The following document contains information on Cypress products. The document has the series name, product name, and ordering part numbering with the prefix “MB”. However, Cypress will offer these products to new and existing customers with the series name, product name, and ordering part number with the prefix “CY”.

How to Check the Ordering Part Number

1. Go to www.cypress.com/pcn.
2. Enter the keyword (for example, ordering part number) in the **SEARCH PCNS** field and click **Apply**.
3. Click the corresponding title from the search results.
4. Download the Affected Parts List file, which has details of all changes

For More Information

Please contact your local sales office for additional information about Cypress products and solutions.

About Cypress

Cypress is the leader in advanced embedded system solutions for the world's most innovative automotive, industrial, smart home appliances, consumer electronics and medical products. Cypress' microcontrollers, analog ICs, wireless and USB-based connectivity solutions and reliable, high-performance memories help engineers design differentiated products and get them to market first. Cypress is committed to providing customers with the best support and development resources on the planet enabling them to disrupt markets by creating new product categories in record time. To learn more, go to www.cypress.com.

AN205292

FM3 MB9A310K/110K 微型控制器, 与 EEPROM TYPE A

本应用笔记主要介绍 FM3 MB9A310K/110K 系列芯片仿真 EEPROM 的使用。

目录

1 概要 1
1.1 目的....1
1.2 术语及缩略语 1
1.3 章节概要....1
2 系统硬件环境 2
3 开发工具 2
4 仿真 EEPROM 函数....2
4.1 宏 2
4.2 函数列表....4
4.3 函数原型....4
5 中断函数 7
5.1 函数列表....7
5.2 函数原型....8
6 库函数概要 8
6.1 芯片的使用情况....8
6.2 ROM 的使用....10
6.3 EEPROM 性能比较 11
6.4 仿真 EEPROM 性能的介绍 11
7 使用的样例 12
7.1 推荐的初始化处理 12
7.2 库调用实例....13
8 使用注意事项....14
9 附加信息 14
修改记录....15
全球销售和设计支持....16

1 概要

1.1 目的

本应用笔记主要介绍 FM3 MB9A310K/110K 系列芯片仿真 EEPROM 的使用。

其中主要介绍五个函数的使用情况，返回值的意义，执行的状态以及操作的例子。

1.2 术语及缩略语

Type A: FM3 MB9A310K/110K series MCU

1.3 章节概要

本文档主要包括以下几个章节：

第二章阐述了系统使用的硬件环境

第三章介绍了开发时使用的软，硬件工具

第四章介绍了仿真 EEPROM 提供给用户的所有函数

第五章介绍了仿真 EEPROM 用到的中断函数

第六章介绍了仿真 EEPROM 的使用条件和占用的资源

第七章介绍了怎样使用仿真 EEPROM 的函数

第八章介绍了使用仿真 EEPROM 过程中的注意事项

2 系统硬件环境

本章介绍仿真 EEPROM 的硬件环境和占用资源

- 芯片 CPU：Cypress MB9A310K/110K 系列芯片
- CPU 频率：40MHz
- 最小指令周期：25ns
- 占用 RAM：2.82K 字节
- 代码占用空间：6.612K 字节

3 开发工具

本章介绍系统开发时用到的工具

表 1. 系统开发环境

名字	描述	型号	厂商	备注
IAR	PC 软件	embedded workbench for ARM, version:6.21.1.2846		
J-LINK	仿真工具		SEGGER	

4 仿真 EEPROM 函数

本节介绍仿真 EEPROM 的所有函数。

4.1 宏

4.1.1 函数执行状态宏

函数执行状态宏用于反应函数执行的状态，表 1 详细列出了所有的函数执行的状态：

表 2. 函数执行状态

名字	描述	代码中定义的值
OK	函数执行正常	0
NG	函数执行错误（每个函数的原因将在每个函数详细描述）	1
BUSY	另一个函数正在执行，导致函数无法执行	2
PARA_ERROR	函数输入的地址或大小错误（具体会在每个函数详细描述）	4
SYSTBUSY	函数执行太频繁，系统忙	5

