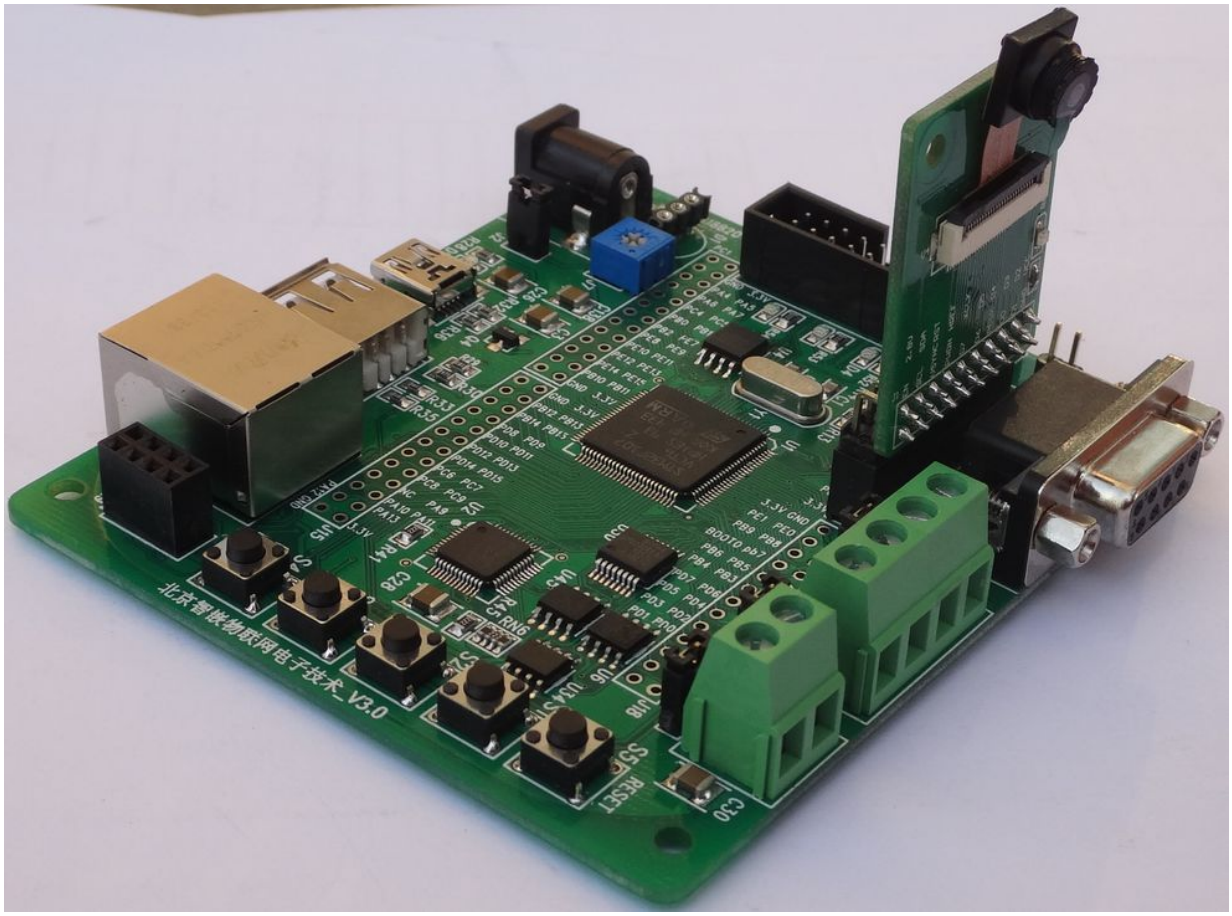
智嵌 以太网实现远程更新固件例程使用手册

版本号：A

拟制人：赵工

时 间 ：2013 年 7 月 1 日



目 录

1 本文档编写目的

本使用手册是针对“以太网实现远程更新固件例程”而编写的。

2 基础知识(该部分摘自互联网)

