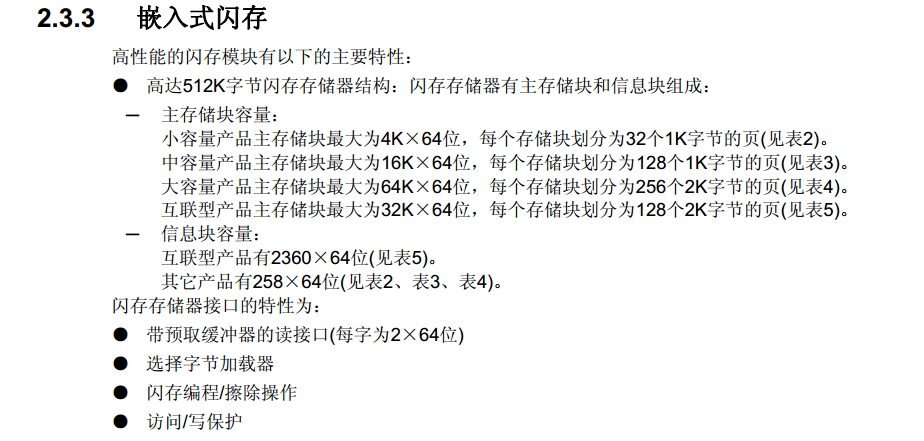
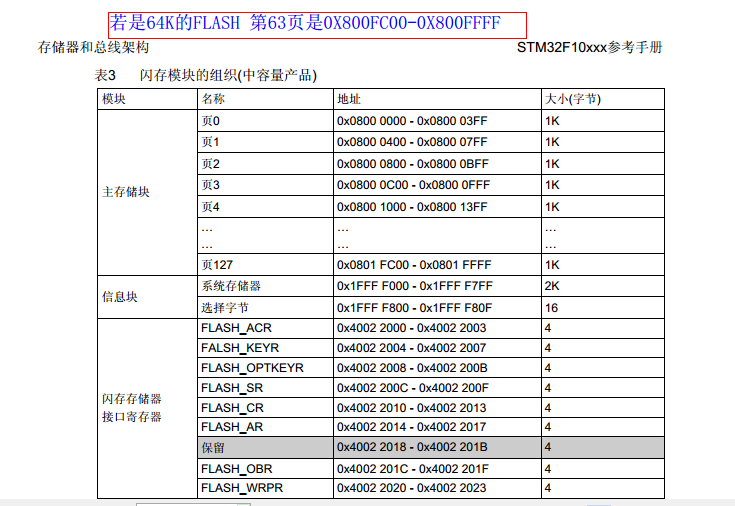
STM32的FLASH是用来存储主程序的，ST公司为了节约成本，没有加入 EEPROM，但是许多场合下我们需要用EEPROM；不过FLASH的容量还是可观的，我们可以利用FLASH模拟EEPROM。

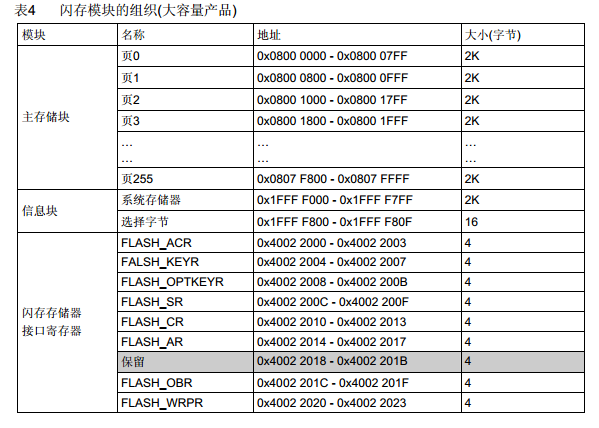
      根据《STM32F10X闪存编程》中的介绍，FLASH除了保存用户代码的部分，其余部分我们是可以利用其作为数据存储使用的。stm32的FLASH分为主存储块和信息块。主存储块用于保存具体的程序代码和用户数据，信息块用于负责由stm32出厂是放置2KB的启动程序（Bootloader）并锁死，用户无法更改。选项字节存储芯片的配置信息及对主存储块的保护信息。STM32的FLASH主存储块按页组织，有的产品每页1KB，有的产品每页2KB。页面典型的用途就是用于按页擦除FLASH。从这点来看，页面有点像通用FLASH的扇区。