在函数执行时会返回不同的状态，针对不同的状态用户应该做什么，下面的文档给出了一些建议供用户参考：

OK ---- 函数执行很成功

NG ---- 用户可能需要重新调用 check 函数初始化一下

BUSY ---- 其他程序正在执行，可以继续执行直到返回 OK

PARA_ERROR ---- 输入的地址超过了定义值或则输入的大小不在定义列表

SYSTBUSY ---- 系统正在擦除，等待 10ms 后再执行

备注：只有写和读来个操作会返回 SYSTBUSY。

4.1.2 Check 函数检测结果

Check 函数会检测 EEPROM 的状态，然后返回相应的值。

表 3. Check 函数检测结果

名字	描述	代码中定义的值
CONSISTENT	EEPROM 状态正常，可以正常使用	0
DEFINE	EEPROM 还没有被定义	1
REDEFINE	EEPROM 的标志丢失，需要重新定义	2
RESTORE	EEPROM 的状态丢失，需要恢复	3

检测出的 EEPROM 的状态存放在这四个状态变量里，不同的变量代表 EEPROM 需要处理的状态不同；接下来的文档介绍在不同的状态时用户需要执行的操作：

CONSISTENT ---- EEPROM 状态良好，用户调用需要的函数即可

DEFINE ---- EEPROM 需要被调用，请调用函数 EEPROM_Define(uint8_t ucSize)

REDEFINE ---- EEPORM 损坏的数据不能恢复需要再定义，请调用函数 EEPROM_Define(uint8_t ucSize)

RESTORE ---- EEPROM 状态丢失，请调用函数 EEPROM_Restore ()

关于详细的调用情况请参考图 9。

4.1.3 EEPROM 大小

EEPROM 的大小由用户定义（128 字节到 2048 字节），详细的大小列表请参见下面的表格：

表 4. EEPROM 大小列表

名字	描述	代码中定义的值
E2P_128B	EEPROM 最大可以存放 128 个字节	0x08
E2P_256B	EEPROM 最大可以存放 256 个字节	0x10
E2P_512B	EEPROM 最大可以存放 512 个字节	0x20
E2P_1024B	EEPROM 最大可以存放 1024 个字节	0x40
E2P_2048B	EEPROM 最大可以存放 2048 个字节	0x80

不同大小的 EEPROM，用户可以访问的地址不相同。接下来的文档列出了不同大小的 EERPOM，用户可以访问的地址范围：

128 字节 ---- 0 到 127

256 字节 ---- 0 到 255

512 字节 ---- 0 到 511

1024 字节 ---- 0 到 1023

2048 字节 ---- 0 到 2047

备注：当用户定义了 EEPROM 的大小，用户能使用的地址已被设定，如果用户调用了超过范围的地址系统会该你参数错误的反馈（PARA_ERROR）. 在使用中请用户参照上面的范围使用。

4.2 函数列表

本节主要介绍仿真 EEPROM 的所有函数以及每个函数的意思。具体参见下表

函数原型	描述	备注
uint8_t EEPROM_Define(uint8_t ucSize)	在定义大小规定的范围内定义一个 EEPROM	-
uint8_t EEPROM_B_Write(uint16_t WtAdrr, uint8_t WDat)	写一个字节的数据到 EEPROM 里，当 flash 的一个 sector 被写满时数据会被搬运到下个 sector；而且旧的 sector 会被擦除	-
uint8_t EEPROM_B_Read(uint16_t RAddr, uint8_t *ucData)	从 EEPROM 里读取一个字节的数据	-
uint8_t EEPROM_Check(uint8_t ucSize, uint8_t *ucData)	检测 EEPROM 里存放数据的空间是否正常以及存在 RAM 里的 EEPROM 的系统标志是否正常。 返回的状态量表明：没有被使用的空间被占用空白了，空间及状态都正常，被损坏的空间能被恢复或被损坏的空间不能被恢复 当每次上电或复位时，这个函数都需要被调用。从而存在 RAM 里的系统标志等也同时被初始化。	在使用 EEPROM 函数前，请先调用这个函数
uint8_t EEPROM_Restore (void)	恢复 EEPROM 包括恢复存在 RAM 里的系统标志，这个函数用于恢复由于突发状况将 EEPROM 破坏掉	-

4.3 函数原型

4.3.1 EEPROM_Define 函数

这个函数定义仿真 EEPROM。

函数	uint8_t EEPROM_Define(uint8_t ucSize)
参数	ucSize: EEPROM 的大小, 参考表 4
返回值	返回函数执行的状态, 参考表 2
描述	定义一个仿真EEPROM及它的大小
备注	EEPROM大小被定义后操作的EEPROM地址不能超过定义的大小

这个函数定义了 EEPROM 的状态和大小，函数的返回值表示函数执行的状态见下表：

表 5. 定义函数的返回值

返回状态值	描述	应对方法
OK	定义操作成功	N/A
NG	EEPROM 占据的 flash 的操作超时或先前执行的函数对 flash 的操作超时	等待一会再调用
BUSY	更高优先级的函数正在执行	等待一会再调用
PARA_ERROR	定义的大小值不是规定的值	运用 EEPROM 大小列表里的大小值，参考表 4

对于这个函数更详细的使用，请参考图 9.

