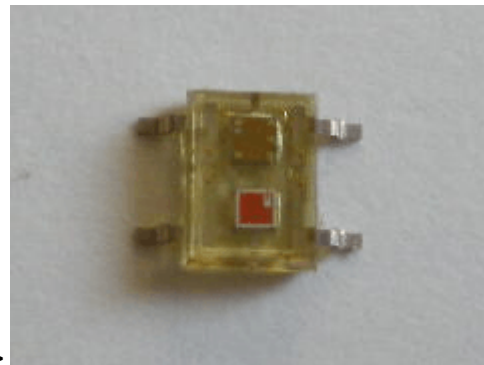
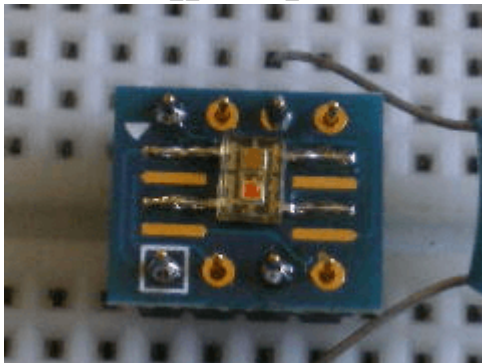


マルツパーツ館で1個283円で販売しています。(2011.1.2時点)

照度センサー 使用した温度センサーは、浜松フォトニクス社製の「S9705」です。詳細については、以前に製作した簡易照度計V2(S9705)を参照してください。S9705は、照度に応じたパルス(100Lux=50kHz)を出力するのでPICではTIMERレジスタを使用して周波数をカウントします。動作電圧は、2.7V~5.5Vです。



<照度センサー S9705 の概観と変換基板への実装例>



赤く見える部分がセンサー部です。

秋月電子通商で2個400円で販売しています。(2011.1.2時点)

◎SDC(SD Memory Card) FAT16対応のSDCを使用します。動作電圧は、2.7V~3.6Vです。

動作原理(ソフトウェア)

記録と停止

- 記録:停止中に、スイッチ(SW_START_STOP)を押下すると記録を開始します(LED点灯)
- 停止:記録中に、スイッチ(SW_START_STOP)を押下すると記録を停止します(LED消灯)

