温度計(温度制御機能付き)

概要

温度センサ(LM35DZ)を利用した温度計を製作しました。温度表示だけでは面白くありませんので、 以下のような機能を実装しました。

- 現在温度、最小温度、最大温度、平均温度を表示する。
- 温度制御機能を実装する。(設定した閾値を上回ると制御出力が出る)

動作原理

- 温度センサーにはLM35DZを使用します。
- その出力を増幅せずにPIC入力とします。(10倍程度に増幅するのが一般的のようですが…)
- PICのA/D変換のVref+を+2.5Vとします。
- これで精度が2.44mV(2.5V÷1024)になります。
- LM35DZの出力は、温度係数が10.0mV/℃ なので約0.25℃単位の測定が出来ます。(充分ですね)
- 温度表示画面では、現在温度、最小温度、最大温度、平均温度を計算し1秒周期で表示します。
- 温度制御画面では、閾値(threshold)の変更や、温度制御(上限制御のみ実装)をするか否かの設定が出来

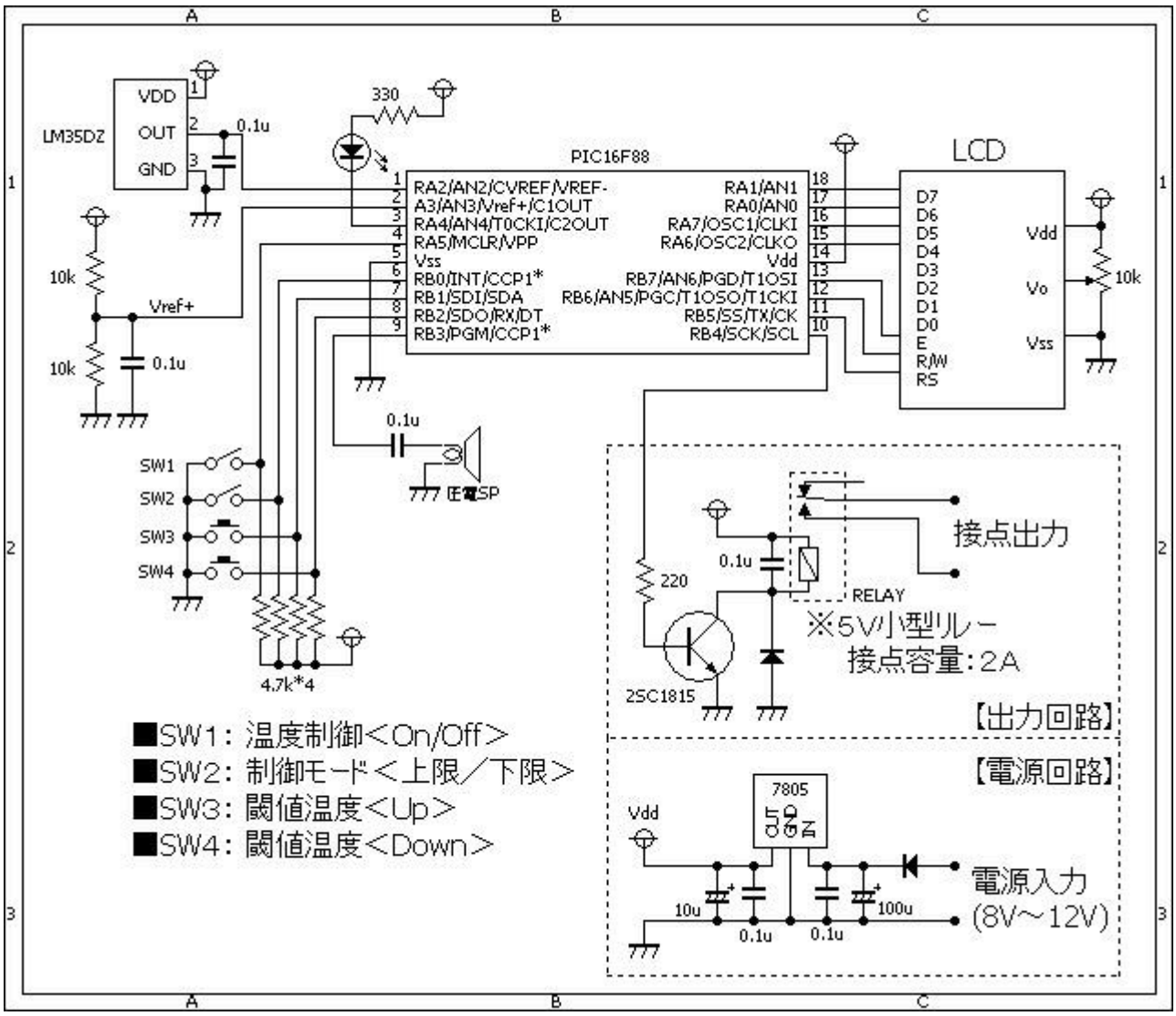
ます。

- 閾値を上回ると、ブザー音(0.1秒)、LED(点滅周期が短くなる)、リレーONを出力します。
- 閾値を下回ると、ブザー音(0.1秒)、LED(点滅周期が遅くなる)、リレーOFFを出力します。
  ヒステリシス制御します。例えば、閾値が22℃である場合、
  リレーがONとなるのは、現在温度が23℃以上になったときです。
  リレーがOFFとなるのは、現在温度が21℃以下になったときです。
- 閾値は、PICのEEPROMに記憶しますので電源をOFFにしても、次回ON時には閾値を再設定する必要はあり

ません。

※温度制御には、制御の安定化のために、ヒステリシス制御を採用したかったのですが、mikro-Cのフリー制限(2Kワード)を超えてしまい諦めました。→何とか実装できました。 ※同様に、温度制御(下限制御)も実装を諦めました。

回路図



ソースコード

スイッチ入力、平均温度等**の**見直**しをした**最終版

ThermoMeter2.c

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

#define       RELAY        PORTB.F4

#define       SW1            PORTA.F5
#define       SW2            PORTB.F0
#define       SW3            PORTB.F1
#define       SW4            PORTB.F2

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

static unsigned    int        nowData, maxData, minData, aveData,
```

```
thresholdData;
static unsigned    char    buf[8], sw3, sw4;

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

void   interrupt(){
    if (PIR1.TMR1IF == 1) {
        PIR1.TMR1IF = 0;
        PORTA.F4 = ~PORTA.F4;
        if (sw3 == 0)
            sw3 = (SW3 == 0) ? 1 : 0;
        if (sw4 == 0)
            sw4 = (SW4 == 0) ? 1 : 0;
    }
}

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

