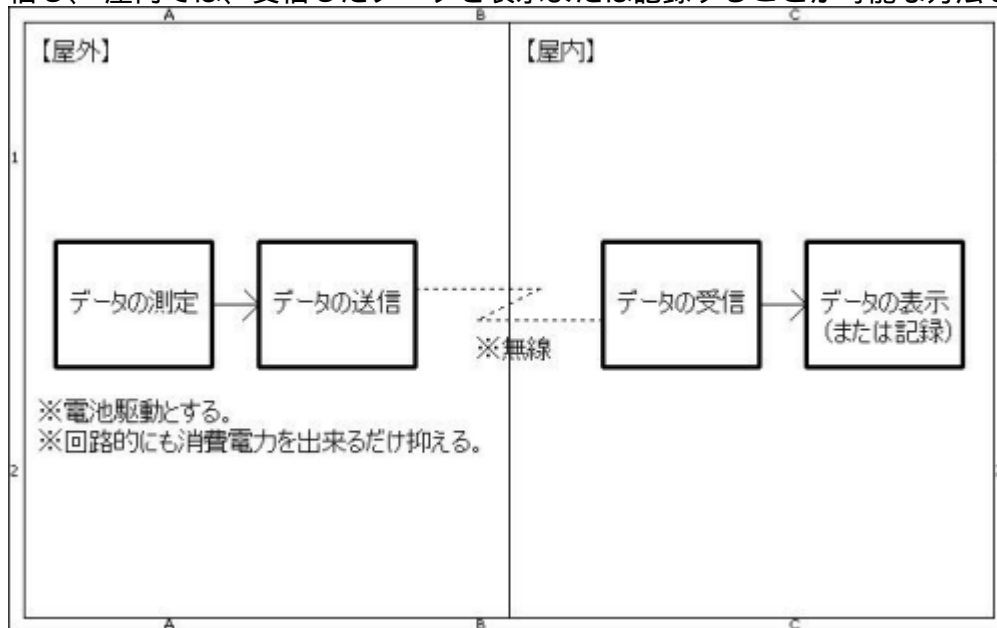


赤外線データ通信(USART)

概要

屋内から、少し離れた場所(数m)のデータを測定し、その結果を長期に渡り記録したい場合が多々あります。しかし、電源や記録媒体迄を含め、屋外に持ち出すには困難な場合(雨天、設置場所の狭さ、電源供給不可等)があります。

そこで、屋外での計測そのものは出来るだけコンパクトにし、その測定結果は、無線(赤外線など)で送信し、屋内では、受信したデータを表示または記録することが可能な方法を考えてみました。



