KF8F3132——Flash 读写样例程序

引言

本应用笔记提供了 KF8F3132—Flash 读写相关的配置信息以及如何能够快速的理解并上手使用该模块的一些配置方式。

本应用笔记须与 KF8F3132 数据手册结合使用。

寄存器

寄存器使用说明：

OSCCTL（系统控制寄存器）

寄存器OSCCTL：系统频率控制寄存器(地址:2FH)

	bit7							bit0
复位值 0011 ----	CLKOE	IRCS2	IRCS1	IRCS0	-	-	-	-
	R/W	R/W	R/W	R/W	U	U	U	U

图注：R＝可读　W＝可写　P＝可编程　U＝未使用
-＝读为0　x＝状态未知

TR0（P0 方向控制寄存器）

寄存器TR0：P0口方向控制寄存器(地址：25H)

	bit7							bit0
复位值 --11 1111	-	-	TR05	TR04	TR03	TR02	TR01	TR00
	U	U	R/W	R/W	R/W	R/W	R/W	R/W

TR1（P1 口方向控制寄存器）

TR1：P1口方向控制寄存器(地址：27H)

	bit7							bit0
复位值 --11 1111	-	-	TR15	TR14	TR13	TR12	TR11	TR10
	U	U	R/W	R/W	R/W	R/W	R/W	R/W

TR2（P2 口方向控制寄存器）

寄存器2.13： TR2: P2口方向控制寄存器(地址: 26H)

	bit7							bit0
复位值 1111 1111	-	-	-	-	TR23	TR22	TR21	TR20
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P0LR（P0 口输出锁存控制寄存器）

寄存器P0LR: P0口输出锁存寄存器(地址：45H)

	bit7							bit0
复位值 --xx xxxx	-	-	P0LR5	P0LR4	P0LR3	P0LR2	P0LR1	P0LR0
	U	U	R/W	R/W	R/W	R/W	R/W	R/W

P1LR（P1 口输出锁存控制寄存器）

寄存器P1LR: P1口输出锁存寄存器(地址: 47H)

	bit7							bit0
复位值 --xx xxxx	-	-	P1LR5	P1LR4	P1LR3	P1LR2	P1LR1	P1LR0
	U	U	R/W	R/W	R/W	R/W	R/W	R/W

P2LR（P2 口输出锁存控制寄存器）

寄存器2.12: P2LR: P2口输出锁存寄存器(地址: 46H)

	bit7							bit0
复位值 xxxx xxxx	-	-	-	-	P2LR3	P2LR2	P2LR1	P2LR0
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

NVMADDRH（NVM 地址指针高 8 位）

NVMADDRL（NVM 地址指针低 8 位）

NVMDATAH（NVM 数据寄存器高 8 位）

NVMDATAL（NVM 数据寄存器低 8 位）

INTCTL（中断控制寄存器）

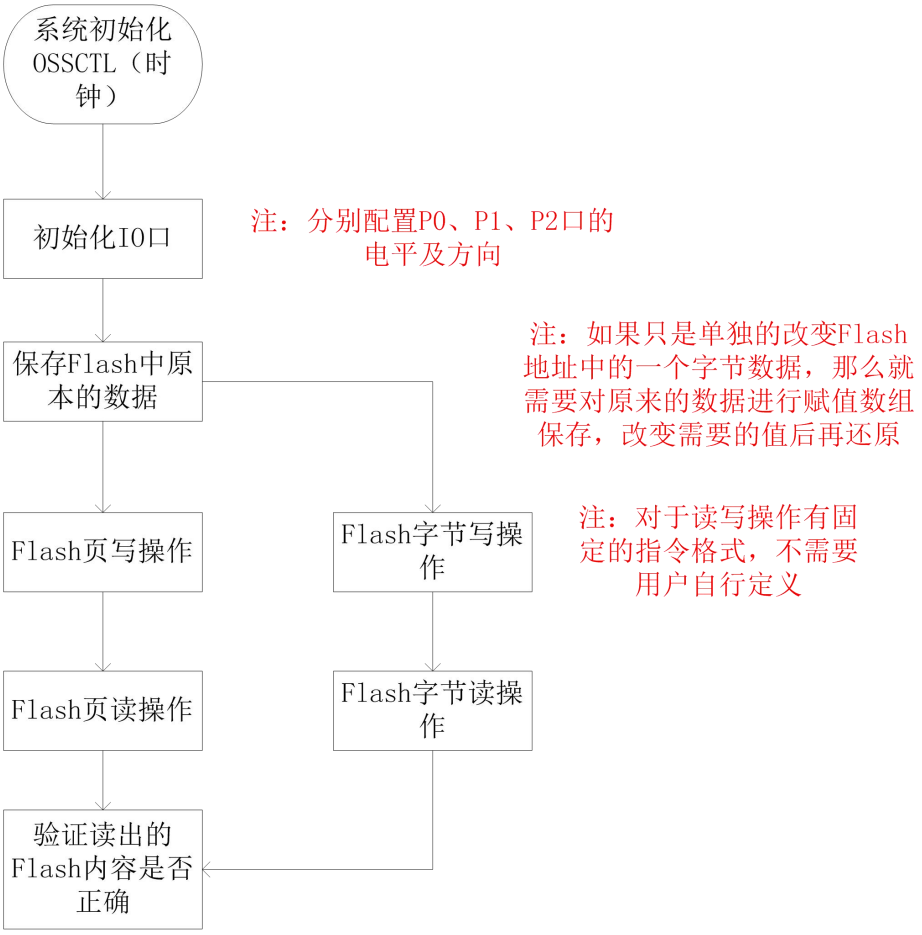
寄存器INTCTL: 中断控制寄存器(地址: 0BH)

