温度データロガー

概要

温度センサーからのデータを一定周期で読み取り、RS232C（TTLレベル）として出力します。 周期は、1秒間隔、1分間隔、5分間隔、10分間隔の4モードから選ぶ事が出来ます。 また（PIC12F683内部に温度データを蓄積（EEPROMに記録）する機能も備えています。 その記憶容量は、128データ（1データは10ビットなので2バイトを使用）となります。

温度センサーについて

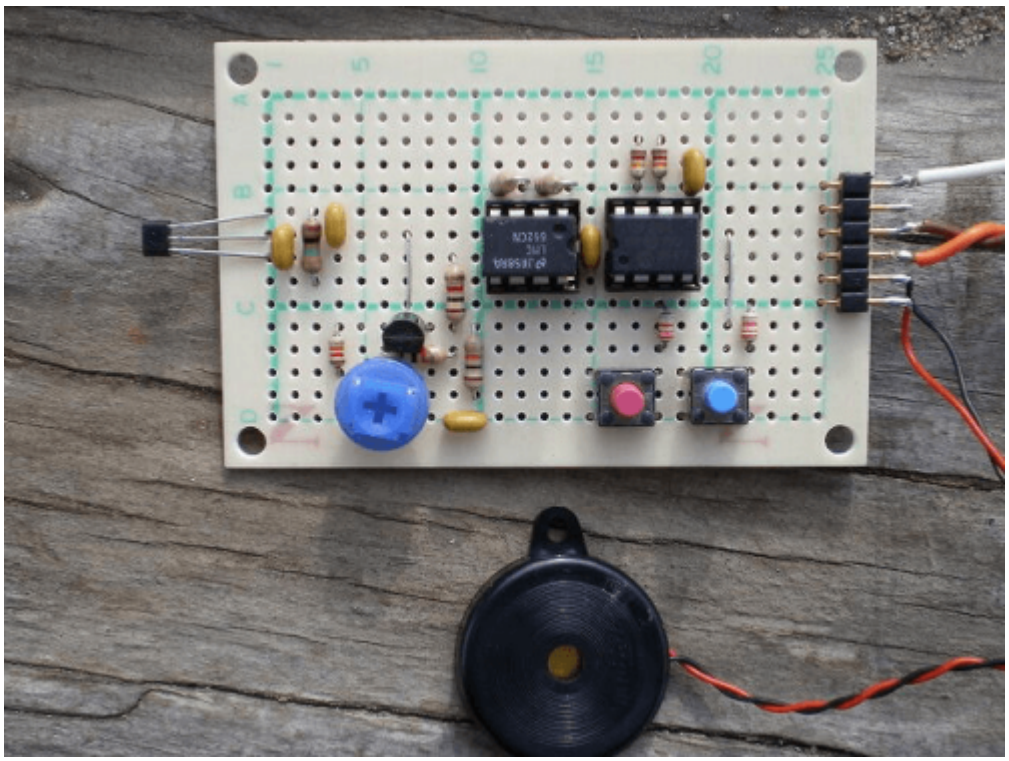
温度センサーには、高精度温度センサ（S8100B (SEIKO)を使用しました。 このセンサーの主な特長は、

- センサ・定電流回路・オペアンプを内蔵
- 電源（5V10μAtype(低消費)
- 測定温度範囲：-40～+100℃
- リニア出力電圧（-8.1mV/K(-8.1mV/℃)→今回は8mVで計算しています。
  Ta=-20℃（1.900V
  Ta=+30℃（1.497V
  Ta=+80℃（1.085V
- リニアリティ（直接性）：1.0(-20℃～+80℃)
- 再現性：±0.3%
- Vssを寒準とした温度電圧出力
- 低消費電流：10μA(25℃)typ

