MM32W0xxxB MCU OTA 应用说明文档

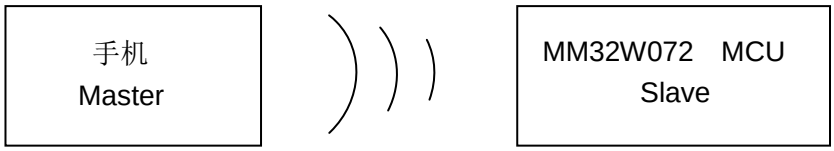
灵动微电子推出 MM32W0xxxB 蓝牙系列 MCU， MM32W072xxB 是超低功耗的单模蓝牙芯片，射频采用 2.4GHz ISM 频段的频率，2MHz 信道间隔，符合蓝牙规范。MM32W072xxB 使用高性能的 ARM® Cortex®-M0 为内核的 32 位微控制器，最高工作频率可达 48MHz，内置高速存储器，丰富的增强型 I/O 端口和外设连接到外部总线。MM32W072xxB 系列包含 1 个 12 位 ADC、4 个 16 位通用定时器、1 个 32 位通用定时器、1 个高级 PWM 定时器，还包含标准的通信接口：2 个 UART 接口、1 个 I2C 接口、1 个 SPI 接口和 1 个 USB 接口。

MM32W072xxB 产品系列工作电压为 2.3V~3.6V，工作温度范围为-40℃~+85℃。多种省电工作模式保证低功耗应用的要求。

MM32W0xxxB MCU OTA 应用说明文档是为了用户快速上手、了解 MM32W 系列 MCU 针对 OTA 升级应用方案配置参考文档；MM32W0xxxB MCU 在 OTA 升级应用领域拥有广泛的应用，针对不同用户在使用 MM32W0xxxB 过程中可能会遇到的疑问，本文都有相关的解答，并且提供针对 OTA 的外设使用样例程序，可以帮助用户快速实现产品设计的验证，节省用户的开发时间。

本文通过手机实现对 MM32W072 MCU 的 APP 代码更新，不涉及手机端的 APP 编程。

图 1. OTA 实现图例



目录

1.OTA 意义

1) 修复产品缺陷。

2) 丰富产品功能，增加用户粘性。

3) 迭代的产品升级，也有助于快速切入市场，降低整体开发成本。

4) 缩短产品生产时间 。

2.OTA 固件升级需求

1) 广泛和种类各异的设备：一个标准 OTA 接口可确保其固件升级架构通用于不同的节点。

2) 不断变化的需求和新功能：在基于 MCU 的应用中，OTA 固件升级功能不仅能够更新固件，而且还能重新配置片上硬件资源。

3) 紧迫的产品上市需求： OTA 固件升级可实现渐进式部署。

3.实现 OTA 程序

OTA 固件升级其实就是 IAP 应用编程，要完成固件升级需要设计两个程序，一个为 Bootloader 程序，另外一个为 Application 程序。

OTA 升级设计很重要的一个工作就是首先规划好 Flash 区域的布局，必须清晰地知道 Bootloader，Application 以及下载的.bin 文件在 Flash 中放置的位置。

Bootloader 角色主要完成的：

1） 读取固件参数信息；

2） 判断是否需要更新 Application 程序，如果不需要直接跳到 Application 区域执行；

3） 如果需要更新，则将固件搬到 Application 区域，并且更新固件参数信息（表示已将更新过该固件了），最后跳到 Application 区域执行。

Application 角色主要完成的：

1） BLE 协议栈部分；

1） 发送 BLE 请求查询上位机是否是最新固件信息；

2） 和当前固件做对比，如果需要更新，就进行下载；

3） 将下载的固件写到规划好的 OTA 固件存储区域；

4.需要准备工具

1、Demo Application Code

W072OTA_APP Code：应用程序

W072OTA_BOOT Code：启动程序

2、相应 Bin 文件

应用程序 Bin 文件

启动程序 Bin 文件

3、Bin 文件合并工具

二进制文件合并工具

4、Hex 转 Bin 文件工具

hex2bin.exe

5、手机 APP

MG OTA

5.分区说明

OTA 主要的任务就是将更新版本代码通过 BLE 下载到 OTA 区域，然后由 BOOT 将 OTA 区域的 Code 转载到 Run Code 区域，并且擦除 OTA 区域的 Code。