2.1 IAP

IAP，全称是“In-Application-Programming”，中文解释为“在程序中编程”。IAP 是一种通过微控制器的对外接口（如 USART，IIC，CAN，USB，以太网接口甚至是无线射频通道），对正在运行程序的微控制器进行内部程序的更新的技术（注意这完全有别于 ICP 或者 ISP 技术）。ICP（In-Circuit Programming）技术即通过在线仿真器对单片机进行程序烧写，而 ISP 技术则是通过单片机内置的 bootloader 程序引导的烧写技术。无论是 ICP 技术还是 ISP 技术，都需要有机械性的操作如连接下载线，设置跳线帽等。若产品的电路板已经层层密封在外壳中，要对其进行程序更新无疑困难重重，若产品安装于狭窄空间等难以触及的地方，更是一场灾难。但若进引入了 IAP 技术，则完全可以避免上述尴尬情况，而且若使用远距离或无线的数据传输方案，甚至可以实现远程编程和无线编程。这绝对是 ICP 或 ISP 技术无法做到的。某种微控制器支持 IAP 技术的首要前提是其必须是基于可重复编程闪存的微控制器。STM32 微控制器带有可编程的内置闪存，同时 STM32 拥有在数量上和种类上都非常丰富的外设通信接口，因此在 STM32 上实现 IAP 技术是完全可行的。

实现 IAP 技术的核心是一段预先烧写在单片机内部的 IAP 程序。这段程序主要负责与外部的上位机软件进行握手同步，然后通过外设通信接口将来自于上位机软件的（APP）程序数据接收后写入单片机内部指定的闪存区域，然后再跳转执行新写入的程序，即 APP 程序，最终就达到了程序更新的目的。

在 STM32 微控制器上实现 IAP 程序之前首先要回顾一下 STM32 的内部闪存组织架构和其启动过程。STM32 的内部闪存地址起始于 0x8000000，一般情况下，程序文件就从此地址开始写入。此外 STM32 是基于 Cortex-M3 内核的微控制器，其内部通过一张“中断向量表”来响应中断，程序启动后，将首先从“中断向量表”取出复位中断向量执行复位中断程序完成启动。而这张“中断向量表”的起始地址是 0x8000004，当中断来临，STM32 的内部硬件机制亦会自动将 PC 指针定位到“中断向量表”处，并根据中断源取出对应的中断向量执行中断服务程序。最后还需要知道关键的一点，通过修改 STM32 工程的链接脚本可以修改程序文件写入闪存的起始地址。

在 STM32 微控制器上实现 IAP 方案，除了常规的串口接收数据以及闪存数据写入等常规操作外，还需注意 STM32 的启动过程和中断响应方式。图 1 显示了 STM32 常规的运行流程。

