# 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：

代码清单 44-3 擦除扇区

```
/**
  * @brief  擦除指定的页
  * @param  Page_Address: 要擦除的页地址.
  * @retval FLASH Status:
  *           可能的返回值: FLASH_BUSY, FLASH_ERROR_PG,
  *           FLASH_ERROR_WRP, FLASH_COMPLETE or FLASH_TIMEOUT.
  */
FLASH_Status FLASH_ErasePage(uint32_t Page_Address)
{
    FLASH_Status status = FLASH_COMPLETE;
    /* 检查参数 */
    assert_param(IS_FLASH_ADDRESS(Page_Address));
    /*...此处省略 XL 超大容量芯片的控制部分*/
    /* 等待上一次操作完成 */
    status = FLASH_WaitForLastOperation(EraseTimeout);

    if (status == FLASH_COMPLETE) {
        /* 若上次操作完成，则开始页擦除 */
        FLASH->CR|= CR_PER_Set;
        FLASH->AR = Page_Address;
        FLASH->CR|= CR_STRT_Set;

        /* 等待操作完成 */
        status = FLASH_WaitForLastOperation(EraseTimeout);

        /* 复位 PER 位 */
        FLASH->CR &= CR_PER_Reset;
    }

    /* 返回擦除结果 */
    return status;
}
```

该函数包含以Page_Address输入参数获得要擦除的地址。内部根据该参数配置FLASH_AR地址，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和FLASH_ProgramHalfWord函数可按字、半字节单位写入数据：（见下一页PPT）

# 读写内部FLASH

代码清单 44-4 写入数据

```
/**
  * @brief  向指定的地址写入一个字的数据（32 位）
  * @param  Address: 要写入的地址
  * @param  Data: 要写入的数据
  * @retval FLASH Status:
        可能的返回值: FLASH_ERROR_PG,
  *           FLASH_ERROR_WRP, FLASH_COMPLETE or FLASH_TIMEOUT.
  */
FLASH_Status FLASH_ProgramWord(uint32_t Address, uint32_t Data)
{
    FLASH_Status status = FLASH_COMPLETE;
    __IO uint32_t tmp = 0;

    /* 检查参数 */
    assert_param(IS_FLASH_ADDRESS(Address));
    /*...此处省略 XL 超大容量芯片的控制部分*/
    /* Wait for last operation to be completed */
    status = FLASH_WaitForLastOperation(ProgramTimeout);

    if (status == FLASH_COMPLETE) {
        /* 若上次操作完成，则开始页入低 16 位的数据（输入参数的第 1 部分） */
        FLASH->CR |= CR_PG_Set;

        *(__IO uint16_t*)Address = (uint16_t)Data;
        /* 等待上一次操作完成 */
        status = FLASH_WaitForLastOperation(ProgramTimeout);

        if (status == FLASH_COMPLETE) {
            /* 若上次操作完成，则开始页入高 16 位的数据（输入参数的第 2 部分） */
            tmp = Address + 2;

            *(__IO uint16_t*) tmp = Data >> 16;

            /* 等待操作完成 */
            status = FLASH_WaitForLastOperation(ProgramTimeout);

            /* 复位 PG 位  */
            FLASH->CR &= CR_PG_Reset;
        } else {
            /* 复位 PG 位 */
            FLASH->CR &= CR_PG_Reset;
        }
    }

    /* 返回写入结果 */
    return status;
}
```

从函数代码可了解到，使用指针进行赋值操作前设置了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