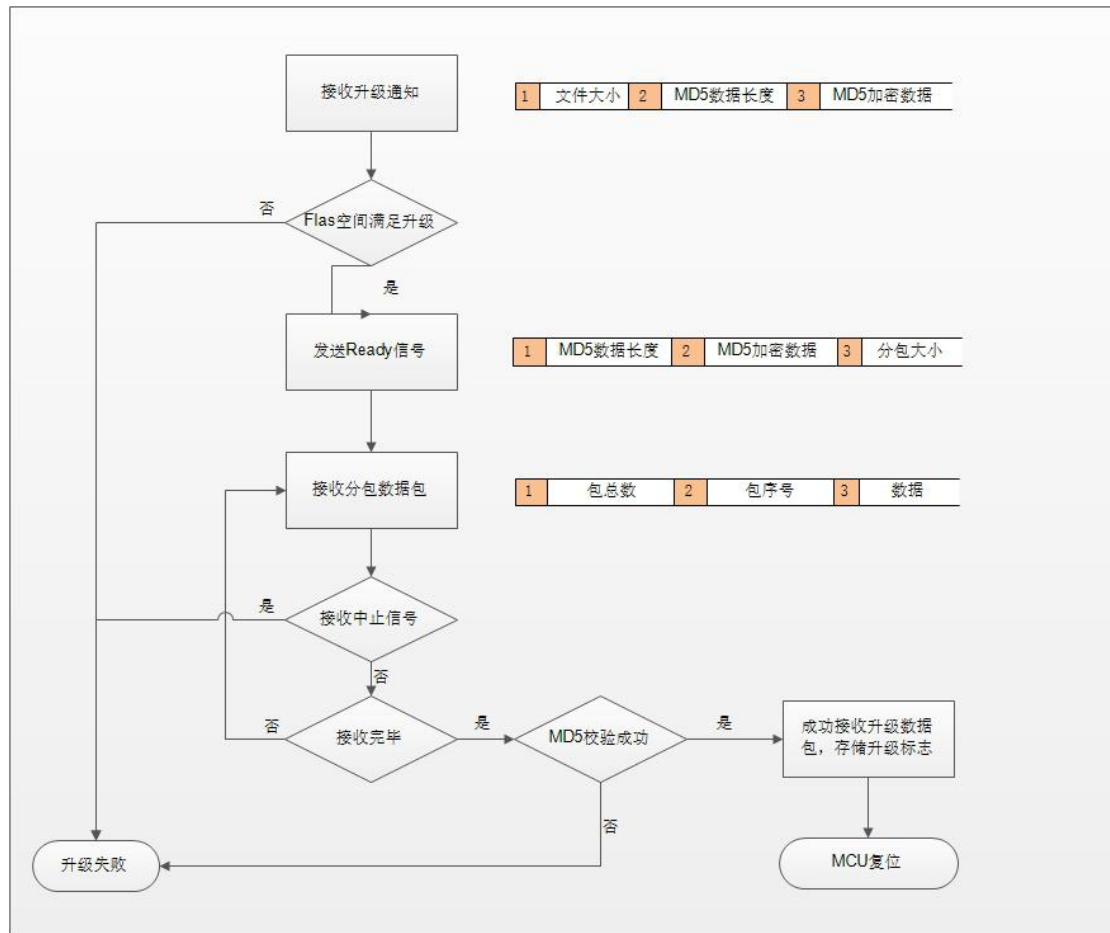
3、JLINK 下载按钮擦除 FLASH 区间



开始写 APP 关于固件升级部分

固件接收流程

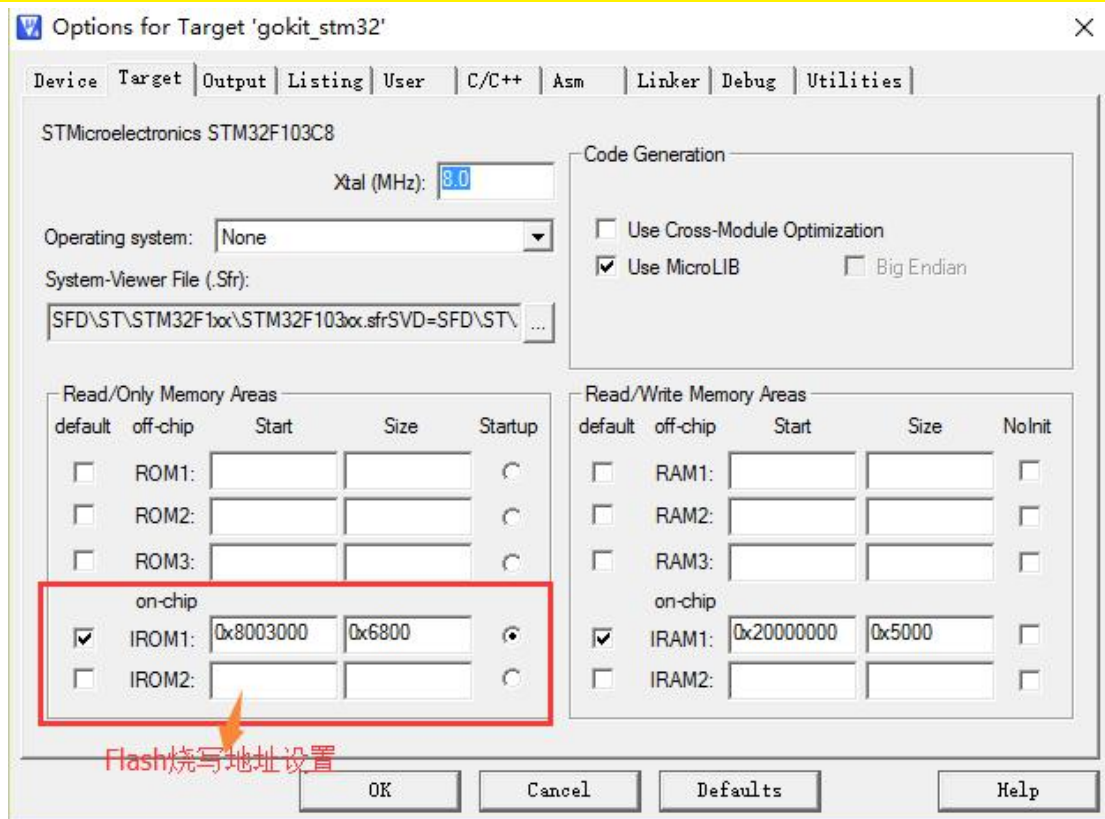
做好 BOOTLOADER 工作后，我们开始写 APP 的代码。APP 固件的编写要注意硬件版本号和软件版本号，软件版本号作为升级迭代很重要的标志。APP 代码我们只需要在 GOKIT 微信宠物屋代码基础上增加大数据接受即接受云端新固件功能即可。**需要注意的是，中断向量地址偏移的定义，这个地方需要我们尤其注意，我在开发过程中在这个地方排查了好长时间。