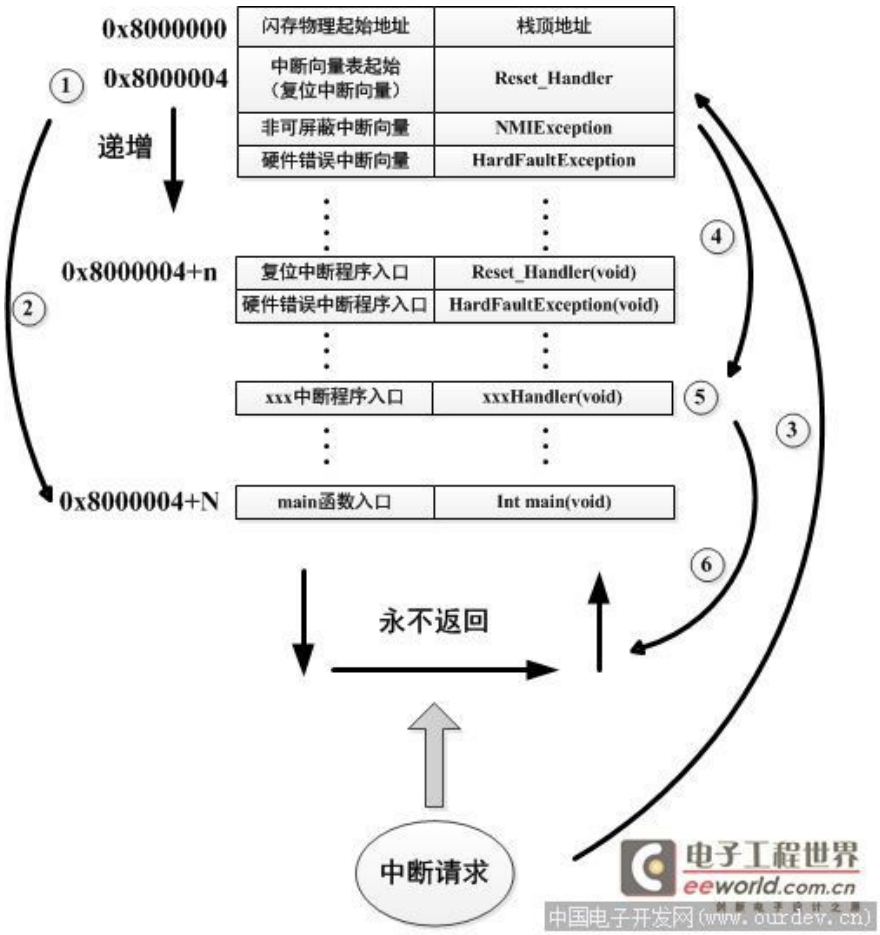


图 1 STM32 常规程序运行流程

对图 1 解读如下：

1、STM32 复位后，会从地址为 0x8000004 处取出复位中断向量的地址，并跳转执行复位中断服务程序，如图 1 中标①所示。

2、复位中断服务程序执行的最终结果是跳转至 C 程序的 main 函数，如图 1 中标号②所示，而 main 函数应该是一个死循环，是一个永不返回的函数。

3、在 main 函数执行的过程中，发生了一个中断请求，此时 STM32 的硬件机制会将 PC 指针强制指回中断向量表处，如图 1 中标号③所示。

4、根据中断源进入相应的中断服务程序，如图 1 中标号⑤所示。

5、中断服务程序执行完毕后，程序再度返回至 main 函数中执行，如图 1 中标号⑥所示。

若在 STM32 中加入了 IAP 程序，则情况会如图 2 所示：

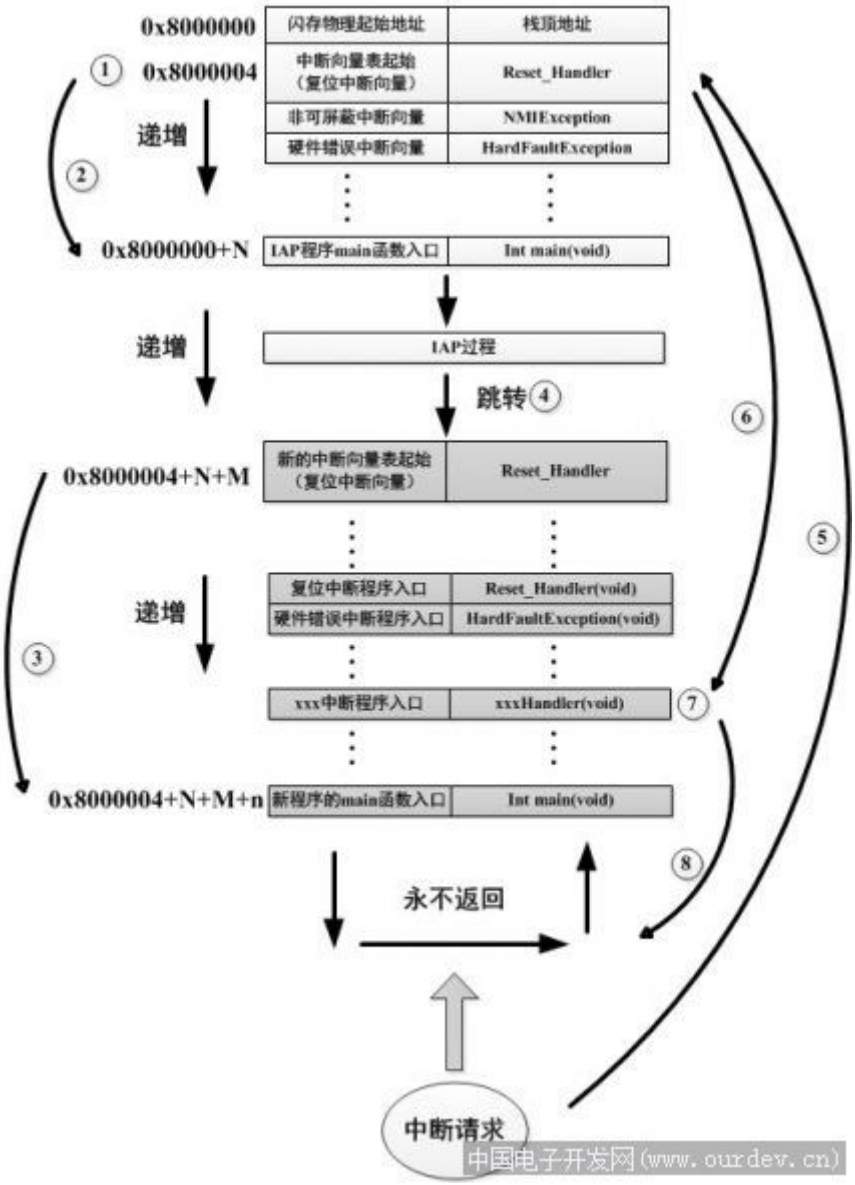


图 2 加入 IAP 后的程序运行流程图

对图 2 的解读如下:

1、 STM32 复位后，从地址为 0x8000004 处取出复位中断向量的地址，并跳转执行复位中断服务程序，随后跳转至 IAP 程序的 main 函数，如图 2 中标号①、②所示。这个过程和图 1 相应部分是一致的。

2、执行完 IAP 过程后（STM32 内部多出了新写入的程序，图 2 中以灰色底纹方格表示，地址始于 0x8000004+N+M）跳转至新写入程序的复位向量表，取出 APP 程序的复位中断向量的地址，并跳转执行 APP 程序的复位中断服务程序，随后跳转至新程序的 main 函数，其过程如图 2 的标号③所示。APP 程序的 main 函数应该也具有永不返回的特性。同时应该注意在 STM32 的内部存储空间在不同的位置上出现了 2 个中断向量表。