在本次提供的样例程序中 Flash(128KB)区域的大致分配示例如表 1 所示，主要用到了前 68KB 的存储空间，包括 4KB 的 BOOT Code 空间、32KB 的 Run Code 空间、32KB 的 OTA Zone 空间，在开发的过程中，可适当根据 BOOT Code 的 Size 和 APP Code 的 Size 调整空间。

上电运行的时候有 BOOT 检测并选择正确的动作，当检测到 OTA Code (B Zone)有合法的 OTA 数据时，则完成数据从 OTA Code (B Zone)到 Run Code (A Zone)的搬移过程，并同时将 OTA 的数据完全擦除 (为了保证 BLE OTA 升级过程的顺利，需要确保 Flash 所有未使用区域已经被完全擦除)。

使用手机 App 连接上 W072 BLE 蓝牙设备，通过特征值对应的句柄将需要升级的二进制文件分别以 16 字节为 1 个包依次发送给 W072 BLE 蓝牙设备，当 W072 BLE 蓝牙设备接收完成所有数据后，自动复位并升级成功。

表 1.地址空间分配

Flash Address	Size	Area
0x0800 0000 ~ 0x0800 1000	4 KB	BOOT Code
0x0800 1000 ~ 0x0800 9000	32KB	Run Code (A Zone)
0x0800 9000 ~ 0x08011000	32 KB	OTA Code (B Zone)
……	……	……

6.代码说明

6.1 APP 程序

APP 程序中主要包括了系统时钟的配置、SPI 初始化配置、低功耗模式及相关中断配置、中断向量表的映射、相关 OTA flash 地址配置以及蓝牙协议相关程序。本应用指导手册只对和 OTA 相关的部分做分析，关于蓝牙使用编程等请参考灵动微电子官网文档 SoftWare Programming Guide_Ver1.0。

APP Code 是要下载到 Flash 的从 0x0800 9000 开始的 32KB 空间以内，中断向量表的映射放在 RAM 的 0x2000 00C0 空间，要在 Keil 中做如图 2 所示的 IROM 和 IRAM 配置和如图 3 所示的预定义。