記録を繰り返す毎に、ファイル名(log_0001.txt~log_9999.txt)を切り替えていきます。 起動時に、


```
//*****  
*  
/*  
    『温度&照度ログ』  
    温度センサ→LM74(ナショセミ)  
    照度センサ→S9705(浜松ホトニクス)  
*/  
//*****  
*  
  
// マクロの定義  
#define BYTE          unsigned short  
#define WORD          unsigned int  
#define DWORD         unsigned long  
sbit SW              at RC0_bit;  
sbit SW_Direction    at TRISC0_bit;  
sbit LED             at RC1_bit;  
sbit LED_Direction   at TRISC1_bit;  
//LM74  
sbit LM74_CS at RA5_bit;  
sbit LM74_S0 at RA3_bit;  
sbit LM74_SC at RA4_bit;  
sbit LM74_CS_Direction at TRISA5_bit;  
//sbit LM74_S0_Direction at TRISA3_bit;  
sbit LM74_SC_Direction at TRISA4_bit;  
#define CR          0x0D  
#define LF          0x0A  
//*****  
*  
// 関数および変数の宣言  
extern void main();  
extern void init_sdc();  
extern void init_timer();  
extern int temperature_measurement();  
extern int temperature_convert(int temperature);  
extern void temperature_convert_2(int temperature, char *msg);  
extern DWORD illuminance_measurement();  
extern DWORD illuminance_convert(DWORD illuminance);  
extern void illuminance_convert_2(DWORD illuminance, char *msg);  
extern void run();  
extern short trigger();  
extern void interrupt();  
extern DWORD clock;  
extern short start_flg;  
//*****  
*  
char buf[40];  
WORD write_cnt = 0;  
char* file_name = "log_???.txt";  
//*****  
*
```

```
// メイン関数です。
void main()
{
    OSCCON.IRCF2 = 1;
    OSCCON.IRCF1 = 1;
    OSCCON.IRCF0 = 1;
    ANSELH.ANS11 = 0;
    ANSELH.ANS10 = 0;
    ANSELH.ANS9 = 0;
    ANSELH.ANS8 = 0;
    ANSEL.ANS7 = 0;
    ANSEL.ANS6 = 0;
    ANSEL.ANS5 = 0;
    ANSEL.ANS4 = 0;
    ANSEL.ANS3 = 0;
    TRISA = 0b11111111;
    TRISB = 0b11111111;
    TRISC = 0b11111111;
    //
    LED_Direction = 0;
    LED = 0;
    SW_Direction = 1;
    //
    LM74_CS_Direction = 0;
//    LM74_SO_Direction = 1;
    LM74_SC_Direction = 0;
    LM74_SC = 0;
    LM74_CS = 1;
    //
    init_sdc();
    init_timer();
    //
    while (1) {
        if (SW == 1) {
            continue;
        }
        while (SW == 0) {
            Delay_ms(100);
        }
        //
        clock = 0;
        start_flg = 1;
        LED = 1;
        //
        write_cnt++;
        WordToStr(write_cnt, buf);
        file_name[4] = (buf[1] == ' ') ? '0' : buf[1];
        file_name[5] = (buf[2] == ' ') ? '0' : buf[2];
        file_name[6] = (buf[3] == ' ') ? '0' : buf[3];
        file_name[7] = (buf[4] == ' ') ? '0' : buf[4];
        Mmc_Fat_Assign(file_name, 0xA0);
    }
}
```

```
        Mmc_Fat_Delete();
        Mmc_Fat_Set_File_Date(2010, 12, 25, 12, 34, 56);
        Mmc_Fat_Assign(file_name, 0xA0);
        Mmc_Fat_Rewrite();
        //
        run();
        //
        LED = 0;
        start_flg = 0;
    }
}
//*****
*
// 測定タイミングをチェックする関数です。
short trigger()
{
    while ((clock % 10) != 0) {
        Delay_ms(1);
        if (SW == 0) {
            while (SW == 0) {
                Delay_ms(100);
            }
            return (-1);
        }
    }
    return (0);
}
//*****
*
void ByteToStr2(unsigned short number, char *output)
{
    ByteToStr(number, output);
    output[0] = (output[1] == ' ') ? '0' : output[1];
    output[1] = output[2];
    output[2] = 0x00;
}
//*****
*
// 測定&記録を行う関数です。
void run()
{
    DWORD    tmp;
    BYTE    hh, mm, ss;
    int    temperature;