3、在新程序 main 函数执行的过程中，一个中断请求来临，PC 指针仍会回转至地址为 0x8000004 中断向量表处，而并不是 APP 程序的中断向量表，如图 2 中标号⑤所示。注意到这是由 STM32 的硬件机制决定的。

4、根据中断源跳转至对应的中断服务，如图 2 中标号⑥所示。注意此时是跳转至了 APP 程序的中断服务程序中。

5、中断服务执行完毕后，返回 main 函数。如图 2 中标号⑧所示。

从上述两个过程的分析可以得知，对将使用 IAP 过程写入的程序要满足 2 个要求：

1、APP 程序必须从 IAP 程序之后的某个偏移量为 x 的地址开始；

2、必须将 APP 程序的中断向量表相应的移动，移动的偏移量为 x；

将中断向量表移动的方法是在程序中加入函数：

void NVIC_SetVectorTable(u32 NVIC_VectTab, u32 Offset)；

其中参数 NVIC_VectTab 为中断向量表起始位置，而参数 Offset 则为地址偏移量，如将中断向量表移至 0x8002000 处，则应调用该函数如下：

void NVIC_SetVectorTable(0x8000000, 0x2000)；

同时有必要提醒读者注意的是，此函数只会修改 STM32 程序中用于存储中断向量的结构体变量，而不会实质地改变中断向量表在闪存中的物理位置，详情请研究该程序原型。

有了以上准备后就可以着手设计一个 IAP 方案了，如下：

1、STM32 复位后，利用一个按键（也可以不用）的状态进行同步，当按键按下时表示将要进行 IAP 过程；

2、在 IAP 过程中，通过上位机软件向 STM32 的外设通讯接口（比如 USART1、网口等）发送所要更新的 APP 程序文件，STM32 接收到数据后转而从 0x8002000 地址开始写入收到的数据；