图 2. IROM 和 IRAM 配置

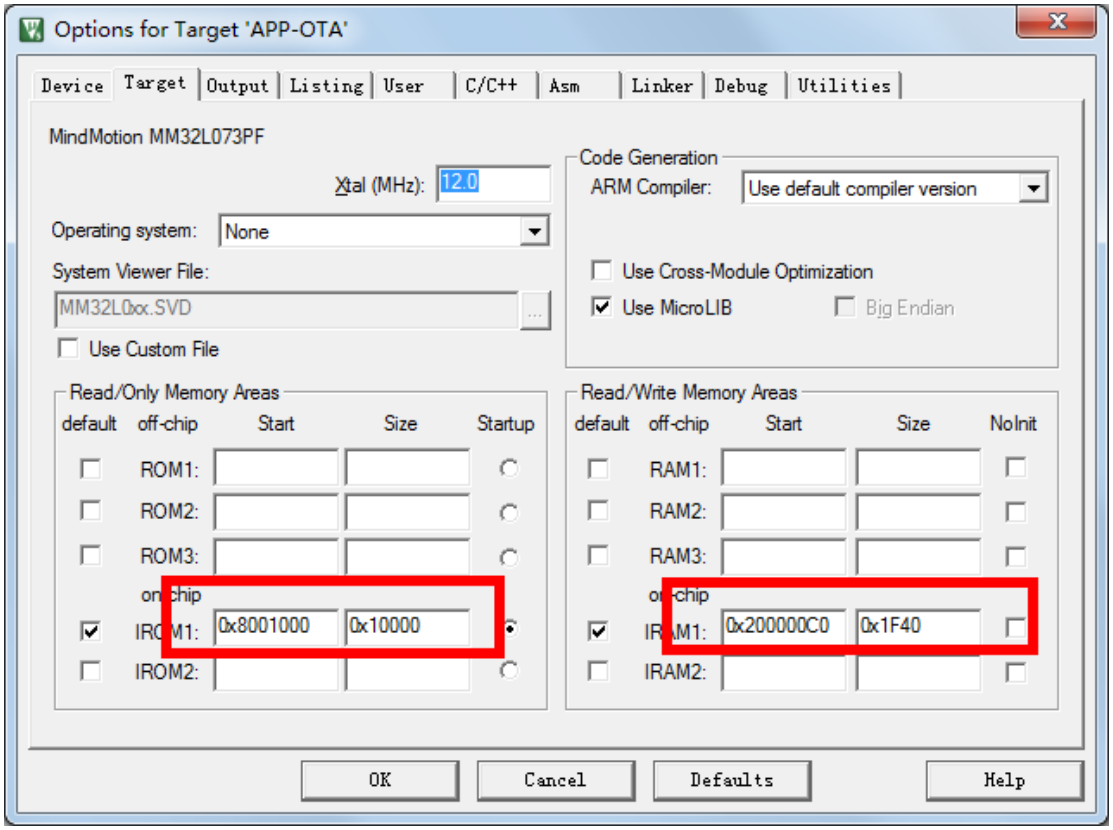
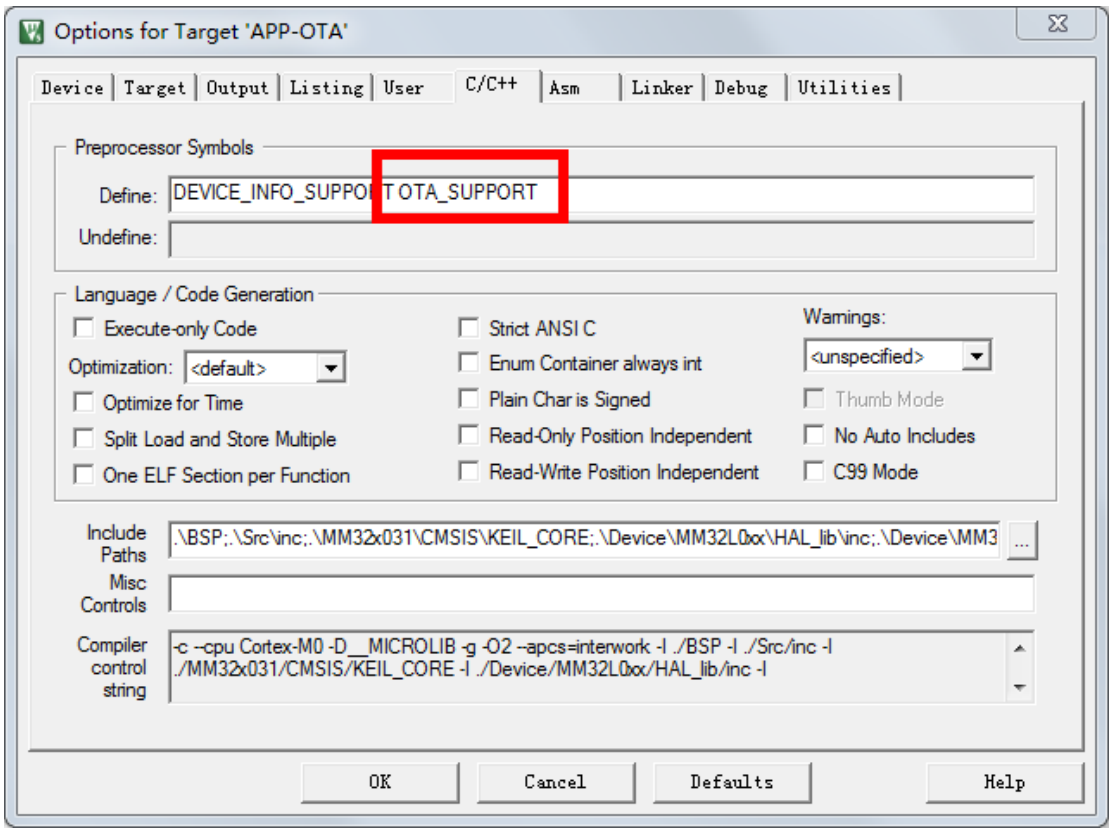


