Iap,全名为in applacation programming

功能：可通过某种通信方式，将新程序传输到片子上来更新程序

实现iap有两个很重要的前提,首先,单片机程序能对自身的内部flash进行擦写,第二,单片机要有能够和外部进行通讯的方式,无论是网络还是别的方式,只要能传输数据就行

　　通常实现 IAP 功能时，即用户程序运行中作自身的更新操作，需要在设计固件程序时编写两个项目代码，第一个项目程序不执行正常的功能操作，而只是通过某种通信方式(如 USB、 USART)接收程序或数据，执行对第二部分代码的更新；第二个项目代码才是真正的功能代码。这两部分项目代码都同时烧录在 User Flash 中，当芯片上电后，首先是第一个项目代码开始运行，它作如下操作：

1）检查是否需要对第二部分代码进行更新

2）如果不需要更新则转到 4）

3）执行更新操作

4）跳转到第二部分代码执行

要做iap首先我们要知道stm32的启动流程,流程如下

单片机从0x80000000位置启动,并将该地址当成系统栈顶地址

运行到中断向量表中,默认的中断向量表为0x80000004,该位置存放复位中断

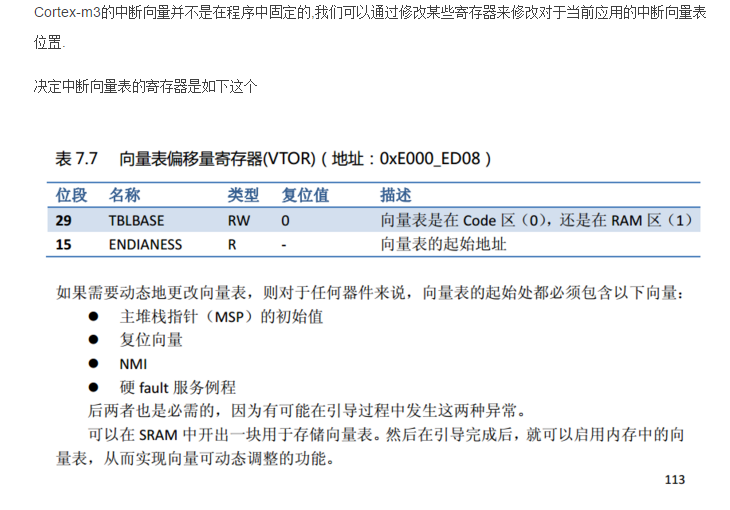
跳转到复位中断处理函数当中,进行系统初始化,然后运行main函数

　　当我们准备用iap的时候,单片机内部是有着两套程序的,这个时候我们就需要在iap中

　　和app中分别放置两套中断向量表,当iap代码中将app烧写到flash中之后,跳转到app的中断向量表中,程序就可以正常执行了,当然需要修改某些系统设置,使得在app和iap阶段单片机可见的中断向量表只能有一套(具体请查看stm32芯片的启动代码)

　　而当需要从app跳转到iap的时候,只需要将app的中断向量表修改成iap的中断向量表,同时主动跳转到iap的reset中断处理程序,这样就能再次开始iap流程.

这样,在系统中就需要我们确定几个东西,第一个是iap程序的中断向量表,为0x80000004位置(80000000存放的是msp的初始值),第二个是app程序的中断向量表,该位置需要根据iap程序的长度计算,比如iap占用了64K,那么512K的芯片而言,就还有448K的空间存放app程序,448K的最开始放置中断向量表,位置就应该是0x08000000+0x10004的位置.



////////////////////////////////////////////////////////////////////////////////////

学习笔记

F103C8 的SRAM有20K ，定义串口接收数组的时候应该小于这个数。也就是说APP程序必须小于这个大小。

\_\_attribute\_\_ ((at(0X20001000))

首先,\_\_attribute\_\_,这个是 用来指定变量或结构位域的特殊属性,该关键字后的双括弧中的内容是属性说明。   
然后是at关键字,该关键字可以用来设置变量的绝对地址,也就是你可以通过这个关键字,指定某个变量处于内存里面的某个给定的地址.   
  
综合起来,就是设置变量处于0X20001000这个地址.

其实就是写两个程序，一个用于判断是执行还是更新（1号），一个是功能程序（2号）。

将判断程序下到片子上后，功能程序可以通过串口下载，1号程序将其存入到SRAM中，在将他写入到flash中，地址为规定好了一个地址后面，地址前面是存放1号程序的。再设置他的中断向量表的偏移。之后

关于偏移地址需要设置一致，中断向量偏移地址决定了APP程序的复位运行位置

写入flash的时候需要设置写入长度，根据容量不同而设置不同的大小

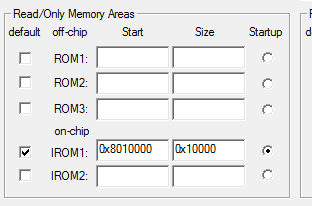
步骤：

首先需要写一个IAP程序

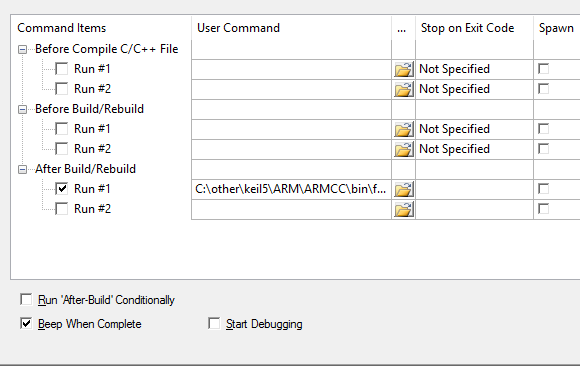
这个IAP 里面，需要通过串口接收发来的程序，并将其保存到flash里面。之后将PC指针跳转APP程序

然后需要一个APP程序，在该程序头加上类似于

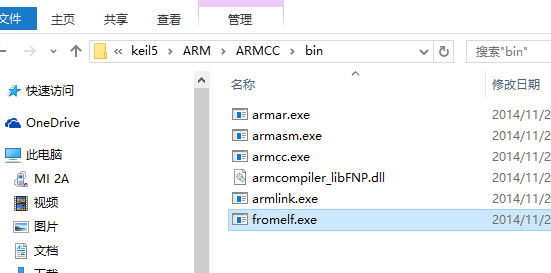
SCB->VTOR = 0x08000000 | 0x10000;

的向量表偏移，根据IAP的制定的长度填写

前面的10000表示IAP的长度，后面的10000表示还剩余的长度

之后再设置编译BIN文件

第一个是选择编译器C:\other\keil5\ARM\ARMCC\bin\fromelf.exe --bin -o



选择生成的BIN文件F:\study\stm32\procedure\IAP\_example\APP\RTE\APP.bin

选择你工程里面的F:\study\stm32\procedure\IAP\_example\APP\Objects\App1.axf