3、再次复位后，跳转 0x8002004 地址开始运行 APP 程序；

注意事项：

（1）利用 IAP 写入的 APP 程序最好是.bin 格式的文件，但不能是.hex 格式的文件；

（2）向 STM32 发送 APP 程序文件时尽量慢一些，因为 STM32 对 FLASH 的写入速度往往跟不上通讯外设接口的速度；

（3）建议在 STM32 和上位机之间设计一套握手机制和出错管理机制，这样可以大幅提高 IAP 的成功率；

2.2 APP

2.1 节所述的 IAP 程序必须通过其它手段，如 JTAG 或 ISP 烧入，我们可以把 IAP 程序称为 Bootloader 程序。APP 程序即是我们通过 IAP 程序下载的应用程序。IAP 和 APP 程序分别存放在 STM32 FLASH 的不同地址范围，一般从最低地址区开始存放 IAP，紧跟其后的就是 APP 程序（注意，如果 FLASH 容量足够，是可以设计很多 APP 程序的）。

3 IAP 的实现

本文档介绍两种 IAP 的实现方式，第一种是借助开发板上的按键实现 IAP 和 APP 程序的切换；第二种是通过在 APP 程序中设置更新标志，并在 IAP 程序开始处检查该标志来判断是否要进行程序的更新，如果不需要更新则直接跳转到 APP 程序，如果需要更新则继续运行 IAP，并开始下载新的 APP 程序。

第一种在理论上实现了 IAP 对 APP 的更新，但是仍然需要对开发板机械操作才能完成，基本不具有实际应用价值，但也不失为学习 IAP 技术的有效方法之一。第二种方式不需要对开发板做任何机械的操作即可以实现程序的更新，这对于实现远程更新程序具有重大意义。

3.1 按键方式实现 IAP

1、 打开“以太网实现远程更新固件例程（按键方式）”文件夹，通过 JLINK 将“IAP(按键方式)”下载到开发板，按住 4 个 USER 按键其中之一，再复位，则进入到 IAP 程序，此时开发板等待远程连接（通过 HTTP）。

2、 打开浏览器，在地址栏输入“192.168.1.253”，回车，则会出现如图 3 所示界面：

图 3

在 ID 栏输入“1234”，在 Password 栏输入“1234”，然后点击“Login”，登录成功后，如图 4 显示：

图 4 选择待下载的目标 BIN 文件

在上图中点“浏览”，选择要要下载的 bin 文件：“以太网实现远程更新固件例程（裸机）\以太网实现远程更新固件例程（按键方式）\LED 灯闪烁实验（裸机）\Project\RVMDK\BIN”（关于 bin 文件的生成见后文），然后点击“Updata”即可。下载完成后，出现如图 5 界面：

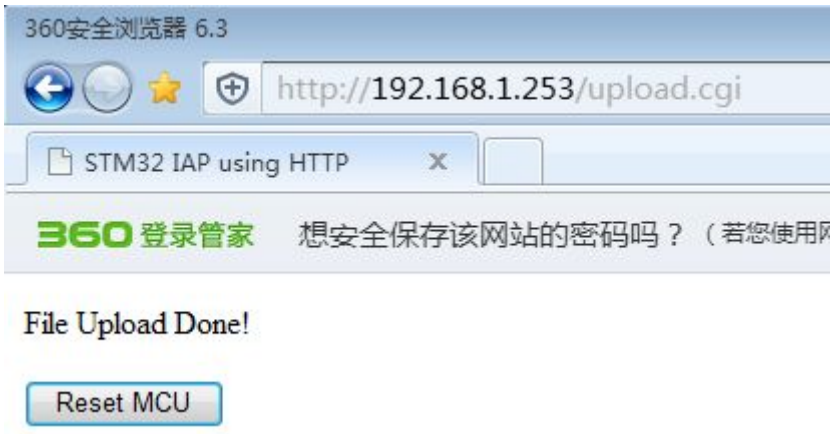
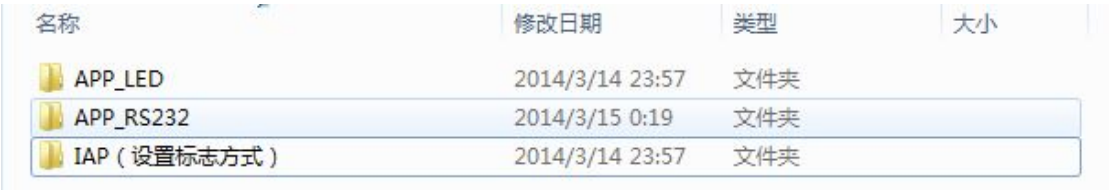