    DWORD    illuminance;
    //
    while (1) {
        if (trigger() == -1) {
            return;
        }
    }
    //
```

```
    tmp = clock;
    tmp /= 10;
    hh = tmp / 3600;
    mm = (tmp % 3600) / 60;
    ss = (tmp % 3600) % 60;
    ByteToStr2(hh, buf);
    buf[2] = ':';
    ByteToStr2(mm, &buf[3]);
    buf[5] = ':';
    ByteToStr2(ss, &buf[6]);
    //
    temperature = temperature_measurement();
    temperature = temperature_convert(temperature);
    temperature_convert_2(temperature, &buf[8]);
    //
    illuminance = illuminance_measurement();
    illuminance = illuminance_convert(illuminance);
    illuminance_convert_2(illuminance, &buf[14]);
    //
    buf[25] = CR;
    buf[26] = LF;
    Mmc_Fat_Write(buf, 27);
    //
    Delay_ms(100);
}

}

//*****
*
// 温度測定関数
int temperature_measurement()
{
    DWORD    temperature;
    short    cnt;
    //
    temperature = 0;
    LM74_CS = 0;
    for (cnt = 0; cnt < 16; cnt++) {
        LM74_SC = 1;
        if (LM74_S0 == 1) {
            temperature |= 1;
            temperature <<= 1;
        } else {
            temperature |= 0;
            temperature <<= 1;
        }
    }
    LM74_SC = 0;
}
LM74_CS = 1;
temperature >>= 1;
return (temperature);
}
```

```
//*****  
*  
// 温度換算関数(1)  
int    temperature_convert(int temperature)  
{  
    double d;  
    int    tmp;  
    //  
    if ((temperature & 0b1000000000000000) == 0) {  
        d = temperature >> 3;  
        d = d * 0.0625 * 10;  
    } else {  
        d = (temperature >> 3) & 0x0FFF;  
        d = (d - 4096) * 0.0625 * 10;  
    }  
    tmp = d;  
    return (tmp);  
}  
//*****  
*  
// 温度換算関数(2)  
void    temperature_convert_2(int temperature, char *msg)  
{  
    if (temperature >= 0) {  
        IntToStr(temperature, msg);  
        msg[0] = ' ';  
        msg[1] = ' ';  
        msg[2] = msg[3];  
        msg[3] = msg[4];  
        msg[4] = '.';  
    } else {  
        IntToStr(temperature, msg);  
        msg[0] = ' ';  
        msg[1] = '-';  
        msg[2] = (msg[3] == '-') ? ' ' : msg[3];  
        msg[3] = (msg[4] == '-') ? ' ' : msg[4];  
        msg[4] = '.';  
    }  
}  
//*****  
*  
// 照度測定関数  
short    freq_flg = 0;  
//  
DWORD    illuminance_measurement()  
{  
    DWORD    freq = 0;  
    BYTE    tmp_TMR0H, tmp_TMR0L;  
    //  
    T0CON.T0CS = 1;  
    T0CON.PSA = 1;
```

```

T0CON.TMR0ON = 0;
T0CON.T08BIT = 0;
INTCON.TMR0IF = 0;
TMR0H = 0;
TMR0L = 0;
freq = 0;
freq_flg = 1;
while (freq_flg != 0) {
    if (INTCON.TMR0IF == 1) {
        INTCON.TMR0IF = 0;
        freq++;
    }
}
if (INTCON.TMR0IF == 1) {
    INTCON.TMR0IF = 0;
    freq++;
}
tmp_TMR0L = TMR0L;
tmp_TMR0H = TMR0H;
freq = (freq * 65536) + (((DWORD)tmp_TMR0H) * 256) +
(DWORD)tmp_TMR0L;
return (freq * 10);
}
//*****
*
// 照度換算関数(1)
DWORD illuminance_convert(DWORD illuminance)
{
    return (illuminance / 5);
}
//*****
*
// 照度換算関数(2)
void illuminance_convert_2(DWORD illuminance, char *msg)
{
    LongToStr(illuminance, msg);
    msg[0] = msg[1];
    msg[1] = msg[2];
    msg[2] = msg[3];
    msg[3] = msg[4];
    msg[4] = msg[5];
    msg[5] = msg[6];
    msg[6] = msg[7];
    msg[7] = msg[8];
    msg[8] = '.';
}
//*****
*
//■□□□を初期化する関数です。
sfr sbit Mmc_Chip_Select at RB7_bit;
sfr sbit Mmc_Chip_Select_Direction at TRISB7_bit;

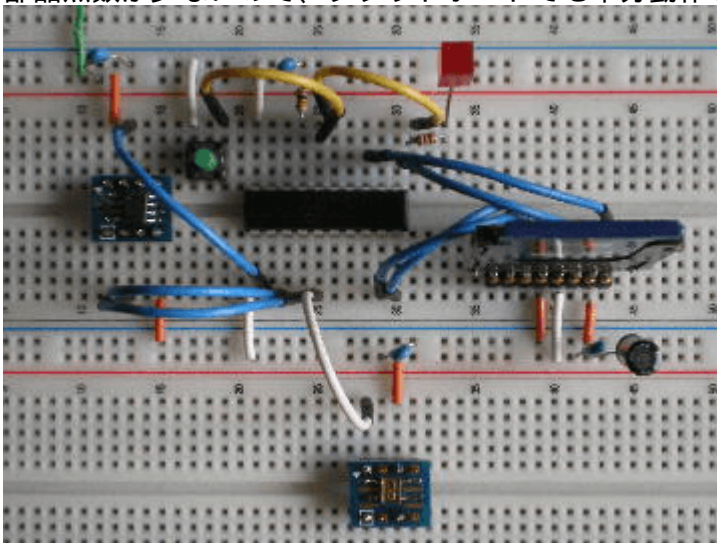
```