**STM32 标准库默认中断向量地址偏移为 0x0，但是我们 APP 实际的偏移是 0x3000。如果不修改，APP 也可以正常加载运行，但是不会相应中断。所以，我们需要根据实际 APP 下载的起始地址，对中断向量地址偏移做定义。按照协议规定，我们去实现大数据整个流程，具体如下：



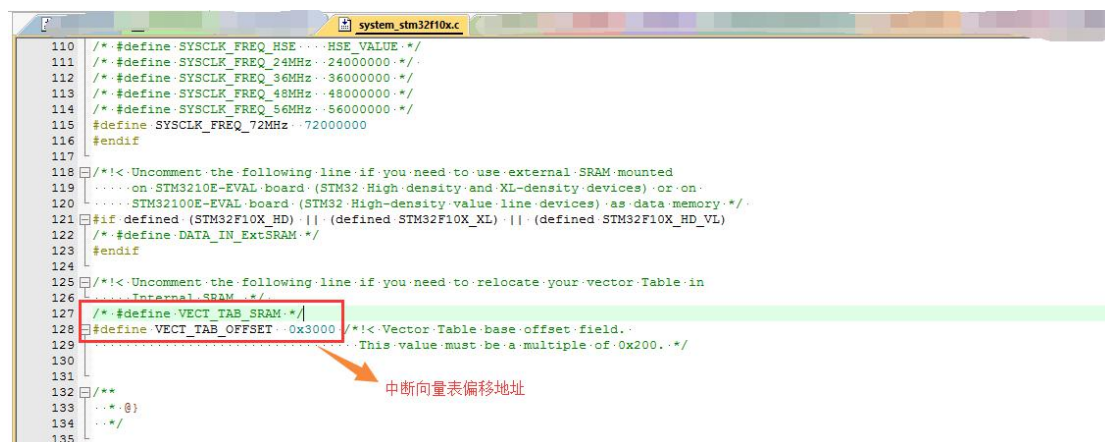
编译器设置

同样，因为硬件 FLASH 空间限定，我们需要对 APP 的固件大小做严格的限制。本方案，针对 GOKIT 我们可允许的最大固件为 26KB。需要升级的新固件同样最大可支持 26KB。