图 3. 宏定义添加



由于 MM32W072 是 Cortex-M0 的内核，没有 M3 中的 SCB->VTOR 寄存器来配置中断向量表的起始地址，可以用如图 4 的方法来实现中断向量表从 Flash 到 RAM 的映射：

图 4. 中断向量表重映射

```
void CodeNvcRemap(void)
{
   uint32_t i = 0;
    for(i = 0; i < 48; i++)
    {
        *((uint32_t*)(0x20000000 +  (i << 2)))=*(__IO uint32_t*)(APPLICATION_ADDRESS + (i<<2));
    }
    RCC->APB2ENR |= RCC_APB2ENR_AFIOEN;
    SYSCFG->CFGR |= 0x03;
}
```

关于 Flash 地址的定义如下图 5 所示，改变 BOOT 空间和 OTA 空间的大小，只需要修改此处的定义即可。

图 5.存储空间分配

```
#define APPLICATION_ADDRESS     (uint32_t)0x08001000
#define FLASH_E2PROM_ADDR_BASE (0x08000000)
#define FLASH_BOOT_ROM_SIZE    (4*1024)
#define FLASH_E2PROM_ADDR_USR  (FLASH_E2PROM_ADDR_BASE + 4*1024)
#define FLASH_E2PROM_ADDR_OTA  (FLASH_E2PROM_ADDR_BASE + 36*1024)
#define FLASH_E2PROM_ADDR_OTA_KB (32)
```

6.2 BOOT 程序

BOOT 程序主要是将 Flash 从 0x0800 9000 地址开始的 OTA Code(B Zone) Copy 到 Flash 从 0x0800 1000 地址开始的 Run Code(A Zone)，完成后 Clear B Zone 的 Code，PC 指针开始从 0x0800 9000 的地址开始执行 APP Code.

程序中关于 A Zone 和 B Zone 的偏移地址定义如下图 6 所示。

图 6.存储空间分配

```
#define BANK_ID_A (0)
#define BANK_ID_B (1)

#define BANK_BASE           0x08000000
#define BANK_A_OFFSET       (1024*4)
#define BANK_B_OFFSET       (BANK_A_OFFSET + 1024 * 32)  /*1KB user data area*/
#define BANK_B_SIZE_KB_MAX  (32)
```

