零死角玩转STM32

读写内部FLASH

淘宝：firestm32.taobao.com

论坛： www.firebbs.cn

扫描进入淘宝店铺

主讲内容

01 STM32的内部FLASH简介

02 对内部FLASH的写入过程

03 查看工程的空间分布

04 操作内部FLASH的库函数介绍

05 实验：读写内部FLASH

参考资料:《零死角玩转STM32》

“读写内部FLASH”章节

读写内部FLASH

操作内部FLASH的库函数

为简化编程，STM32标准库提供了一些库函数，它们封装了对内部FLASH写入数据操作寄存器的过程。

1. FLASH解锁、上锁函数

```
1
2  #define FLASH_KEY1               ((uint32_t)0x45670123)
3  #define FLASH_KEY2               ((uint32_t)0xCDEF89AB)
4  /**
5    * @brief  Unlocks the FLASH control register access
6    * @param  None
7    * @retval None
8    */
9  void FLASH Unlock(void)
10 {
11     if ((FLASH->CR & FLASH_CR_LOCK) != RESET) {
12         /* Authorize the FLASH Registers access */
13         FLASH->KEYR = FLASH KEY1;
14         FLASH->KEYR = FLASH_KEY2;
15     }
16 }
17
18 /**
19   * @brief  Locks the FLASH control register access
20   * @param  None
21   * @retval None
22   */
23 void FLASH_Lock(void)
24 {
25     /* Set the LOCK Bit to lock the FLASH Registers access */
26     FLASH->CR |= FLASH_CR_LOCK;
27 }
```

解锁的时候，它对FLASH_KEYR寄存器写入两个解锁参数，上锁的时候，对FLASH_CR寄存器的FLASH_CR_LOCK位置1。

读写内部FLASH

2.设置操作位数及擦除扇区

解锁后擦除扇区时可调用FLASH_EraseSector完成：

```
26 FLASH_Status FLASH_EraseSector(uint32_t FLASH_Sector, uint8_t VoltageRange)
27 {
28     uint32_t tmp_psize = 0x0;
29     FLASH Status status = FLASH COMPLETE;
30
31     /* Check the parameters */
32     assert_param(IS_FLASH_SECTOR(FLASH_Sector));
33     assert param(IS VOLTAGERANGE(VoltageRange));
34
35     if (VoltageRange == VoltageRange 1) {
36         tmp_psize = FLASH_PSIZE_BYTE;
37     } else if (VoltageRange == VoltageRange_2) {
38         tmp_psize = FLASH_PSIZE_HALF_WORD;
39     } else if (VoltageRange == VoltageRange 3) {
40         tmp_psize = FLASH_PSIZE_WORD;
41     } else {
42         tmp_psize = FLASH_PSIZE_DOUBLE_WORD;
43     }
44     /* Wait for last operation to be completed */
45     status = FLASH_WaitForLastOperation();
46
47     if (status == FLASH_COMPLETE) {
48         /* if the previous operation is completed, proceed to erase the sector */
49         FLASH->CR &= CR_PSIZE_MASK;
50         FLASH->CR |= tmp psize;
51         FLASH->CR &= SECTOR MASK;
52         FLASH->CR |= FLASH CR SER | FLASH Sector;
53         FLASH->CR |= FLASH_CR_STRT;
54
55         /* Wait for last operation to be completed */
56         status = FLASH WaitForLastOperation();
57
58         /* if the erase operation is completed, disable the SER Bit */
59         FLASH->CR &= (~FLASH_CR_SER);
60         FLASH->CR &= SECTOR MASK;
61     }
62     /* Return the Erase Status */
63     return status;
64 }
```

该函数包含两个输入参数，分别是要擦除的扇区号和工作电压范围，选择不同电压时实质是选择不同的数据操作位数，参数中可输入的宏在注释里已经给出。函数根据输入参数配置PSIZE位，然后擦除扇区，擦除扇区的时候需要等待一段时间，它使用FLASH_WaitForLastOperation等待，擦除完成的时候才会退出FLASH_EraseSector函数。

读写内部FLASH

3.写入数据

对内部FLASH写入数据不像对SDRAM操作那样直接指针操作就完成了，还要设置一系列的寄存器，利用FLASH_ProgramWord、FLASH_ProgramHalfWord和FLASH_ProgramByte函数可按字、半字及字节单位写入数据：

```
16 FLASH Status FLASH ProgramWord(uint32 t Address, uint32 t Data)
17 {
18     FLASH_Status status = FLASH_COMPLETE;
19
20     /* Check the parameters */
21     assert_param(IS_FLASH_ADDRESS(Address));
22
23     /* Wait for last operation to be completed */
24     status = FLASH WaitForLastOperation();
25
26     if (status == FLASH COMPLETE) {
27/* if the previous operation is completed, proceed to program the new data */
28         FLASH->CR &= CR_PSIZE_MASK;
29         FLASH->CR |= FLASH PSIZE WORD;
30         FLASH->CR |= FLASH_CR_PG;
31
32         *(__IO uint32_t*)Address = Data;
33
34         /* Wait for last operation to be completed */
35         status = FLASH_WaitForLastOperation();
36
37         /* if the program operation is completed, disable the PG Bit */
38         FLASH->CR &= (~FLASH CR PG);
39     }
40     /* Return the Program Status */
41     return status;
42 }
```

读写内部FLASH

3.写入数据

```
16 FLASH Status FLASH ProgramWord(uint32 t Address, uint32 t Data)
17 {
18     FLASH_Status status = FLASH_COMPLETE;
19
20     /* Check the parameters */
21     assert_param(IS_FLASH_ADDRESS(Address));
22
23     /* Wait for last operation to be completed */
24     status = FLASH WaitForLastOperation();
25
26     if (status == FLASH COMPLETE) {
27/* if the previous operation is completed, proceed to program the new data */
28         FLASH->CR &= CR_PSIZE_MASK;
29         FLASH->CR |= FLASH PSIZE WORD;
30         FLASH->CR |= FLASH_CR_PG;
31
32         *(__IO uint32_t*)Address = Data;
33
34         /* Wait for last operation to be completed */
35         status = FLASH_WaitForLastOperation();
36
37         /* if the program operation is completed, disable the PG Bit */
38         FLASH->CR &= (~FLASH_CR_PG);
39     }
40     /* Return the Program Status */
41     return status;
42 }
```

从函数代码可了解到，使用指针进行赋值操作前设置了数据操作宽度，并设置了PG寄存器位，在赋值操作后，调用了FLASH_WaitForLastOperation函数等待写操作完毕。HalfWord和Byte操作宽度的函数执行过程类似。

THANKS

论坛： www.firebbs.cn

淘宝：firestm32.taobao.com

扫描进入淘宝店铺