void   IntToStrEx(int number, char *buf)
{
        IntToStr(number, buf);
        buf[7] = 0x00;
        buf[6] = buf[5];
        buf[5] = '.';
}

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

void   buzzer()
{
    Pwm_Start();
    Delay_ms(100);
    Pwm_Stop();
}

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

void   ThermoCntl()
{
    //温度制御をするかしないかを判断する。
    if (SW2 == 1) {
        Lcd_Custom_Out(1, 9, "<----->");     //温度制御しない。
        RELAY = 0;
        T1CON.T1CKPS1 = 1;
    } else {
        Lcd_Custom_Out(1, 9, "<Upper>");     //温度制御する。
        //制御出力をONにするかを判断する。
```

```
        if ((nowData > thresholdData) && (RELAY == 0)) {
            buzzer();
            RELAY = 1;
            T1CON.T1CKPS1 = 0;
            T1CON.T1CKPS0 = 1;
        }
        //制御出力をOFFにするかを判断する。ヒステリシス制御！
        if ((nowData < (thresholdData - 10)) && (RELAY == 1)) {
            buzzer();
            RELAY = 0;
            T1CON.T1CKPS1 = 1;
            T1CON.T1CKPS0 = 0;
        }
    }
    //
    if (sw3 == 1) {     //閾値を＋１する。
        sw3 = 0;
        if (thresholdData < 990)
            thresholdData = ((thresholdData / 10) + 1) * 10;
        Eeprom_Write(0, thresholdData & 0xFF);
        Eeprom_Write(1, ((thresholdData >> 8) & 0xFF));
    }
    //
    if (sw4 == 1) {     //閾値をー１する。
        sw4 = 0;
        if (thresholdData > 0)
            thresholdData = ((thresholdData / 10) - 1) * 10;
        Eeprom_Write(0, thresholdData & 0xFF);
        Eeprom_Write(1, ((thresholdData >> 8) & 0xFF));
    }
    //閾値温度を表示する。
    IntToStrEx(thresholdData, buf);
    Lcd_Custom_Out(2, 1, "threshold->");
    Lcd_Custom_Out(2, 12, &buf[3]);
}

