

LCDモニタ(3.3V駆動)

概要

以前にもLCDモニターを製作しました。しかし5V駆動のため、接続相手も5V駆動である必要がありました。最近、3.3V駆動の回路も多いので3.3Vで駆動するLCDモニターを製作しました。

<仕様>

- 電源は、3.3V駆動とします。
- 通信速度は、4800bps・9600bps・19200bps・38400bpsの選択を可能とします。
- 動作モードは、受信モードと送信モードの2種類とします。
- 送信モードの出力は、2系統用意します。

動作原理

<起動時処理>

- LCDに、負電圧(約2V)を供給するために、内蔵のCCPをPWMモードで起動します(10kHz・duty50%)
- 通信速度をSW1とSW2で選択します。
 - 9600bps・SW1=1・SW2=1
 - 19200bps・SW1=0・SW2=1
 - 38400bps・SW1=1・SW2=0
 - 4800bps・SW1=0・SW2=0
- 起動モードをSW3で選択します。
 - 受信モード・SW3=1
 - 送信モード・SW3=0

<受信処理>

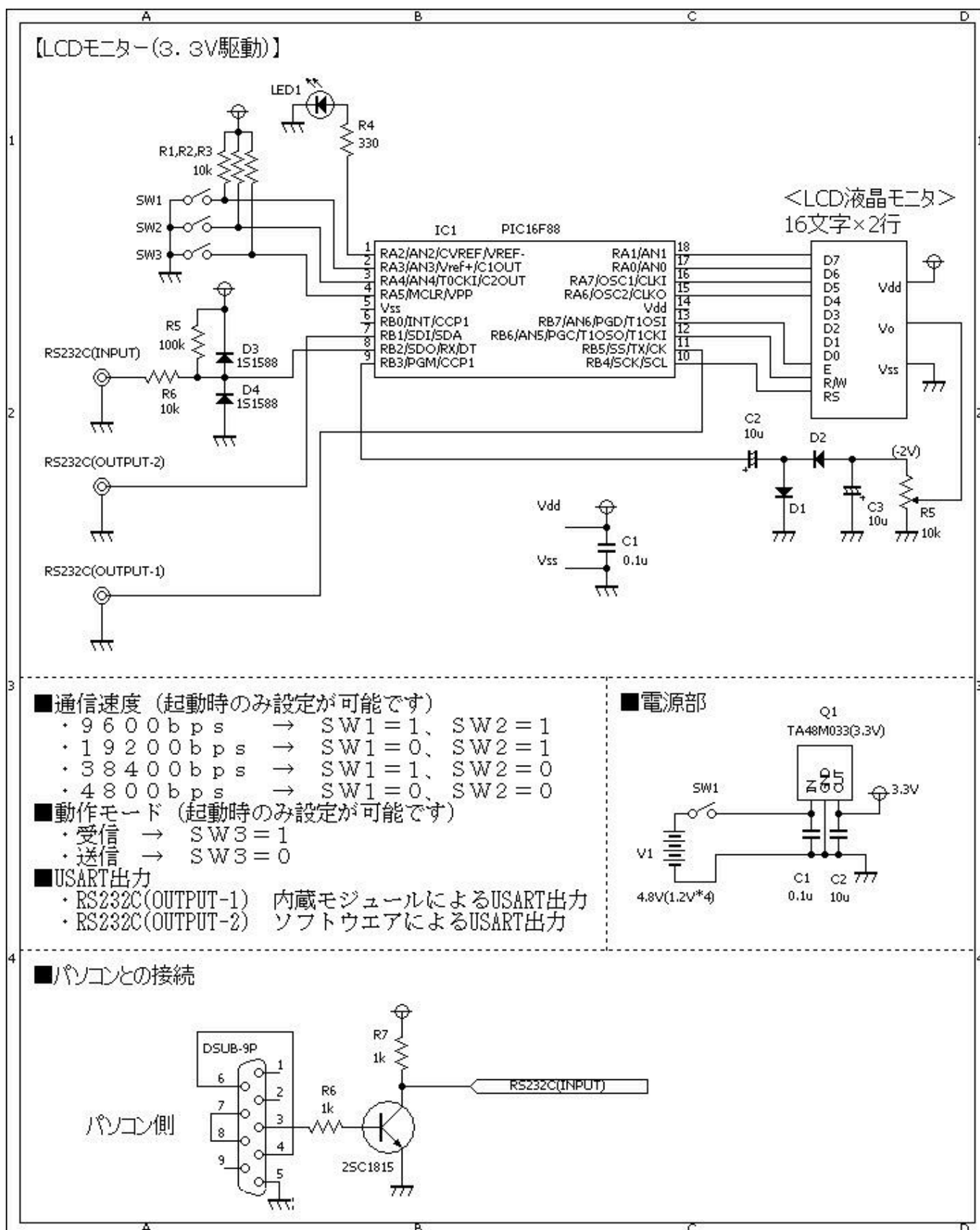
- データを受信します。
- データが、CR(Carriage Return)またはLF(Line Feed)であれば、カーソルを1行目の1文字目に戻します。
- データが、FF(Form Feed)であれば、画面を消去して、カーソルを1行目の1文字目に戻します。
- それ以外のデータであれば、画面に順次表示していきます。
- 表示方向は、次のようになります。
 - 1行目(上段)の1文字目(左端)から16文字目(右端)に向かって表示します。
 - 2行目(下段)の1文字目(左端)から16文字目(右端)に向かって表示します。
 - その後、1行目(上段)の1文字目(左端)に戻ります。

