温度データロガ□V3

概要

前回製作した、温度データロガ□V2を改良しました。 <改良ポイント>

- 測定周期の時間間隔を、1秒または1分の選択を可能とする。
- 測定周期の時間精度を、実用上問題ないレベルに上げる。

動作原理

PIC12F683は、ピン数が少ないので、起動時のスイッチの状態で、測定周期を選択させるようにしました。

- 測定周期1分:スイッチを押さずに電源を入れる。
- 測定周期1秒:スイッチを押しながら電源を入れる。

測定周期の精度を上げるために、リアルタイムクロックモジュール[RTC-8564NB]を搭載しました。

- 秒データまたは分データの変化を検出します。

<処理の流れ>

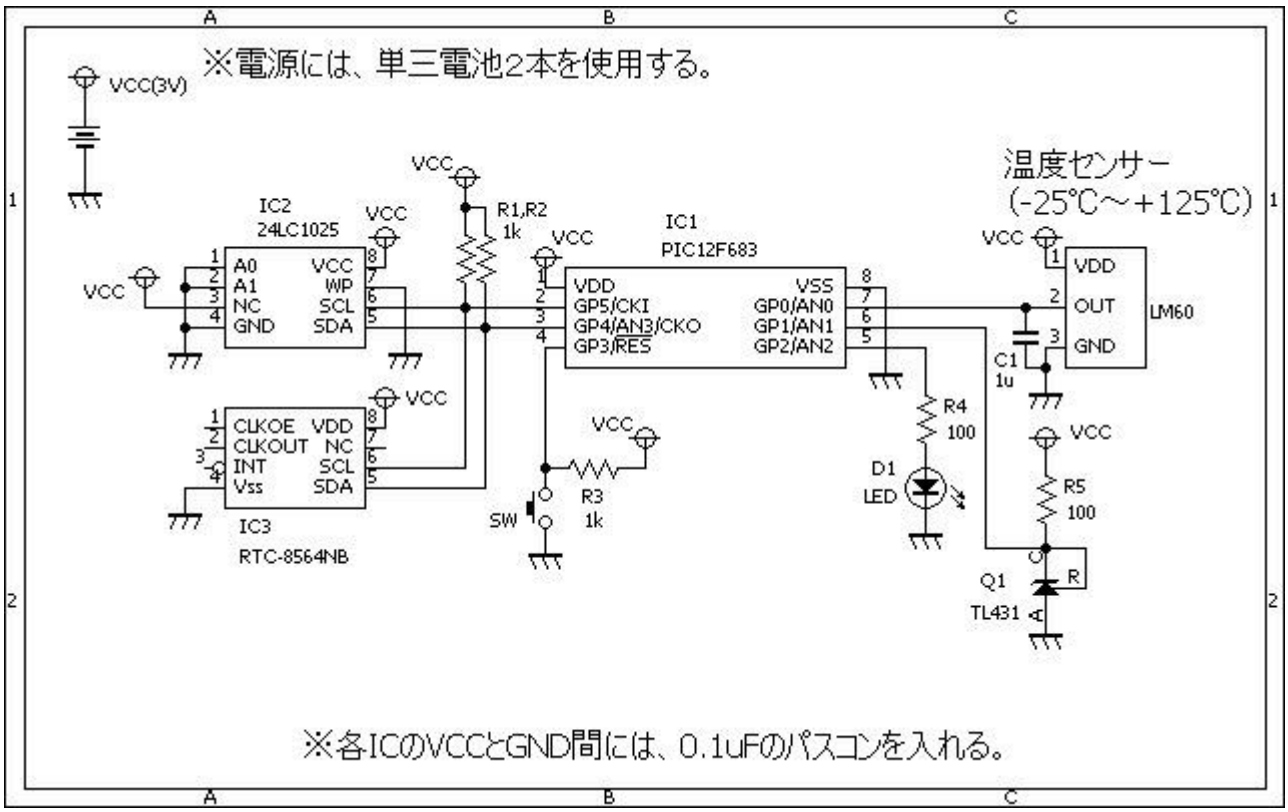
1. 内臓モジュールを初期化する。
2. スイッチが押された状態(0)であれば、測定周期を1秒とする。
3. スイッチが離された状態(1)であれば、測定周期を1分とする。
4. スイッチが押されるのを待つ。
5. 温度を100回測定し、その平均を求める。
6. 結果をEEPROMに書き込む。
   ※格納フォーマットは、1データ8バイト構成で、次のようになります。
   “29.8”+CR+LF
   “-12.4”+CR+LF
7. 正確な測定周期(1秒または1分)を得る。※下記参照
8. スイッチが押されるまで、3.5.を繰り返す。
9. スイッチが押されると測定を停止し、4.に戻る。