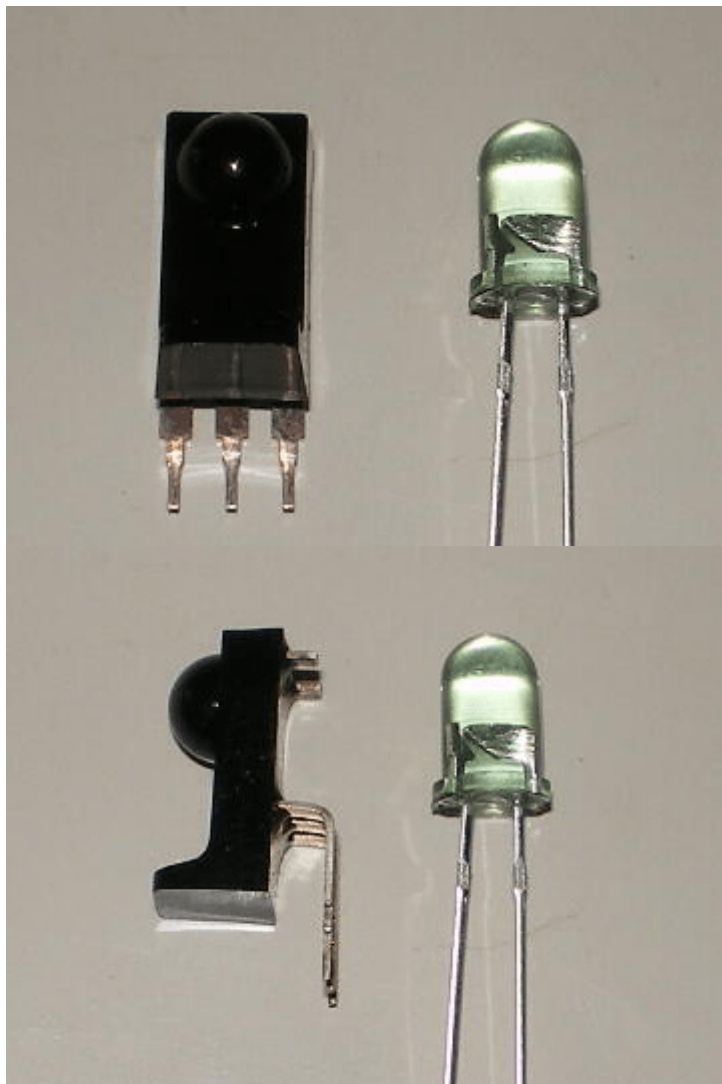
<仕様>

- 通信速度は、1200bpsとする。(スタートビット、データ8ビット、ストップビット)
- 送信モードでは、アナログデータを4チャンネル分測定し、赤外線で送信する。(約1秒周期)
- 受信モードでは、受信したデータをLCDに表示させる。
- 到達距離は、赤外線の送受信モジュールに左右されるが、約3m~10mとする。
- 同一ソフトで、送受のモードを切り替えを可能とする。
- ノイズ対策をある程度考慮する□(38kHzキャリア変調方式)

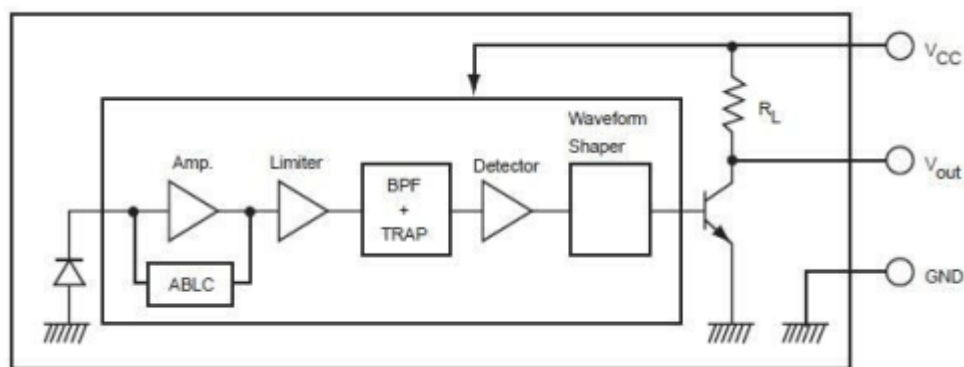
動作原理