4.3.2 EEPROM_B_Write 函数

这个函数写一个字节的数据到仿真 EEPROM 的地址上。

函数	uint8_t EEPROM_B_Write(uint16_t WtAdrr, uint8_t WDat)
参数	WtAdrr: 写一个字节的数据到定义的大小内的地址上(小于或等于定义的 EEPROM 大小) WDat: 被写的数据
返回值	返回函数执行的状态，参考表2
描述	存储一个字节的数据到仿真EEPROM
备注	N/A

函数执行的状态会通过返回值返回给用户，具体的返回值代表的意思请参考下表：

表 6.写函数的返回值

返回状态值	描述	应对方法
OK	写函数执行成功	N/A
NG	读错误或 flash 操作超时	再调用一个 check 函数（初始化步骤）
BUSY	检测，定义或恢复函数正在执行	等待一会再调用
PARA_ERROR	输入的 EEPROM 地址超过了定义的 EEPROM 大小	修改写的地址到规定的范围之内
SYSTBUSY	Flash 正在被擦除时持续不停的写会影响擦除	延迟 10 毫秒后再写

接下来是一些使用写函数的样列：

图 1. 连续写的样例

```
for(j=0;j<486;j++)     //486 is a sample, any value that in range is ok
{
    i = EEPROM_B_Write(j,j);
    if(i == SYSTBUSY)
    {
      Delay(200);      //10ms
      i = EEPROM_B_Write(j,j);
    }
}
```

图 2. 单个写的样例

```
i = EEPROM_B_Write(0x21,0x55);     //write 0x55 to address 0x21
```

备注：返回值“i”反应操作状态，关于更详细的描述请参考表 2.

4.3.3 EEPROM_B_Read 函数

读函数是从 EEPROM 里读出一个字节的数据。

函数	uint8_t EEPROM_B_Read(uint32_t RAddr,unsigned char *ucData)
参数	RAddr: 在定义范围内的仿真 EEPROM 的地址(小于等于定义的 EEPROM 大小) *ucData: 读出来的数据
返回值	函数执行的返回值，参考表 2
描述	从 EEPROM 里读出一个字节的数据
备注	N/A

下表详细描述了读函数的执行状态：

表 7. 写函数的返回值

返回状态值	描述	应对方法
OK	读操作成功	N/A
NG	读错误或对 flash 的操作超时	调用 check 函数（重新初始化一次）
BUSY	检测，定义或恢复函数正在执行	等待一会再调用函数
PARA_ERROR	输入的地址超过了定义的 EEPROM 的大小	修改读的地址让其在定义的大小范围之内
SYSTBUSY	Flash 正在擦除时持续不停的读会影响擦除操作	延时 10 毫秒后再读

下图是调用读函数的样例：

图 3. 读函数的样例

```
unsigned char ReadDat;              //save read data

i = EEPROM_B_Read(250,&ReadDat);    //250 is sample value
```

备注：读的地址要在 EEPROM_Define （）函数定义的 EEPROM 的大小范围之内。

4.3.4 EEPROM_Check 函数

这个函数检查 EEPROM 的状态和初始化储存在 RAM 里的全局变量。

函数	uint8_t EEPROM_Check(uint8_t ucSize, uint8_t *ucData)
参数	ucSize: 用户打算定义的 EEPROM 的大小，参考表 4 *ucData: 返回的 EEPROM 的状态，参考表 3
返回值	函数执行的结果，参考表 2
描述	检查 EEPROM 有没有被定义，数据被破坏了能不能恢复等
备注	N/A

Check 函数检测 EEPROM 的状态。函数执行的结果如下表：

表 8. 检测函数的返回值

返回的状态	描述	应对方法
OK	检测函数执行成功	N/A
NG	Flash 的执行超时	等待一会再调用函数
BUSY	系统正在执行其他的函数	等待一会再调用函数
PARA_ERROR	输入的大小值不在规定的列表内，具体参考表 4	输入在列表范围内的 EEPROM 大小值

备注：用户需要根据检测的结果来调用相应的函数。具体的使用请参考图 9.