1、设置 FLASH 固件下载地址



2、中断向量偏移地址设置



运行日志

APP 接收云端固件数据

1、模组请求发送大数据即云端固件，如下图所示：

SAgentToMCU: ff ff 0 2b 19 bf 0 0 0 0 5b 10 0 20 63 61 31 63 36 64 38 32 30 63 33 30 39 65 35 66 66 34 62 31 33 61 36 33 65 35 32 35 34 61 37 62 72
 [181230] MCU : ff ff 0 5 1a bf 0 0 de
 updateFileSize = 23312
 FileMD5Len = 32 MD5: 63 61 31 63 36 64 38 32 30 63 33 30 39 65 35 66 66 34 62 31 33 61 36 33 65 35 32 35 34 61 37 62
 MD5_Hex: ca 1c 6d 82 0c 30 9e 5f f4 b1 3a 63 e5 25 4a 7b
 MCU Ready
 SAgentToMCU: ff ff 0 5 1c bf 0 0 e0
 MCU : ff ff 0 6 12 bf 0 0 2 d9

模组发起升级请求,MCU解析出固件大小和MD5, 为升级做预准备
 MCU准备就绪, 下一步准备接收数据分片

2、MCU 接收大数据分片，累计计算 MD5

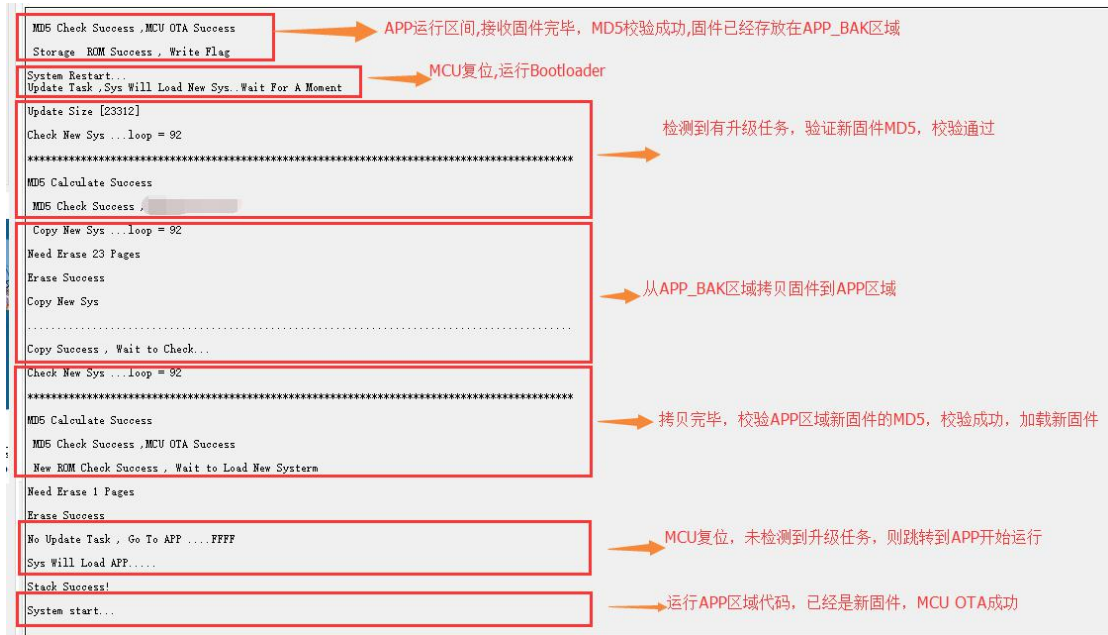
[illegible]

3、MCU 接收大数据完毕，验证 MD5,存储 FLAG，MCU 复位

[illegible]

Bootloder 执行升级任务

- 1、Bootloader 检测到升级任务，开始执行升级
- 2、第一遍读出固件，计算 MD5，验证固件正确性
- 3、MD5 校验成功，从 APPBAK 拷贝数据到 APP
- 4、拷贝完毕，校验 APP 区域新固件 MD5，校验成功则清空 FLAG 标志，跳转到 APP 执行新固件，校验失败则重启 MCU



Log content and annotations:

- MD5 Check Success ,MCU OTA Success
Storage ROM Success , Write Flag → APP运行区间,接收固件完毕, MD5校验成功,固件已经存放在APP_BAK区域
- System Restart...
Update Task ,Sys Will Load New Sys. Wait For A Moment → MCU复位,运行Bootloader
- Update Size [23312]
Check New Sys ...loop = 92

MD5 Calculate Success
MD5 Check Success , → 检测到有升级任务, 验证新固件MD5, 校验通过
- Copy New Sys ...loop = 92
Need Erase 23 Pages
Erase Success
Copy New Sys
Copy Success , Wait to Check... → 从APP_BAK区域拷贝固件到APP区域
- Check New Sys ...loop = 92