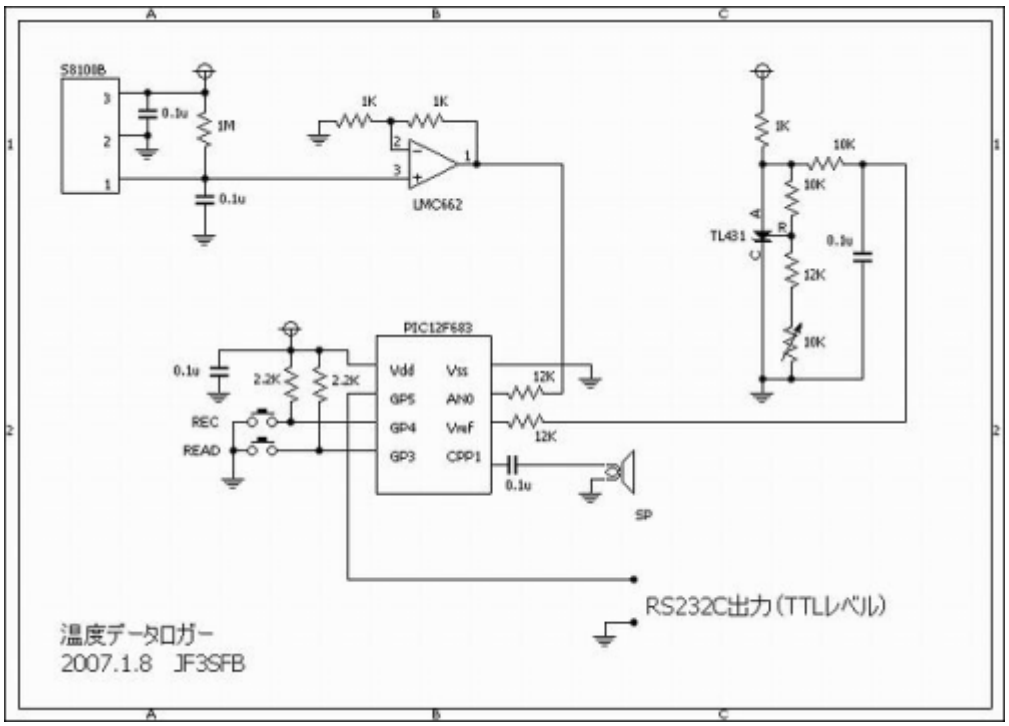
となっています。

ハードウエア構成

1. S8100Bによって温度を電圧に変換します。
2. この出力電圧をオペアンプ（LMC662CN（を利用して2倍に増幅します。
3. PICでは（A/D変換機能を使ってデータを取り込みます。
   PICのA/D変換（10ビット＝1024）で使用する、基準電圧(TL431)を、
   1024×4（4096→4.096V
   にして、1ビットあたりの精度を、4mVとします。
4. PICで補正されたデータをRS232C（TTLレベル）として出力します。
5. 記録SW（REC（と読出SW（READ（を設けています。



回路図



処理説明

1. 周期的に、A/D変換して温度データを得ます
    - 周期設定 （立ち上げ時．□□GPIO.F4□GPIO.F3□
      4通りの設定ができる
      1秒間隔□GPIO.F4□“1”□GPIO.F3□“1”□128秒（約2分）

1分間隔（GPIO.F4＝“0”，GPIO.F3＝“1”）128分（約2時間）
5分間隔（GPIO.F4＝“1”，GPIO.F3＝“0”）640分（約10時間）
10分間隔（GPIO.F4＝“0”，GPIO.F3＝“0”）1280分（約20時間）

2. 得られたデータに4mVを掛けます
3. 最も電圧が高くなる、-40℃では、4.120mVとなります。
(1.900mV×2倍)－(8mV×2倍×(40℃－20℃))→4.120mV
4. 補正します（小数点第１の位まで求めます）
(((4120 - temp) * 10) / 16) - 400
5. 結果をRS232C出力します
6. 記録と読出
   - 記録モード（GPIO.F4）
   データをEEPROMに記録します
   記憶容量は128データ（1データは2バイト）
   - 読出モード （GPIO.F3）
   EEPROMより読出してUSARTに出力します