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

void   main()
{
    unsigned    int         ad2, tmp;
    unsigned    char    cnt, flg;
    //
    OSCCON = 0b01110000;     //内臓クロックを8Mhzに設定する。
    CMCON = 0b00000111;         //コンパレータは使用しない。
    ANSEL = 0b00000100;         // A/D変換はAN2を使用する。
    TRISA = 0b00101100;
    TRISB = 0b00000111;
    //タイマー１を設定する。
```

```
    PIE1.TMR1IE = 1;
    PIR1.TMR1IF = 0;
    T1CON.T1CKPS1 = 1;
    T1CON.T1CKPS0 = 0;
    T1CON.TMR1ON = 1;
    //
    ADCON1.VCFG1 = 1;
    ADCON1.VCFG0 = 0;
    //
    Lcd_Custom_Config(&PORTA, 6, 7, 0, 1, &PORTB, 5, 6, 7);
    Lcd_Custom_Cmd(LCD_CURSOR_OFF);
    Lcd_Custom_Cmd(Lcd_Clear);
    //
    Pwm_Init(5000);         // 5Khz
    tmp = (PR2 << 2) / 2;
    CCPR1L = tmp >> 2;
    CCP1CON.F6 = tmp & 0b00000001;
    CCP1CON.F7 = (tmp & 0b00000010) >> 1;
    buzzer();
    //
    RELAY = 0;
    nowData = 0;
    maxData = 0;             // 0
    minData = 1000;          // 100
    aveData = 0;             // 0
    thresholdData = Eeprom_Read(1);
    thresholdData = thresholdData << 8;
    thresholdData = thresholdData | Eeprom_Read(0);
    if ((thresholdData < 0) || (thresholdData > 990))
        thresholdData = 0;
    cnt = 0;
    sw3 = sw4 = 0;
    flg = 0;
    //
    INTCON.PEIE = 1;     // これ以降の処理で割り込みを許可する。
    INTCON.GIE = 1;          // これ以降の処理で割り込みを許可する。
    //
    while (1) {
        for (cnt = 0; cnt < 60; cnt++) {
            ad2 = Adc_Read(2);
            nowData = (int)((double)ad2 * 2.44);
            maxData = maxData < nowData ? nowData : maxData;
            minData = minData > nowData ? nowData : minData;
            aveData += nowData;
            //画面をクリアする。
//            Lcd_Custom_Cmd(Lcd_Clear);
            //現在温度を表示する。
            IntToStrEx(nowData, buf);
            Lcd_Custom_Out(1, 1, "now");
            Lcd_Custom_Out(1, 4, &buf[3]);
            //温度制御画面に切り替えるかをチェックする。
```

```
            if (SW1 == 0) {
                ThermoCntl();
                //
                Delay_ms(1000);
                continue;
            }
            //最大温度を表示する。
            IntToStrEx(maxData, buf);
            Lcd_Custom_Out(2, 8, " max");
            Lcd_Custom_Out(2, 12, &buf[3]);
            //最小温度を表示する。
            IntToStrEx(minData, buf);
            Lcd_Custom_Out(2, 1, "min");
            Lcd_Custom_Out(2, 4, &buf[3]);
            //平均温度を表示する。
            if (flg == 0)
                tmp = aveData / (cnt + 1);
            else
                tmp = aveData / (cnt + 2);          //前回の平均温度を考慮する。
            IntToStrEx(tmp, buf);
            Lcd_Custom_Out(1, 9, "ave");
            Lcd_Custom_Out(1, 12, &buf[3]);
            //
            Delay_ms(1000);
        }
        aveData = aveData / 60;          //前回の平均温度を考慮する。（引き継ぐ）
        flg = 1;
    }
}

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

(旧版) ヒステリシス制御に対応した版です。

ThermoMeter2old2.c

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

#define       RELAY        PORTB.F4

#define       SW1            PORTA.F5
#define       SW2            PORTB.F0
#define       SW3            PORTB.F1
#define       SW4            PORTB.F2

