实验六 SPI总线

杨庆龙
1500012956

2018.4.25

1 实验目的

- 了解SPI总线的基本时许
- 了解串行FLASH芯片的基本原理
- 掌握串行FLASH芯片的基本用法

2 实验原理

2.1 SPI总线简介

2.1.1 传输线

- MOSI:主设备到从设备的数据线
- MISO:从设备到主设备的数据线
- SPCK:主设备驱动的时钟信号
- NSS:从设备选择线

2.1.2 优点

- 全双工
- 协议灵活
- 接口简单
- 信号单向传输

2.1.3 缺点

- 管脚较多
- 没有流控制信号，没有应答机制
- 只有一个主设备
- 数据传输距离比较近

配置寄存器如表1

Table 1: SPI配置寄存器 SPI0CFG

D7	D6	D5	D4	D3	D2	D1	D0
CKPHA	CKOPL	BC2	BC1	BC0	SPIFRS2	SPIFRS1	SPIFRS0

2.2 SPI接口

C8051F020的SPI控制器可工作在主模式或从模式下，相关控制使用XBR设置，各寄存器功能如下

- CKPHA:SPI时钟相位
- CKPOL:SPI时钟极性
- BC2-0:获得当前帧已发送的比特数
- SPIFR2-0:用来设置帧大小
- SPIF:中断标识，软清除
- WCOL:写入碰撞位，软清除
- MODF:主模式碰撞位，软清除
- RXOVRN:接收溢出，软清除
- TXBSY:发送忙标识，自动清除
- SLVSEL:选中标识，NSS为低时置1
- MSTEN:主模式允许位
- SPIEN:SPI允许位

使用SPI0CKR设置SPI时钟频率

$$f = \frac{SYSCLK}{2 \times SPI0CKR + 1}$$

2.3 SPI Flash的使用

3 思考题

1. 还是保持03FH，但如果写入其他值，就可能会出问题。因为flash只能写入0，不能写入bit1，所以当我们要在同一个位置写两次时，就可能会出问题。
2. 可能会得到错误的结果，如果实际数据长度比所需数据长，那只能得到部分bit，如果更短，则会用后续数据补上去。都会影响后续数据的读入。

4 源码

```
#include <C8051F020.h>
#include <stdio.h>
#include "time.h"
#include "SPI.h"
#include "communicate.h"

void main() {
  unsigned char v;
  unsigned char c;
  unsigned int addr;
  unsigned int v1;
  unsigned int v2;
  WDTCN = 0xDE;
  WDTCN = 0xAD;
  sysclk_init();
  P6 = 0x80;
  uart0_init();
  spi_init();
  while(1){
```

```
20          do {
21            c = getchar();
22          }while((c == '') || (c == '\r') || (c == '\n'));
23          scanf("%lx", &addr);
24          switch(c) {
25            case 'd':
26            ////////////////////////
27            break;
28            case 'w':
29            scanf("%bx", &v1);
30            flash_write(addr,v1);
31            printf("\r\n␣W␣%lx␣%bx␣OK\r\n",addr,v1);
32            break;
33            case 'c':
34            flash_erease(addr);
35            printf("\r\n␣C␣%lx␣OK\r\n",addr);
36            break;
37          }
38        }
39      }
```