程序中关于跳转到 0x0800 1000 地址开始执行 APP 程序如下图 7 所示。

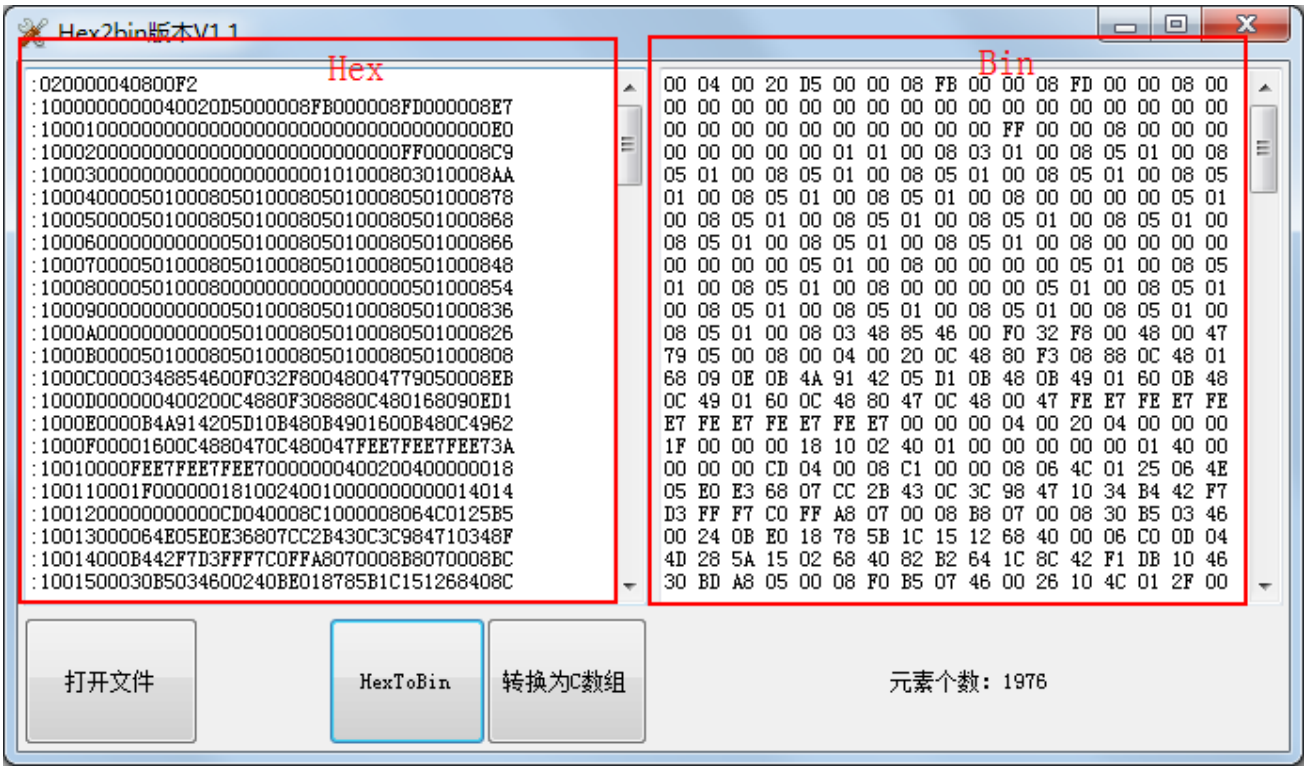
图 7.跳转程序

```
/***********A区起始地址为0x08001000*********************/
#define APPLICATION_ADDRESS     (uint32_t)0x08001000

typedef  void (*pFunction)(void);

/* Private function prototypes -----------------------------------------------*/
void Jump2App(uint32_t AppAddr)
{
    pFunction Jump_To_Application;
    uint32_t JumpAddress;

    JumpAddress = *(__IO uint32_t*) (AppAddr + 4);
    Jump_To_Application = (pFunction) JumpAddress;

    /* Initialize user application's Stack Pointer */
    __set_MSP(*(__IO uint32_t*) APPLICATION_ADDRESS);

    /* Jump to application */
    Jump_To_Application();
}
```

6.3 Hex 文件转 Bin 文件

在完成 APP 和 BOOT 的程序编译后，需要将 Code 生成的 Hex 文件转换成对应的 Bin 文件，方便将两部分 Code 下载到两段不同的 Flash 存储空间中。Hex 转 Bin 可通过 Keil MDK 来生成，也可以通过第三方软件来生成。如图 8 是通过第三方 Tool 来完成转换。