<正確な測定周期(1秒または1分)を得る方法>

1. RTCより、秒データを取得する(S1)
2. RTCより、秒データを取得する(S2)
3. S1とS2が同じ値であれば、2.に戻る。
4. S1とS2が異なる値であれば、1秒経過したとみなす。

回路図

前回製作した、温度データロガ□V2に、[RTC-8564NB]を、I2C接続しただけです。



ソースコード

ThermoLoggerV2.c

```
//*****************************************************************************
*
/*
   『温度データロガ(ThermoLoggerV2)』
  ■概要
    1分周期：スイッチ(SW)を押さずに起動する。
        約11日間の温度データを記録します。
          １３１０７２バイト÷8バイト÷60÷24時間

    1秒周期：スイッチ(SW)を押しながら起動する。
        約4.5時間の温度データを記録します。
          １３１０７２バイト÷8バイト÷3600

    格納フォーマットは、1データ8バイト構成で、次のようになります。
□□□□”△△□□□□”□CR□LF
□□□□”△□□□□□”□CR□LF
*/
//*****************************************************************************
*

#define       SW        GPIO.F3
#define       LED        GPIO.F2

#define       ON        1
```

```
#define       OFF        0

#define       CR        0x0D
#define       LF        0x0A

#define       ACK        1
#define       NO_ACK    0

#define       STAT_START        1
#define       STAT_STOP        0

#define       CYCLE_SEC        1
#define       CYCLE_MIN        60

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

unsigned short    RTC_8564_Read(unsigned short addr)
{
    unsigned short dat;
    //
    Soft_I2C_Start();
    Soft_I2C_Write(0xA2);
    Soft_I2C_Write(addr);
    Soft_I2C_Start();
    Soft_I2C_Write(0xA3);
    dat = Soft_I2C_Read(NO_ACK);
    Soft_I2C_Stop();
    //
    return (dat);
}

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

void   RTC_8564_Write(unsigned short addr, unsigned short dat)
{
    Soft_I2C_Start();
    Soft_I2C_Write(0xA2);
    Soft_I2C_Write(addr);
    Soft_I2C_Write(dat);
    Soft_I2C_Stop();
}

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

void   EEPROM_24LC1025_Page_Write(unsigned long addr, unsigned short
*buf, unsigned short len)
{
    unsigned    short    cnt;
```

```
    //
    Soft_I2C_Start();
    if ((addr & 0x10000) == 0)
        Soft_I2C_Write(0xA0);
    else
        Soft_I2C_Write(0xA8);
    Soft_I2C_Write((addr >> 8) & 0xFF);
    Soft_I2C_Write(addr & 0xFF);
    for (cnt = 0; cnt < len; cnt++) {
        Soft_I2C_Write(buf[cnt]);
    }
    Soft_I2C_Stop();
}

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

void main()
{
    static    unsigned    char    buf[10], cnt;
    static    unsigned    double    ad;
    static    unsigned    long    addr;
    static    unsigned    short    oldDat, newDat, runStatus,
cycleMode;
    //
    CMCON0 = 0b00000111;
    ANSEL.ANS0 = 1;
    ANSEL.ANS1 = 0;
    ANSEL.ANS2 = 0;
    ANSEL.ANS3 = 0;
    ADCON0.VCFG = 1;
    TRISIO = 0b00001011;
    OSCCON = 0b01110000;
    //
    Soft_I2C_Config(&GPIO, 4, 5);
    //
    RTC_8564_Write(0x00, 0x00);
    //
    runStatus = STAT_STOP;
    if (SW == 1)
        cycleMode = CYCLE_MIN;
    else
        cycleMode = CYCLE_SEC;
    //
    for (cnt = 0; cnt < 10; cnt++) {
        LED = ON;
        Delay_ms(50);
        LED = OFF;
        Delay_ms(50);
    }
```