MD5 Calculate Success
MD5 Check Success ,MCU OTA Success
New ROM Check Success , Wait to Load New System → 拷贝完毕, 校验APP区域新固件的MD5, 校验成功, 加载新固件
- Need Erase 1 Pages
Erase Success
No Update Task , Go To APPFFFF → MCU复位, 未检测到升级任务, 则跳转到APP开始运行
- Sys Will Load APP.....
Stack Success!
System start... → 运行APP区域代码, 已经是新固件, MCU OTA成功

5、升级成功，通过日志查看软件版本号



Log content and annotations:

- No Update Task , Go To APPFFFF → 固件验证并拷贝完毕,跳转到APP
- Sys Will Load APP.....
Stack Success!
System start...
SYSCLKSource is PL!
SYS clock =72MHz
HCLK clock =72MHz
PCLK1 clock =36MHz
PCLK2 clock =72MHz
SADCLK_Frequencyclock =36MHz
Gokit Init Ok
Gokit Start Success ,SoftVersion = 02030002 → 验证软件版本号, 升级成功
- [2044] MCU : ff ff 0 11 5 0 0 0 4 0 0 0 0 5 0 2a 1f 0 0 68
GAgentToMCU: ff ff 0 5 6 0 0 0 b

云端操作

添加新固件

1、创建新固件



Socket control interface details:

- Header: Socket control | 申请发布 | 技术支持: 400-6525-488 | 800099639 | 意见反馈
- Left Menu: 产品管理 (产品信息, 数据点, 虚拟设备, 运行状态, 产品开发资源), 服务 (固件升级(OTA), 产测工具)
- Main Content: 固件列表
- Buttons: + 创建新固件 (highlighted)
- Legend: 蓝色: 已验证固件. 红色: 未验证固件, 可能存在风险.
- Table:

版本名称	硬件版本	软件版本	固件类型	推送状态	创建时间
wxl...	02030101	02030007	mcu	未开始	2016-05-09 17:38
GQQ...	02030100	02030013	mcu	未开始	2016-04-28 16:51
GQQ...	02030100	02030003	mcu	未开始	2016-04-28 15:14
GOK...	02030100	02030008	mcu	未开始	2016-04-20 15:38

2、编辑新固件，软件版本号和固件保持一致，固件格式为 bin

版本名称: wx_test *

固件类型: MCU * → 选择MCU类型

推送方式: V4.1 * ?

选择固件: gokit_mcu_stm32.bin ? 重新上传 → 上传新固件,bin格式

硬件版本号: 02030101 ? → 硬件版本号

软件版本号: 02030007 ? → 软件版本号和固件保持一致

3、保存完成

下载固件

1、点击要升级的固件开始升级

Socket_control 申请发布 技术支持: 400-6525-488 800099639 [意见反馈](#)

产品管理
产品信息
数据点
虚拟设备
运行状态
产品开发资源

服务
固件升级 (OTA)
产测工具

固件列表 + 创建新固件

蓝色: 已验证固件, 红色: 未验证固件, 可能存在风险。

版本名称	硬件版本	软件版本	固件类型	推送状态	创建时间
wx_test	02030101	02030007	mcu	未开始	2016-05-09 17:38
GQ0...	02030100	02030013	mcu	未开始	2016-04-28 16:51
GQ0...	02030100	02030003	mcu	未开始	2016-04-28 15:14
GOK...	02030100	02030008	mcu	未开始	2016-04-20 15:38

机智云官网 | 关于我们 | 常见问题 | 服务协议

2、复查固件信息无误，点击验证固件

固件详细信息

版本名称: wx_test

固件类型: mcu

推送方式: v4.1

固件: gokit_mcu_stm32.bin [下载](#)

硬件版本号: 02030101

软件版本号: 02030007

创建时间: 2016-05-09 17:38

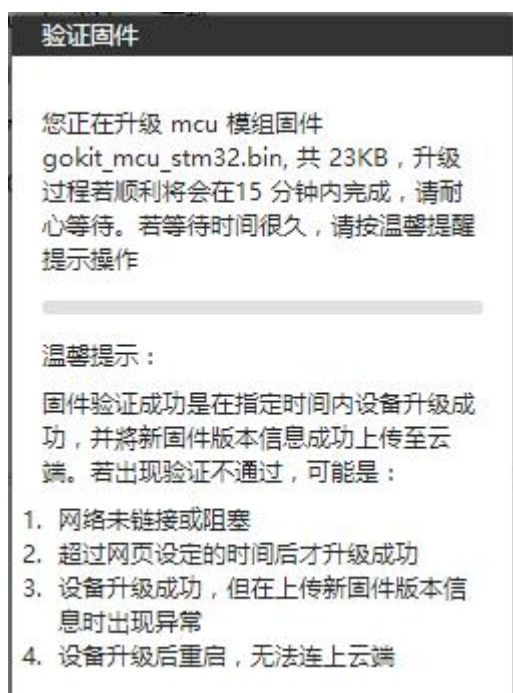
可以对固件做处理

[编辑固件](#) [删除固件](#)