ソースコード

ThermoMeter.c

```
//*****************************************************************************
*
/*
          【温度計】
  ・高精度３端子CMOS温度センサ　S8100B
     ：センサ・定電流回路・オペアンプを内蔵
     ：電源:5V10μAtype(低消費)
     ：リニア電圧出力:-8.0mV/℃
     ：測定温度範囲:-40～+100℃
       ：リニア出力電圧　-8.1mV/K(-8.1mV/℃)
　　　Ta=-20℃　1.900V
　　　Ta=+30℃　1.497V
　　　Ta=+80℃　1.085V
       ：リニアリティ（直接性）：1.0(-20℃～+80℃)
       ：再現性：±0.3%
　　Vssを寒準とした温度電圧出力
       ：低消費電流：10μA(25℃)typ
  ・2倍に増幅する。使用オペアンプはLMC662CN
  ・補正した結果をUSART(RS232C)で出力する。
  ・記録モード（GPIO.F4）
       ：データをEEPROMに書き込む
       ：記憶容量は128データ（1データは2バイト）
  ・再生モード （GPIO.F3）
     　　EEPROMより読み込みUSARTに出力する。
  ・周期設定（立ち上げ時.　　GPIO.F4，GPIO.F3）
       ：4通りの設定ができる。
       ：1秒間隔（GPIO.F4＝"1"，GPIO.F3＝"1"）　128秒（約2分）
       ：1分間隔（GPIO.F4＝"0"，GPIO.F3＝"1"）　128分（約2時間）
       ：5分間隔（GPIO.F4＝"1"，GPIO.F3＝"0"）　640分（約10時間）
       ：10分間隔（GPIO.F4＝"0"，GPIO.F3＝"0"）　1280分（約20時間）
```

```
  ・履歴
      ：2007.1.2　初期バージョン作成
*/
//*****************************************************************
*

static unsigned    int        timeCnt, cycleTime;
static unsigned    short    startFlag;

void   interrupt(){
    if (INTCON.T0IF == 1) {         //約30Hz = (1 / 8000000) * 4 * 256 *
256
        INTCON.T0IF = 0;
    }
    if (PIR1.TMR1IF == 1) {         //約4Hz = (1 / 8000000) * 4 * 8 * 256
* 256
        PIR1.TMR1IF = 0;
        //
        timeCnt++;
        if (timeCnt >= cycleTime) {     //最大で約16000秒(約4.5時間)のカウントが
可能
            timeCnt = 0;
            startFlag = 1;
        }
    }
}

//*****************************************************************
*

void   Pwm_Change_DutyEx(unsigned int duty_ratio)
{
    CCPR1L = duty_ratio >> 2;
    CCP1CON.F6 = duty_ratio & 0b00000001;
    CCP1CON.F7 = (duty_ratio & 0b00000010) >> 1;
}

//*****************************************************************
*

void   Soft_Uart_Write_String(char *buf)
{
    short    len, i;
    len = strlen(buf);
    for (i = 0; i < len; i++) {
        INTCON.GIE = 0;
        Soft_Uart_Write(buf[i]);
        INTCON.GIE = 1;
        Delay_ms(2);
    }
}
```

```
//*****************************************************************************
*

void   buzzer(unsigned short cnt, unsigned short interval)
{
    short    i, j;
    for (i = 0; i < cnt; i++) {
        Pwm_Start();
        for (j = 0; j < interval; j++)
            Delay_ms(1);
        Pwm_Stop();
        for (j = 0; j < interval; j++)
            Delay_ms(1);
    }
}

//*****************************************************************************
*

void main()
{
    static    unsigned    char    buf[10];
    static    unsigned    short    recFlag, lowData, hiData;
    static    unsigned    int        ad0, temp, cnt, len, address;
    //
    OSCCON = 0b01110000;    //クロックは8Mhz
    CMCON0 = 0b00000111;    //コンパレータは使用しない。
    ANSEL = 0b00000011;        // AN0□AN1を使用する。
    TRISIO = 0b00011011;
    OPTION_REG = 0b10000111;
    PIE1.TMR1IE = 1;
    PIR1.TMR1IF = 0;
    T1CON = 0b00110001;