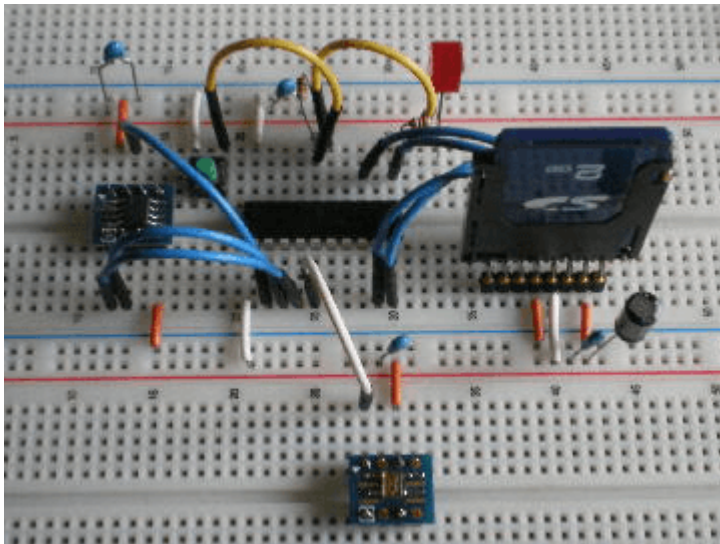
```
//
void  init_sdc()
{
    short  cnt;
    //
    SPI1_Init_Advanced(_SPI_MASTER_OSC_DIV64,
    _SPI_DATA_SAMPLE_MIDDLE, _SPI_CLK_IDLE_LOW, _SPI_LOW_2_HIGH);
    if (Mmc_Fat_Init()) {
        while (1) {
            LED = 1;
            Delay_ms(100);
            LED = 0;
            Delay_ms(100);
        }
    }
    SPI1_Init_Advanced(_SPI_MASTER_OSC_DIV4,
    _SPI_DATA_SAMPLE_MIDDLE, _SPI_CLK_IDLE_LOW, _SPI_LOW_2_HIGH);
    for (cnt = 0; cnt < 3; cnt++) {
        LED = 1;
        Delay_ms(500);
        LED = 0;
        Delay_ms(500);
    }
}
//*****
*
//■100msecの周期割り込みを発生させる関数です。
void  init_timer()
{
    // CCPの設定
    PIE1.CCP1IE = 1;
    PIR1.CCP1IF = 0;
    CCP1CON.CCP1M3 = 1;
    CCP1CON.CCP1M2 = 0;
    CCP1CON.CCP1M1 = 1;
    CCP1CON.CCP1M0 = 1;
    CCPR1L = 0x50;          // 100msec...クロックが16Mhzの時
    CCPR1H = 0xC3;        //
    100msec... (1÷16000000)*4*8*50000(0xC350)
    // TIMER1の設定
    PIE1.TMR1IE = 0;
    PIR1.TMR1IF = 0;
    TMR1L = 0;
    TMR1H = 0;
    T1CON.TMR1CS = 0;
    T1CON.T1CKPS0 = 1;
    T1CON.T1CKPS1 = 1;
    T1CON.TMR1ON = 1;
    //
    INTCON.PEIE = 1;
```

```
    INTCON.GIE = 1;
}
//*****
*
// 割り込み関数です。
DWORD   clock = 0;
short   start_flg = 0;
//
void     interrupt()
{
    if (PIR1.CCP1IF == 1) {
        PIR1.CCP1IF = 0;
        //
        if (start_flg == 1) {
            clock++;
        }
        if (freq_flg == 2) {
            freq_flg = 0;
            T0CON.TMR0ON = 0;
        }
        if (freq_flg == 1) {
            freq_flg = 2;
            T0CON.TMR0ON = 1;
        }
    }
}
//*****
*
```