	bit7							bit0
复位值 0000 0000	AIE	PUIE	T0IE	INT0IE	P0IE	T0IF	INT0IF	P0IF
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

NVMCTL0（NVM 控制寄存器 0）

NVMCTL1（NVM 控制寄存器 1）

Flash 读写样例程序框图



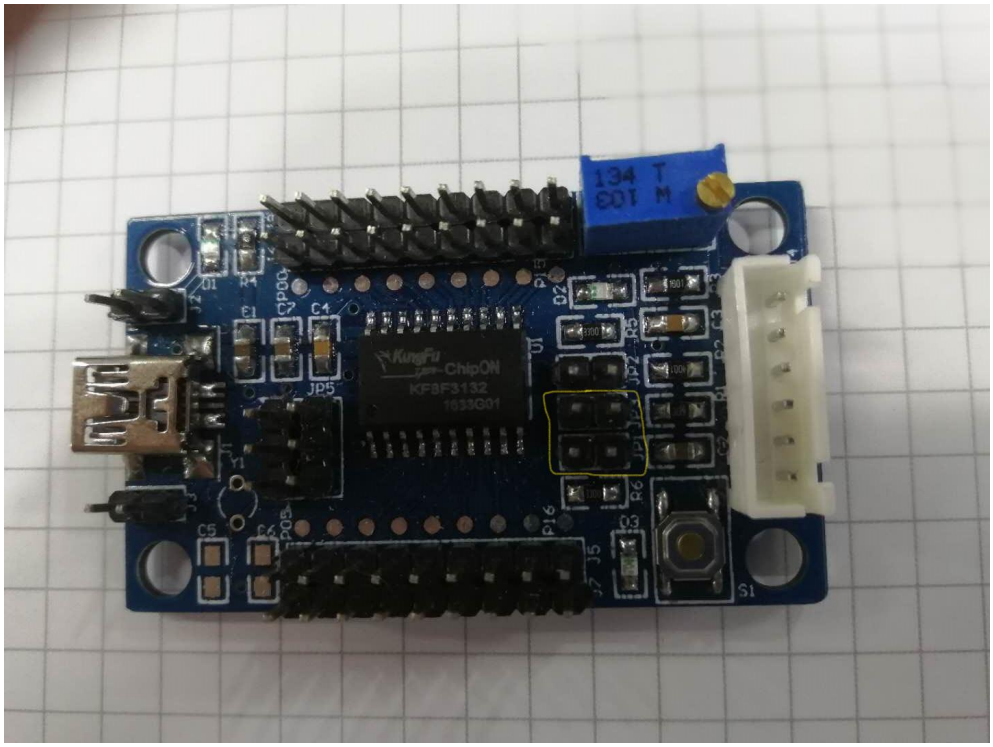
Flash 读写样例简述：

开发环境：ChipON IDE

功能简述：演示 Flash 页写、页读、单字写、单字读函数功能，并验证操作的结果。如果读写结果错误，则点亮 LED3，操作结束后 LED2 点亮，此时可以断电，使用 PRO 读取 Flash 空间的内容。

硬件连接：JP2 和 JP3 连接（图中黄色框中的插针接跳线帽）

特殊说明：当前芯片的 Flash 可自写范围是：800H～0FDFH。为防止程序空间和 Flash 自写空间冲突，用户需要在项目的 occupancy.txt 文件中声明 Flash 自写的起始和结束地址。