無線によるデータ通信には、最も安価な“赤外線方式”を採用しました。また、ノイズ対策をも考慮し、リモコン等で使われている“38kHzの搬送波(キャリア)”で送信します。市販されている、“赤外線リモコン受信モジュール”は、殆どのものが標準で、アンプ、バンドパスフィルタ、広域トラップ回路等を内蔵し、出力はTTLレベルとなっているので、とても使いやすくなっています。

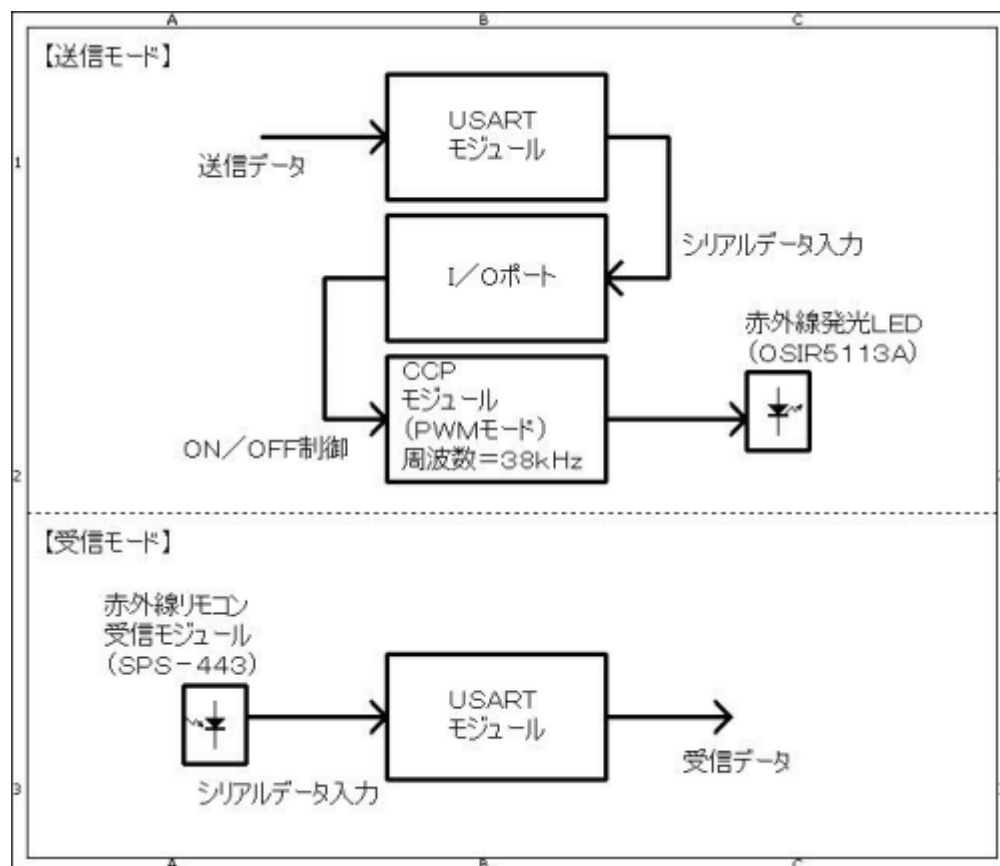
<赤外線リモコン受信モジュールと赤外線発光LED> ※黒っぽいものが、赤外線リモコン受信モジュール(SPS-443)です□ ※SPS-443以外にも□SPS-440□SPS-442□SPS-444□SPS-448等が販売されています。透明(少し緑色)のものが、赤外線発光LED(OSIR5113A)です。