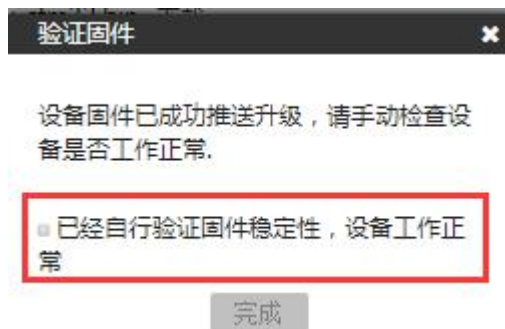
3、确保需要升级的设备处于联网状态 Z，添加模组 MAC



4、推送过程



5、固件升级完成，查看 MCU 运行日志，验证固件正确性



6、固件验证通过，添加规则，关于规则在此不赘述

固件列表 > 固件推送



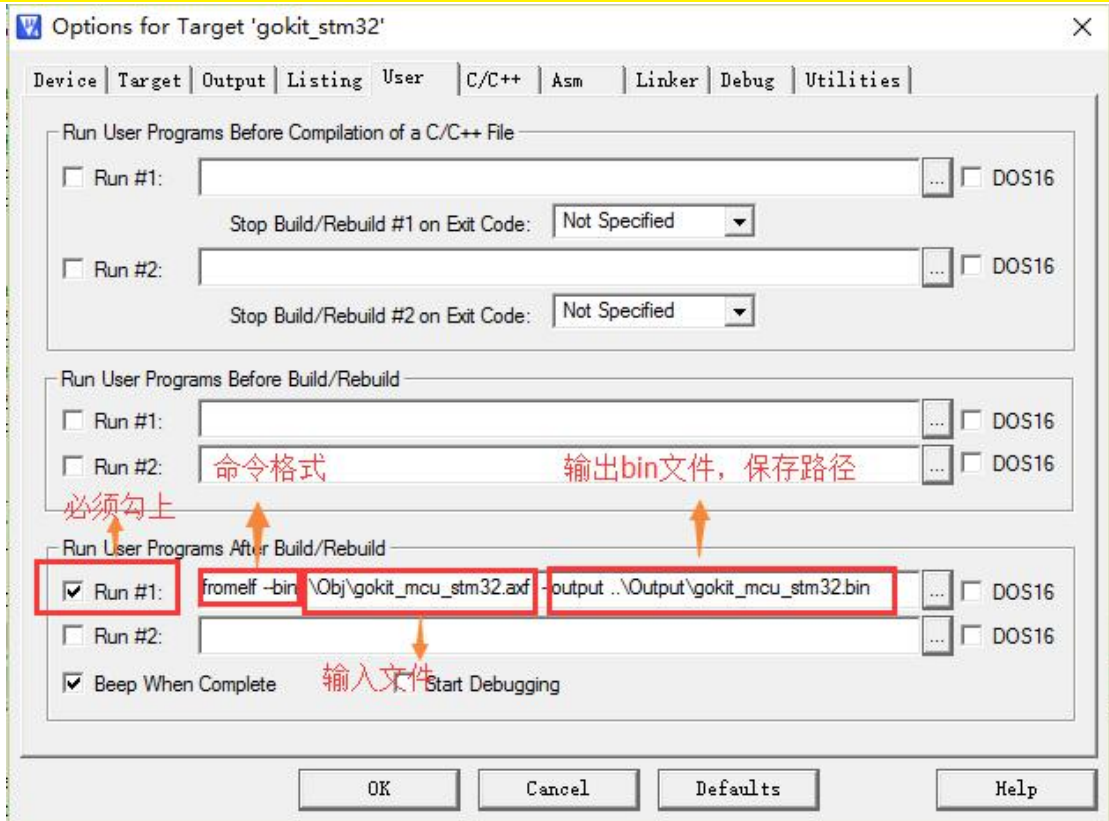
Keil 生成 bin 文件

本方案只测试了 bin 文件的远程升级, hex 文件未作研究, 所以我们需要通过 Keil 将我们的固件编译生成 bin 文件。Keil 自带了工具软件 fromelf.exe, 只要进行适当配置便可以输出 bin 文件了, 至于 fromelf 工具语法在此不在赘述, 可参考链接:

