

简介

HolyChip 部分Flash MCU采用了Flash作类EEPROM的设计方式，MCU的ROM空间均可以用作类EEPROM数据存储（IAP操作），用户在数据存储时的灵活性大大提升。

在IAP操作时，如果不做好相关的保护措施，可能会造成程序运行异常。本手册主要介绍IAP的使用注意事项。

- 本技术手册适用芯片：HC88F0671、HC88F0661、HC88F0651、HC89S003F4、HC89S105C6、HC89S105C8、HC89S105K6、HC89S105K8、HC89S105AC8、HC89S105AK8、HC89F0411P、HC89F0421、HC89F0431、HC89F0531、HC89F0541、HC89F301、HC89F302、HC89F303、HC89F3421、HC89F3531、HC89F3541。
- 相关数据手册、工具及技术文档下载网址：<http://www.holychip.cn/>。

目录

1、BOR 电压以及 BOR 消抖时间设置	3
1.1 原理说明	3
1.2 BOR 电压设置方式	3
2、Option 设置 IAP 擦写保护设置	4
2.1 原理说明	4
2.2 操作说明	4
3、IAP 操作前判断 MCU 电压	7
4、中断	7
5、双区域保存存储数据	8
5.1 实现原理	8
5.2 操作说明	8
6、IAP 函数绝对地址编译 函数绝对地址编译及指针调用	9
6.1 实现原理	9
6.2 操作说明	9
7、版本说明	11

1、BOR 电压以及 BOR 消抖时间设置

1.1 原理说明

MCU 工作电压为 2.0V-5.5V，如果关闭 BOR 时，MCU 在 1.5V-2.0V 可能会工作异常，程序可能会跑飞从而导致误擦除。如果将 BOR 电压设置为 2.0V 及 2.0V 以上，可以保证 MCU 工作在一个稳定的电压范围内。

1.2 BOR 电压设置方式

BOR 电压检测控制寄存器 BORDC

位编号	7	6	5	4	3	2	1	0
R/W	R/W	R/W	R	R	R	R/W	R/W	R/W
复位值	1	0	0	0	0	0	0	0
位符号	BOREN	BOR_DBC_EN	-			BORVS[2:0]		

位编号	位符号	说明
7	BOREN	BOR 使能位 0: 禁止 BOR 1: 允许 BOR
6	BOR_DBC_EN	BOR 消抖使能位 0: 不使能 1: 使能
5-3	-	保留位（读为 0，写无效）
2-0	BORVS[2:0]	BOR 检测电压点选择位 000: 1.8V 001: 2.0V 010: 2.4V 011: 2.6V 100: 3.0V 101: 3.6V 110: 3.9V 111: 4.2V

BOR 电压检测控制寄存器 BORDC

位编号	7	6	5	4	3	2	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位值	0	0	0	0	0	0	0	0
位符号	BORDBC[7:0]							

位编号	位符号	说明
-----	-----	----

7-0	BORDBC[7:0]	BOR 消抖控制位 消抖时间 = BORDBC[7:0] * 8T _{CPU} + 2 T _{CPU} 注：需要使能 BOR_DBC_EN，否则 BOR 不消抖。
-----	-------------	---

在设置 BOR 电压以及 BOR 消抖时间时，BOR 电压至少设置到 2.0V，在系统允许的情况下，BOR 电压设置的越高越好并且尽量将消抖时间设置较短从而保证 MCU 工作在一个可靠的电压环境内。