<送信処理>

- 英字の“A”～“Z”のデータを送信し、1秒スリープします。
- 英字の“a”～“z”のデータを送信し、1秒スリープします。
- 数字の“0”～“9”のデータを送信し、1秒スリープします。
- これらの処理を繰り返します。
送信出力は、2系統あります。

- 内蔵モジュールによる出力(Txピン)
- ソフトウェアによる出力(RB1ピン) 4800bpsと9600bpsのみに対応しています。

回路図



ソースコード

lcd_monitor_v3.c

```
//*****
*
/*
   モニタ(3.3V駆動)

■ 通信速度
000000000000→000000000000
000000000000→000000000000
000000000000→000000000000
000000000000→000000000000

■ 動作モード
受信モード→000000
受信したデータを16文字2行の32の表示エリアに
順次表示していきます。
データが、00または00の時には、画面をクリアせずに、カーソル
を1行目の1文字目に戻します。
データが、00の時には、画面をクリアし、カーソルを1行目の
1文字目に戻します。
送信モード→000000
テストデータの送信を繰り返します。
: 英字(大文字)“A”“B”“C”
: 英字(小文字)a“b”“c”
: 数字      “0”~“9”

*/
//*****
*

#define SW1 PORTA.F3
#define SW2 PORTA.F4
#define SW3 PORTA.F5

#define CR 0x0D
#define LF 0x0A
#define FF 0x0C

#define LED PORTA.F2
#define ON 1
#define OFF 0

//*****
*

void Pwm_Change_DutyEx(unsigned int duty_ratio)
{
    CCPR1L = duty_ratio >> 2;
    CCP1CON.CCP1Y = duty_ratio & 0b00000001;
```

```
    CCP1CON.CCP1X = (duty_ratio & 0b00000010) >> 1;
}

//*****
*

void    recv()
{
    static    unsigned    short    rd, cnt;
    //
    cnt = 0;
    //
    while (1) {
        LED = OFF;
        if (Usart_Data_Ready() == 0)
            continue;
        LED = ON;
        rd = Usart_Read();
        //
        if ((rd == CR) || (rd == LF)) {
            Lcd_Custom_Cmd(LCD_FIRST_ROW);
            cnt = 0;
            continue;
        }
        if (rd == FF) {
            Lcd_Custom_Cmd(LCD_FIRST_ROW);
            Lcd_Custom_Cmd(LCD_CLEAR);
            cnt = 0;
            continue;
        }
        //
        switch (cnt) {
            case 0:
                Lcd_Custom_Cmd(LCD_FIRST_ROW);
                Lcd_Custom_Chrcp(rd);
                break;
            case 16:
                Lcd_Custom_Cmd(LCD_SECOND_ROW);
                Lcd_Custom_Chrcp(rd);
                break;
            default:
                Lcd_Custom_Chrcp(rd);
                break;
        }
        //
        if (cnt < 31) {
            cnt++;
        } else {
            cnt = 0;
        }
    }
}
```

```
}

//*****
*

void send()
{
    static unsigned short cnt;
    //
    while (1) {
        //'A'.....'Z'
