doi: 10.7690/bgzdh.2014.09.026

基于 STM32 平台的 IAP 设计实现

唐小平，廖美英，张祥
（绵阳市维博电子有限责任公司传感器技术部，四川 绵阳 621000）

摘要：针对闪存系统对在应用中编程(in-application programming，IAP)的强烈需求，介绍基于 STM32 平台的 IAP 设计原理以及实现方法。分析 IAP 设计的重要性，以 STM32F103RC 为例，介绍 IAP 程序中的 FLASH 规划、用户程序条件，IAP 实现原理以及在产品中应用 IAP 详细设计的实现过程。结果表明：IAP 应用在产品项目中的成功实现，能使现场产品固件的更新更为便捷，可进行大面积推广使用。

关键词：STM32 平台；在应用中编程；FLASH 规划；STM32F103RC；Cortex-M
中图分类号：TP311.1　　**文献标志码：**A

Implementation of IAP Based on STM32 Platform

Tang Xiaoping, Liao Meiying, Zhang Xiang
(*Department of Sensor Technology*, *Mianyang Weibo Electronics Co*., *Ltd.*, *Mianyang* 621000, *China*)

Abstract: Aiming at the strong demand of flash memory in application programming (IAP), this paper describes the design principle and implementation based on the STM32 IAP platform. Analyze the importance of IAP, citing STM32F103RC as typical, introduce the FLASH program planning, the user's program conditions in IAP, the principle of realization and the detailed realization process of IAP design in product. The results show that the successful implementation of IAP application in product project makes updating of on-site products' application more convenient, could be used for large area application.

Keywords: STM32 platform; in-application programming; FLASH programming; STM32F103RC; Cortex-M.

0　引言

在应用中编程(in-application programming，IAP)是用户自己的程序在运行过程中对 User Flash 的部分区域进行烧写，可在产品发布后方便地通过预留的通信口对产品中的固件程序进行更新升级。通讯接口可以是 SPI、I2C、UART、USB、CAN 和以太网等。随着用户对产品现场适应能力的要求越来越高，大部分基于闪存的系统对于 IAP 的需求也越来越强。笔者以 STM32F103RC 为例，介绍了 IAP 在产品中应用的详细设计和实现过程。

1　STM32 系列产品简介

STM32 系列单片机是 ST 公司推出的 32 位产品线，基于 ARM Cortex-M 内核，具有高性能、低成本和低功耗的特点。从入门级 STM32F0(Cortex-M0)、超低功耗型 STM32L1(Cortex-M3)、主流型 STM32F1(Cortex-M3)，到高性能 STM32F2(Cortex-M3)、混合信号 STM32F3(Cortex-M4)、超高新能 STM32F4（Cortex-M4），FLASH 空间大小范围为 16～1 024 K，RAM 空间范围为 4～256 K，CPU 主频率范围为 16～180 MHz，能满足不同客户的需求。

2　IAP 程序中 FLASH 规划

IAP 程序规划为 8 K byte FALSH 空间，其中 1 页空间用于更新用户程序模块(Application 简称“APP”)标志信息块。APP 标志信息包括 APP 是否需要更新标志、APP 产品更新通讯地址信息、更新产品的 CPU 信息、更新产品程序版本、更新产品程序大小和更新时间等，用于查看产品的程序信息。IAP 占用 FLASH 空间规划如表 1 所示，不同页容量的 CPU 为了实现 IAP 功能需要占用 9～32 K FLASH，余下的空间供 APP 使用。用户可根据 APP 程序大小，加上 9～32 K 额外 FLASH 开销，选择合适的 CPU 容量，实现 IAP 更新 APP 功能。

表 1　IAP 不同页容量的 CPU 占用 FLASH 空间(K)

每页容量	IAP	状态标志
1	8	1
2	8	2
4	8	4
16	16	16

下面以 STM32F103RC 为例，来说明 IAP 设计。STM32F103RC 基于 ARM Cortex-M3 内核，CPU 时钟频率 72 MHz，1.25DMips/MHz，256 K Falsh，48

收稿日期：2014-06-04；修回日期：2014-07-14
作者简介：唐小平(1976—)，男，四川人，学士学位，从事智能电量隔离传感器的设计与开发。

K RAM，Flash 每页容量为 2 K。

3　用户程序条件[1]

如图 1 所示，在 STM32F103RC 微控制器中，用户必须从 FLASH 基地址(0x08000000)开始存放 IAP 应用程序，此时用户程序 APP 必须从 0x08002800(根据规划)开始，最大程序地址到 0x0803FFFF 结束，用户程序最大为 246 K bytes，用户程序向量表必须位于地址 0x08002800。状态标志地址空间从 0x08002000 开始，到 0x080027FF 结束，为 1 页 FLASH 大小。