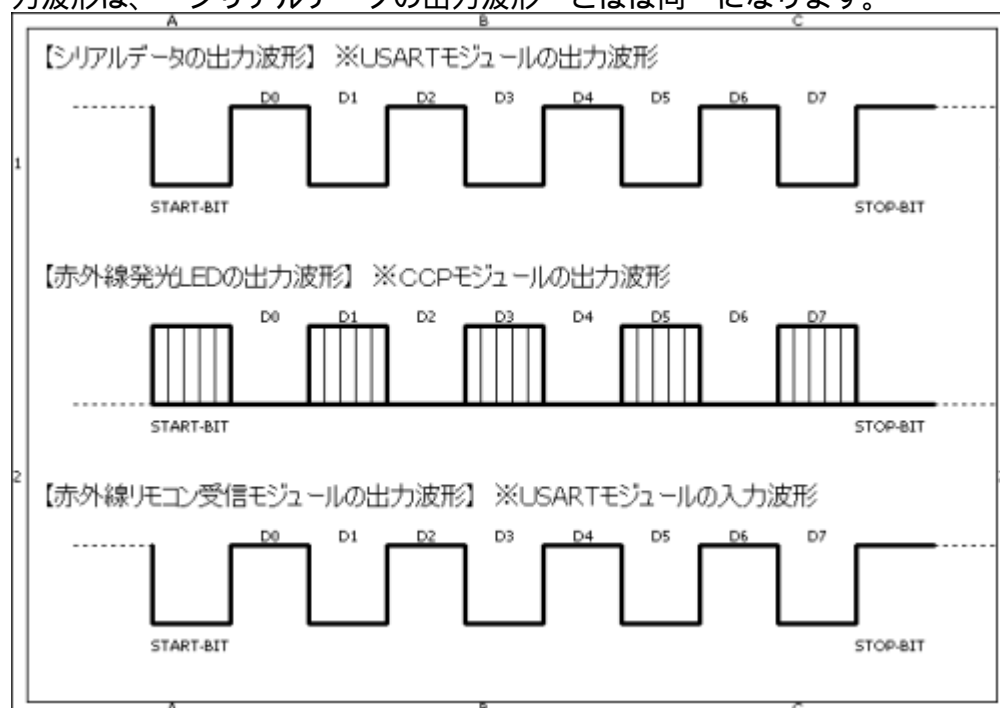
<SPS-443のブロックダイアグラム> アンプ+バンドパスフィルタ+広域トラップ回路を内蔵し、インバータ蛍光灯対策が行われています。また□ TTL出力のため使いやすい構造になっています。



<データの送信と受信のブロックダイアグラム> データの送信側では□PICのUSARTモジュールの出力を、PICのI/Oポートで直接入力し、その状態(“1”、“0”)に応じて□CCPモジュール(PWMモード、発振周波数=38kHz)をON/OFFし、CCPモジュールの出力で、赤外線発光LEDをON/OFFさせる。データの受信側では、赤外線リモコン受信モジュールの出力を、PICのUSARTモジュールで受信する。