        LED = ON;
        Usart_Write(FF);
        Soft_Uart_Write(FF);
        for (cnt = 0; cnt < 26; cnt++) {
            Usart_Write(cnt + 'A');
            Soft_Uart_Write(cnt + 'A');
            Delay_ms(10);
        }
        LED = OFF;
        Delay_ms(1000);
        //'a'.....'z'
        LED = ON;
        Usart_Write(FF);
        Soft_Uart_Write(FF);
        for (cnt = 0; cnt < 26; cnt++) {
            Usart_Write(cnt + 'a');
            Soft_Uart_Write(cnt + 'a');
            Delay_ms(10);
        }
        LED = OFF;
        Delay_ms(1000);
        //'0'.....'9'
        LED = ON;
        Usart_Write(FF);
        Soft_Uart_Write(FF);
        for (cnt = 0; cnt < 10; cnt++) {
            Usart_Write(cnt + '0');
            Soft_Uart_Write(cnt + '0');
            Delay_ms(10);
        }
        LED = OFF;
        Delay_ms(1000);
    }
}

//*****
*

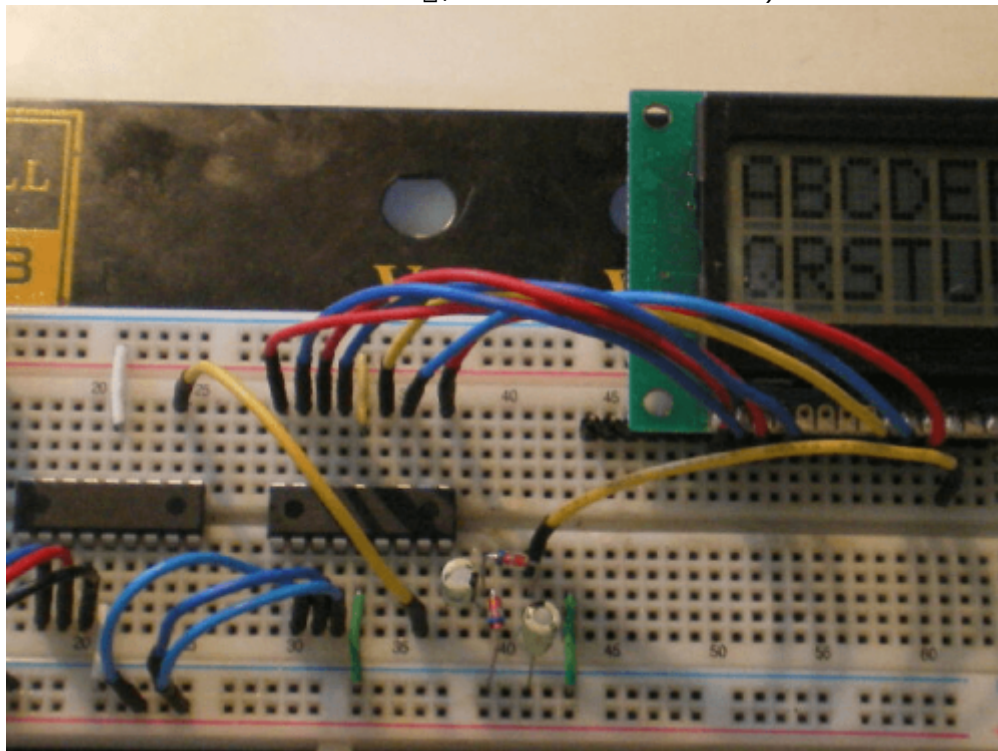
void main()
{
```

```
OSCCON = 0b01110000;
CMCON  = 0b00000111;
ANSEL  = 0b00000000;
TRISA  = 0b00111000;
TRISB  = 0b00000100;
//□□□のマイナス電源用
Pwm_Init(10000);    //10kHz
Pwm_Change_DutyEx((PR2 * 4) / 2);    //50%
Pwm_Start();
//
Lcd_Custom_Config(&PORTA, 1, 0, 7, 6, &PORTB, 4, 6, 7);
//    Lcd_Custom_Cmd(LCD_CURSOR_OFF);
Lcd_Custom_Cmd(LCD_CLEAR);
LED = ON;
Lcd_Custom_Out(1, 1, "Lcd Monitor v3");
if ((SW1 == 1) && (SW2 == 1)) {
    Lcd_Custom_Out(2, 9, " 9600bps");
}
if ((SW1 == 0) && (SW2 == 1)) {
    Lcd_Custom_Out(2, 9, "19200bps");
}
if ((SW1 == 1) && (SW2 == 0)) {
    Lcd_Custom_Out(2, 9, "38400bps");
}
if ((SW1 == 0) && (SW2 == 0)) {
    Lcd_Custom_Out(2, 9, " 4800bps");
}
//
if ((SW1 == 1) && (SW2 == 1)) {
    Uart_Init(9600);
    Soft_Uart_Init(PORTB, 0, 1, 9600, 0);
}
if ((SW1 == 0) && (SW2 == 1)) {
    Uart_Init(19200);
    Soft_Uart_Init(PORTB, 0, 1, 19200, 0);
}
if ((SW1 == 1) && (SW2 == 0)) {
    Uart_Init(38400);
    Soft_Uart_Init(PORTB, 0, 1, 38400, 0);
}
if ((SW1 == 0) && (SW2 == 0)) {
    Uart_Init(4800);
    Soft_Uart_Init(PORTB, 0, 1, 4800, 0);
}
Delay_ms(500);
Lcd_Custom_Cmd(LCD_CLEAR);
LED = OFF;
//
if (SW3 == 1) {
    recv();
} else {
```

```
send();  
}  
}  
  
//*****  
*
```