4.3.5 EEPROM_Restore 函数

这个函数恢复 EEPROM 系统标志和状态量。

函数	uint8_t EEPROM_Restore (void)
参数	无
返回值	返回函数执行的结果，具体的返回值请参考表 2
描述	恢复 EEPROM 的状态和被破坏的存储空间
备注	恢复EEPROM存在RAM的全局变量，EEPROM内部储存数据的空间以及EERPOM的状态

函数恢复由异常情况导致的 EPROM 数据丢失及状态被破坏。下表描述函数的执行状态：

表 9. 恢复函数的返回值

返回的状态值	描述	应对的方法
OK	恢复执行成功	N/A
NG	数据不能被恢复，现在执行的或先前执行的 flash 操作超时	再调用一次 check 函数
BUSY	系统正在调用其他的函数	等待一会再调用函数

对于详细的使用情况请参考图 9.

5 中断函数

5.1 函数列表

函数	描述	备注
void BT0_67_IRQHandler(void)	定时器定时检查 flash 的寄存器来判断 flash 的擦除是否执行完了	N/A

5.2 函数原型

5.2.1 BT0_67_IRQHandler 函数

这个函数是用于定时的去检查 flash 的擦除是否完成了。

函数	void BT0_67_IRQHandler(void)
参数	无
返回值	无
描述	这个中断函数用户只需要放在系统的 base timer 的中断函数里就可以了
备注	在调用整个EEPROM函数时请不要关掉base timer的中断

备注：这个函数被 BT0_7_IRQHandler ()调用从而实现 EEPROM timer 中断函数的使能。详细的使用情况请参考图 8.

6 库函数概要

6.1 芯片的使用情况

这个 EEPROM 的函数库能被使用在拥有两个独立 flash 的 FM3 芯片上，具体会使用到的芯片的特性如下：

- Work flash 拥有的 sector 中包含 SA2（范围在 0x200c4000 到 0x200c5fff 即可）
- Work flash 拥有的 sector 中包含 SA3（范围在 0x200c6000 到 0x200c7fff 即可），参考图 4
- 拥有 base timer 且拥有通道 6 和 7（寄存器参考图 5）
- Work flash 的芯片操作算法如图 6

下图展示了 work flash 中被用到的 sector，work flash 的自动算法和用到 base timer 的寄存器：

图 4. EEPROM 用到的 sectors

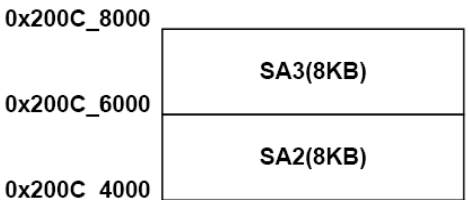


图 5. Base Timer 的寄存器

■ **Timer Control Register (High-order bytes of TMCR)**

bit	15	14	13	12	11	10	9	8
Field	res	CKS2	CKS1	CKS0	res		EGS1	EGS0
Attribute	R/W	R/W	R/W	R/W	R/W		R/W	R/W
Initial value	0	0	0	0	0b00		0	0

■ **Timer Control Register 2 (Low-order bytes of TMCR)**

bit	7	6	5	4	3	2	1	0
Field	T32	FMD2	FMD1	FMD0	OSEL	MDSE	CTEN	STRG
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0

■ **Status Control Register (STC)**

bit	7	6	5	4	3	2	1	0
Field	res	TGIE	res	UDIE	res	TGIR	res	UDIR
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0

bit	15	0
Field	PCSR [15:0]	
Attribute	R/W	
Initial value	0xXXXX	

■ **Register configuration**

bit	15	14	13	12	11	10	9	8
Field	SEL67_3	SEL67_2	SEL67_1	SEL67_0	SEL45_3	SEL45_2	SEL45_1	SEL45_0
Attribute	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0

图 6. Work Flash 的算法

Command	Number of writes	1st write		2nd write		3rd write		4th write		5th write		6th write	
		Address	Data	Address	Data	Address	Data	Address	Data	Address	Data	Address	Data
Read/ Reset	1	0xXXX	0xF0	--	--	--	--	--	--	--	--	--	--
Write	4	0xAA8	0xAA	0x554	0x55	0xAA8	0xA0	PA	PD	--	--	--	--
Flash erase	6	0xAA8	0xAA	0x554	0x55	0xAA8	0x80	0xAA8	0xAA	0x554	0x55	0xAA8	0x10
Sector erase	6	0xAA8	0xAA	0x554	0x55	0xAA8	0x80	0xAA8	0xAA	0x554	0x55	SA	0x30
Sector erase suspended	1	0xXXX	0xB0	--	--	--	--	--	--	--	--	--	--
Sector erase restarting	1	0xXXX	0x30	--	--	--	--	--	--	--	--	--	--

