

簡易湿度計V3(オートレンジ)

概要

以前に製作した、簡易湿度計の強化見直しを図りました。

<強化ポイント>

- 印加信号の発生回路を省き、PICの内蔵モジュールで代替を行う。
- レンジ切り替えを行い精度の向上を図る。
- 調整を不要とする。

動作原理

湿度センサーに直列に接続した抵抗(R:10kΩ)に、信号(500Hz矩形波)を印加すると、湿度に応じて、湿度センサーのインピーダンス®が変化するため、抵抗(R)の両端電圧(V)が変化します。これらの条件(結果)より、演算を行い湿度を求めます。

動作原理(ハードウェア)

湿度センサ HS-15Pを使用します。本センサーの詳細については、簡易湿度計を参照して下さい。

印加信号発生 PIC内蔵のCPPモジュールをPWMモードで使用し、約500Hzの矩形波(デューティ50%)を発生させます。この信号をHS-15Pの入力信号(V1)として印加します。この信号をPICでA/D変換して取り込みます。

信号の増幅 HS-15Pの出力信号(V2)を、インピーダンス変換(ボルテージフォロア)した後(V2同等)で、11倍に増幅(V3)します。11倍に増幅する前と後の信号を各々PICでA/D変換して取り込みます。

動作原理(ソフトウェア)

信号の取り込み V1 V2 V3の電圧をA/D変換して取り込みます。取り込む際には、信号の矩形波の立ち上がっている期間を、5回取り込みその平均値を求めます。

インピーダンスの算出 HS-15Pのインピーダンス®は、オームの法則で求めることができます。インピーダンス®=(V1-V2)÷(V2÷10kΩ)

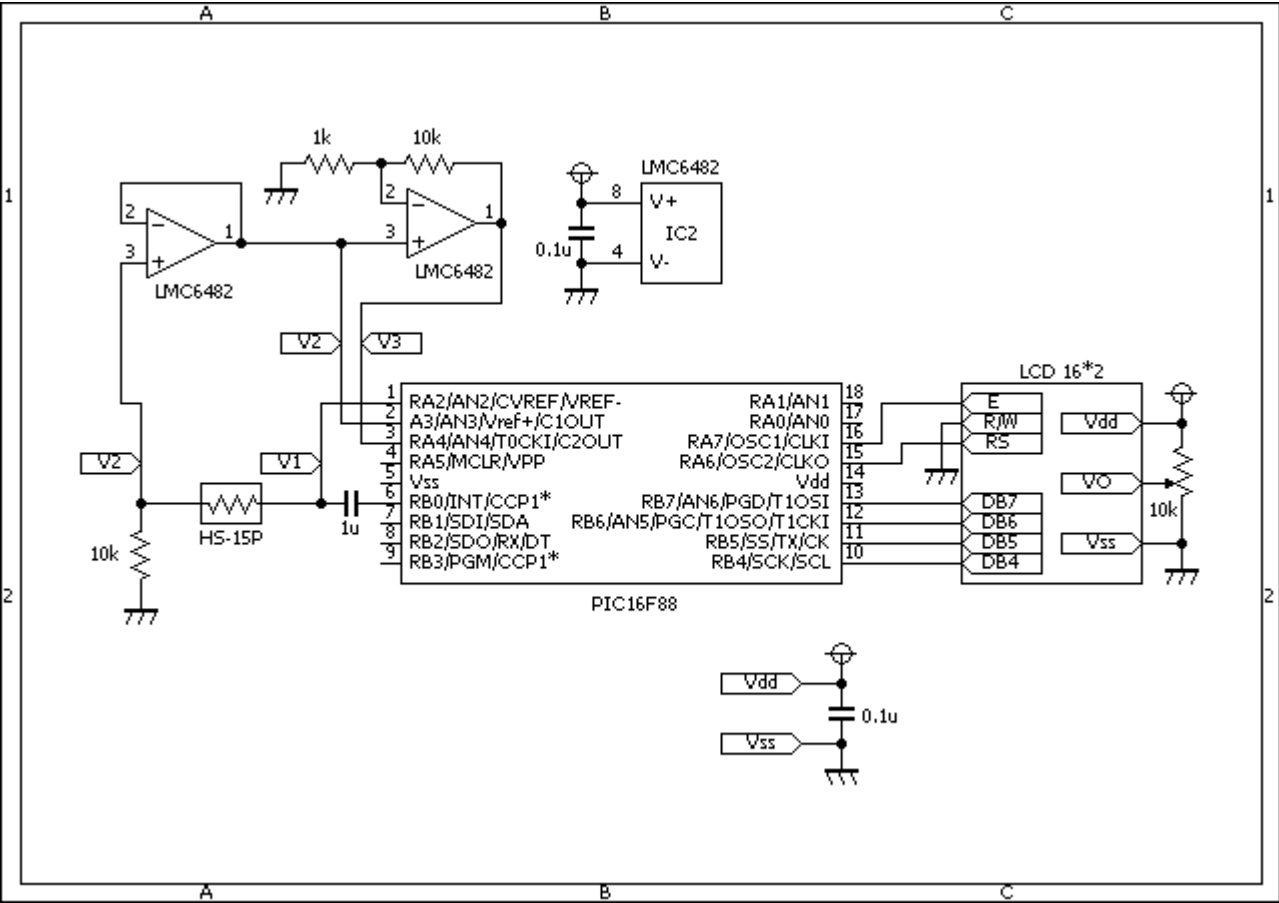
レンジ切り替え 11倍された電圧(V3)の値が、4900mV以下であれば、信号が飽和(saturate)していないと判断し、上記の式の V2の代わりに V3を1/11した値を、V2とみなし計算します。