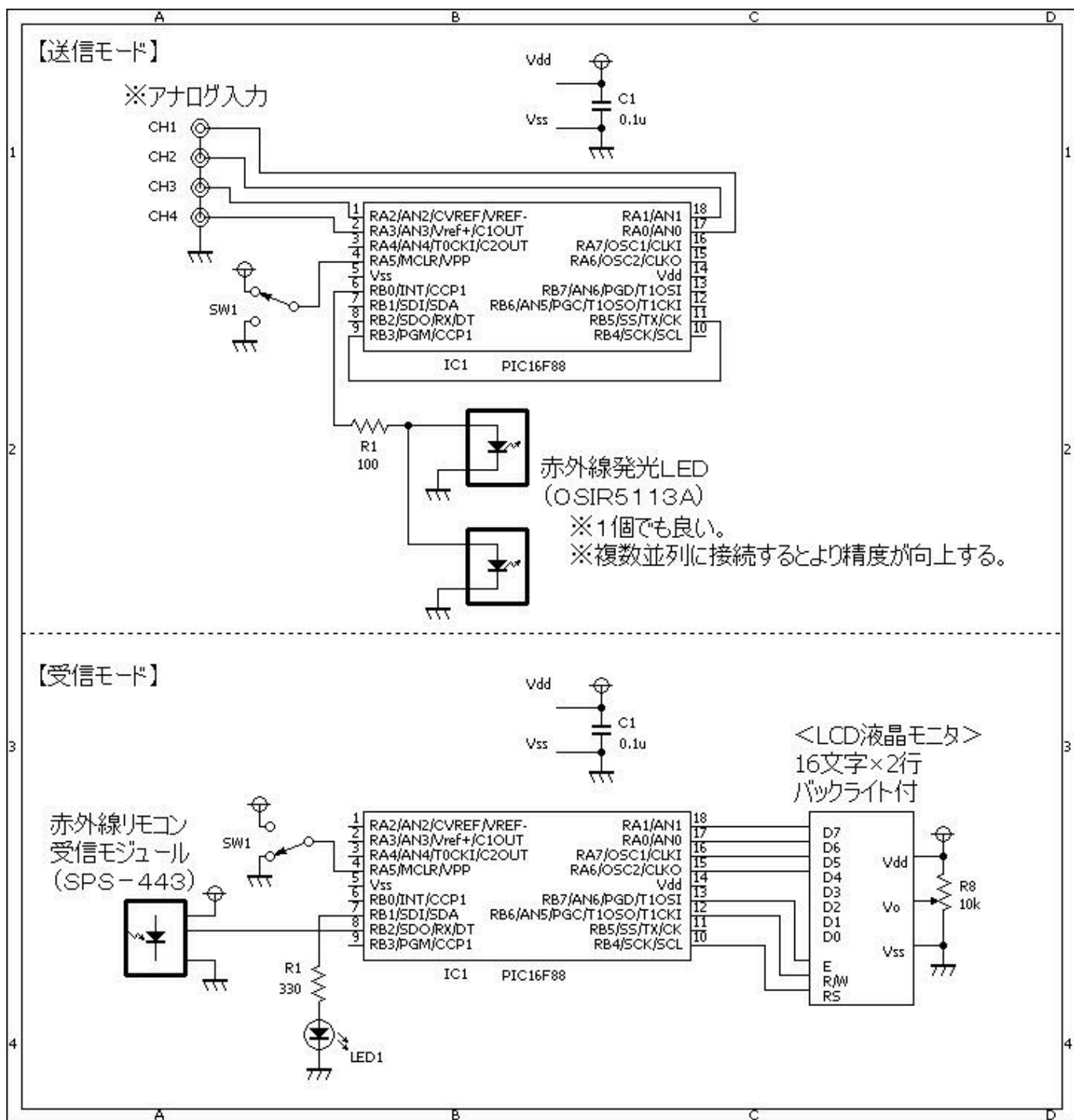


<出力波形> シリアルデータの出力波形は、“調歩同期式”の基本的な波形です。赤外線発光LEDの出力波形は、“38kHzキャリア変調”された波形になります。赤外線リモコン受信モジュールの出力波形は、“シリアルデータの出力波形”とほぼ同一になります。