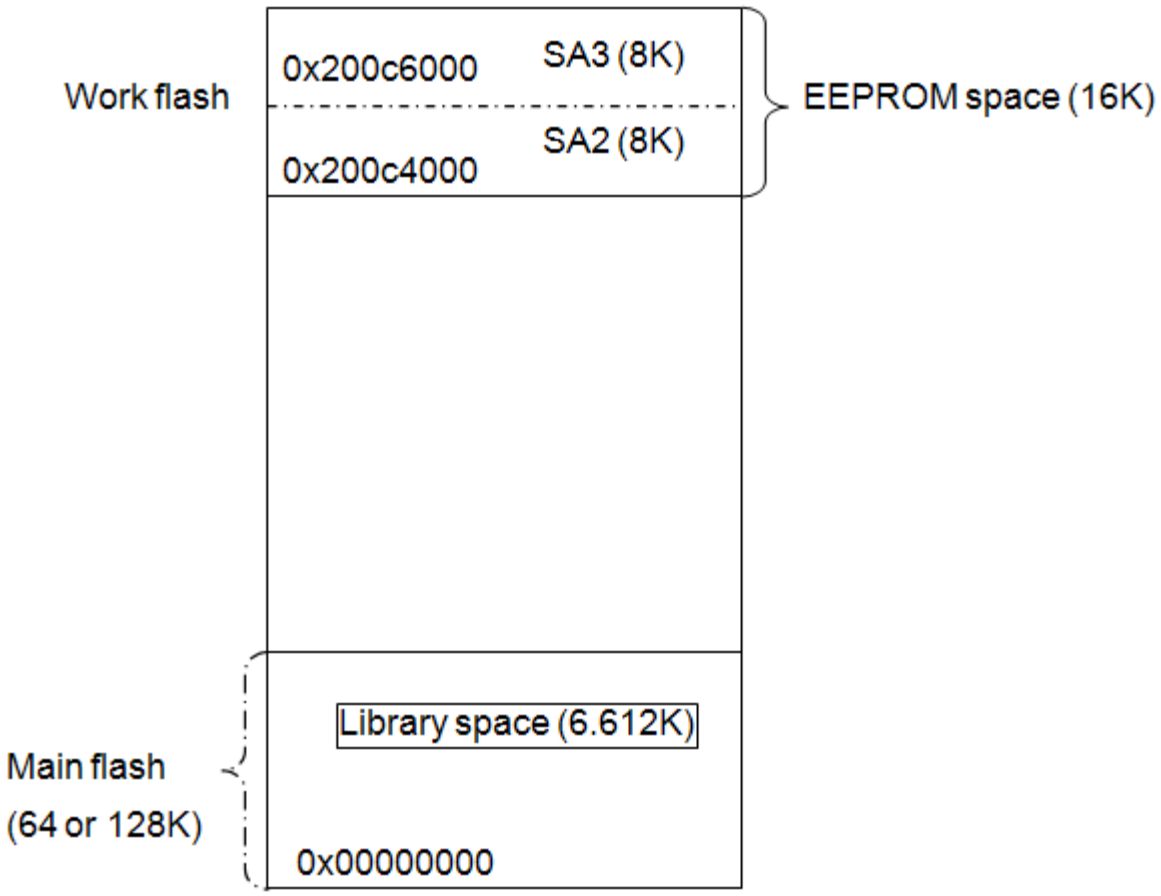
X: Any value
PA: Write address
SA: Sector address (Specify any address within the address range of the sector to erase)
PD: Write data

MB9A310K/110K 系列 FM3 芯片（类型 5）完全满足以上的特性. 关于更详细的这些特性的资料，请参考 MB9A310K 或 MB9A110K 的 flash programming 手册和外围手册.

