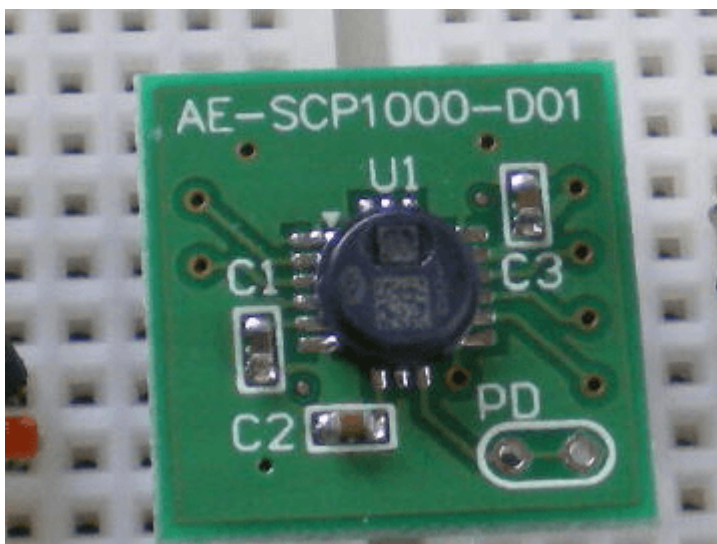


簡易気圧&温度計(SCP1000)

概要

気圧と温度が測定できるセンサーが手に入りましたので、早速、簡易な「気圧&温度計」を製作してみました。使用したセンサーモジュールは、VTIテクノロジー社製の気圧(絶対圧)センサ「SCP1000」を、更に使いやすくした、8ピンDIP型の「SCP1000モジュール」です。(秋月電子にて販売)



<SCP1000のモジュールの概観>

<SCP1000モジュールの特長>

- 扱いやすい8ピンDIP型になっており、気圧計、高度計などに最適
- 電源電圧<2.4~3.3V>
- インターフェース<SPI>
- 測定範囲<30kPa~120kPa(300ヘクトパスカル~1200ヘクトパスカル)>

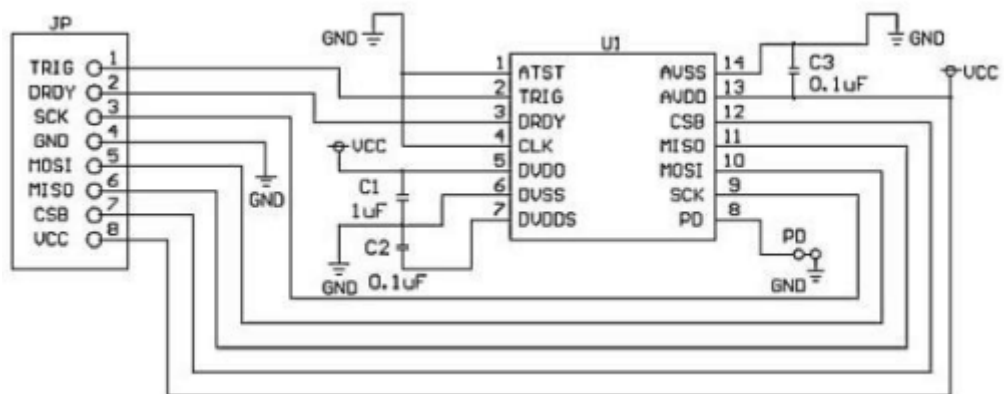
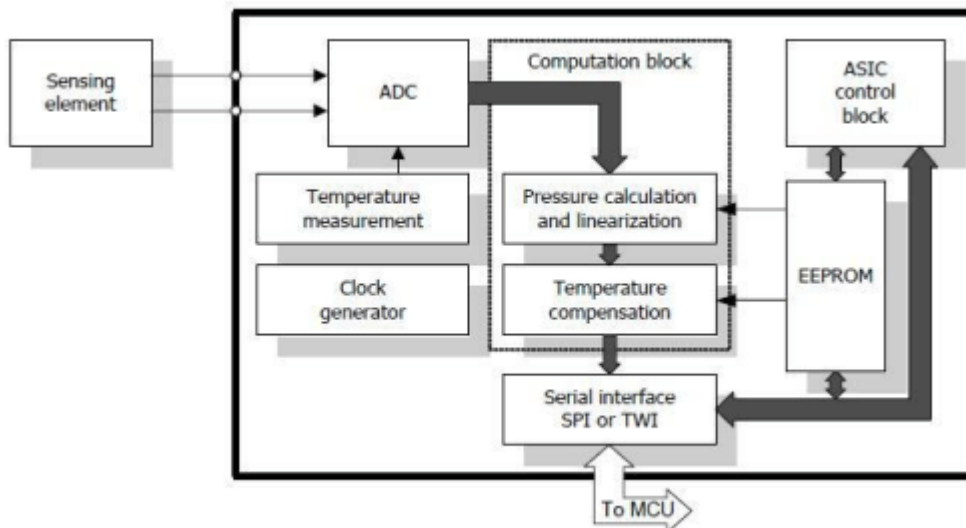
動作原理