回路図

送信も受信もプログラムは同じです□SW1によって双方を切り替えます。赤外線発光LEDは、1個でも動作しますが、複数を並列に接続すると精度が向上します。



ソースコード

[remoteControl_v1.c](#)

```
//*****
*
/*
< 赤外線データ通信□□□□□□□□
*/
//*****
*
```

```
#define      SW_MODE      PORTA.F5
#define      EOF          0x1A

#define      LED          PORTB.F1

//*****
*

void  Pwm_Change_DutyEx(unsigned int duty_ratio)
{
    CCP1L = duty_ratio >> 2;
    CCP1CON.CCP1Y = duty_ratio & 0b00000001;
    CCP1CON.CCP1X = (duty_ratio & 0b00000010) >> 1;
}

//*****
*

void  Usart_Write_Ex(unsigned short data)
{
    Usart_Write(data);
    //
    while (1) {
        while (PORTB.F3 == 1) {
            if (TXSTA.TRMT == 1)
                return;
        }
        Pwm_Start();
        while (PORTB.F3 == 0)
            ;
        Pwm_Stop();
    }
}

//*****
*

void  Usart_Write_Str_Ex(unsigned short* pData)
{
    while (*pData != 0x00) {
        Usart_Write_Ex(*pData);
        Delay_ms(10);
        pData++;
    }
}

//*****
*

void  dataSendProc()
```

```
{
    static unsigned short buf[32];
    static double ad;
    //
    while (1) {
        ad = Adc_Read(0);
        ad = ad * 4.8828125;
        WordToStr(ad, buf);
        Usart_Write_Str_Ex("1:");
        Usart_Write_Str_Ex(&buf[1]);
        Usart_Write_Str_Ex("mV");
        Usart_Write_Ex(EOF);
        //
        ad = Adc_Read(1);
        ad = ad * 4.8828125;
        WordToStr(ad, buf);
        Usart_Write_Str_Ex("2:");
        Usart_Write_Str_Ex(&buf[1]);
        Usart_Write_Str_Ex("mV");
        Usart_Write_Ex(EOF);
        //
        ad = Adc_Read(2);
        ad = ad * 4.8828125;
        WordToStr(ad, buf);
        Usart_Write_Str_Ex("3:");
        Usart_Write_Str_Ex(&buf[1]);
        Usart_Write_Str_Ex("mV");
        Usart_Write_Ex(EOF);
        //
        ad = Adc_Read(3);
        ad = ad * 4.8828125;
        WordToStr(ad, buf);
        Usart_Write_Str_Ex("4:");
        Usart_Write_Str_Ex(&buf[1]);
        Usart_Write_Str_Ex("mV");
        Usart_Write_Ex(EOF);
        //
        Delay_ms(100);
    }
}

//*****
*

void dataRecvProc()
{
    static unsigned short dt, cnt, buf[32];
    //
    while(1) {
        cnt = 0;
```

```
while (1) {
    if (Usart_Data_Ready() != 1) {
        continue;
    }
    //
    dt = Usart_Read();
    if (dt != EOF) {
        buf[cnt] = dt;
        cnt++;
    } else {
        buf[cnt] = 0x00;
        LED = ~LED;
        break;
    }
    //
    if (cnt == 16) {
        cnt = 0;
        continue;
    }
}
//
switch (buf[0]) {
case '1':
    Lcd_Custom_Out(1, 1, &buf[2]);
    break;
case '2':
    Lcd_Custom_Out(1, 9, &buf[2]);
    break;
case '3':
    Lcd_Custom_Out(2, 1, &buf[2]);
    break;
case '4':
    Lcd_Custom_Out(2, 9, &buf[2]);
    break;
}
}

//*****
*

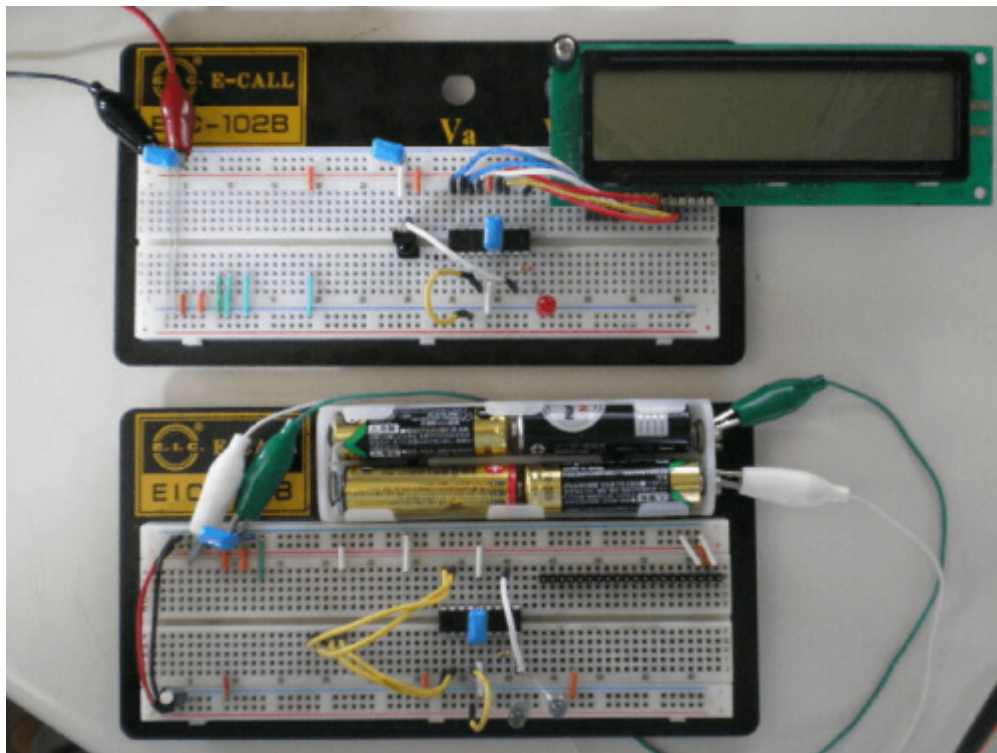
void main()
{
    static    unsigned    short    dt, cnt, buf[32];
    static    unsigned    int      ad;
    //
    OSCCON = 0b01110000;    // クロックを8Mhzに設定する。
    TRISA   = 0b11111111;
    TRISB   = 0b00001100;
    //
    LED = 0;
}
```

```
//
if (SW_MODE == 1) {    //データ送信モード
    ANSEL = 0b00001111;
    //
    Pwm_Init(38000);    // 38kHz duty=50%
    Pwm_Change_DutyEx((PR2 * 4) / 2);
    Pwm_Stop();
} else {                //データ受信モード
    ANSEL = 0b00000000;
    Lcd_Custom_Config(&PORTA,1,0,7,6,&PORTB,4,6,7);
    Lcd_Custom_Cmd(LCD_CURSOR_OFF);
    Lcd_Custom_Cmd(LCD_CLEAR);
    for (cnt = 0; cnt < 16; cnt++) {
        Lcd_Custom_Chrc(1, cnt + 1, 0xFF);
        Delay_ms(50);
    }
    for (cnt = 0; cnt < 16; cnt++) {
        Lcd_Custom_Chrc(2, cnt + 1, 0xFF);
        Delay_ms(50);
    }
    Delay_ms(1000);
    Lcd_Custom_Cmd(LCD_CLEAR);
}
//
Usart_Init(1200);
//
while (1) {
    if (SW_MODE == 1) {    //データ送信モード
        dataSendProc();
    } else {                //データ受信モード
        dataRecvProc();
    }
}
}
```