备注：这个 EEPROM 库完全占用了 work flash 的两个 sector（0x200c4000 到 0x200c7fff），当用户在使用这个库函数时请不要在用到这些地址的 flash。

6.2 ROM 的使用

ROM 描述了库函数使用到的芯片的空间，如下图：



库函数的代码占用的 flash 大小是 6.612K，所在的位置由用户的系统自动分配。其中 EEPROM 占用了 16K 的 work flash。

6.3 EEPROM 性能比较

下表列出了传统 I2C EERPOM 与仿真 EEPROM 性能上的比较：

表 10. 仿真 EEPROM 的性能

EEPROM 性能比较							
类型		传统 I2C	仿真 EEPROM 版本 1.0.0				
最大的时钟频率		100KHz	MCU @40MHz				
大小		-	2048B	1024B	512B	256B	128B
单字节写	正常	100us	76us	76us	76us	76us	76us
	最大	-	10ms	5.5ms	2.6ms	1ms	1ms
持续写 189 个字节	正常	13ms	23.8ms	23.8ms	16.9ms	15.5ms	15.1ms
	最大	-	33.3ms	28.1ms	20.1ms	19.1ms	19ms
持续写 377 个字节	正常	21ms	47.1 ms	41.9ms	33.1ms	31.7ms	31.4ms
	最大	-	451ms	258ms	36ms	32.7ms	32.4ms
单字节读	正常	200us	8.2us	8.2us	8.2us	8.2us	8.2us
	最大	-	-	-	-	-	-