インピーダンス 湿度への換算 算出したインピーダンス®を、下表のテーブルを使用して、簡易的な方法で湿度に変換します。尚、このテーブルで採用している値は、HS-15Pの25℃の時の特性値です。

湿度範囲		インピーダンス範囲		1/10単位のインピーダンス
20%	40%	8MΩ	220kΩ	389kΩ

湿度範囲		インピーダンス範囲		1/10単位のインピーダンス
40%	60%	220kΩ	23kΩ	9.85kΩ
60%	80%	23kΩ	3.5kΩ	975Ω
80%	100%	3.5kΩ	750Ω	138Ω

回路図



ソースコード

[hygro_meter_v3.c](#)

```
//*****
*
/*
< 簡易湿度計□□□オートレンジ >
*/
//*****
*
//LCD
sbit LCD_RS at RA6_bit;
sbit LCD_EN at RA7_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 TRISA6_bit;
sbit LCD_EN_Direction at TRISA7_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;
//
#define BYTE    unsigned short
#define WORD    unsigned int
#define DWORD   unsigned long
//*****
*
// ■ 関数宣言
extern void     main();
extern WORD     ADC_Get_Sample_Average(unsigned short channel);
//*****
*
// ■ A/D変換 (5回平均) 関数
WORD     ADC_Get_Sample_Average(unsigned short channel)
{
    WORD  ad;
    short cnt;
    //
    while (ADC_Get_Sample(2) > (1024 * 0.1)) {
    }
    while (ADC_Get_Sample(2) < (1024 * 0.9)) {
    }
    //
    ad = 0;
    for (cnt = 0; cnt < 5; cnt++) {
        ad += ADC_Get_Sample(channel);
    }
    return (ad / 5);
}
//*****
*
// ■ 抵抗 → 湿度変換関数(HS-15P)
struct tbl {
    short  s;
    short  e;
    long   s_r;
    long   e_r;
    long   n_r;
} hygro_tbl[4] = {
    {20, 40, 8000000, 220000, 389000},
    {40, 60, 220000, 23000, 9850},
    {60, 80, 23000, 3500, 975},
    {80, 100, 3500, 750, 138}
};

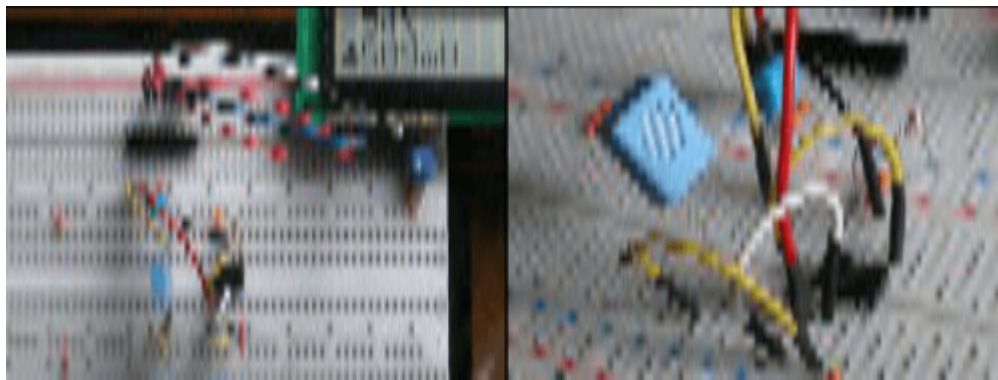
```

```
short  hygro_cnv(long h)
{
    if ((hygro_tbl[0].s_r >= h) && (hygro_tbl[0].e_r <= h)) {
        h = (h - hygro_tbl[0].e_r) / hygro_tbl[0].n_r;
        h = hygro_tbl[0].e - h;
        return (h);
    }
    if ((hygro_tbl[1].s_r >= h) && (hygro_tbl[1].e_r <= h)) {
        h = (h - hygro_tbl[1].e_r) / hygro_tbl[1].n_r;
        h = hygro_tbl[1].e - h;
        return (h);