動作確認

部品点数が少ないので、ブレッドボードでも十分動作を確認することができます。





記録したログファイルの内容です。各行に記録された内容は、左側から、測定経過時間、温度、照度で

時間	温度	照度
00:00:01	14.9	243.92
00:00:02	14.9	242.36
00:00:03	15.0	299.92
00:00:04	14.9	253.82
00:00:05	14.9	257.08
00:00:06	15.0	385.20
00:00:07	15.0	421.50
00:00:08	15.0	421.16
00:00:09	15.0	421.18
00:00:10	15.0	420.74
00:00:11	15.0	420.40
00:00:12	14.9	420.56
00:00:13	15.0	420.58
00:00:14	15.0	420.72
00:00:15	15.0	418.76
00:00:16	15.0	419.88
00:00:17	15.0	420.22

す。

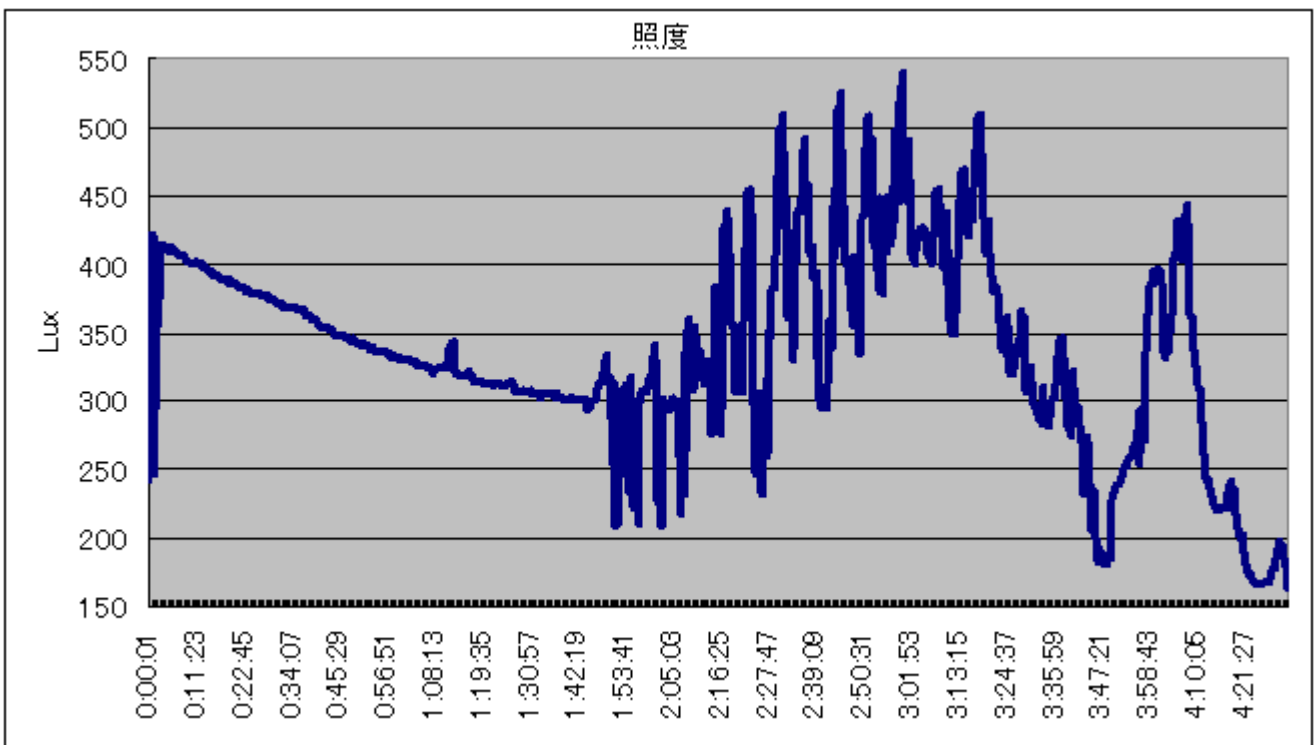
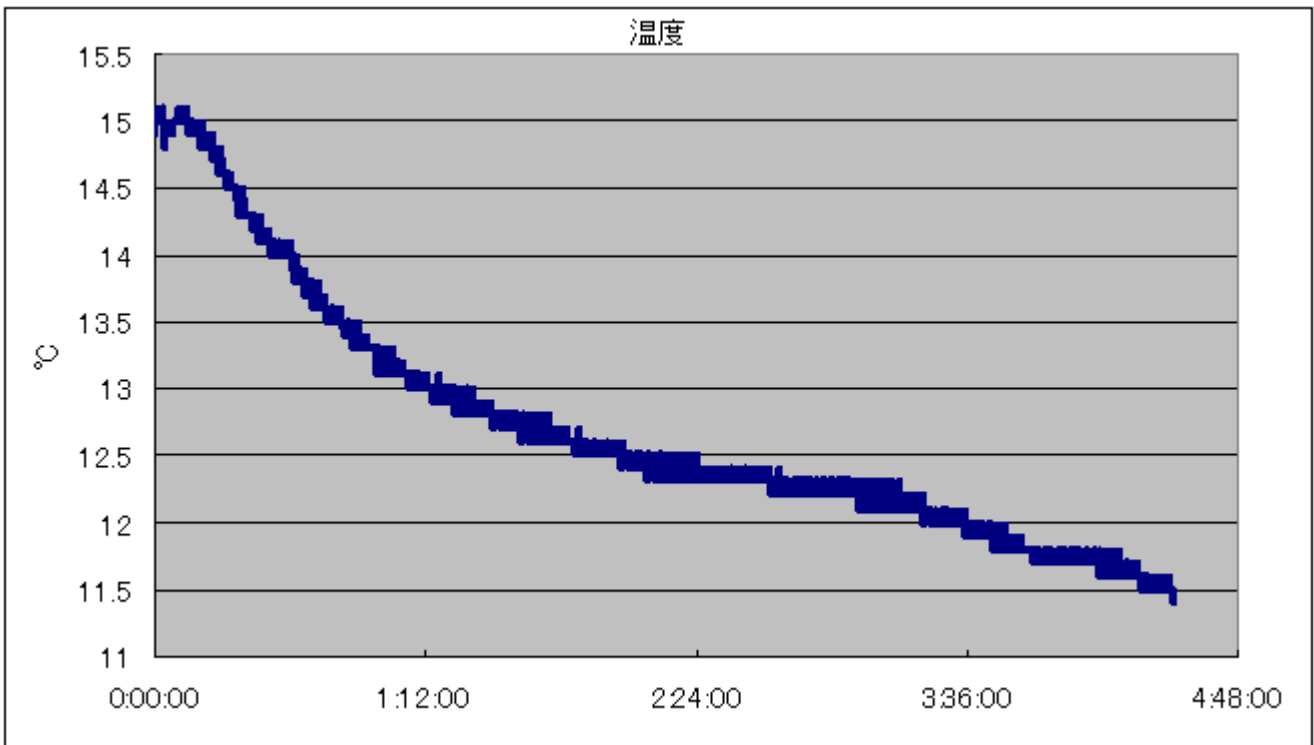