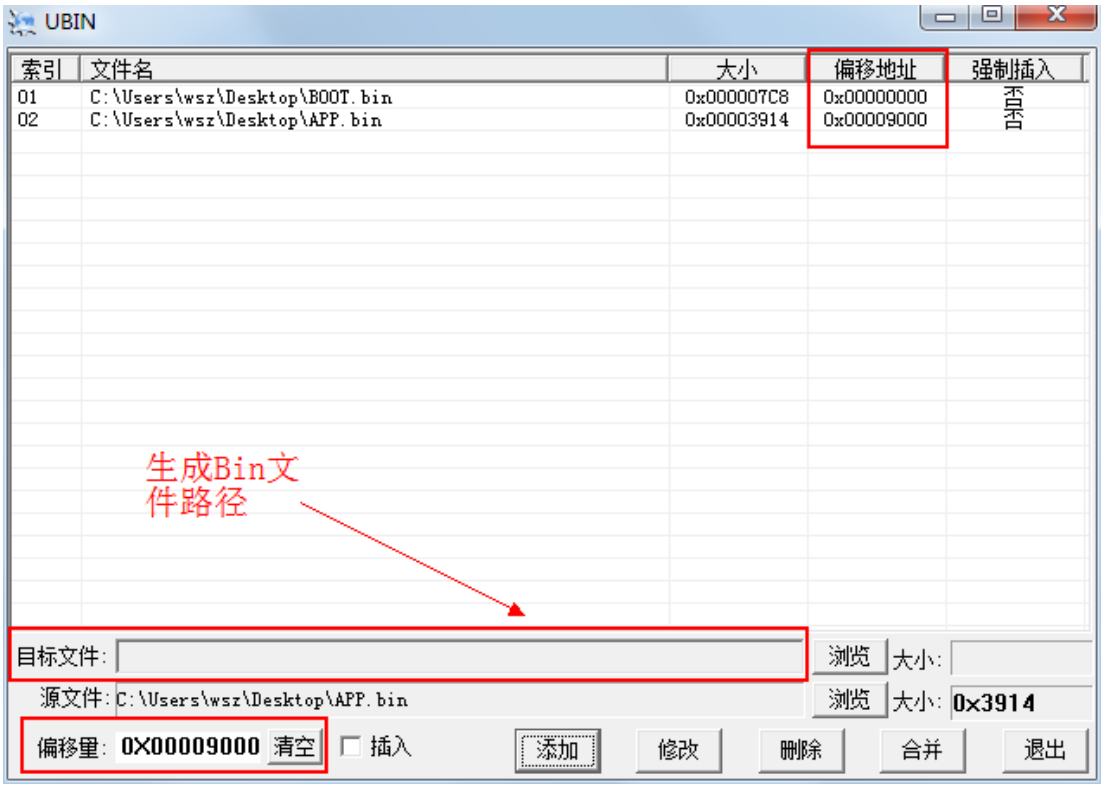
图 8.hex 转 bin



6.4 Bin 文件合并

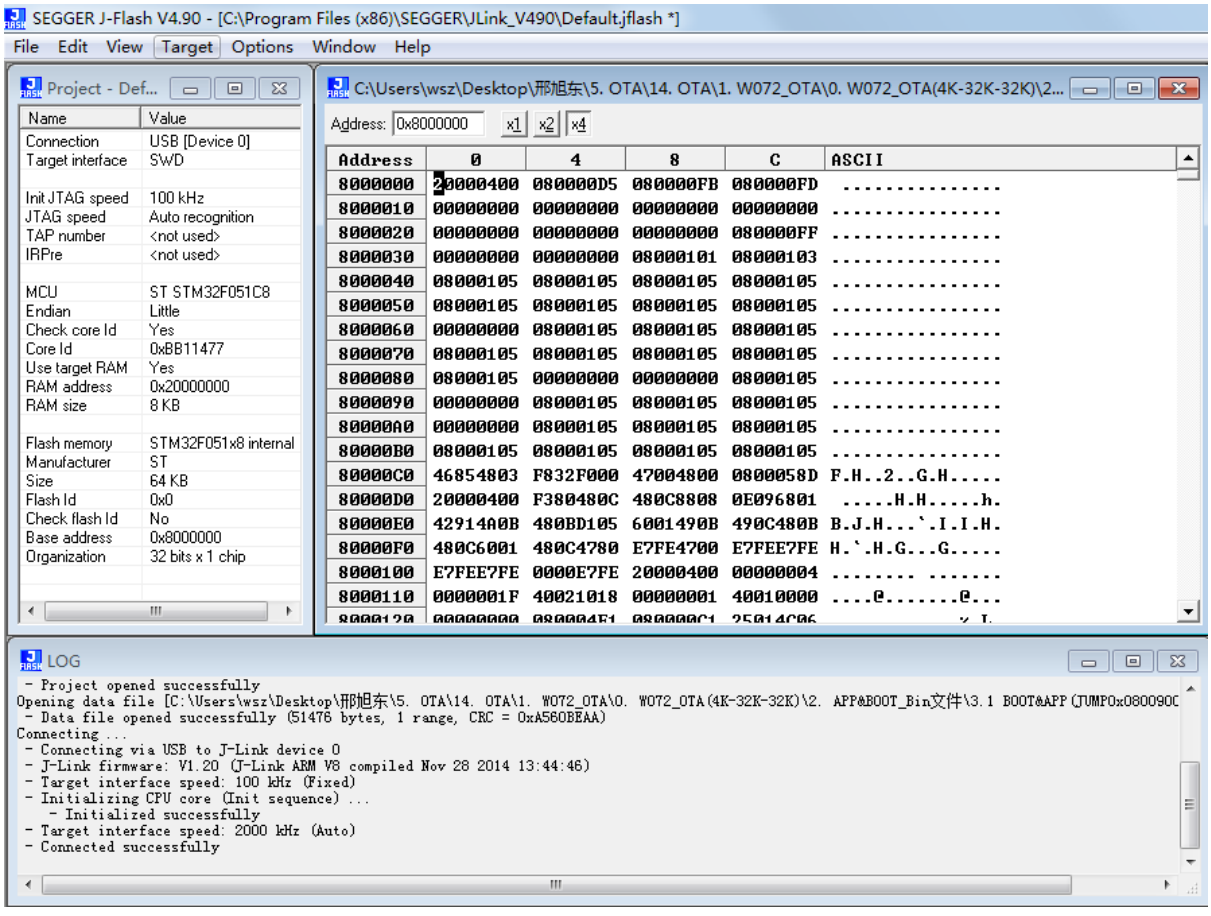
完成 Hex to Bin 文件的转换之后，需要将 APP Bin 和 BOOT Bin 合并成一个 Bin 文件，根据在 Flash 中划分的存储空间，将 BOOT Bin 文件存放在起始地址为 0x0800 0000 开始的 4KB Flash 空间中，将 APP Bin 文件存放在起始地址为 0x0800 9000 开始的 32KB Flash 空间中。Bin 文件的合并利用第三方 Tools 来完成，如图 9。