图 5

然后点击“Reset MCU”，这时即可运行所下载的 APP 程序（注意点击复位的时候，四个 USER 按键不能按下，否则又进入到 IAP 程序了）。

3.2 设置更新标志实现 IAP

这种方式中，IAP 程序和 APP 程序都必须运行网络协议（LWIP），本文档介绍两个 APP 程序的实现（要分开下载）：APP_LED 和 APP_RS232，在光盘中的位置为“开发板例程\裸机版\以太网实现远程更新固件例程（裸机）\以太网实现远程更新固件例程（设置标志方式）”:如下图：



1、 打开“以太网实现远程更新固件例程（设置标志方式）”文件夹，用 JLINK 将“IAP（设置标志方式）”程序下载到开发板（方法和 3.2 节相同），然后再将“APP_LED”程序下载到开发板（第一次要用 JLINK 下载，以后就可以通过网口下载）。

2、 复位开发板，此时可以看到 APP_LED 程序在运行。

3、 如果需要对 APP_LED 程序更新，则可以在浏览器的地址栏输入“192.168.1.253”回车，输入 ID 和 Password（都是“1234”），选择要下载的 bin 文件，以 APP_LED 为例路径如下：“以太网实现远程更新固件例程（裸机）\以太网实现远程更新固件例程（设置标志方式）\APP_LED\Project\RVMDK\BIN”。下载完成后，点击复位即可运行所下载的 APP 程序。

注意：因为本 IAP 程序不检测是否已经下载了 APP 程序，而是直接跳转到 APP 程序地址了（0x08010000），如果在这个地址处还没有装载 APP 程序，这时整个程序就会死掉，永远不会返回了。所以第一次使用时需要用 JLINK 把 IAP 和 APP 程序都下载到开发板(只是存放的地址不同)。当然读者也可以在 IAP 程序中增加判断在 APP 的地址处是否已经下载了 APP 程序的代码，如果还没有下载，则不跳转。

3.3 IAP 和 APP 程序的注意点

1、 在 IAP 程序中，跳转到 APP 程序前要关闭总中断以及复位中断向量表和时钟，如：

```
CLI();          //关闭总中断
NVIC_DeInit();
RCC_DeInit();
```

2、 在 IAP 程序中中断向量表的偏移值为 0x0，如：

```
void NVIC_Configuration(void)
{
  NVIC_InitTypeDef   NVIC_InitStructure;

  /* Set the Vector Table base location at 0x08000000 */
  NVIC_SetVectorTable(NVIC_VectTab_FLASH, 0x0);

  /* 2 bit for pre-emption priority, 2 bits for subpriority */
  NVIC_PriorityGroupConfig(NVIC_PriorityGroup_4);

  /* Enable the Ethernet global Interrupt */
  NVIC_InitStructure.NVIC_IRQChannel = ETH_IRQn;
  NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority = 2;
  NVIC_InitStructure.NVIC_IRQChannelSubPriority = 0;
  NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE;
  NVIC_Init(&NVIC_InitStructure);

}
```

3、 在 APP 程序中，要打开全局中断（如果用到中断的话），如：

```
/* Setup STM32 system (clocks, GPIO, NVIC) */
NVIC_DeInit();
RCC_DeInit();
NVIC_Configuration();
System_Setup();
USART_Configuration();
SEI(); //开全局中断
```