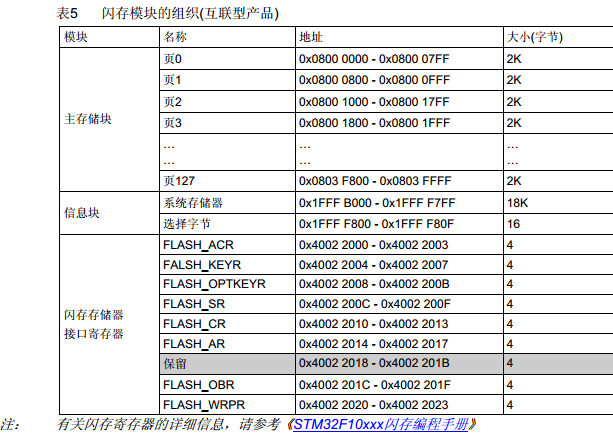
     通常情况下程序也不会把FLASH写满；在没满的时候我们可以把最后一或两页用来模拟EEPROM；这样我们就可以不用在外部另外来加EEPROM了。下面是STM32F103中文手册关于FLASH的截图；由于我用的是STM32C8T6做的实验；手册中没有给STM32C8T6的FLASH的地址信息，容量是64K，可以算出第63页地址是0X800FC00-0X800FFFF。











由于太晚了直接上代码：

头文件：

#ifndef  \_\_ST\_FLASH\_H

#define  \_\_ST\_FLASH\_H

/\*\*@file StFlash.h

  \*@brief stm32 flash的读写操作

  \*    使用：

  \*        \* 此文件主要是针对STM32F103系列的芯片，注意不同容量大小的芯片的地址范围不一样

  \*        \* 使用Read和Write函数进行读写，具体参数和返回值见函数说明

  \*        \* 可以利用flash模拟EEPROM使用

  \*

  \*@author  DHS（746769845@qq.com）

  \*

  \*/

#include "stm32f10x.h"

#include "stm32f10x\_flash.h"

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

//用户根据自己的需要设置

#define STM32\_FLASH\_SIZE  64 //所选STM32的FLASH容量大小(单位为K)

#define STM32\_FLASH\_WREN 1              //使能FLASH写入(0，不是能;1，使能)

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

#if STM32\_FLASH\_SIZE<256

#define STM\_SECTOR\_SIZE 1024 //字节

#else

#define STM\_SECTOR\_SIZE 2048

#endif

//FLASH起始地址

#define STM32\_FLASH\_BASE 0x08000000 //STM32 FLASH的起始地址

//FLASH解锁键值

class STFLASH

{

private:

bool mUseHalfWord;//

uint32\_t mStartAddress;//

public:

STFLASH(uint32\_t startAddress=(0x08000000+1000),bool useHalfWord=true);

//读取指定地址的半字(16位数据)

//faddr:读地址(此地址必须为2的倍数!!)

//返回值:对应数据.

u16 ReadHalfWord(u32 faddr);

//WriteAddr:起始地址

//pBuffer:数据指针

//NumToWrite:半字(16位)数

void Write\_NoCheck(u32 WriteAddr,u16 \*pBuffer,u16 NumToWrite) ;

//从指定地址开始写入指定长度的数据

//WriteAddr:起始地址(此地址必须为2的倍数!!)

//pBuffer:数据指针

//NumToWrite:半字(16位)数(就是要写入的16位数据的个数.)

void Write(u32 WriteAddr,u16 \*pBuffer,u16 NumToWrite);

//从指定地址开始读出指定长度的数据

//ReadAddr:起始地址

//pBuffer:数据指针

//NumToWrite:半字(16位)数

void Read(u32 ReadAddr,u16 \*pBuffer,u16 NumToRead) ;

};

#endif

程序文件：

#include "StFlash.h"

/\*\*@file StFlash.h

\*@brief stm32 flash的读写操作

\* 使用：

\* \* 此文件主要是针对STM32F103系列的芯片，注意不同容量大小的芯片的地址范围不一样

\* \* 使用Read和Write函数进行读写，具体参数和返回值见函数说明

\* \* 可以利用flash模拟EEPROM使用

\*

\*@author DHS（746769845@qq.com）

\*

\*/

u16 STMFLASH\_BUF[STM\_SECTOR\_SIZE/2];//最多是2K字节

STFLASH::STFLASH(uint32\_t startAddress,bool useHalfWord)

{

if(startAddress%STM\_SECTOR\_SIZE!=0)//不是页的开始,将开始处设置为下一个页开始的地方

startAddress+=(STM\_SECTOR\_SIZE-(startAddress%STM\_SECTOR\_SIZE));

mStartAddress=startAddress;

mUseHalfWord=useHalfWord;

}

//读取指定地址的半字(16位数据)

//faddr:读地址(此地址必须为2的倍数!!)