图 1　闪存存储器使用

4　IAP 实现原理[2]

IAP 实现原理流程如图 2 所示。

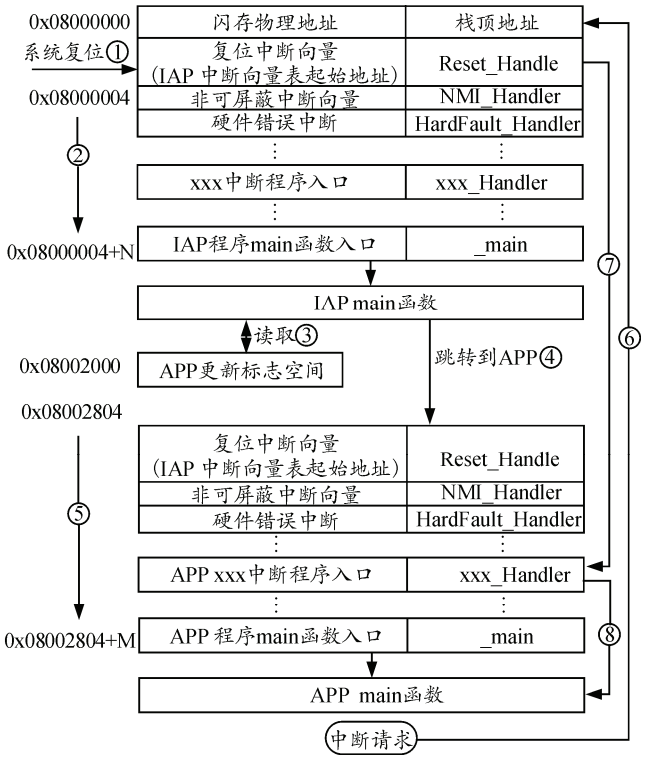

图 2　IAP 实现原理流程

在图 2 IAP 实现原理流程中，STM32F103RC 微控制器复位后，先从 0x08000004 地址取出 IAP 复位中断向量的地址，如标号①所示，并跳转到 IAP 复位中断服务程序。在 IAP 复位中断服务程序执行完之后，会跳转到 IAP 中 main 函数，如图标号②所示。在 IAP 的 main 函数中从 0x08002000 开始地址读取 APP 更新标志，如标号③所示，判断是否需要更新 APP 程序。如果需要，执行 APP 程序更新；如果不需要更新，就跳转到 APP 程序入口，如标号④所示，从 0x08002804 地址取出 APP 复位中断向量的地址，并跳转到 APP 复位中断服务程序。在 APP 复位中断服务程序执行完之后，会跳转到 APP 中 main 函数，执行用户后台程序，如标号⑤所示。在 APP 程序模块后台程序执行过程中，如果 CPU 得到 1 个中断请求，PC 指针仍强制跳转到地址 0x08000004 中断向量表处，如标号⑥所示。程序再根据在 APP 程序模块设置的中断向量表偏移量，跳转到对应中断源新的中断服务程序中，如标号⑦所示。在执行完中断服务程序后，程序返回后台程序继续运行，如标号⑧所示。

5　IAP 设计实现

5.1　实现方法

在本例 IAP 设计中，通信接口采用 UART，通讯波特率 19200BPS，通信协议采用 MODBUS(RTU)协议。

从图 1 闪存存储器使用情况看，IAP 程序空间和 APP 程序空间相互独立。首先将 IAP 程序空间的目标代码烧写到 Flash 存储器的开始区域，随后这个驱动程序使用通讯接口从 PC 端下载二进制 BIN 文件到 STM32F103RC 的内部闪存中，然后跳转去执行新下载的程序。

在设计 IAP 程序模块时，单片机 I/O 口只初始化 UART 端口，JTAG(或 SWD)端口，外部硬件看门狗端口，其他选用复位值；中断只允许串口中断和 1 个定时器中断，其他禁止。

在设计 APP 程序时，首先要初始化中断向量的偏移地址，SCB->VTOR=0x08000000|0x2800；在程序的编译器中设置 APP 程序的 FLASH 基地址为 0x08002800。把 APP 生成的 hex 文件通过编译工具转化为 BIN 文件，方便上位机解析下载该文件。

在设计上位机 IAP 软件时，上位机首先解析 BIN 文件的大小，再把解析的文件按 1 页(文中示例为 2 K)大小组合为 1 帧数据依次发送给下位机。下位机接收完 1 帧数据后按页写入 FLASH。同时，上位机与下位机同时对该页数据进行校验，上位机下载完 BIN 文件后，读取每页校验码，判断比较校验码是否正确。如果有校验码不正确的页，重新下载