图 9.bin 文件合并



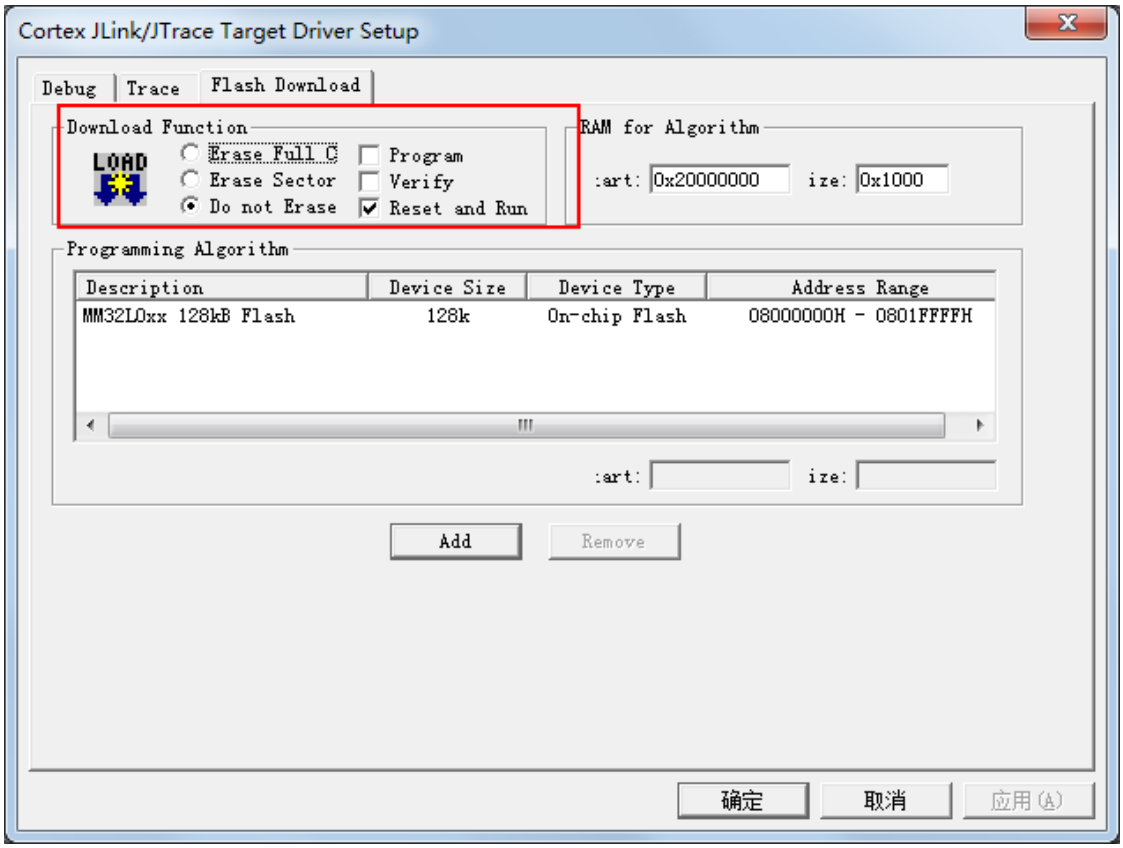
调试将生成的 Bin 文件通过 J-Flash 来下载到 Flash，如图 10。