Flash 读写样例程序：

MCU 初始化：

```
void init_mcu()
{
      OSCCTL = 0x70;                                    //设置为8M
      /*********端口初始化***********/
      TR0 = 0;
      P0LR = 0;             //P0口配置配置为低电平输出

      TR1 = 0;              //P1口配置为输出模式
      P1LR = 0X06;

      TR2 = 0;              //P2设置为低电平输出
      P2LR = 0;
}
```

Flash 字节读：

```
unsigned int FLASH_READ_ONE(unsigned int address)
{
     // FLASH_READ_BUF;  需要定义全局的该变量，获取值用于返回，或直接使用
     //; 参数的使用
     NVMADDRH=(unsigned char)(address>>8);
     NVMADDRL=(unsigned char)address;
     __asm
          ;//备份中断使能寄存器
               BANKSEL _INTCTL
               MOV       R1,_INTCTL
          ;//关闭中断，操作不可打断
               CLR       _INTCTL,_AIE
               JNB       _INTCTL,_AIE
               JMP       $-2
          ;//时钟频率备份及降频操作,建议降频到1M,此时中断已被关
               MOV       R0,#0x30
               MOV       R2,_OSCCTL
               MOV       _OSCCTL,R0
          ;//硬件使能读操作
               BANKSEL _NVMCTL0
               MOV  R5,#0x81
               MOV  _NVMCTL0,R5
               NOPZ
               NOPZ
               NOPZ
               NOPZ
          ;//时钟与中断使能的还原
               MOV       _OSCCTL,R2
               AND  R1,#0xC0  ;//中断使能仅关系高2位
               ORL  INTCTL,R1
          ;//操作结果赋值到变量，用于返回
               BANKSEL   _NVMDATAL
               MOV  R6,_NVMDATAL
               MOV  R7,_NVMDATAH
               BANKSEL _FLASH_READ_BUF
               MOV (_FLASH_READ_BUF),R6
               MOV (_FLASH_READ_BUF+1),R7
     __endasm;
     return    FLASH_READ_BUF;
}
```

Flash 字节写：

```
void FLASH_WRITE_ONE(unsigned int address,unsigned int value)
{
      // 读出当前页到数据缓存，要求缓存大小必须满足写地址的偏移量要求
      FLASH_READ_Page(address&0xFFE0);
      // 修改待写数据在按照页排序中位置数据结果
      FLASH_BUFFER[address&0x1F] = value;
      // 整页数据回写
      FLASH_WRITE_Page(address&0xFFE0);
}
```

Flash 页读：

```
void FLASH_READ_Page(unsigned int address)
{
     unsigned char length=32;
     //; 参数的使用
     NVMADDRH=(unsigned char) (address>>8);
     NVMADDRL=address;

     __asm
     ;//备份中断使能寄存器
          BANKSEL _INTCTL
          MOV       R1,_INTCTL
     ;//关闭中断，操作不可打断
          CLR       _INTCTL,_AIE
          JNB       _INTCTL,_AIE
          JMP       $-2
     ;//时钟频率备份及降频操作,建议降频到1M,此时中断已被关
          MOV       R0,#0x30
          MOV       R2,_OSCCTL
          MOV       _OSCCTL,R0
     ;//读取结果的存放起始RAM地址，即数组缓存区
FLASH_BUFFER[x]
          MOV       R3,#_FLASH_BUFFER
     __endasm;

     while(length--)      // 仅使用R0，不改变R1
     {
          __asm
          ;//硬件读
          BANKSEL _NVMCTL0
          MOV  R5,#0x81
          MOV  _NVMCTL0,R5
          NOPZ
          NOPZ
          NOPZ                    ;//至少2条
          NOPZ
          ;//读结果处理
          BANKSEL   _NVMDATAL
          MOV  R6,_NVMDATAL
          MOV  R7,_NVMDATAH

          BANKSEL _FLASH_BUFFER
          ST        [R3],R6
          INC       R3
          ST        [R3],R7
          INC       R3
          ;//
指向下一操作地址，建议每次操作内容在一个块中，此时不需要处理_BADDR
L进位的_BADDRH+1操作
          BANKSEL _NVMADDRL
          INC       _NVMADDRL
          JNB       _PSW,_Z
          INC       _NVMADDRH
          __endasm;
     }

     __asm
          ;//时钟与中断使能的还原
          BANKSEL _OSCCTL
          MOV       _OSCCTL,R2
          AND  R1,#0xC0   ;//中断使能仅关系高2位
          ORL  _INTCTL,R1
     __endasm;
}
```