SCP1000との通信は、SPI(Serial Peripheral Interface)方式で行います。実際の制御は、SCP1000のレジスタにデータを、書き込んだり読み込んだりすることにより行います。

<処理の流れ>

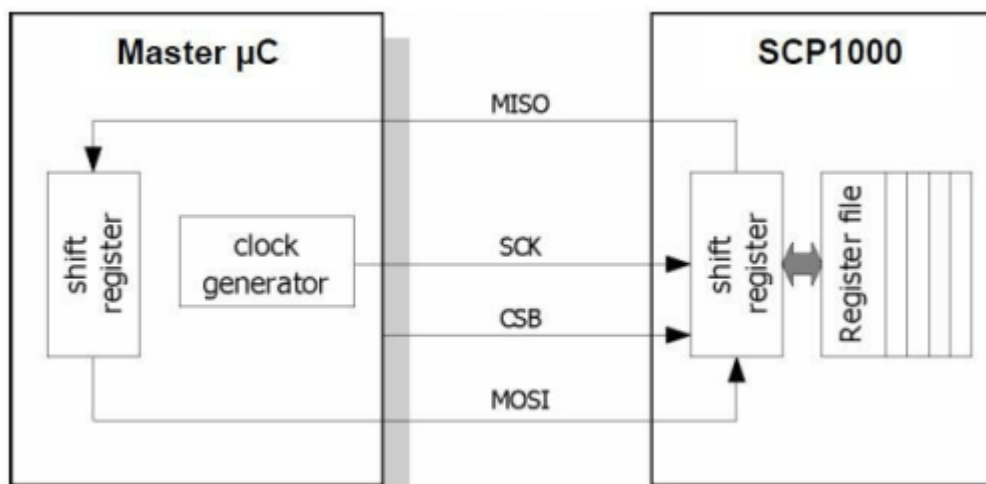
1. LCDを初期化します。
2. SPIを初期化します。
3. SCP1000をソフトウェアリセットします。
4. SCP1000のオペレーションモードを設定します。(高精度モード)
5. SCP1000のリビジョン番号を取得します。(必ず必要なわけではありません)
6. SCP1000のステータスを取得し、「スタートアップ状態」、「データレディー状態」であるかをチェックします。
7. 気圧を取得し、パスカル値に変換し、LCDに表示します。
8. 温度を取得し、値に変換し、LCDに表示します。
9. 6.に戻ります。