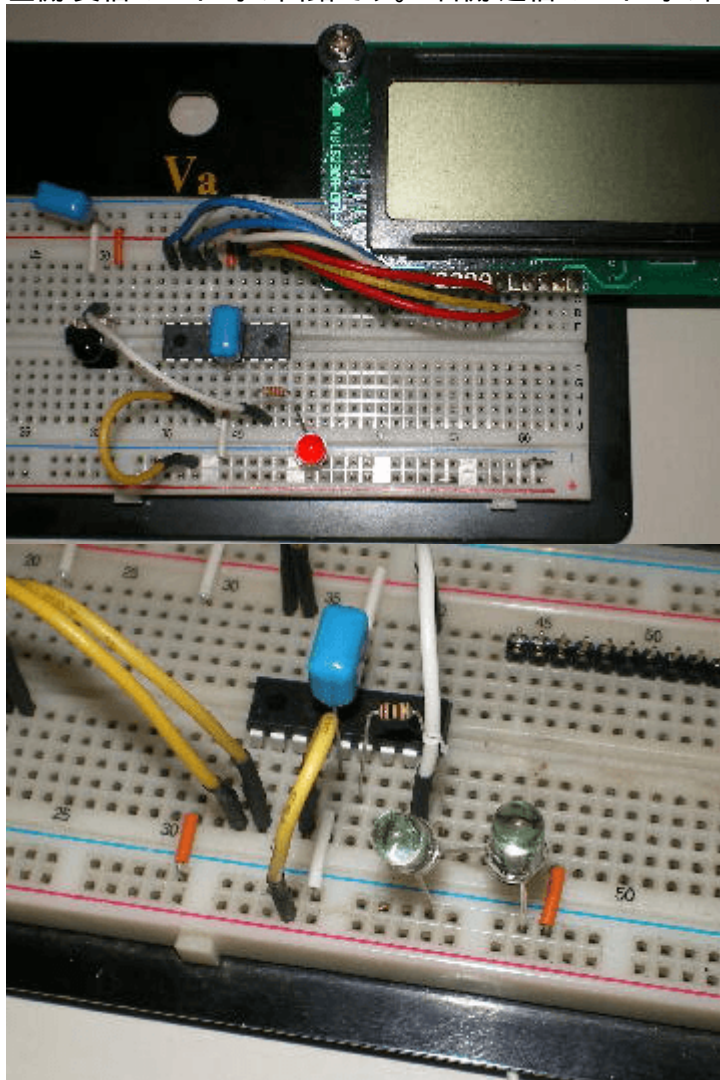
//*****
*

動作確認

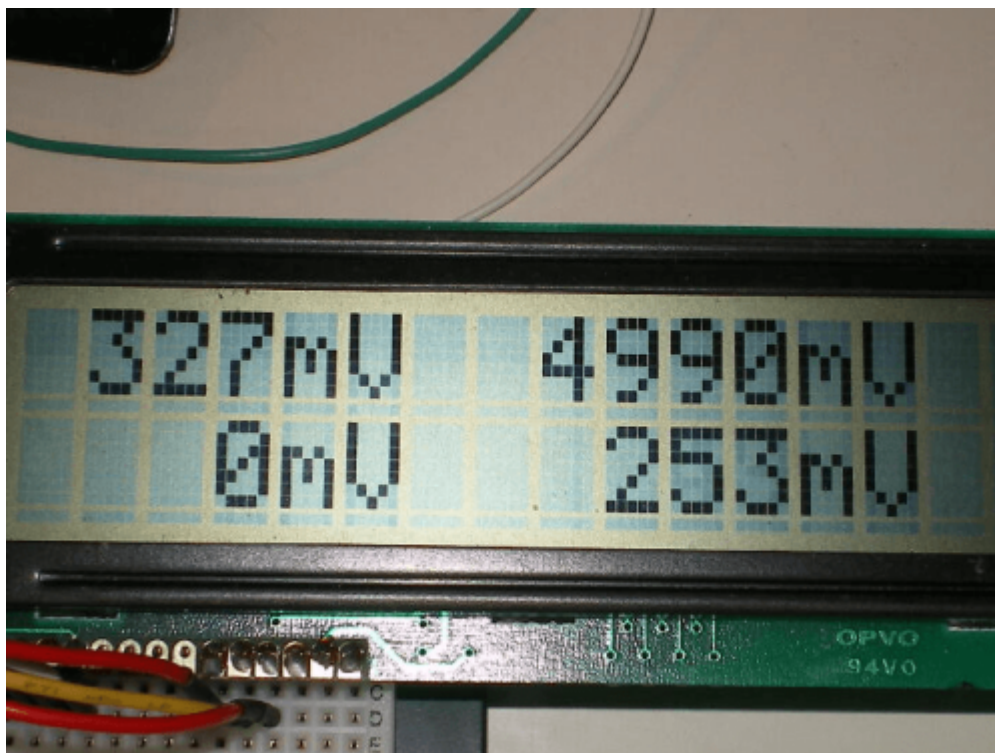
上側:受信モード時の回路です。 下側:送信モード時の回路です。



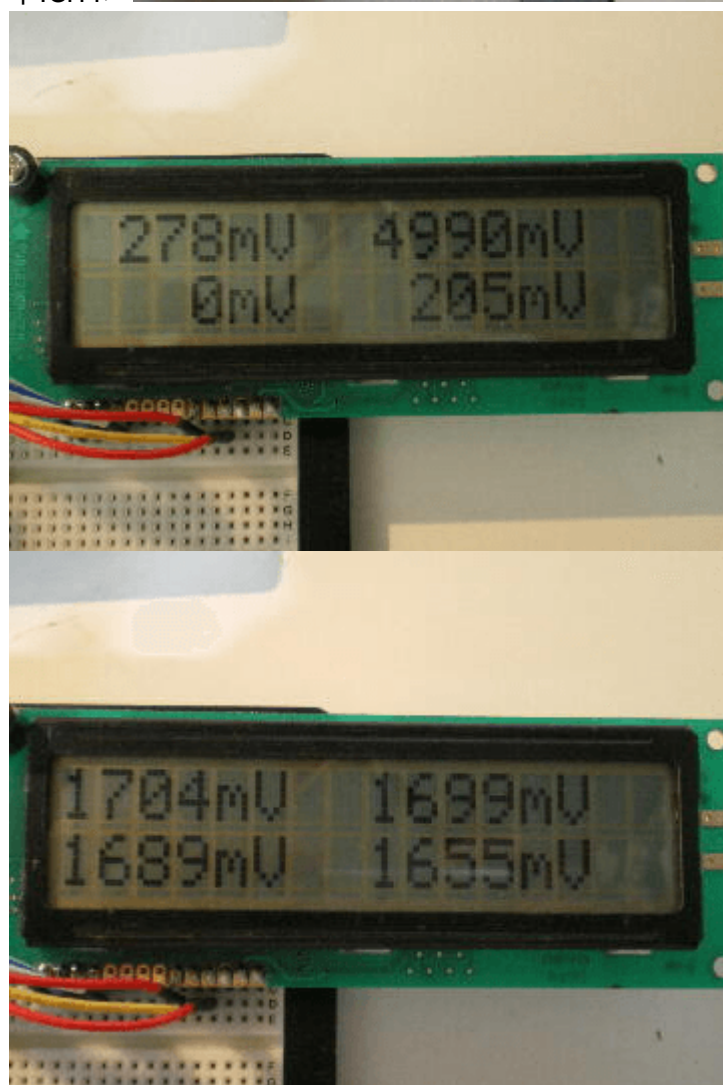
左側:受信モード時の回路です。 右側:送信モード時の回路です。

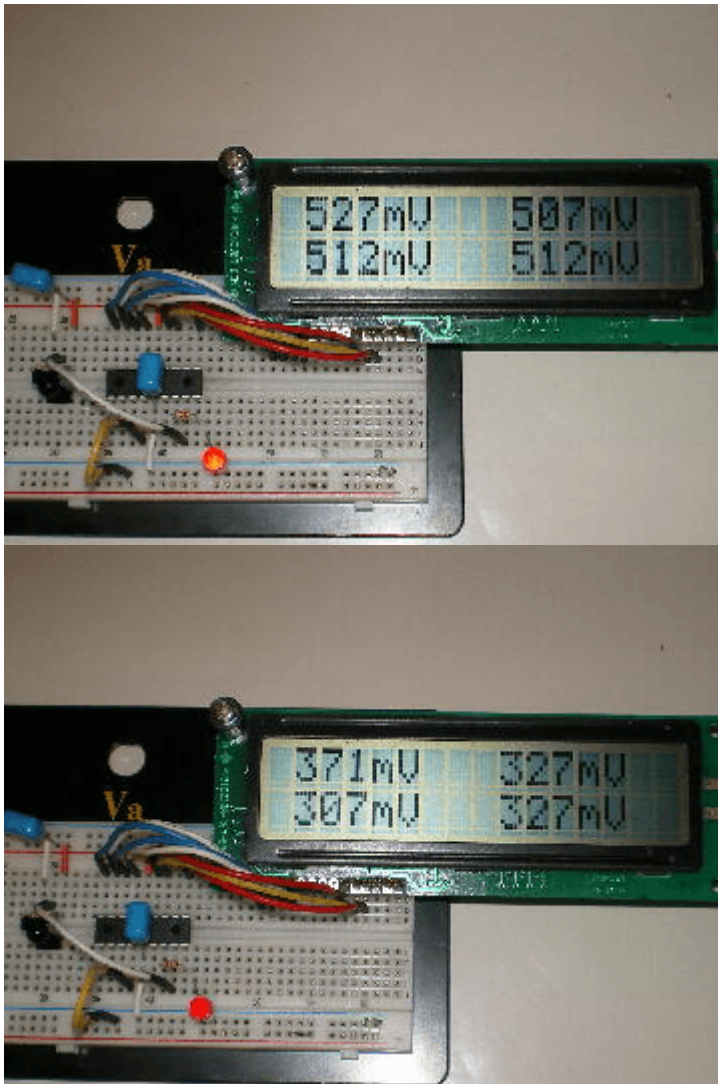


測定データを受信し、LCDに表示したところです□ LCDの表示内容 <左上:CH1□右上:CH2□左下:CH3□右



下:CH4>





如何ですか? 測定器と表示(記録)器が分離(無線)しているので、身近な測定には大いに重宝するのではないのでしょうか!?

From:

<http://deepsky.jp/wiki/> - うごくといいな

Permanent link:

<http://deepsky.jp/wiki/doku.php?id=elechobby:picdic:pic16f88:121&rev=1588212467>

Last update: 2025/10/17 14:27