BOR 使能后的功耗约为 8uA，如使用者需要低功耗，建议在进入掉电模式之前关闭 BOR，掉电模式唤醒后再使能 BOR，从而达到节省功耗的目的。

2、Option 设置 IAP 擦写保护设置

2.1 原理说明

在 Option 中设置相关的 IAP 擦写保护，使能程序区数据保护位，可以有效的保证程序区数据不会被改写或误擦除。

2.2 操作说明

2.2.1 Keil 操作方式

以 HC88F0671 的“Flash 做类 EEPROM 读写”范例为例。

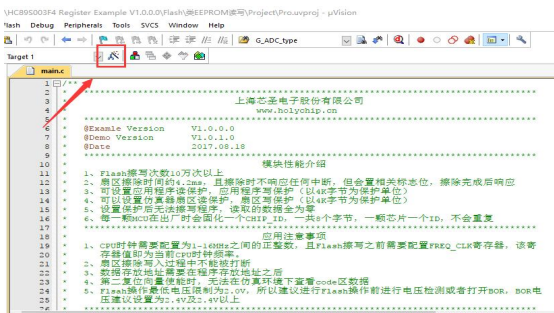


图 2-1

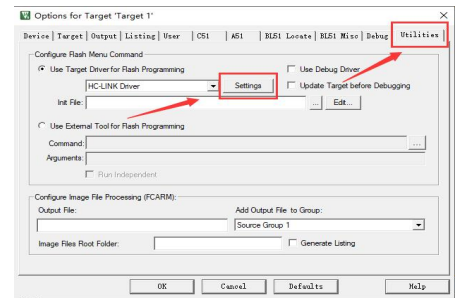


图 2-2

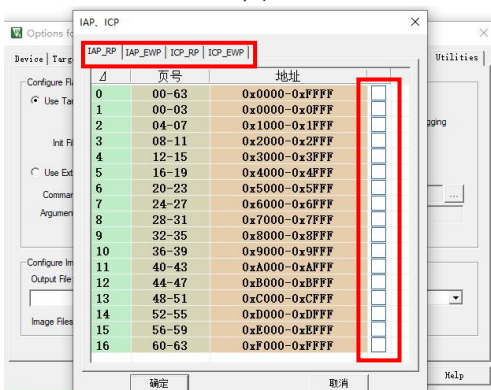


图 2-4

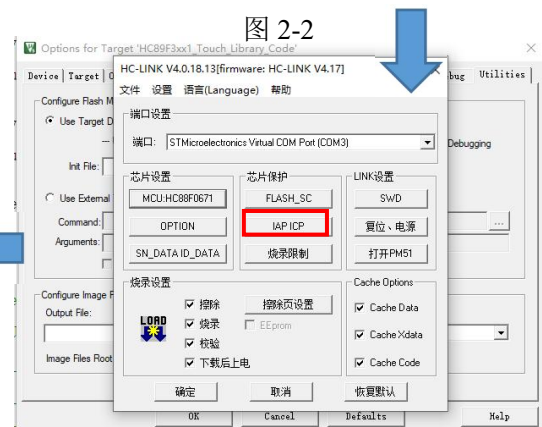


图 2-3

操作步骤:

- 1、KEIL 中打开 Option。(如图 2-1)
- 2、点击 Utilities 选项卡中的 Settings 选项。(如图 2-2)
- 3、点击保护配置选项卡。(如图 2-3)
- 4、根据需求勾选相应的扇区后点击确定。(如图 2-4)

2.2.2 HC-PM51 操作方式

图 2-5

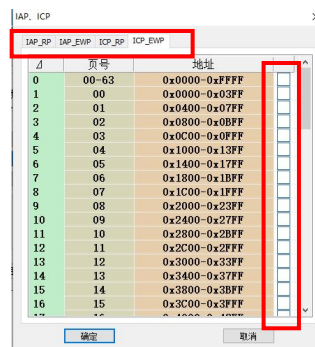


图 2-6

操作步骤:

- 1、选择芯片型号，配置代码选项，加载代码后点击 IAP/ICP 设置选项。(如图 2-5)
- 2、选择保护类型和需要保护的地址。(如图 2-6)

```
Rebuild target 'Target 1'
compiling main.c...
..\user\main.c(106): warning C294: unreachable code
linking...
Program Size: data=28.0 xdata=0 code=585
creating hex file from "Pro"...
"Pro" - 0 Error(s), 1 Warning(s).
Build Time Elapsed: 00:00:03
```

2.2.3、保护区域计算方法

图 2-8

HolyChip Flash 系列 MCU 均为 128 个字节一个扇区， $8 \times 128 = 1K$ 为 1 页。

KEIL 编译后会生成编译字节大小，以 HC88F0671 的“Flash 做类 EEPROM 读写”范例为例，本例程编译的字节为 585 个（如图 2-8），由于 HC88F0671 以 4K 字节为保护为单位，所以在勾选时勾选 0、1、2、3 页（如图 2-5 和图 2-7）。部分 MCU 以 1K 字节为保护单位，用户根据需求选择即可。