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

```
unsigned   int        nowData, maxData, minData, aveData,
thresholdData;
unsigned   char    buf[10];

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

void   interrupt(){
    if (PIR1.TMR1IF == 1) {
        PIR1.TMR1IF = 0;
        PORTA.F4 = ~PORTA.F4;
    }
}

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

void   IntToStrEx(int number, char *buf)
{
        IntToStr(number, buf);
        buf[7] = 0x00;
        buf[6] = buf[5];
        buf[5] = '.';
}

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

void   buzzer()
{
    Pwm_Start();
    Delay_ms(100);
    Pwm_Stop();
}

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

void   ThermoCntl()
{
```

```
    //温度制御をするかしないかを判断する。
    if (SW2 == 1) {
        Lcd_Custom_Out(1, 9, "<----->");     //温度制御しない。
        RELAY = 0;
        T1CON.T1CKPS1 = 1;
    } else {
        Lcd_Custom_Out(1, 9, "<Upper>");     //温度制御する。
        //制御出力をONにするかを判断する。
        if ((nowData > thresholdData) && (RELAY == 0)) {
            buzzer();
            RELAY = 1;
            T1CON.T1CKPS1 = 0;
        }
        //制御出力をOFFにするかを判断する。ヒステリシス制御！
        if ((nowData < (thresholdData - 10)) && (RELAY == 1)) {
            buzzer();
            RELAY = 0;
            T1CON.T1CKPS1 = 1;
        }
    }
    //
    if (SW3 == 0) {     //閾値を＋１する。
        if (thresholdData < 990)
            thresholdData = ((thresholdData / 10) + 1) * 10;
        Eeprom_Write(0, thresholdData & 0xFF);
        Eeprom_Write(1, ((thresholdData >> 8) & 0xFF));
    }
    //
    if (SW4 == 0) {     //閾値をー１する。
        if (thresholdData > 0)
            thresholdData = ((thresholdData / 10) - 1) * 10;
        Eeprom_Write(0, thresholdData & 0xFF);
        Eeprom_Write(1, ((thresholdData >> 8) & 0xFF));
    }
    //閾値温度を表示する。
    IntToStrEx(thresholdData, buf);
    Lcd_Custom_Out(2, 1, "threshold->");
    Lcd_Custom_Out(2, 12, &buf[3]);
}

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

void   main()
{
    unsigned    int         ad2, tmp;
    unsigned    char    cnt;
    //
    OSCCON = 0b01110000;     //内臓クロックを8Mhzに設定する。
    CMCON = 0b00000111;         //コンパレータは使用しない。
```

```
    ANSEL = 0b00000100;         // A/D変換はAN2を使用する。
    TRISA = 0b00101100;
    TRISB = 0b00000111;
    //タイマー1を設定する。
    PIE1.TMR1IE = 1;
    PIR1.TMR1IF = 0;
    T1CON.T1CKPS0 = 1;
    T1CON.T1CKPS1 = 1;
    T1CON.TMR1ON = 1;
    //
    ADCON1.VCFG1 = 1;
    ADCON1.VCFG0 = 0;
    //
    Lcd_Custom_Config(&PORTA, 6, 7, 0, 1, &PORTB, 5, 6, 7);
    Lcd_Custom_Cmd(LCD_CURSOR_OFF);
//    Lcd_Custom_Cmd(Lcd_Clear);
    //
    Pwm_Init(5000);    // 5Khz
    tmp = (PR2 << 2) / 2;
    CCPR1L = tmp >> 2;
    CCP1CON.F6 = tmp & 0b00000001;
    CCP1CON.F7 = (tmp & 0b00000010) >> 1;
    buzzer();
    //
    RELAY = 0;
    nowData = 0;
    maxData = 0;
    minData = 1000;
    thresholdData = Eeprom_Read(1);
    thresholdData = thresholdData << 8;
    thresholdData = thresholdData | Eeprom_Read(0);
    if ((thresholdData < 0) || (thresholdData > 990))
        thresholdData = 0;
    cnt = 0;
    //
    INTCON.PEIE = 1;     //これ以降の処理で割り込みを許可する。
    INTCON.GIE = 1;     //これ以降の処理で割り込みを許可する。
    //
    while (1) {
        aveData = 0;
        for (cnt = 0; cnt < 60; cnt++) {
            Lcd_Custom_Cmd(Lcd_Clear);
            ad2 = Adc_Read(2);
            nowData = (int)((double)ad2 * 2.44);
            maxData = maxData < nowData ? nowData : maxData;
            minData = minData > nowData ? nowData : minData;
            aveData += nowData;
            //現在温度を表示する。
            IntToStrEx(nowData, buf);
            Lcd_Custom_Out(1, 1, "now");
            Lcd_Custom_Out(1, 4, &buf[3]);
```