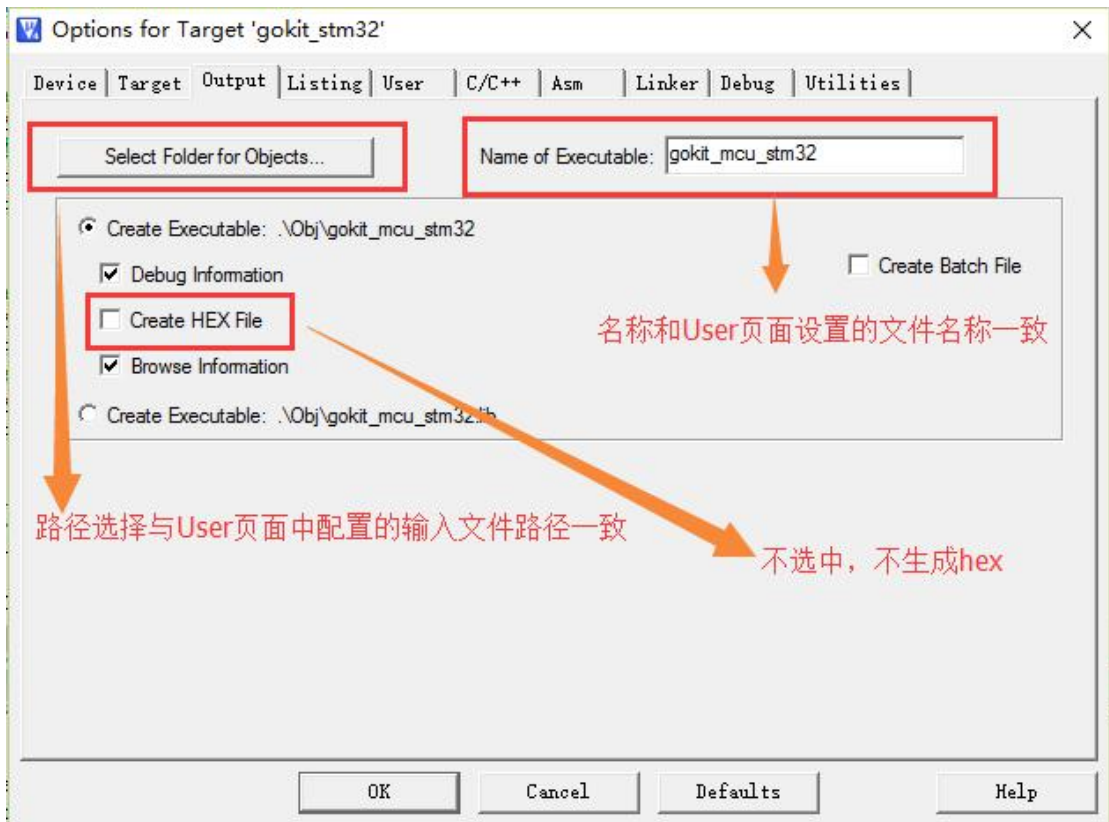
<http://forum.eepw.com.cn/thread/225710/1/>

下面以 Gokit 微信宠物屋工程为例, 编译后能生成 .bin 格式的文件:

1. 选择 User 标签页, 并进行如下图一样的配置:



2. 根据 User 页的配置还要配置 Output 页面，具体如下：



3. 点击 OK 确定，然后再重新编译则会按照上图中的配置路径生成 .bin 格式的文件了：

```
Build Output
Build target 'gokit_stm32'
linking...
Program Size: Code=22824 RO-data=268 RW-data=196 ZI-data=2876
User command #1: fromelf --bin .\Obj\gokit_mcu_stm32.axf --output ..\Output\gokit_mcu_stm32.bin
".\Obj\gokit_mcu_stm32.axf" - 0 Error(s), 0 Warning(s).
```