3、IAP 操作前判断 MCU 电压

为防止 MCU 上电瞬间电压不稳导致程序跑飞造成误操作，所以建议每次 IAP 操作前采用 ADC 或者 LVD 检测 MCU 当前电压，如果电源低于 2.0V 则不进行 IAP 操作。具体的操作方式详见各 MCU 数据手册及例程。

4、中断

IAP 操作前，建议关闭中断（EA=0），确保在 IAP 操作期间不会被中断影响，待 IAP 操作完成后，再将中断恢复。

5、双区域保存存储数据

5.1 实现原理

在执行 IAP 操作时，不可避免的会遇到数据擦除结束后，尚未写数据就掉电的情况，所以建议采用双区域保存数据的方式，即使一个区域的数据被擦除，也可以保证另一个区域的数据被正常读取。

5.2 操作说明

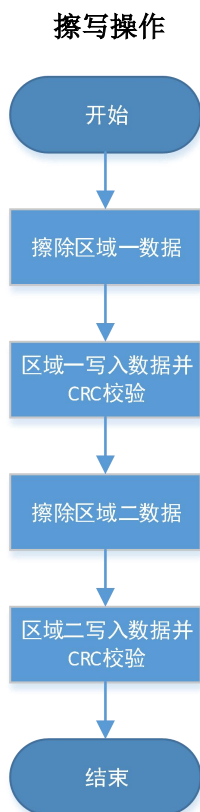


图 5-1

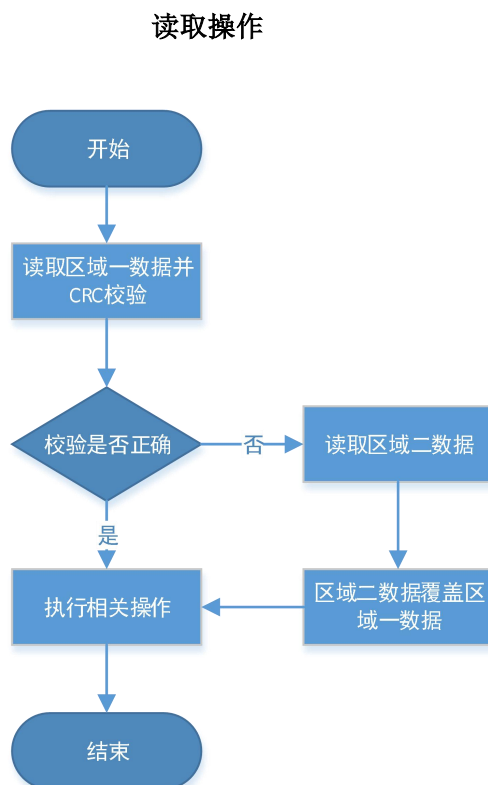


图 5-2

擦写操作：

更新数据时，先更新区域一内数据，更新完成后再更新区域二内数据。（如图 5-1）

读取操作：

读取数据时，先读取区域一内数据，并进行 CRC 校验，如果校验错误，则丢弃区域一的数据，读取区域二内数据。（如图 5-2）

6、IAP 函数绝对地址编译 函数绝对地址编译及指针调用

6.1 实现原理

在 Option 中设置相关的 IAP 擦写保护，使能用户程序所在扇区保护位后，使用绝对地址编译将 IAP 操作函数放到未设置 IAP 保护的区域，并使用函数指针的方式调用函数，保证 MCU 不会误擦除用户代码。

6.2 操作说明

以 HC88F0671 的“Flash 做类 EEPROM 读写”范例为例，

- 1、打开 Flash 做类 EEPROM 读写，在主函数里分别找到 FLASH 擦除(Flash_EraseBlock)和写 (FLASH_WriteData)的函数，打开 Project -> Options for Target 菜单，选中 BL51 Locate 选项卡，在 Code: 中输入: ?PR?_Flash_EraseBlock?MAIN(0X2000),?PR?_FLASH_WriteData?MAIN(0X2080)。格式为: ?PR?_函数名?MAIN(要编译的地址)，注意，输入的文本（包括标点符号）必须是英文格式，否则编译程序会报错，然后点击 OK，程序编译后，在 M51 文件中可以找到相关信息，如图