图 10.程序下载



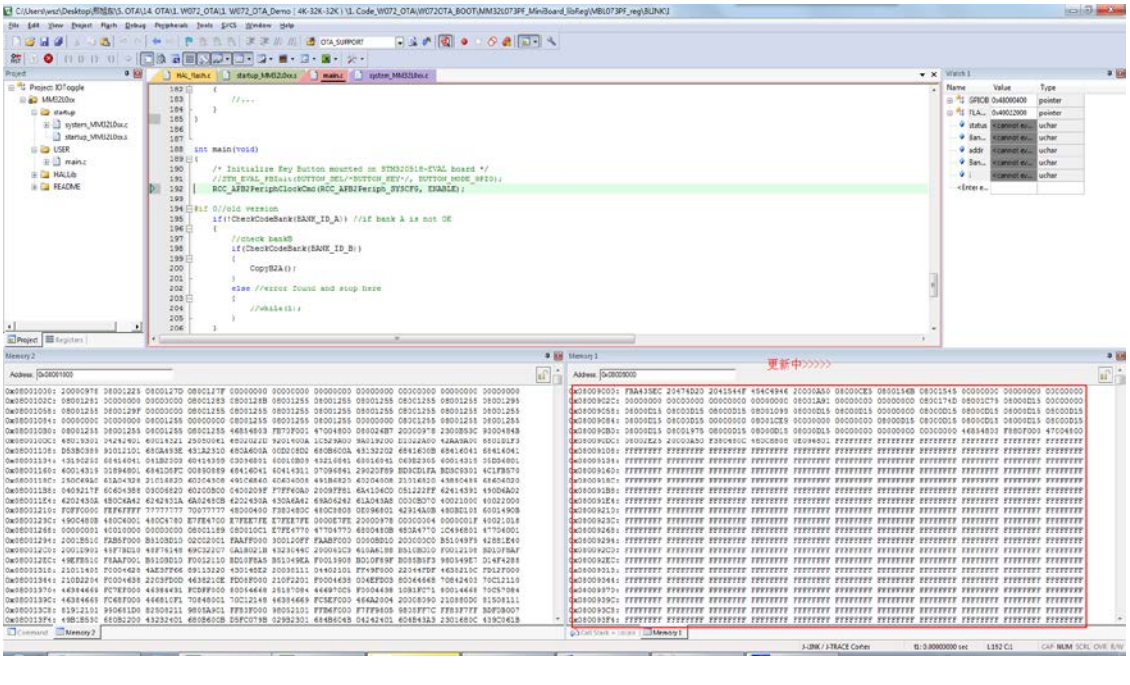
用 Keil5 IDE 打开 W072OTA_BOOT 工程，做图 11 配置后进入 Debug 模式。

图 11.调试设置



打开 MG OTA 手机 APP，点击 ConnectButton，等待手机与 BLE 蓝牙设备连接成功，Buttion 由 Connect 切换成 Disconnect，表示连接成功，rssi=-xx=>>表示连接后的广播强度，点击 OTA-B 或者 OTA-A 对应右侧的 NO Button,则开始更新响应的固件程序，通过在 Keil5 Debug 模式下 Run ,在 Memory 窗口观察从 0x0800 9000 开始的 Code 的更新变化，如图 12 所示。

图 12.数据搬运及擦除



使用手机 APP 连接上 BLE 蓝牙设备，通过特征值对应的句柄将需要升级的二级制文件分别以 16 字节为 1 个包依次发送给 BLE 设备，当 BLE 设备接收完成所有数据后，自动复位并升级成功。

协议中处理 OTA 数据的代码请参考函数 void OTA_Proc(u8 *data, u16 len)//每个接收到的 OTA 数据包的处理。