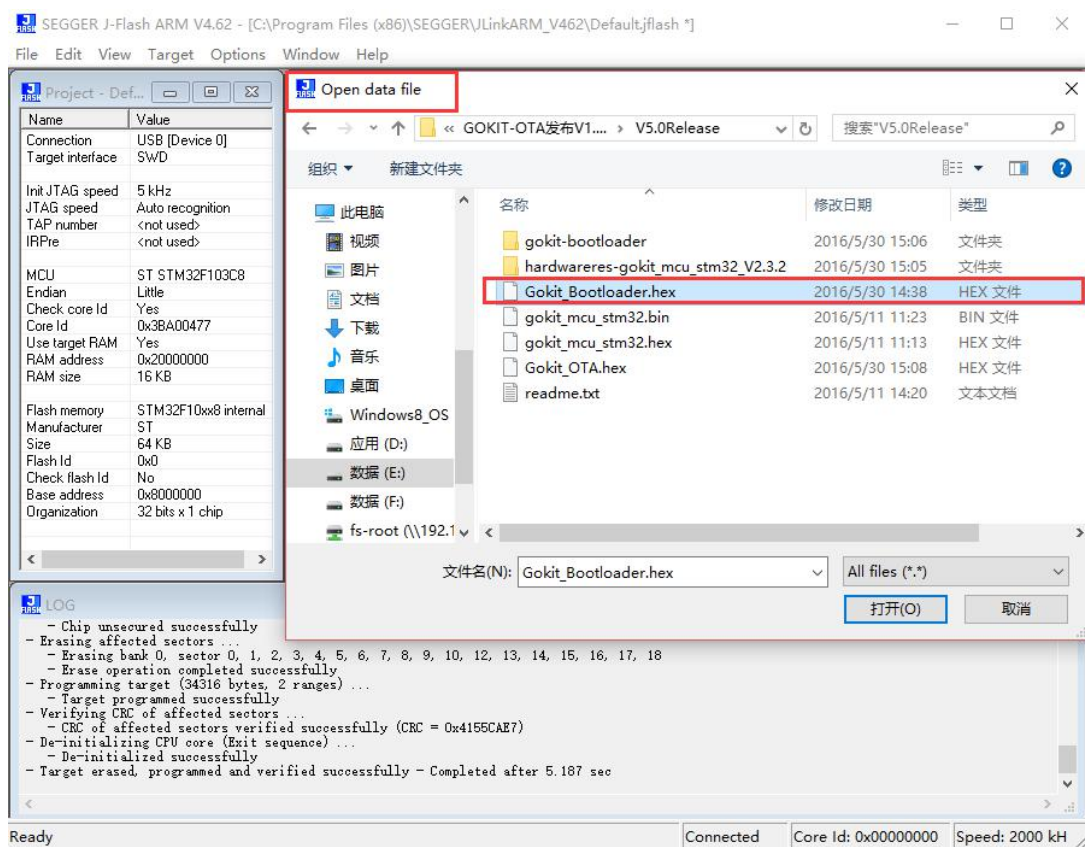
合并 hex

本方案会编译生成两个 hex 文件，但是为方便用户下载，在此提供 hex 合并的操作，按照如下操作，我们可以将 bootloader 和 app 两个固件合并成一个固件，注意是 hex 格式文件，这样，用户只需要一个下载一个 hex 文件，即可实现带有远程 OTA 功能的微信宠物工程实现。