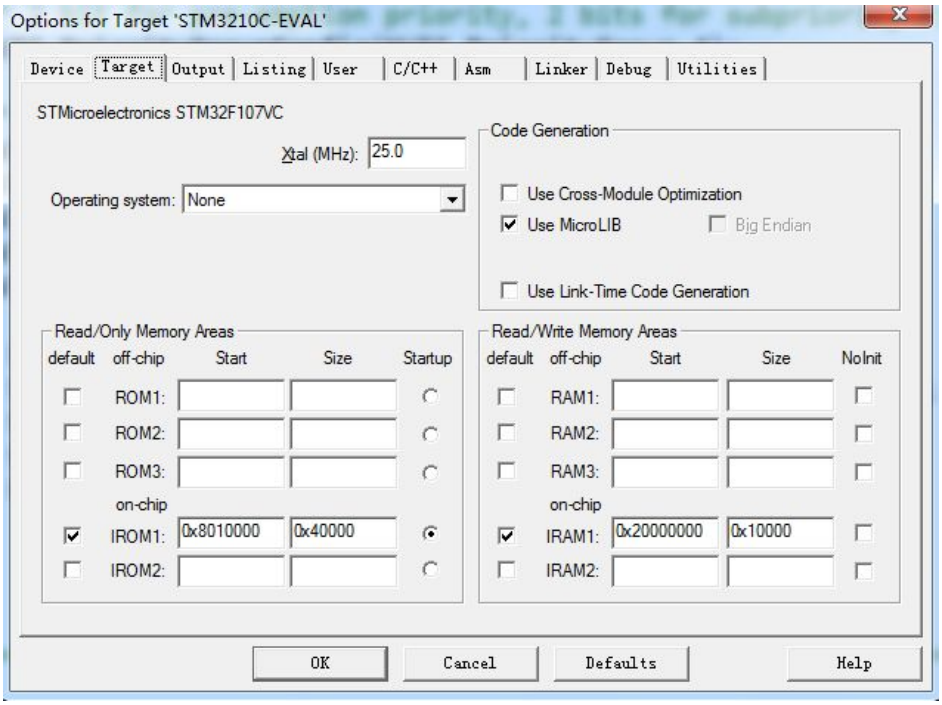
4、 在 APP 程序中，中断向量表的偏移值为 0x10000（这个值要和 APP 程序的起始地址对应，本示例中的 APP 起始地址都是 0x08010000），如：

```
void NVIC_Configuration(void)
{
    NVIC_InitTypeDef   NVIC_InitStructure;

    /* Set the Vector Table base location at 0x08000000 */
    NVIC_SetVectorTable(NVIC_VectTab_FLASH, 0x10000);

    /* 2 bit for pre-emption priority, 2 bits for subpriority */
    NVIC_PriorityGroupConfig(NVIC_PriorityGroup_4);


    NVIC_InitStructure.NVIC_IRQChannel = USART2_IRQn;
    NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority = 3;
    NVIC_InitStructure.NVIC_IRQChannelSubPriority = 0;
    NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE;
    NVIC_Init(&NVIC_InitStructure);
}
```

4 BIN 文件的生成

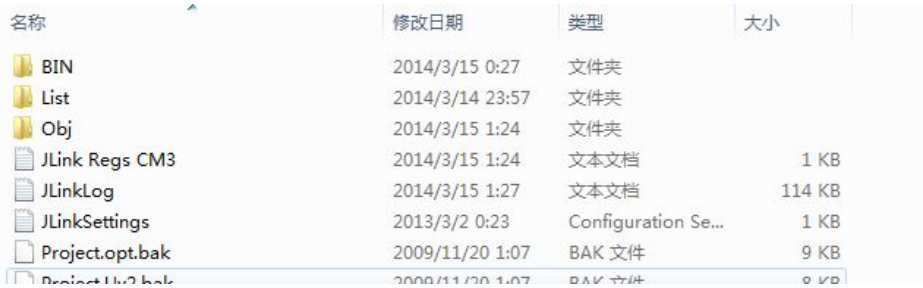
1、 设置 APP 程序起始地址

打开示例工程（如 APP_LED），在工程的“Option for Target….”界面中的“Target”页里将“IROM”的“Start”列改为 APP 程序起始的地址，如下图中将程序起始位置设为 0x8010000（也可以是其他地址）。

