回复: 155

**XIVN1987**

(64877775)

出0 入17汤圆

# 分享离线SWD编程器代码 [复制链接]

发表于 2018-9-16 16:08:04 | 只看该作者

1楼 电梯直达

本帖最后由 XIVN1987 于 2018-9-16 16:12 编辑

SWD离线编程器，其实很简单，，

因为关键代码国外的大侠都已经给实现了，，我们只需要简单拼接一下就OK啦

下面我就说下怎样通过拼接代码实现离线编程器：

1、首先，既然是SWD编程器，那首先当然是要实现SWD时序协议了
由于单片机都没有SWD外设，所以只能用GPIO模拟实现SWD时序，，这部分功能已经由ARM公司的CMSIS-DAP代码实现

2、然后就是基于CMSIS-DAP，实现通过DAP读写目标芯片的内存、内核寄存器，，这部分功能已经由DAPLink里面的swd_host.c文件实现

同时，swd_host.c还实现了另一个对实现编程器至关重要的函数：

```
01.  uint8_t swd_flash_syscall_exec(const program_syscall_t *sysCallParam, uint32_t entry, uint32_t arg1,
     uint32_t arg2, uint32_t arg3, uint32_t arg4)
```

复制代码

它的作用是通过DAP在目标芯片上执行

那么，我们只要把编程算法（一段在目标芯片上执行的代码，里面有Flash_Erase、Flash_Write两个函数）通过SWD写入目标芯片的SRAM，然后再通过SWD调用目标芯片SRAM里面的Flash_Erase、Flash_Write两个函数，不就能实现通过SWD给目标芯片编程了吗？？

所以，程序的主体结构就是：

其中target_flash_init()的主要作用就是把芯片的编程算法下载到目标芯片的SRAM中去

好了，SWD编程器已经实现

不过还有一个问题：要下载到目标芯片SRAM中去的编程算法从哪里来？？

我们知道，Keil针对每一颗芯片都有一个Flash编程算法，这个算法存在一个后缀为.FLM的文件里面，，要是我们能把.FLM文件里面的算法内容抽取出来给我们用，，那不就完美了吗

3、其实这个功能也已经有国外大神给实现了，GitHub上的FlashAlgo项目里面有个flash_algo.py文件，它就是用来实现这个功能的

工程示例代码：

另外，这个工程我也已经上传到github上了，，希望坛友能顺便去给加个星，，谢谢啦

https://github.com/XIVN1987/DAPProg

回复 举报本楼层

**XIVN1987**

楼主 | 发表于 2018-9-17 09:02:33 | 只看该作者 来自 26楼

(64816906)

出0 入17汤圆

> boboo 发表于 2018-9-17 00:37
> Traceback (most recent call last):
> File "C:%users\FreeRTOS\Desktop\FlashAlgo\flash_algo.py", line ...

我把flash_algo.py给打包成.exe文件了，，把*.FLM文件放到flash_algo.exe文件目录下，，，然后双击执行flash_algo.exe就能生成*.FLM对应的*.c文件



回复 举报本楼层

**XIVN1987**

楼主 | 发表于 2018-9-23 12:09:10 | 只看该作者 来自 66楼

(64287309)

出0 入17汤圆

本帖最后由 XIVN1987 于 2018-9-23 15:25 编辑

Keil_v5\ARM\Flash_Template目录下有个烧录算法模板，，其中部分函数如下：

对比SWD_flash.c的代码发现一些问题：
1、Keil算法中的函数都是正确返回0、错误返回1，而SWD_flash.c认为函数返回0表示出错
2、Keil算法中Init函数的第三个参数说明，Init函数应该是在Erase、Program、Verify之前各分别执行一次，，而SWD_flash.c的实现只在最开始（也就是Erase之前）调用一次Init

所以，很可能SWD_flash.c不是针对Keil的编程算法写的，，DAPLink项目可能实现了自己的编程算法接口，，而我上面那个STM32的Demo之所以能执行成功，可能是因为：
1、所有函数都没检查返回值，
2、可能在STM32的编程算法中Init函数对Erase、Program、Verify这三个操作执行的内容是一样的，，所以执行一遍Init就行了

不过用在另一些芯片上可能就不行了

所以，我对SWD_flash.c做了一些修正，，不过由于没有板子，暂时没法测试，，感兴趣的坛友可用试一下

==========

另外如果想实现在线编程器的话，其实GitHub上也有现成的代码可用参考：https://github.com/mbedmicro/pyOCD

在这个项目下有个叫flash.py的文件，其中部分函数如下：

是不是看起来和SWD_flash.c中的函数非常像啊，，有了这个文件，实现在线编程器就So Easy了！！

不过也有两个小问题：
1、这个项目是基于CMSIS-DAP（DAPLink）的，如果想用JLink做在线下载的话，需要把底层部分换成jlink.py
2、这是个命令行的项目，想要做个带图形界面的在线下载器的话，需要自己添加GUI功能



回复 举报本楼层

(64268592)

出0 入17汤圆

继续填坑

下面两段内容分别来自flash_algo.py和c_blob.tmpl

整个编程烧写过程占用了目标芯片4K SRAM，其中SRAM起始地址为0x20000000，栈顶指向4K SRAM的末尾，，编程算法占用4K SRAM的前1K，，待烧写数据占用4K SRAM的中间2K，，静态变量和栈共用4K SRAM的最后1K

这种设计对绝大多数Cortex-M芯片是没有问题的，，不过有几种情况可能需要调整红线框住的部分：
1、SRAM的起始地址不是0x20000000，这种调整最简单，把entry的值调整成正确值就可用了
2、单片机的SRAM小于4K，这种就比较麻烦，得根据实际情况重新规划SRAM的分配，，然后红线框住的部分可能都要改动
3、编程算法内容大于1K，，有些使用片外SPI Flash的芯片它的编程算法会非常大，1K SRAM装不下，，这种把后面几个部分的地址都往后延就行了

所以，对于有些比较特殊的芯片，，需要先修改一下flash_algo.py和c_blob.tmpl，然后再生成算法文件对应的.c文件，，不过还好，生成是一次行的，，



回复 举报本楼层

**belongfs**

发表于 2018-9-16 16:21:08 | 只看该作者 2楼

(64876991)

出0 入0汤圆

工程的事例是使用什么硬件？

回复 举报本楼层

**XIVN1987**

楼主 | 发表于 2018-9-16 16:25:50 | 只看该作者 3楼

(64876709)

出0 入17汤圆

> belongfs 发表于 2018-9-16 16:21
> 工程的事例是使用什么硬件？

随便一个STM32F103C8的demo板就行，，模拟SWD用的B13、B14两个引脚



回复 举报本楼层

**bigk2000**

发表于 2018-9-16 17:31:34 | 只看该作者 4楼

(64872765)

不错
曾经自己写过SWD协议实现，可以通过寄存器编程