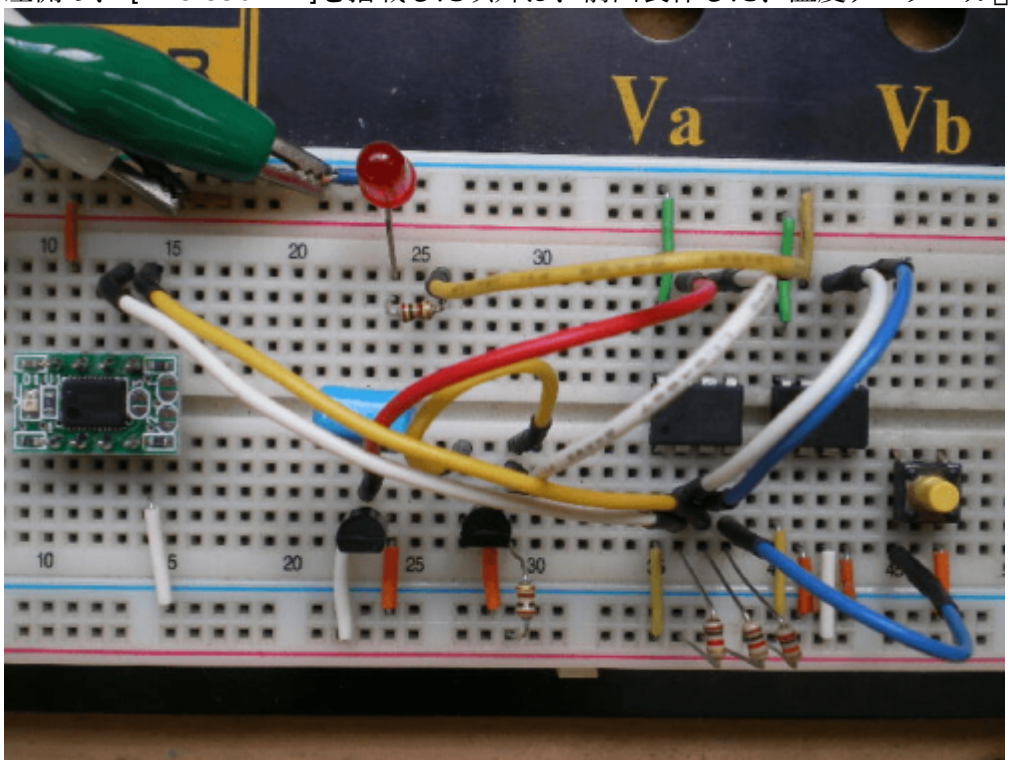
```
    //
    while (1) {
        //スイッチが押されるのを待つ。
        while (SW == 1)
            Delay_ms(10);
        runStatus = STAT_START;
        //RTCの秒または時間を取得する。
        if (cycleMode == CYCLE_MIN)
            oldDat = RTC_8564_Read(0x03);
        else
            oldDat = RTC_8564_Read(0x02);
        addr = 0;
        while (runStatus == STAT_START) {
            LED = ON;
            Delay_ms(100);
            LED = OFF;
            //温度を100回測定し、その平均を求める。
            ad = 0;
            for (cnt = 0; cnt < 100; cnt++) {
                ad += Adc_Read(0);
                Delay_ms(1);
            }
            ad = ad / 100.0;
            ad = ((((ad * 2.4365234375) - 268.0) / 6.25) - 25.0) *
10.0;
            //結果をEEPROMに書き込む。
            IntToStr(ad, buf);
            buf[6] = buf[5];
            buf[5] = '.';
            buf[7] = CR;
            buf[8] = LF;
            EEPROM_24LC1025_Page_Write(addr, &buf[1], 8);
            addr += 8;
            if (addr >=0x20000)
                break;
            //RTCの秒または分が変化したかをチェックする。
            while (1) {
                if (cycleMode == CYCLE_MIN)
                    newDat = RTC_8564_Read(0x03);
                else
                    newDat = RTC_8564_Read(0x02);
                if (oldDat != newDat) {
                    oldDat = newDat;
                    break;
                }
                //
                if (SW == 0) {
                    runStatus = STAT_STOP;
                    break;
                }
                //
```

```
                Delay_ms(10);
            }
        }
        //
        for (cnt = 0; cnt < 10; cnt++) {
            LED = ON;
            Delay_ms(50);
            LED = OFF;
            Delay_ms(50);
        }
    }
}

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

動作確認

左側の、 [RTC-8564NB]を搭載した以外は、前回製作した、温度データロガ□V2と同じです。



※測定周期の精度が向上したので、実用性がかなり向上しました。