<SCP1000のブロックダイアグラム>



<SCP1000の回路図>

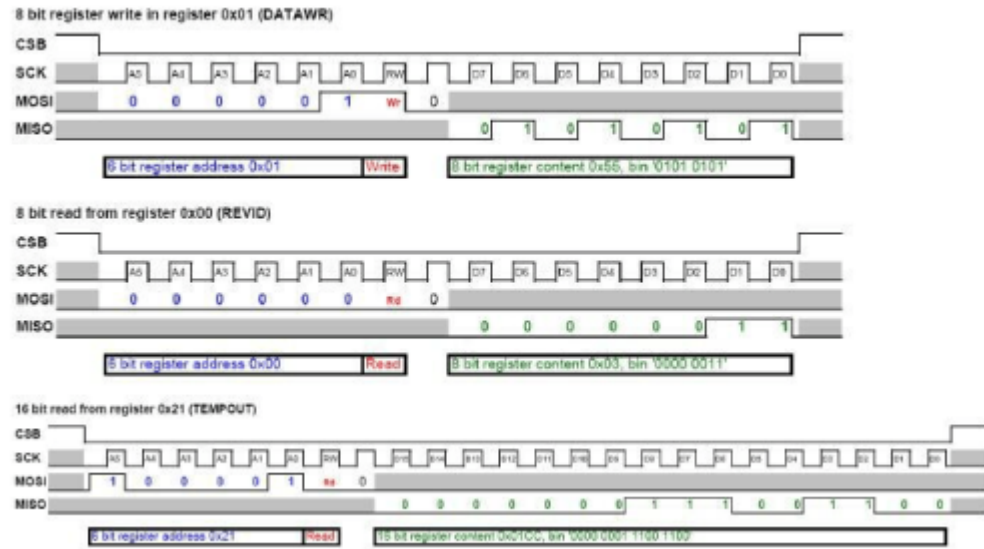
<SCP1000のマスター/スレーブの構成>



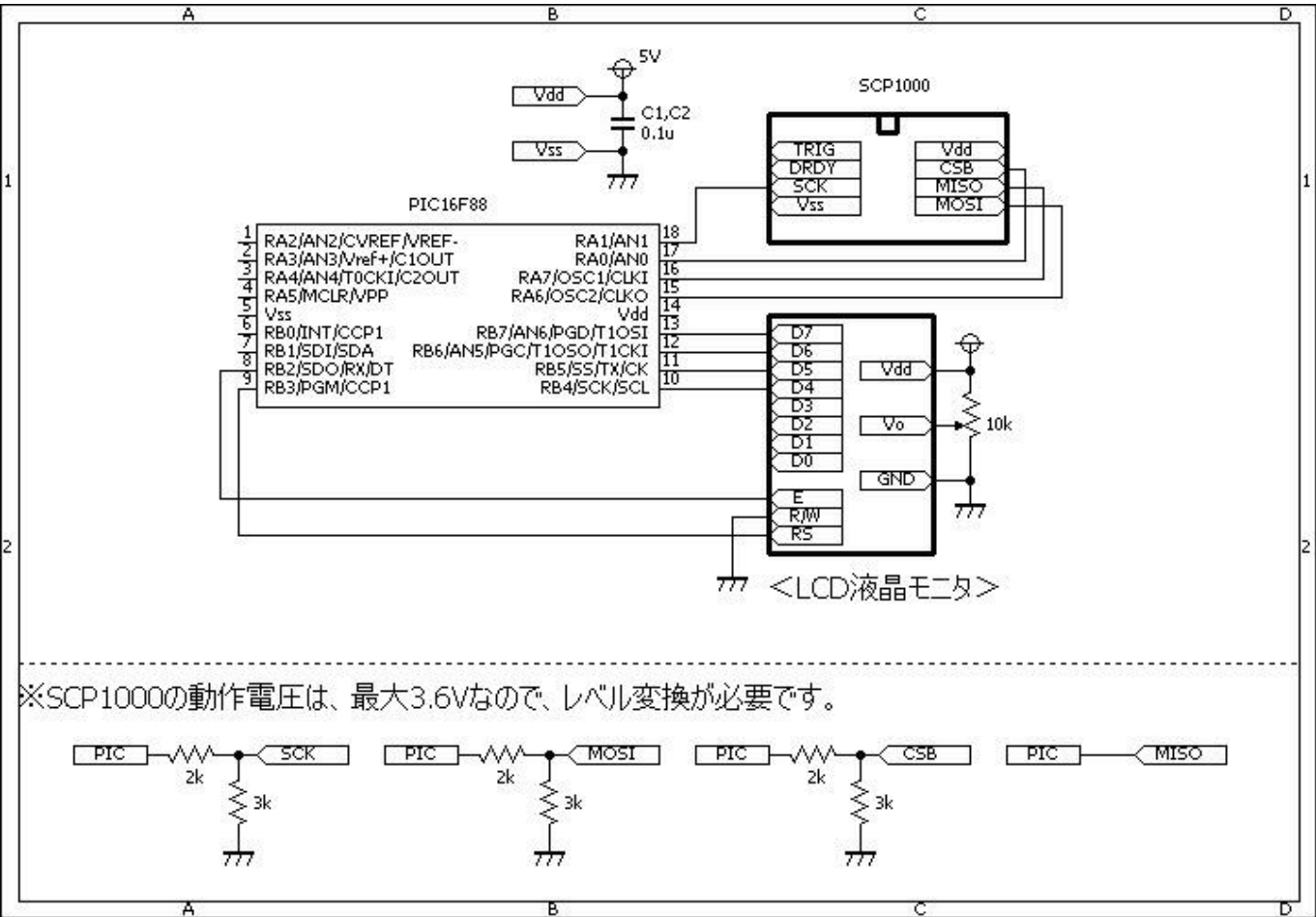
Address		Name	Description	Mode (R, W, RW)	Register access	Width[bits]
SPI	TWI					
0x00		REVID	ASIC revision number	R	Direct	8
0x01		DATAWR	Indirect register access data	RW	Direct	8
0x02		ADDPTR	Indirect register access pointer	RW	Direct	8
0x03		OPERATION	Operation register	RW	Direct	8
0x04		OPSTATUS	Operation status	R	Direct	8
0x06		RSTR	ASIC software reset	W	Direct	8
0x07		STATUS	ASIC top-level status	R	Direct	8
0x1F	0x7F	DATARD8	Pressure output data (MSB) or 8 bit data read from EEPROM	R	Direct	8
0x20	0x80	DATARD16	Pressure output data (LSB) or 8-bit data read from indirect register	R	Direct	16
0x21	0x81	TEMPOUT	14-bit temperature output data	R	Direct	16
	0x00	CFG	Configuration register	RW	Indirect	8
	0x05	TWIADD	TWI address	W	Indirect	8
	0x29	USERDATA1	User data	RW	EEPROM	8
	0x2A	USERDATA2	User data	RW	EEPROM	8
	0x2B	USERDATA3	User data	RW	EEPROM	8
	0x2C	USERDATA4	User data	RW	EEPROM	8