6-3

所示程序已经被编译在相关地址处。

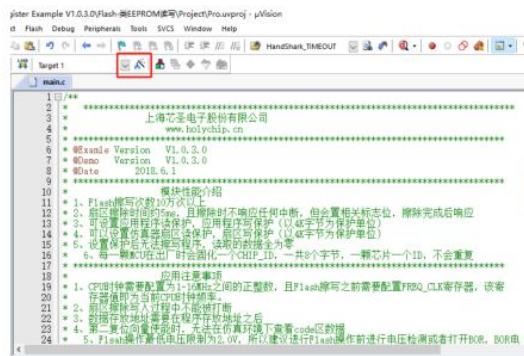


图 6-1

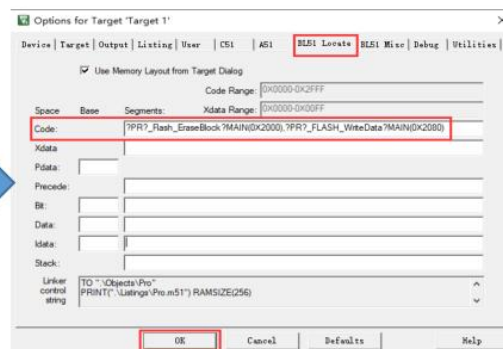


图 6-2

CODE	CODE	MEMORY	
0000H	0003H	ABSOLUTE	
0003H	0019H	UNIT	?C_INITSEG
001CH	0007H		*** GAP ***
0023H	0003H	ABSOLUTE	
0026H	00BAH	UNIT	?PR?MAIN?MAIN
00E0H	008CH	UNIT	?C_C51STARTUP
016CH	0044H	UNIT	?PR?_FLASH_WRITEARR?MAIN
01B0H	0037H	UNIT	?PR?UART1_RPT?MAIN
01E7H	0037H	UNIT	?PR?_FLASH_READARR?MAIN
021EH	0031H	UNIT	?C_LIB_CODE
024FH	1DB1H		*** GAP ***
2000H	0022H	UNIT	?PR?_FLASH_ERASEBLOCK?MAIN
2022H	005EH		*** GAP ***
2080H	0024H	UNIT	?PR?_FLASH_WRITEDATA?MAIN

图 6-3

- 2、使用函数指针调用绝对地址处的函数，详情请参考例程

- (1) 定义函数原型: `void Flash_EraseBlock(unsigned int fui_Address);`
- (2) 定义相应的函数指针变量: `void (*CALL_FLASH_ERASE)(unsigned int i);`
- (3) 函数指针变量赋值, 指向我们定位的绝对地址的函数: `CALL_FLASH_ERASE=Flash_EraseBlock;`
- (4) 函数指针的调用: `(*CALL_FLASH_ERASE)(0x2B00);`

7、版本说明

版本	日期	描述
V1.00	2018/1/25	初版
V1.01	2018/6/26	1、 增加相关型号 2、 修改部分错误

HOLYCHIP 公司保留对以下所有产品在可靠性、功能和设计方面的改进作进一步说明的权利。

HOLYCHIP 不承担由本手册所涉及的产品或电路的运用和使用所引起的任何责任，HOLYCHIP 的产品不是专门设计来应用于外科植入、生命维持和任何 HOLYCHIP 产品产生的故障会对个体造成伤害甚至死亡的领域。如果将 HOLYCHIP 的产品用于上述领域，即使这些是由 HOLYCHIP 在产品设计和制造上的疏忽引起的，用户应赔偿所有费用、损失、合理的人身伤害或死亡所直接或间接所产生的律师费用，并且用户保证 HOLYCHIP 及其雇员、子公司、分支机构和销售商与上述事宜无关。

芯圣电子
2018 年 6 月