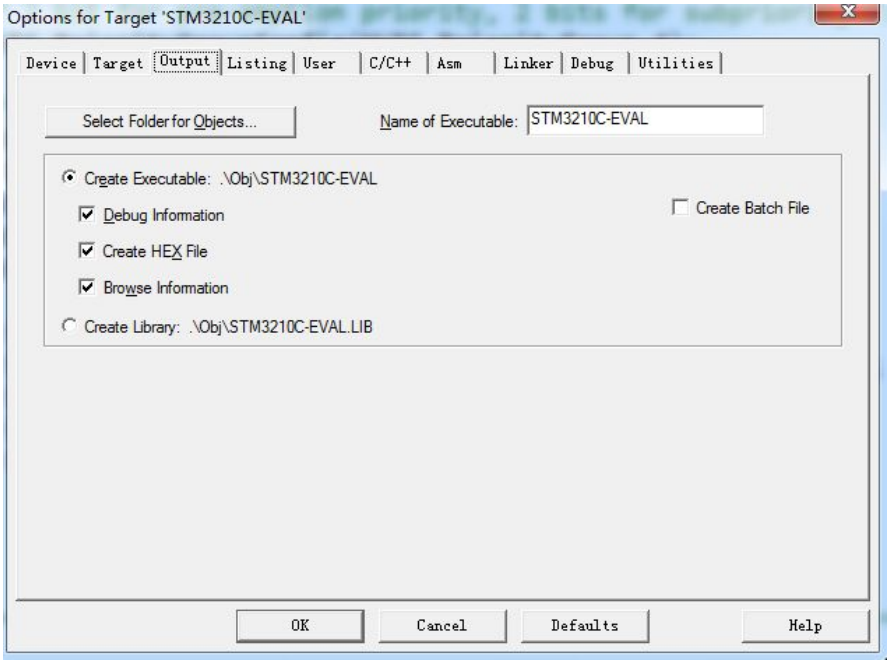


2、 利用 KEIL4 生成 BIN

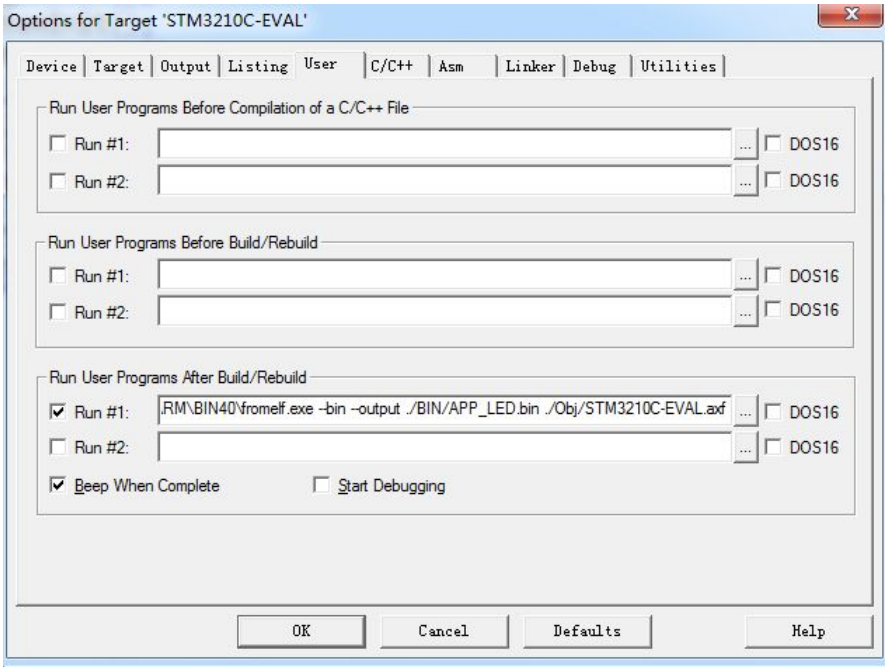
（1） 在“APP_LED\Project\RVMDK”路径下新建文件夹“BIN”：



（2） 在”Output”选项中，设置如下：



（3） 在“User”选项中，设置如下：



将以下红色内容复制到“Run #1”中：

C:\Keil\ARM\BIN40\fromelf.exe --bin --output ./BIN/APP_LED.bin ./Obj/STM3210C-EVAL.axf

点击“OK”退出设置，再对工程编译，然后就可以看到新建的“BIN”文件夹中有“APP_LED.BIN”生成。

----------以下无正文。