<SCP1000のレジスタ>

<SCP1000の通信フォーマット(8ビット書き込み、8ビット読み込み、16ビット読み込み)>



回路図



ソースコード

気圧の取得箇所に誤りがありましたので、訂正版を掲載いたしました。

[pressure_v110.c](#)

```
//*****
*
/*
   「簡易気圧測定器」SCP1000
*/
//*****
*
//SCP1000
#define  OPARATION    0x03
#define  RSTR         0x06
#define  STATUS       0x07
#define  DATARD8      0x1F
#define  DATARD16     0x20
#define  TEMPOUT      0x21
//SPI
sbit  SoftSpi_CSB  at  RA0_bit;
sbit  SoftSpi_SDI  at  RA7_bit;
sbit  SoftSpi_SDO  at  RA6_bit;
```

```
sbit SoftSpi_CLK at RA1_bit;
sbit SoftSpi_CSB_Direction at TRISA0_bit;
sbit SoftSpi_SDI_Direction at TRISA7_bit;
sbit SoftSpi_SDO_Direction at TRISA6_bit;
sbit SoftSpi_CLK_Direction at TRISA1_bit;
//LCD
sbit LCD_RS at RB3_bit;
sbit LCD_EN at RB2_bit;
sbit LCD_D7 at RB7_bit;
sbit LCD_D6 at RB6_bit;
sbit LCD_D5 at RB5_bit;
sbit LCD_D4 at RB4_bit;
sbit LCD_RS_Direction at TRISB3_bit;
sbit LCD_EN_Direction at TRISB2_bit;
sbit LCD_D7_Direction at TRISB7_bit;
sbit LCD_D6_Direction at TRISB6_bit;
sbit LCD_D5_Direction at TRISB5_bit;
sbit LCD_D4_Direction at TRISB4_bit;
//other
#define BYTE unsigned short
#define WORD unsigned int
#define WORD unsigned long
//*****
*
//SCP1000への1バイトデータの書き込み
void scp1000_write(BYTE addr, BYTE dat)
{
    SoftSpi_CSB = 0;
    SoftSpi_Write((addr << 2) | 2);
    SoftSpi_Write(dat);
    SoftSpi_CSB = 1;
    //
    Delay_ms(100);
}
//*****
*
//SCP1000からの1バイトデータの読み込み
BYTE scp1000_read8(BYTE addr)
{
    BYTE dummy, tmp;
    //
    SoftSpi_CSB = 0;
    SoftSpi_Write(addr << 2);
    tmp = SoftSpi_Read(dummy);
    SoftSpi_CSB = 1;
    //
    Delay_ms(100);
    return (tmp);
}
//*****
*
```

```
//SCP1000からの2バイトデータの読み込み
WORD    scp1000_read16(BYTE addr)
{
    BYTE    dummy;
    WORD    tmp;
    //
    SoftSpi_CSB = 0;
    Soft_Spi_Write(addr << 2);
    tmp = Soft_Spi_Read(dummy) << 8;
    tmp |= Soft_Spi_Read(dummy);
    SoftSpi_CSB = 1;
    //
    Delay_ms(100);
    return (tmp);
}
//*****
*
void    main()
{
    BYTE    buf[16];
    BYTE    tmp;
    WORD    temperature;
    WORD    pressure;
    //
    OSCCON = 0b01110000;
    CMCON  = 0b00000111;
    ANSEL  = 0b00000000;
    TRISA  = 0b10111010;
    TRISB  = 0b00000000;
    //□□□の初期化
    Lcd_Init();
    Lcd_Cmd(_LCD_CURSOR_OFF);
    Lcd_Cmd(_LCD_CLEAR);
    Lcd_Out(1, 1, "Pressure v1.10");
    Delay_ms(1000);
    Lcd_Cmd(_LCD_CLEAR);
    //□□□の初期化
    SoftSpi_Init();
    SoftSpi_CSB = 1;
    Delay_ms(100);
    //ソフトウェアリセット&オペレーションモード設定
    scp1000_write(RSTR, 0x01);
    scp1000_write(OPARATION, 0x0A);
    tmp = scp1000_read8(0x00);
    Lcd_Out(1, 1, "SCP1000 start!");
    Delay_ms(1000);
    Lcd_Cmd(_LCD_CLEAR);
    //
    while (1) {
        //ステータスの取得
```