Flash 页写：

```
void FLASH_WRITE_Page(unsigned int address)
{
    unsigned char length=32;
    //; 参数的使用
    NVMADDRH=(unsigned char) (address>>8);
    NVMADDRL=address;

    __asm
    ;//备份中断使能寄存器
        BANKSEL _INTCTL
        MOV     R1,_INTCTL
    ;//关闭中断，操作不可打断
        CLR     _INTCTL,_PUIE
        CLR     _INTCTL,_AIE
        JNB     _INTCTL,_AIE
        JMP     $-2
    ;//时钟频率备份及降频操作,建议降频到1M,此时中断已被关
        MOV     R0,#0x30
        MOV     R2,_OSCCTL
        MOV     _OSCCTL,R0
    ;//读取结果的存放起始RAM地址，即数组缓存区  FLASH_BUFFER[x]
        MOV     R3,#_FLASH_BUFFER
    __endasm;

    while(length--)          // 仅使用R0，不改变R1
    {
        __asm

        ;//加载待写数据
        BANKSEL _FLASH_BUFFER
        LD      R6,[R3]
        INC     R3
        LD      R7,[R3]
        INC     R3
        BANKSEL _NVMDATAH
        MOV _NVMDATAH,R7
        MOV _NVMDATAL,R6
        ;//硬件写
        MOV  R5 ,#0x84
        MOV  _NVMCTL0,R5
        MOV  R5,#0x69
        MOV  _NVMCTL1,R5
        MOV  R5,#0x96
        MOV  _NVMCTL1,R5
        SET  _NVMCTL0 , 1    ;// 写存在高压，高压还原添加空指令确保后续运行正常
        NOPZ
        NOPZ
        NOPZ
        NOPZ
        NOPZ
        NOPZ                                ;// 建议10条，至少2条
        NOPZ
        NOPZ
        NOPZ
        NOPZ
        MOV  R5,#0X80
        MOV  _NVMCTL0,R5
        ;//指向下一操作地址，这里不考虑高位，特性要求只能操作一页内的数据，不能跨页
        BANKSEL _NVMADDRL
        INC     _NVMADDRL
        __endasm;
    }

    __asm
        ;//时钟与中断使能的还原
        BANKSEL _OSCCTL
        MOV     _OSCCTL,R2
        AND  R1,#0xC0  ;//中断使能仅关系高2位
        ORL  _INTCTL,R1
    __endasm;
}
```

主函数：

```
void main()
{
	unsigned char i=0;
	unsigned int j=0;

	init_mcu();//初始化晶振和IO口

	for(i;i<32;i++)//填写Flash缓存
	{
		FLASH_BUFFER[i]= 0x0E00+i;
	}
	FLASH_WRITE_Page(0x0E00);//整页Flash写操作

	FLASH_READ_Page(0x0E00);//整页Flash读操作
	for(i;i<32;i++)//验证读出的Flash内容是否正确，如果有错误则Led3点亮
	{
		j= FLASH_BUFFER[i];
		if(j!=(0x0E00+i))
			LED3=0;
	}

	FLASH_WRITE_ONE(0x0E40,0x01);//单字Flash写
	j=FLASH_READ_ONE(0x0E40);//单字Flash读
	LED2=0;//操作结束后，Led2点亮

	if(j!=1)//单字读写操作验证，有错误则点亮Led3
		LED3=0;
	while(1)
	{

	}
}
```

注意事项：

1、在写 Flash 时，必须先对每个页的第一块进行写操作，以擦除本页的数据，如果没有对第一块进行写操作，直接写后面块则本页的所有数据都不会被擦除。

2、写 Flash 时，只能对 Flash 成块写入数据，不允许跨区操作。

3、不能单独将一个字节（或字）的数据写入某块的一个字节（或字）中，如果写入 Flash 中的数据没有 16 个字或不能被 16 所整除，需要将块中不需要写入数据的单元写入 0 或者其他值，否则可能会导致写入的数据出错。

4、如果原来的 Flash 保存有数据，现在需要修改原数据中的一个字或者几个字，其他单元的值不变，则需要先将其他对应块中其他数据读出来保存，然后再将需要修改的数据和之前读出的值写入。

5、配置位的 SWRTEN 需配置为 0，才能对 Flash 进行写操作。