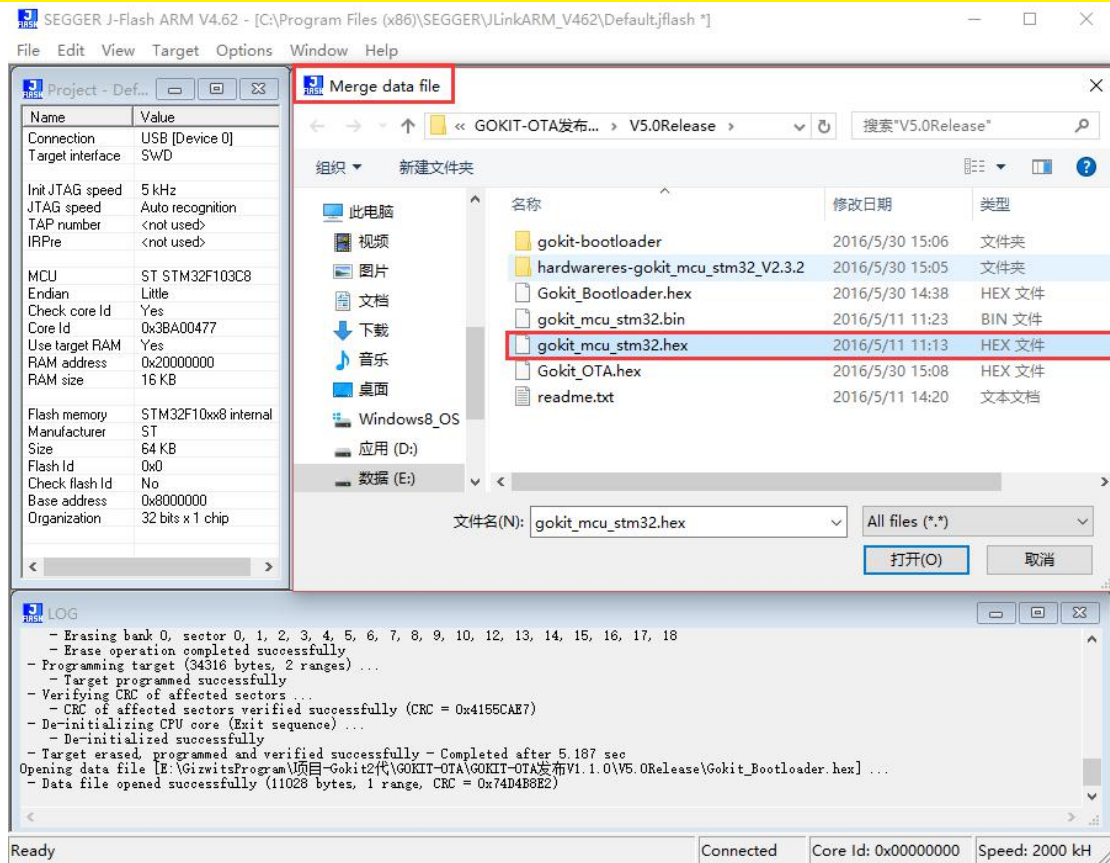
1、准备好需要合并的 hex 文件，如下：

 Gokit_Bootloader.hex	2016/5/9 19:00	HEX 文件	30 KB	注：hex大小 不等于固件实 际大小
 gokit_mcu_stm32.hex	2016/5/10 18:11	HEX 文件	65 KB	

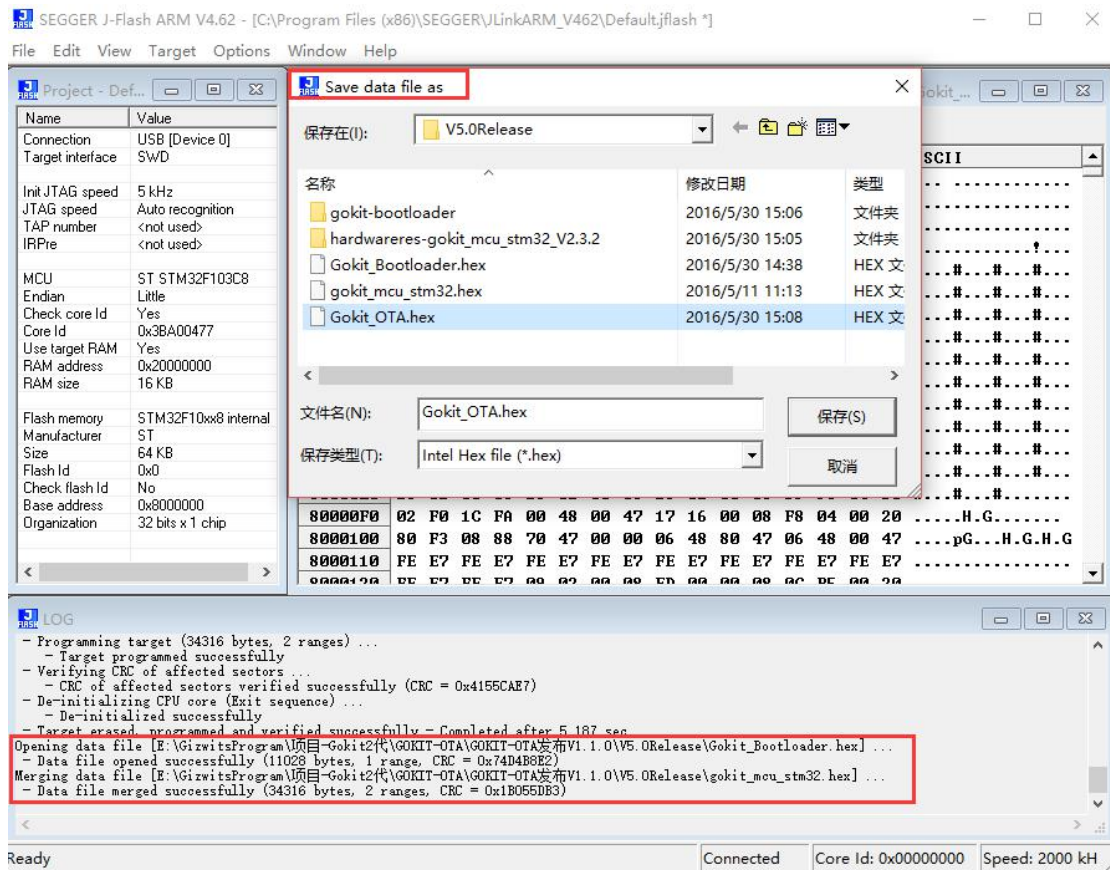
2、开启 JFlash-ARM 工具. 打开第一个文件，顺序无所谓，因为 hex 文件自带地址信息，如下：



3、点击 File 菜单，选择 Merge data File，如下：



4、保存合并后的 hex 文件，如下：



5、生成目标文件，可以用 STMicroelectronics flash loader 下载