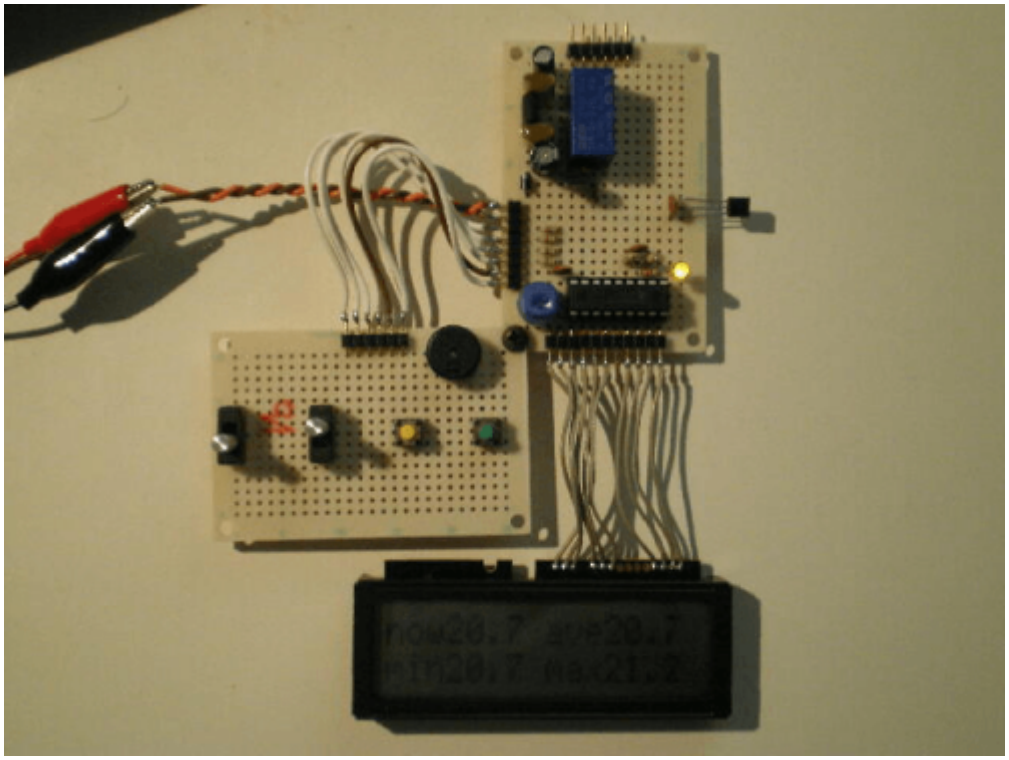
```
            //温度制御をするかをチェックする。
            if (SW1 == 0) {
                ThermoCntl();
                //
                Delay_ms(500);
                continue;
            }
            //最大温度を表示する。
            IntToStrEx(maxData, buf);
            Lcd_Custom_Out(2, 9, "max");
            Lcd_Custom_Out(2, 12, &buf[3]);
