实验 16 - FLASH 读写实验

1. 实验目的

掌握 NRF24LE1 的 FLASH 的读写。

2. 实验内容

按键 S1 按下时，程序向 0x2800~0x2804 地址依次写入 0x01、0x02、0x03、0x04、0x05 5 个字节数据，写完后，再读出数据并通过串口打印出数据。

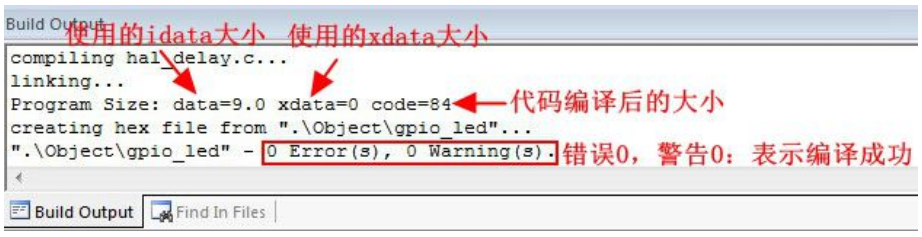
3. 实验原理

NRF24LE1 内部集成 16KB 的 FLASH，页面大小为 512 字节，共 32 页。MCU 对 FLASH 读写必须按照以下顺序进行：

- 设置 FSR 寄存器的 WEN 为高来使能 Flash 擦/写访问，此时 Flash 对于 MCU 的擦/写访问是开放的，直到 FSR 寄存器的 WEN 位为低；
- **要写入数据的 Flash 必须先擦除**，擦除操作只能对整页进行。写页地址（0~31）到 FSR 寄存器可以擦除页。
- 设置 PCON 寄存器的 PMW 位为高来使能程序存储器写模式；
- MCU 通过正常的存储器写操作即可对 Flash 进行编程，字节是按指定地址单个写入的（没有自动递增功能）。

4. 实验步骤

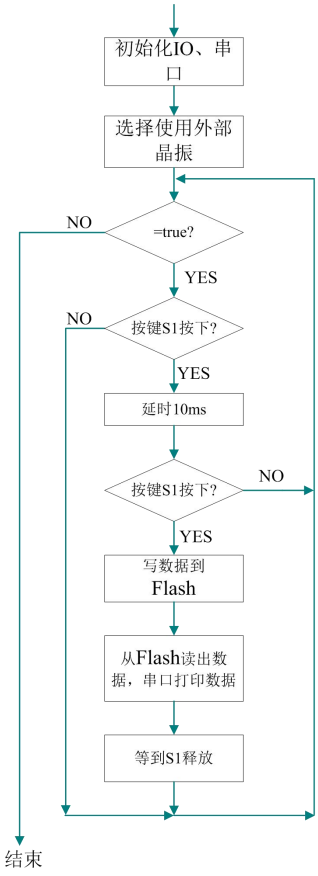
- 在 Keil uVision4 中打开工程“Flash.uvproj”工程；
- 编译工程，注意查看编译输出栏，观察编译的结果，如果有错误，修改程序，直到编译成功为止；



- 将编译生成的 HEX 文件“Flash.hex” (该文件位于工程目录下的“Object”文件夹中)通过编程器下载到开发板中运行。
- 打开串口调试助手，选择串口号，设置波特率为 57600，打开串口，注意不要勾选“HEX 显示”。按下 S1 按键再松开，观察串口输出的数据。

5. 实验程序

5.1. 程序流程



5.2. 程序清单

```
#define   D1      P00   //开发板上的指示灯 D1

/*******************************************************************************
 *描   述 : 配置 IO P0.0 为输出，驱动 LED。P03 输出: UART TXD，P04:输入 UART RXD
```

```
 *入   参 : 无
 *返回值 : 无
*******************************************************************************/
void IO_Init(void)
{
    P0DIR &= ~0x01;     //配置 P0.0 为输出
    P0DIR &= ~0x08;     //P03:输出  UART TXD
    P0DIR |= 0x10;      //P04:输入  UART RXD
    D1 = 1;             //设置 D1 初始状态为熄灭
}
/*******************************************************************************
 *描   述 : 串口打印字符串
 *入   参 : 无
 *返回值 : 无
*******************************************************************************/
void PutString(char *s)
{
  while(*s != 0)
    hal_uart_putchar(*s++);
}

/*******************************************************************************
 *描   述 : 根据起始地址，向 flash 上写入多个字节数据
 *入   参 : a：起始地址   *p：写入数据的存放地址  n：读出的字节数
 *返回值 : 无
*******************************************************************************/
void flash_bytes_write(uint16_t a, const uint8_t *p, uint16_t n)
{
  uint8_t xdata *pb;

  F0 = EA;   //保存中断设置
  EA = 0;    //关闭全局中断

  WEN = 1; //使能 flash 擦写操作

  pb = (uint8_t xdata *)a; //写入数据

  while(n--)
  {
    PCON |= 0x10; //置位 pmw 使能对 flash 的访问
    *pb = *p;      //写入数据
    PCON &= ~0x10; ////清零 pmw 关闭对 flash 的访问
```

```
        pb++;
        p++;

        while(RDYN == 1) //等待操作完成
            ;
    }

    WEN = 0; //关闭 flash 擦写操作

    EA = F0; //恢复中断设置
}

/*****************************************************************************
 *描   述 : 根据起始地址，从 flash 上读取多个字节数据
 *入   参 : a：起始地址   *p：数据存放地址  n：读出的字节数
 *返回值 : 无
*****************************************************************************/
void flash_bytes_read(uint16_t a, uint8_t *p, uint16_t n)
{
    uint8_t xdata *pb = (uint8_t xdata *)a;

    while(n--)
    {
        PCON |= 0x10;    //置位 pmw 使能对 flash 的访问
        *p = *pb;            //写入数据
        PCON &= ~0x10; //清零 pmw 关闭对 flash 的访问

        pb++;
        p++;
    }
}

/*****************************************************************************
 *描   述 : 主函数
 *入   参 : 无
 *返回值 : 无
*****************************************************************************/
void main(void)
{
    uint8_t    i;
    uint32_t    LoopCount = 0;

    hal_clk_set_16m_source(HAL_CLK_XOSC16M);    //使用外部 16MHz 晶振
```

```
    IO_Init();   //初始化 IO
    hal_uart_init(UART_BAUD_57K6); //初始化 UART，波特率 57600
    while(hal_clk_get_16m_source() != HAL_CLK_XOSC16M) //等待时钟稳定
        ;
    EA = 1;             //开启全局中断

    while(1)
    {
        LoopCount++;
        if(LoopCount == 10000)
        {
          D1 = ~D1;   //D1 指示灯闪烁，指示设备工作正常
          LoopCount = 0;
        }

        if(S1 == 0)//  按键 S1 按下?
        {
          delay_ms(10);
          if(S1 == 0)        //确认按键 S1 按下
          {
            hal_flash_page_erase(20);   //写之前先擦除
            delay_ms(30);
            for(i=0;i<5;i++) WriteBuf[i] = i+1;
            flash_bytes_write(0x2800,WriteBuf,5); //写入数据
            flash_bytes_read(0x2800,temp_data,5);//读出数据
            PutString("Flash address 0x2800~0x2804:"); //串口打印数据
            for(i=0;i<5;i++)PutHexString(temp_data[i]);
            while(S1==0);    //等待按键释放
          }
        }
    }
}
```