    INTCON = 0b01100000;
    ADCON0.VCFG = 1;
    //
    Pwm_Init(3000);    // 3Khz
    Pwm_Change_DutyEx(1024 / 2);
    //
    Soft_Uart_Init(GPIO, 3, 5, 9600, 0);
    //
    timeCnt = 0;
    startFlag = 0;
    address = 0;
    recFlag = 0;
    //
    if ((GPIO.F4 == 1) && (GPIO.F3 == 1)) {
        cycleTime = 2;            //1秒
        buzzer(1, 200);
    }
```

```
    if ((GPIO.F4 == 0) && (GPIO.F3 == 1)) {
        cycleTime = 4 * 60;         //1分
        buzzer(2, 200);
    }
    if ((GPIO.F4 == 1) && (GPIO.F3 == 0)) {
        cycleTime = 4 * 300;    //5分
        buzzer(3, 200);
    }
    if ((GPIO.F4 == 0) && (GPIO.F3 == 0)) {
        cycleTime = 4 * 600;    //10分
        buzzer(4, 200);
    }
    //
    INTCON.GIE = 1;     // これ以降の処理で割り込みを許可する。
    //
    while (1) {
        if ((GPIO.F4 == 1) && (GPIO.F3 == 1))
            break;
    }
    Delay_ms(1000);
    buzzer(5, 100);
    //
    while (1) {
        //
        if ((recFlag == 0) && (GPIO.F4 == 0)) {    // write mode
            recFlag = 1;
            address = 0;
            buzzer(2, 100);
        }
        if (GPIO.F3 == 0) {    // read mode
            buzzer(2, 100);
            address = 0;
            for (cnt = 0; cnt < 256; cnt += 2) {
                lowData = Eeprom_Read(address++);
                Delay_ms(20);
                hiData = Eeprom_Read(address++);
                Delay_ms(20);
                ad0 = lowData | ((hiData << 8) & 0x300);
                //
                temp = ad0 * 4;
                temp = (((4120 - temp) * 10) / 16) - 400;
                WordToStr(temp, buf);
                buf[0] = buf[1];
                buf[1] = buf[2];
                buf[2] = buf[3];
                buf[3] = '.';
                Soft_Uart_Write_String(&buf[1]);
                Soft_Uart_Write_String("\n\r");
                Delay_ms(100);
            }
```

```
            address = 0;
            buzzer(3, 100);
        }
        if (startFlag == 0)
            continue;
        startFlag = 0;
        buzzer(1, 100);
        //
        ad0 = Adc_Read(0);
        //
        temp = ad0 * 4;
        temp = (((4120 - temp) * 10) / 16) - 400;
        WordToStr(temp, buf);
        buf[0] = buf[1];
        buf[1] = buf[2];
        buf[2] = buf[3];
        buf[3] = '.';
        Soft_Uart_Write_String(&buf[1]);
        Soft_Uart_Write_String("\n\r");
        //
        if (recFlag == 1) {
            Eeprom_Write(address++, (ad0 & 0xFF));
            Delay_ms(20);
            Eeprom_Write(address++, ((ad0 >> 8) & 0x03));
            Delay_ms(20);
            //
            if (address >= 256) {
                recFlag = 0;
                address = 0;
                buzzer(3, 100);
            }
        }
    }
}

//*****************************************************************************
*
```

動作確認

PCのハイパータミナルを使って動作確認をしました ※PCとの接続は、レベル変換が必要となります。別途作成しましたRS232Cレベル変換ユニットを利用してPCと接続します。