            //最小温度を表示する。
            IntToStrEx(minData, buf);
            Lcd_Custom_Out(2, 1, "min");
            Lcd_Custom_Out(2, 4, &buf[3]);
            //平均温度を表示する。
            IntToStrEx(aveData / (cnt + 1), buf);
            Lcd_Custom_Out(1, 9, "ave");
            Lcd_Custom_Out(1, 12, &buf[3]);
            //
            Delay_ms(500);
        }
    }
}

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

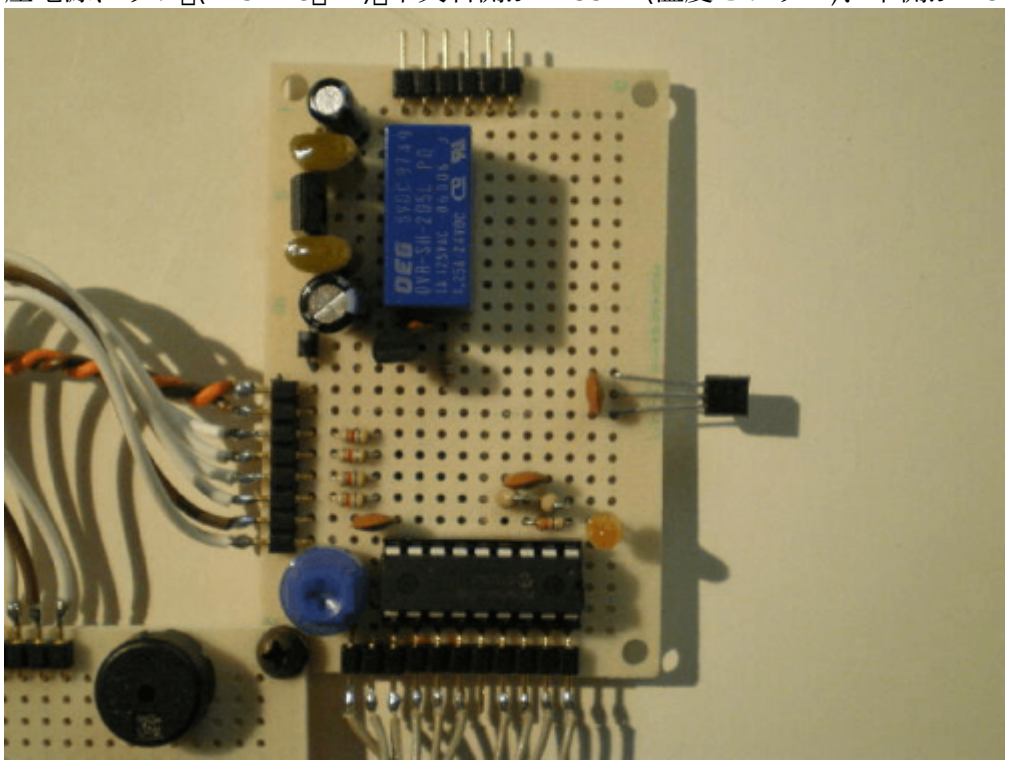
ヒステリシス制御に対応する前の版です。(編者注：保管されておらず、復元不可でした)

動作確認

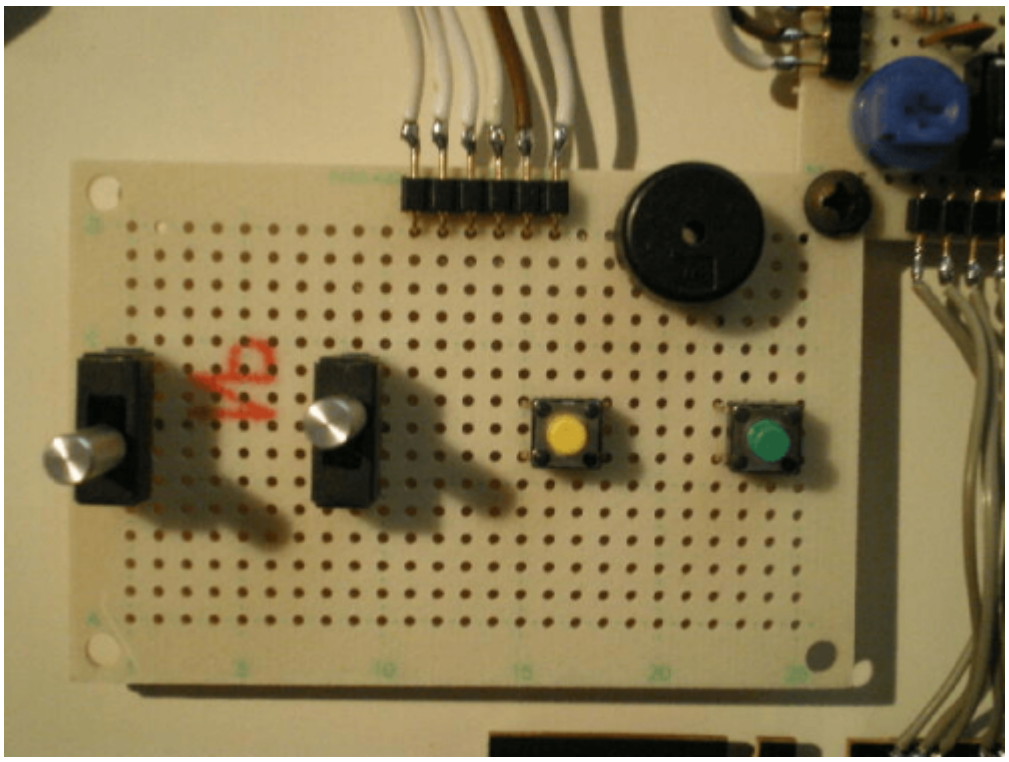
全体構成です。制御部、スイッチ部、表示部より構成されます。



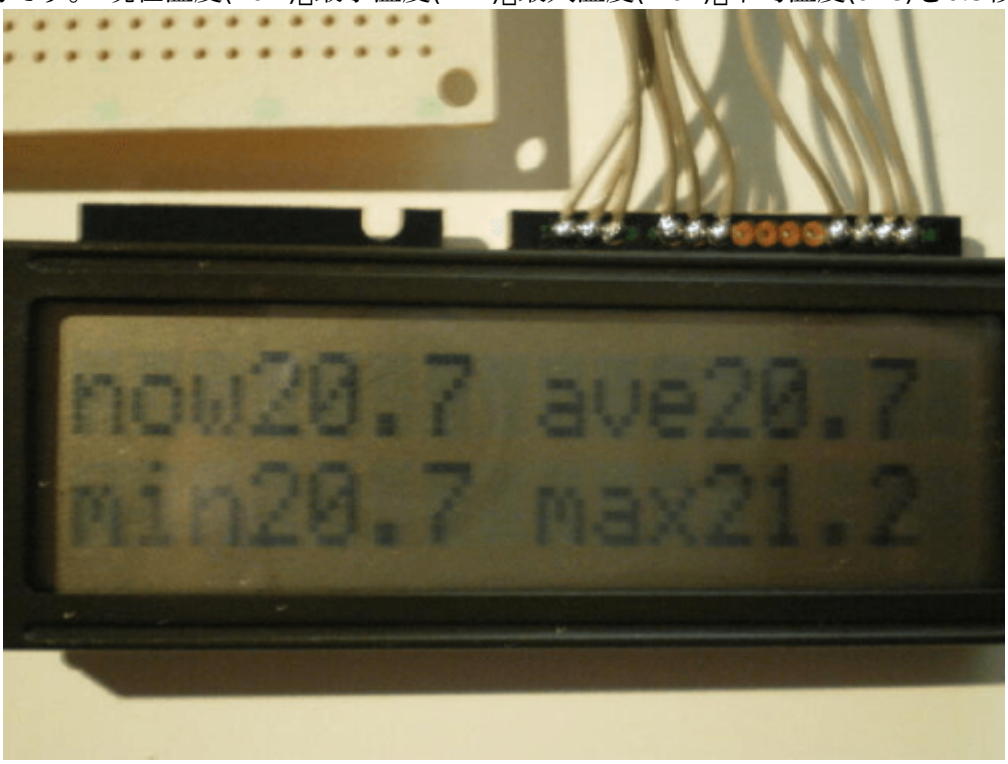
制御部です。 上側が低電圧電源、リレー(125VAC 1A) 中央右側がLM35DZ(温度センサー)、下側がPIC16F88です。



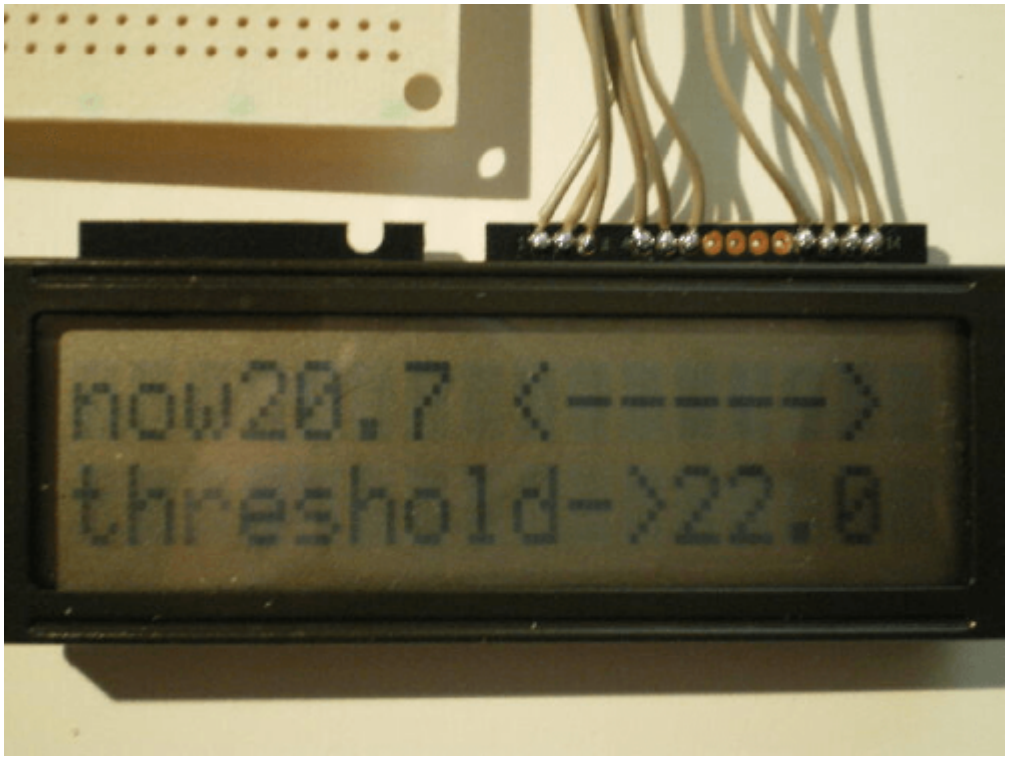