该页数据，直到正确为止。

5.2 IAP 实现关键点

1) 从 APP 程序跳转回来运行 IAP 的时候，初始化时要恢复 RCC，NVIC 为复位状态；并且正确设置 NVIC 向量表，RCC 时钟；

2) 在从 IAP 程序中更新完程序后，将要跳转到 APP 程序之前，必须复位 NVIC 为复位状态，防止在跳转过程中出现中断，导致运行程序失败；

3) 在 APP 的初始化程序部分，初始化时要恢复 RCC，NVIC 为复位状态；并且正确设置 NVIC 向量表，RCC 时钟；

4) 外部硬件看门狗一旦使能就不能禁止，实现 IAP 应用，必须处理好看门狗的喂狗指令。特别需要注意，在擦除、编程大容量 FLASH 时，要充分估计耗时，在运行相关程序时要多次喂狗。

5.3 IAP 下位机实现流程

IAP 下位机实现流程如图 3 所示。

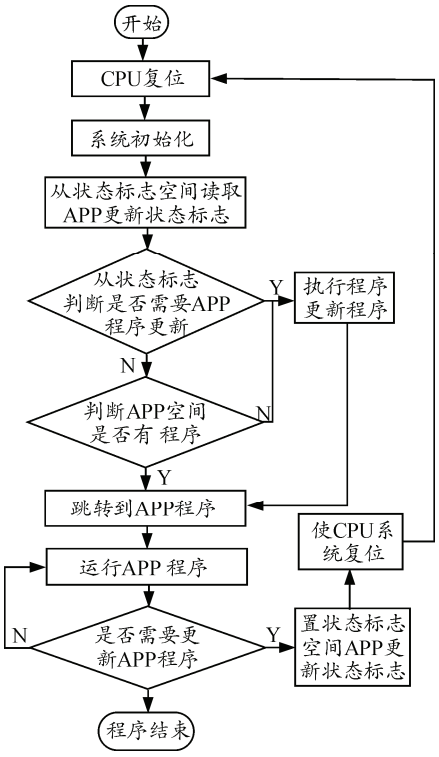

图 3 IAP 实现流程

5.4 IAP 下位机实现程序[3]

```
typedef    void (*pFunction)(void);
pFunction Jump_To_Application;
#define    FLASH_APP_ADDR      0x08002800
mian()
{    uint32 temp32;
   Init_System()
   For(;;){
     if (iapstate.iap_program_flag == 0xaa55){
     for(;;){//运行 IAP 程序，更新程序
     Write_Iap_Flash_State();
     StmWdtClr();
     temp32 = Iap_Boot_Loader_Finish();
     if (temp32){break;}}}
     else if ((((*(__IO uint32 *)FLASH_APP_ADDR)
          &
          0x2FFE0000)==0x20000000)&&(((*(__IO
          uint32*)(FLASH_APP_ADDR+4))&0x0xFF
          FC2800)==0x08000000))
     {STM32_NVIC_DeInit();//跳转 APP 程序
     Jump_To_Application=(pFunction)(*(uint32*)
     (FLASH_APP_ADDR+4));
     __MSR_MSP(*(uint32*)FLASH_APP_ADDR);
     Jump_To_Application();}
     else{
     for(;;){//运行 IAP 程序，初次下载程序
          Write_Iap_Flash_State();
          StmWdtClr();
          temp32 = Iap_Boot_Loader_Finish();
          if (temp32){break;}}}}
```

6 IAP+APP 的 HEX 文件合并下载实现

在 IAP 中实现 APP 程序更新，会导致 1 个 CPU 中有 2 个目标程序模块，编译后会生成 2 个 HEX 文件。在第 1 次下载程序时，一般的做法是先下载 IAP 程序，再下载 APP 程序。对于批量生产的产品，为了减少生产工序，提高生产效率，一般将 2 个 HEX 文件合成 1 个。方法是首先把 IAP 中的 HEX 结束行删除：00000001FF，再把 IAP 的 HEX 文件剩余部分复制到 APP 的 HEX 文件前面，最后实现 2 个 HEX 合并为 1 个文件，进行产品程序的第 1 次下载。

7 结论

笔者在基于 STM32F103RC 的产品平台下，成功实现了在 IAP 程序模块中更新 APP 程序。该方法使现场产品程序固件的更新更为便捷，可以进行大面积推广使用。

参考文献：

[1] ST 公司. 使用 STM32F10xxx 的 USART 实现在应用中编程[S]. 瑞士: ST 公司, 2008: 1-12.

[2] Joseph Yiu. Cortex-M3 权威指南[M]. 宋岩, 译. 北京: 北京航天航空大学出版社, 2009: 83-107.

[3] ST 公司. STM32F10xxx 闪存编程[S]. 瑞士: ST 公司, 2008: 1-21.