6.4 仿真 EEPROM 性能的介绍

这个性能的测量和计算是在正常环境下进行的，在不同的环境下（电压，温度…）这些值有可能会更大或更小。

对于上个章节图表中说的“正常”是指写操作的平均值（在室温下）

对于上个章节图表中说的“最大”是指持续的写导致了模块的搬运和 sector 的擦除

比较的结果是仿真 EEPROM 会比传统 I2C 更快些。

7 使用的样例

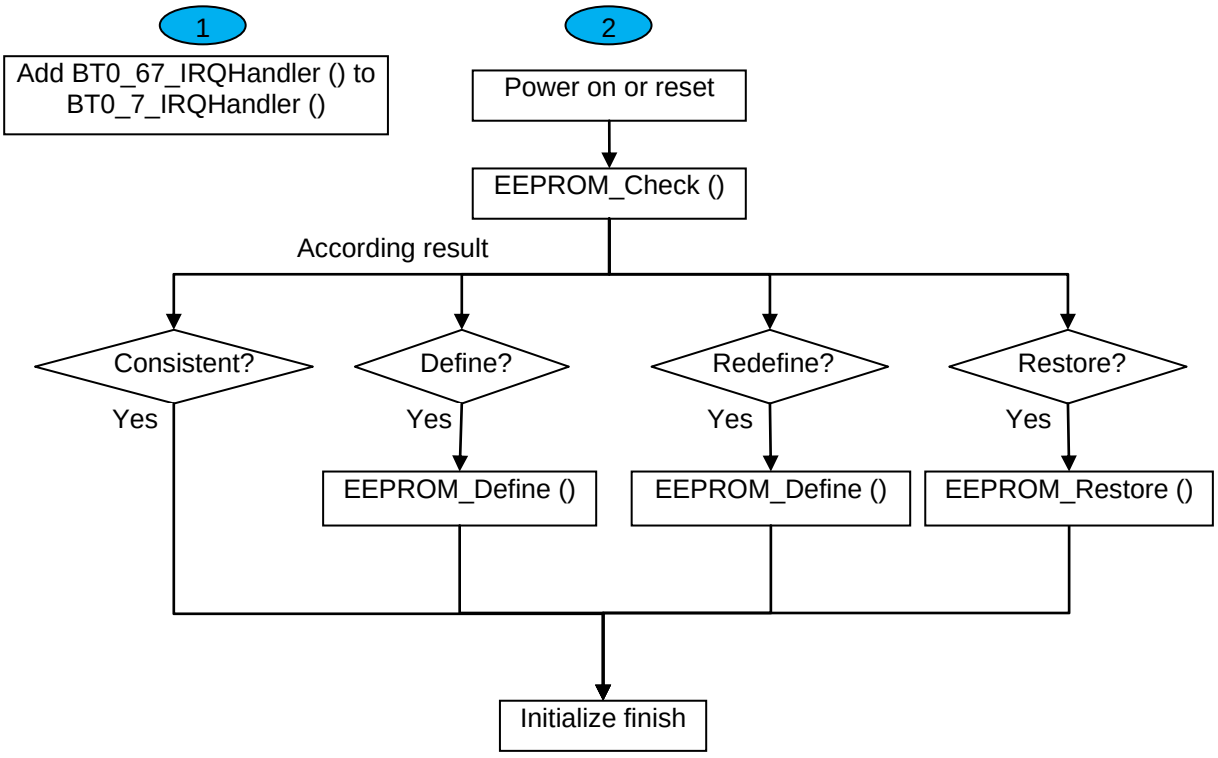
7.1 推荐的初始化处理

每次上电或系统复位，用户都需要调用 Check 函数对 EEPROM 进行初始化。Check 函数是唯一通过软件打开仿真 EEPROM 的方式。

基于 check 的结果，如果返回的结果不是“consistent”，用户需要调用“define”或“restore”函数对 EEPROM 进行定义或恢复。

下图大致描述了 EEPROM 的初始化流程：

图 7. 初始化的流程



下图展示了增加 timer 中断到系统中断函数的处理方法：

图 8. 中断调用例子

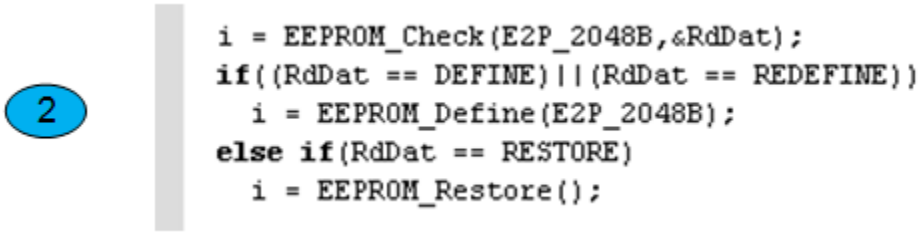
1

```
void BT0_7_IRQHandler(void)
{

  if(bFM3_BT6_RT_STC_UDIR)
    BT0_67_IRQHandler();
}
```

下图展示了步骤 2，EEPROM 初始化的操作例子（check 函数的调用和处理）：

图 9. 初始化例子



7.2 库调用实例

在这个库里面有一个.a 和.h 文件。用户使用的时候需要将这两个文件放到用户的工程里。接下来的步骤会详细介绍怎样使用这个库：

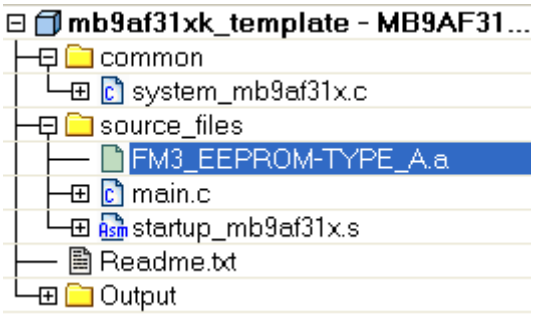
- 增加“FM3_EEPROM-TYPE_A.a”和“FM3_EEPROM-TYPE_A.h”文件到用户的工程文件夹里，如下图：

图 10. 增加库文件



- 增加“FM3_EEPROM-TYPE_A.a”文件到用户的工程里，如下图：

图 11：增加库到工程



- 增加中断函数到工程里，请参考图 8
- 初始化 EEPROM，请参考图 9
- 增加“#include FM3_EEPROM-TYPE_A.h”文件到 main.c 里
- 在用户需要的地方调用需要的库

8 使用注意事项

- 在使用库之前，请先初始化 EEPROM
- 当调用了 define 函数后，请不要立即调用 check 函数
- 如果一个 FM3 芯片和 MB9A310K 一样拥有相同的 work flash 和 timer，这个库函数可以在这款芯片上使用
- 芯片的 base timer 通道 6 和 7 已经被库函数使用了，用户不能在使用库函数的工程里再使用这两个通道的 timer 了
- 如果用户使用了 EEPROM 库，请不要关掉 base timer 的中断
- 芯片的 sector2 和 3 已经被使用了，如果用户在使用仿真 EEPROM 时，不要再使用这两个 sector。Sector 的具体情况请参考图 4
- 如果用户在使用仿真 EEPROM 进行写，擦除或 suspend 的同时又对 work flash 的其他 sector 进行了写，擦除或 suspend 的操作，有可能会引起执行不成功。请不要同时操作

9 附加信息

关于 Cypress 半导体更多的产品信息，请访问以下网站：

http://www.cypress.com/cypress-microcontrollers

http://www.cypress.com/cypress-mcu-product-softwareexamples

修改记录

文档标题：AN205292 - FM3 MB9A310K/110K 微型控制器, 与 EEPROM TYPE A

文档编号：002-05743

修订版	ECN	变更者	提交日期	变更说明
**	-	HUAL	08/09/2012	初稿
*A	5591350	HUAL	01/19/2017	将 Spansion 应用手册“MCU-AN-510060-C-10”转换为 Cypress 格式。