The screenshot shows a Microsoft Excel window titled "Microsoft Excel - 温度&照度ログ2". The menu bar includes "ファイル(F)", "編集(E)", "表示(V)", "挿入(I)", and "書式(O)". The toolbar contains various icons for file operations and editing. The active sheet is named "O17". The data is organized in a table with columns A, B, and C. Column A is labeled "経過時間" (Elapsed Time), Column B is "温度(°C)" (Temperature in °C), and Column C is "照度(Lux)" (Illuminance in Lux). The data spans from row 1 to row 16, with the last row partially cut off.

	A	B	C	D
1	経過時間	温度(°C)	照度(Lux)	
2	0:00:01	14.9	243.92	
3	0:00:02	14.9	242.36	
4	0:00:03	15	299.92	
5	0:00:04	14.9	253.82	
6	0:00:05	14.9	257.08	
7	0:00:06	15	385.2	
8	0:00:07	15	421.5	
9	0:00:08	15	421.16	
10	0:00:09	15	421.18	
11	0:00:10	15	420.74	
12	0:00:11	15	420.4	
13	0:00:12	14.9	420.56	
14	0:00:13	15	420.58	
15	0:00:14	15	420.72	
16	0:00:15	15	418.76	

それをExcelで取り込みグラフ表示させて見ました。

2階の作業部屋に設置して、約4.5時間記録してみました。温度は、暖房を止めたので徐々に下がっていきます。照度は、少しずつ低下しますが、雲の流れの影響で明暗が変化します。これを利用すると雲の流れを分析することにも応用できそうです。



如何ですか? このように簡単な回路ですが、本格的な測定にも十分応用が出来そうですね! 😊

From: <http://www.deepsky.jp/wiki/> - うごくといいな

Permanent link: <http://www.deepsky.jp/wiki/doku.php?id=elechobby:picdic:otherpic:192&rev=1588331009>

Last update: 2025/10/17 14:27