//返回值:对应数据.

u16 STFLASH:: ReadHalfWord(u32 faddr)

{

return \*(vu16\*)faddr;

}

//WriteAddr:起始地址

//pBuffer:数据指针

//NumToWrite:半字(16位)数

void STFLASH::Write\_NoCheck(u32 WriteAddr,u16 \*pBuffer,u16 NumToWrite)

{

u16 i;

for(i=0;i<NumToWrite;i++)

{

FLASH\_ProgramHalfWord(WriteAddr,pBuffer[i]);

WriteAddr+=2;//地址增加2.

}

}

//从指定地址开始写入指定长度的数据

//WriteAddr:起始地址(此地址必须为2的倍数!!)

//pBuffer:数据指针

//NumToWrite:半字(16位)数(就是要写入的16位数据的个数.)

void STFLASH::Write(u32 WriteAddr,u16 \*pBuffer,u16 NumToWrite)

{

u32 secpos; //扇区地址

u16 secoff; //扇区内偏移地址(16位字计算)

u16 secremain; //扇区内剩余地址(16位字计算)

u16 i;

u32 offaddr; //去掉0X08000000后的地址

if(WriteAddr<STM32\_FLASH\_BASE||(WriteAddr>=(STM32\_FLASH\_BASE+1024\*STM32\_FLASH\_SIZE)))return;//非法地址

FLASH\_Unlock(); //解锁

offaddr=WriteAddr-STM32\_FLASH\_BASE; //实际偏移地址.

secpos=offaddr/STM\_SECTOR\_SIZE; //扇区地址 0~127 for STM32F103RBT6

secoff=(offaddr%STM\_SECTOR\_SIZE)/2; //在扇区内的偏移(2个字节为基本单位.)

secremain=STM\_SECTOR\_SIZE/2-secoff; //扇区剩余空间大小

if(NumToWrite<=secremain)secremain=NumToWrite;//不大于该扇区范围

while(1)

{

Read(secpos\*STM\_SECTOR\_SIZE+STM32\_FLASH\_BASE,STMFLASH\_BUF,STM\_SECTOR\_SIZE/2);//读出整个扇区的内容

for(i=0;i<secremain;i++)//校验数据

{

if(STMFLASH\_BUF[secoff+i]!=0XFFFF)break;//需要擦除

}

if(i<secremain)//需要擦除

{

FLASH\_ErasePage(secpos\*STM\_SECTOR\_SIZE+STM32\_FLASH\_BASE);//擦除这个扇区

for(i=0;i<secremain;i++)//复制

{

STMFLASH\_BUF[i+secoff]=pBuffer[i];

}

Write\_NoCheck(secpos\*STM\_SECTOR\_SIZE+STM32\_FLASH\_BASE,STMFLASH\_BUF,STM\_SECTOR\_SIZE/2);//写入整个扇区

}else Write\_NoCheck(WriteAddr,pBuffer,secremain);//写已经擦除了的,直接写入扇区剩余区间.

if(NumToWrite==secremain)break;//写入结束了

else//写入未结束

{

secpos++; //扇区地址增1

secoff=0; //偏移位置为0

pBuffer+=secremain; //指针偏移

WriteAddr+=secremain; //写地址偏移

NumToWrite-=secremain; //字节(16位)数递减

if(NumToWrite>(STM\_SECTOR\_SIZE/2))secremain=STM\_SECTOR\_SIZE/2;//下一个扇区还是写不完

else secremain=NumToWrite;//下一个扇区可以写完了

}

};

FLASH\_Lock();//上锁

}

//从指定地址开始读出指定长度的数据

//ReadAddr:起始地址

//pBuffer:数据指针

//NumToWrite:半字(16位)数

void STFLASH::Read(u32 ReadAddr,u16 \*pBuffer,u16 NumToRead)

{

u16 i;

for(i=0;i<NumToRead;i++)

{

pBuffer[i]=ReadHalfWord(ReadAddr);//读取2个字节.

ReadAddr+=2;//偏移2个字节.

}

}

main文件：

#include "stm32f10x.h"

#include "StFlash.h"/要写入到STM32 FLASH的字符串数组

const u8 TEXT\_Buffer[]={"Flash\_test"};

#define SIZE sizeof(TEXT\_Buffer) //数组长度

#define FLASH\_SAVE\_ADDR 0X0800FC00 //设置FLASH 保存地址(必须为偶数，且其值要大于本代码所占用FLASH的大小+0X08000000)

u8 datatemp[SIZE];

STFLASH flash1;//

int main()

{

SysTick\_Config(72000);

flash1.Write(FLASH\_SAVE\_ADDR,(u16\*)TEXT\_Buffer,SIZE);//写数据，第一次下载程序到32，第二次注释掉此行，断电重新编译下载 //keil watch查看datatemp数组的数据正是之前写进去的数据

flash1.Read(FLASH\_SAVE\_ADDR,(u16\*)datatemp,SIZE);//读数据

while(1) {

}

}