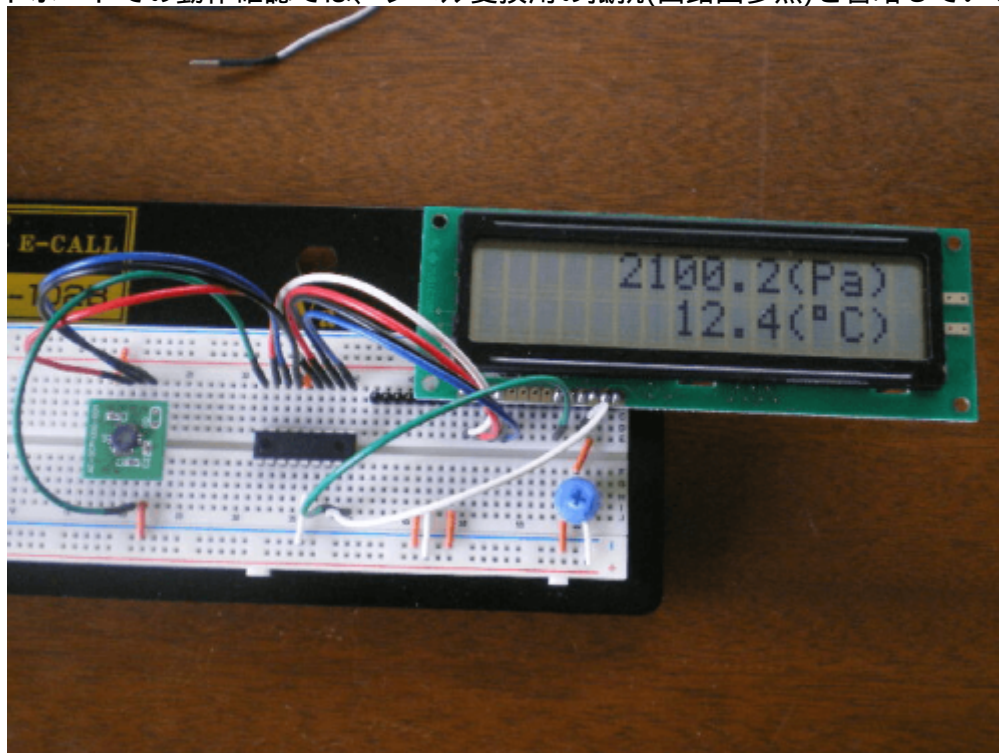
```

    tmp = scp1000_read8(STATUS);
    if ((tmp.F0 != 0) || (tmp.F5 != 1)) {
        continue;
    }
    if (tmp.F4 == 1) {
        Lcd_Chr(2, 16, '*'); //Real time error!
    } else {
        Lcd_Chr(2, 16, ' ');
    }
    //気圧の取得→換算(パスカル)→表示
    pressure = scp1000_read8(DATARD8) & 0x03;
    pressure <=<= 16;
    pressure |= scp1000_read16(DATARD16);
    LongToStr((pressure * 10) / 4, buf);
    buf[0] = buf[1];
    buf[1] = buf[2];
    buf[2] = buf[3];
    buf[3] = buf[4];
    buf[4] = buf[5];
    buf[5] = buf[6];
    buf[6] = buf[7];
    buf[7] = buf[8];
    buf[8] = buf[9];
    buf[9] = '.';
    Lcd_Out(1, 1, buf);
    Lcd_Out(1, 12, "(Pa)");
    //温度の取得→換算( )→表示
    temperature = scp1000_read16(TEMP0UT) & 0x3FFF;
    if ((temperature & 0x2000) == 0) {
        Lcd_Chr(2, 1, ' ');
    } else {
        Lcd_Chr(2, 1, '-');
    }
    temperature &= 0x1FFFF;
    WordToStr((temperature * 10) / 20, buf);
    buf[0] = buf[1];
    buf[1] = buf[2];
    buf[2] = buf[3];
    buf[3] = '.';
    Lcd_Out(2, 7, buf);
    Lcd_Out(2, 12, "\\xDFC");
}

//*****
*
```

動作確認

ブレッドボードでの動作確認では、レベル変換用の抵抗(回路図参照)を省略しているため、ご注意ください。



ださい。

左側=2階のベランダ、右側=家の前の路上 高度差は約5mです。表示内容は、上段がPA(パスカル)表示、



下段が温度表示です。



左側=2階の室内、右側=1階の室内 室内では、殆ど差がでません。



From:

<http://deepsky.jp/wiki/> - うごくといいな

Permanent link:

<http://deepsky.jp/wiki/doku.php?id=elechobby:picdic:pic16f88:134&rev=1588221838>

Last update: 2025/10/17 14:27