スイッチ部です。 左側から、画面切り替え、温度制御ON/OFF 閾値UP 閾値DOWN 黒い丸い物が圧電スピーカです。 右上の青いボリュームは、LCD(液晶)の輝度調整用の物です。



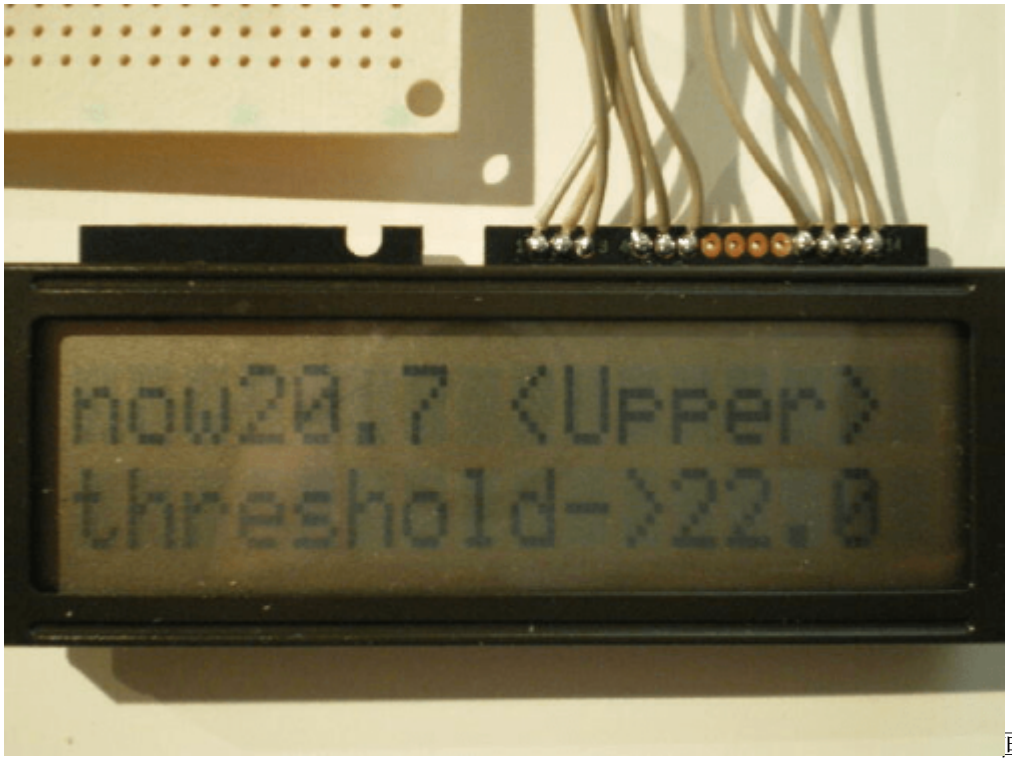
LCD(液晶表示)です。16文字2列の物です。 現在温度(now)、最小温度(min)、最大温度(max)、平均温度(ave)を0.5秒周期で表示しています。



温度制御画面です。 現在温度が、20.7℃、閾値(threshold)が22.0度、温度制御OFFの状態です。



温度制御画面です。 現在温度が、20.7℃、閾値(threshold)が22.0度、温度制御ONの状態です。



画像では紹介できませんが、現在温度が閾値を上回ると、ブザー音(0.1秒)□LED(点滅周期が短くなる)、 リレ□ONとなります。