全球销售和设计支持

赛普拉斯公司拥有一个由办事处、解决方案中心、厂商代表和经销商组成的全球性网络。如果想要查找离您最近的办事处，请访问赛普拉斯所在地。

产品

ARM® Cortex®微控制器	cypress.com/arm
汽车级产品	cypress.com/automotive
时钟与缓冲器	cypress.com/clocks
接口	cypress.com/interface
物联网	cypress.com/iot
存储器	cypress.com/memory
微控制器	cypress.com/mcu
PSoC	cypress.com/psoc
电源管理 IC	cypress.com/pmic
触摸感应	cypress.com/touch
USB 控制器	cypress.com/usb
无线连接	cypress.com/wireless

PSoC® 解决方案

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP

赛普拉斯开发者社区

论坛 | WICED IoT 论坛| 项目 | 视频 | 博客 | 培训 | 组件

技术支持

cypress.com/support

PSoC 是赛普拉斯半导体公司的注册商标。PSoC Creator 是赛普拉斯半导体公司的商标。 此处引用的所有其他商标或注册商标都归其各自所有者所有。

赛普拉斯半导体
198 Champion Court
San Jose, CA 95134-1709

电话 :408-943-2600
传真 :408-943-4730
网站地址 :www.cypress.com

©赛普拉斯半导体公司，2012-2017 年。本文件是赛普拉斯半导体公司及其子公司，包括 Spansion LLC（“赛普拉斯”）的财产。本文件，包括其包含或引用的任何软件或固件（“软件”），根据全球范围内的知识产权法律以及美国与其他国家签署条约由赛普拉斯所有。除非在本款中另有明确规定，赛普拉斯保留在该等法律和条约下的所有权利，且未就其专利、版权、商标或其他知识产权授予任何许可。如果软件并不附随有一份许可协议且贵方未以其他方式与赛普拉斯签署关于使用软件的书面协议，赛普拉斯特此授予贵方属人性质的、非独家且不可转让的如下许可（无再许可权）（1）在赛普拉斯特软件著作权项下的下列许可权（一）对以源代码形式提供的软件，仅出于在赛普拉斯硬件产品上使用之目的且仅在贵方集团内部修改和复制软件，和（二）仅限于在有关赛普拉斯硬件产品上使用之目的将软件以二进制代码形式的向外部最终用户提供（无论直接提供或通过经销商和分销商间接提供），和（2）在被软件（由赛普拉斯公司提供，且未经修改）侵犯的赛普拉斯专利的权利主张项下，仅出于在赛普拉斯硬件产品上使用之目的制造、使用、提供和进口软件的许可。禁止对软件的任何其他使用、复制、修改、翻译或汇编。

在适用法律允许的限度内，赛普拉斯未对本文件或任何软件作出任何明示或暗示的担保，包括但不限于关于适销性和特定用途的默示保证。赛普拉斯保留更改本文件的权利，届时将不另行通知。在适用法律允许的限度内，赛普拉斯不对因应用或使用本文件所述任何产品或电路引起的任何后果负责。本文件，包括任何样本设计信息或程序代码信息，仅为供参考之目的提供。文件使用人应负责正确设计、计划和测试信息应用和由此生产的任何产品的功能和安全性。赛普拉斯产品不应被设计为、设定为或授权用作武器操作、武器系统、核设施、生命支持设备或系统、其他医疗设备或系统（包括急救设备和手术植入物）、污染控制或有害物质管理系统中的关键部件，或产品植入之设备或系统故障可能导致人身伤害、死亡或财产损失其他用途（“非预期用途”）。关键部件指，若该部件发生故障，经合理预期会导致设备或系统故障或会影响设备或系统安全性和有效性的部件。针对由赛普拉斯产品非预期用途产生或相关的任何主张、费用、损失和其他责任，赛普拉斯不承担全部或部分责任且贵方不应追究赛普拉斯之责任。贵方应赔偿赛普拉斯因赛普拉斯产品任何非预期用途产生或相关的所有索赔、费用、损失和其他责任，包括因人身伤害或死亡引起的主张，并使之免受损失。

赛普拉斯、赛普拉斯徽标、Spansion、Spansion 徽标，及上述项目的组合，WICED，及 PSoC、CapSense、EZ-USB、F-RAM 和 Traveo 应视为赛普拉斯在美国和其他国家的商标或注册商标。请访问 cypress.com 获取赛普拉斯商标的完整列表。其他名称和品牌可能由其各自所有者主张为该方财产。