    }
    if ((hygro_tbl[2].s_r >= h) && (hygro_tbl[2].e_r <= h)) {
        h = (h - hygro_tbl[2].e_r) / hygro_tbl[2].n_r;
        h = hygro_tbl[2].e - h;
        return (h);
    }
    if ((hygro_tbl[3].s_r >= h) && (hygro_tbl[3].e_r <= h)) {
        h = (h - hygro_tbl[3].e_r) / hygro_tbl[3].n_r;
        h = hygro_tbl[3].e - h;
        return (h);
    }
}

//*****
*
// ■ メイン関数
void  main()
{
    long    v1, v2, v3, r;
    char    buf[16];
    int     cnt;
    //
    OSCCON = 0b01110000;
    ANSEL  = 0b00011100;
    TRISA  = 0b00111111;
    TRISB  = 0b00000110;
    //
    ADC_Init();
    //
    Lcd_Init();
    Lcd_Cmd(_LCD_CURSOR_OFF);
    Lcd_Cmd(_LCD_CLEAR);
    Lcd_Out(1, 1, "hygrometer v3");
    Delay_ms(1000);
    Lcd_Cmd(_LCD_CLEAR);
    Lcd_Out(1, 4, "%");
    Lcd_Ch(1, 16, 0xF4);
    Lcd_Out(2, 5, "mV");
    Lcd_Out(2, 14, "mV");
    //
    PWM1_Init(500);           //500Hz
```

```
PWM1_Set_Duty(PR2 / 2);
PWM1_Start();
//
while (1) {
    v1 = 0;
    for (cnt = 0; cnt < 200; cnt++) {
        v1 += ADC_Get_Sample_Average(2);
    }
    v1 = v1 / 200;
    v2 = 0;
    for (cnt = 0; cnt < 200; cnt++) {
        v2 += ADC_Get_Sample_Average(3);
    }
    v2 = v2 / 200;
    v3 = 0;
    for (cnt = 0; cnt < 200; cnt++) {
        v3 += ADC_Get_Sample_Average(4);
    }
    v3 = v3 / 200;
    //
    v1 = (double)v1 * 4.8828125;
    v2 = (double)v2 * 4.8828125;
    v3 = (double)v3 * 4.8828125;
    if (v3 < 4900) {
        v2 = v3 / 11;
        Lcd_Chr(2, 16, '_');
    } else {
        Lcd_Chr(2, 16, ' ');
    }
    r = (double)(v1 - v2) / ((double)v2 / 10000);
    //
    ByteToStr(hygro_cnv(r), buf);
    Lcd_Out(1, 1, buf);
    LongToStr(r, buf);
    Lcd_Out(1, 8, &buf[3]);
    WordToStr(v1, buf);
    Lcd_Out(2, 1, &buf[1]);
    WordToStr(v2, buf);
    Lcd_Out(2, 9, buf);
}
}
//*****
*
```

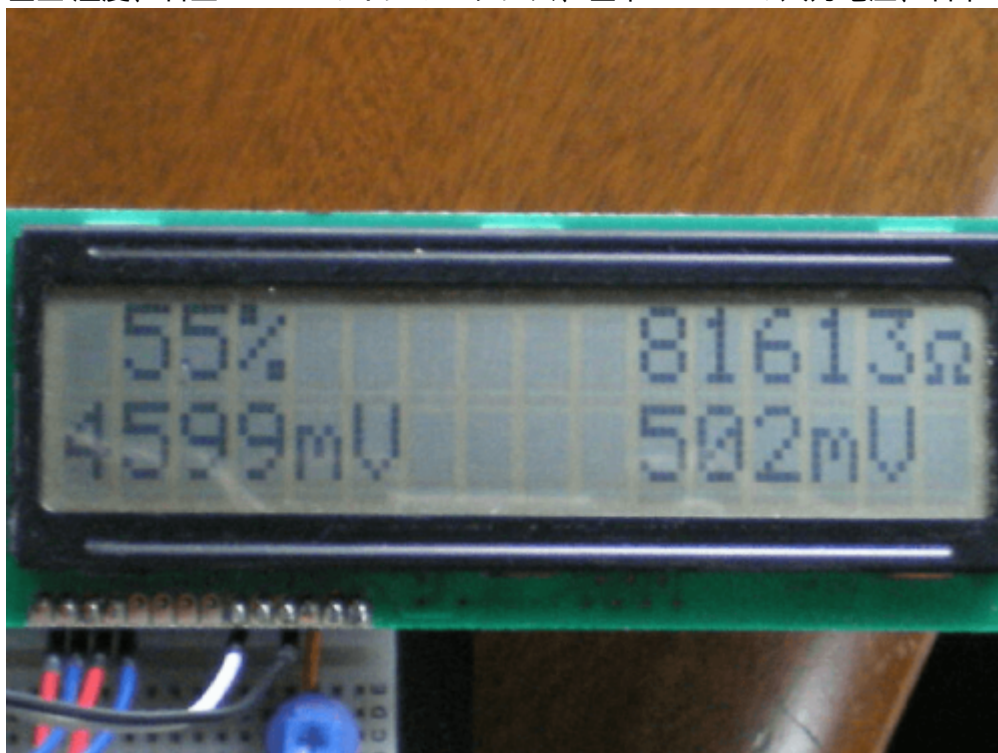
動作確認



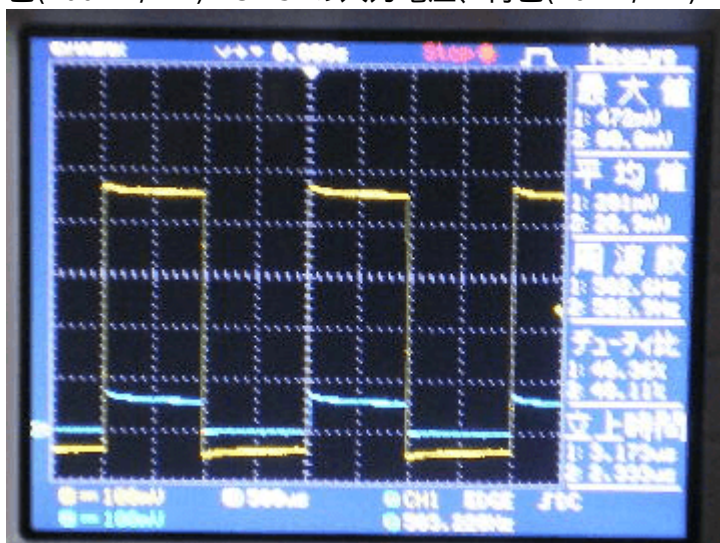
(編者注：壊れた画像しか

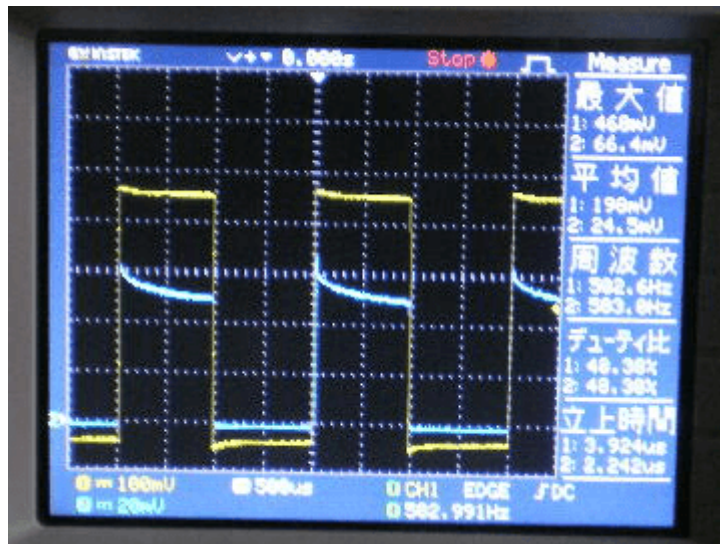
保管されておりませんでした)

左上:湿度、右上:HS-15Pのインピーダンス、左下:HS-15Pの入力電圧、右下:HS-15Pの出力電圧



左側<黄色(100mV/DIV):HS-15Pの入力電圧、青色(100mV/DIV):HS-15Pの出力電圧> 右側<黄色(100mV/DIV):HS-15Pの入力電圧、青色(20mV/DIV):HS-15Pの出力電圧>





<精度向上へ向けての課題>

- 周囲の温度に応じた、温度補償を行う必要があります。
- インピーダンスから湿度への変換方式を見直す必要があります。(対数演算など)

From:

<http://deepsky.jp/wiki/> - うごくといいな

Permanent link:

<http://deepsky.jp/wiki/doku.php?id=elechobby:picdic:pic16f88:151&rev=1588229434>

Last update: 2025/10/17 14:28