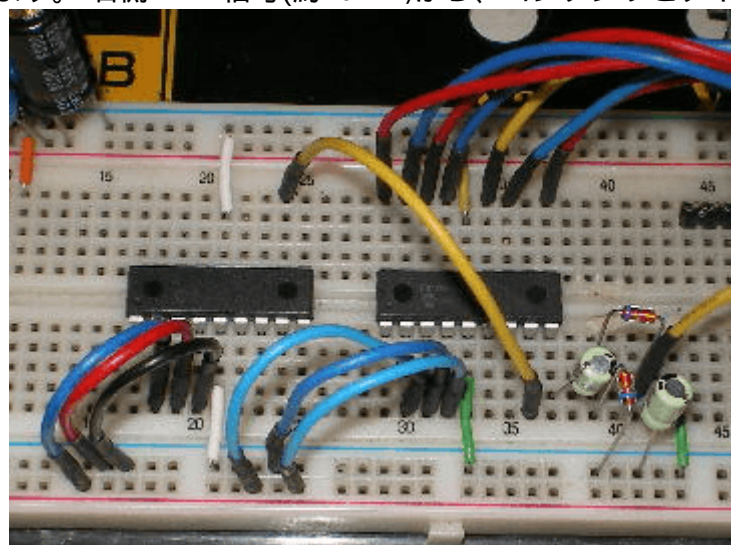
動作確認

左側のPICは送信モードで起動しています(LCDは接続していません) 右側のPICは受信モードで起動して

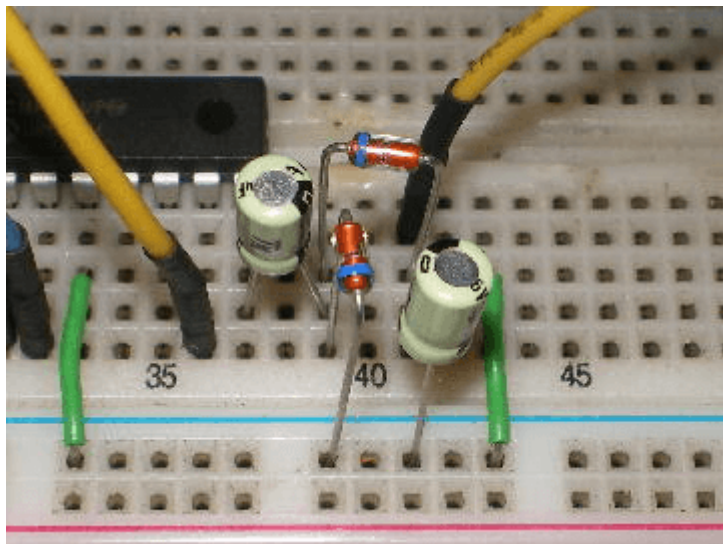


います。

左側:PIC同士は、黄色の線のみで接続されています。 右側:PWM信号(約10kHz)から、コンデンサとダイ



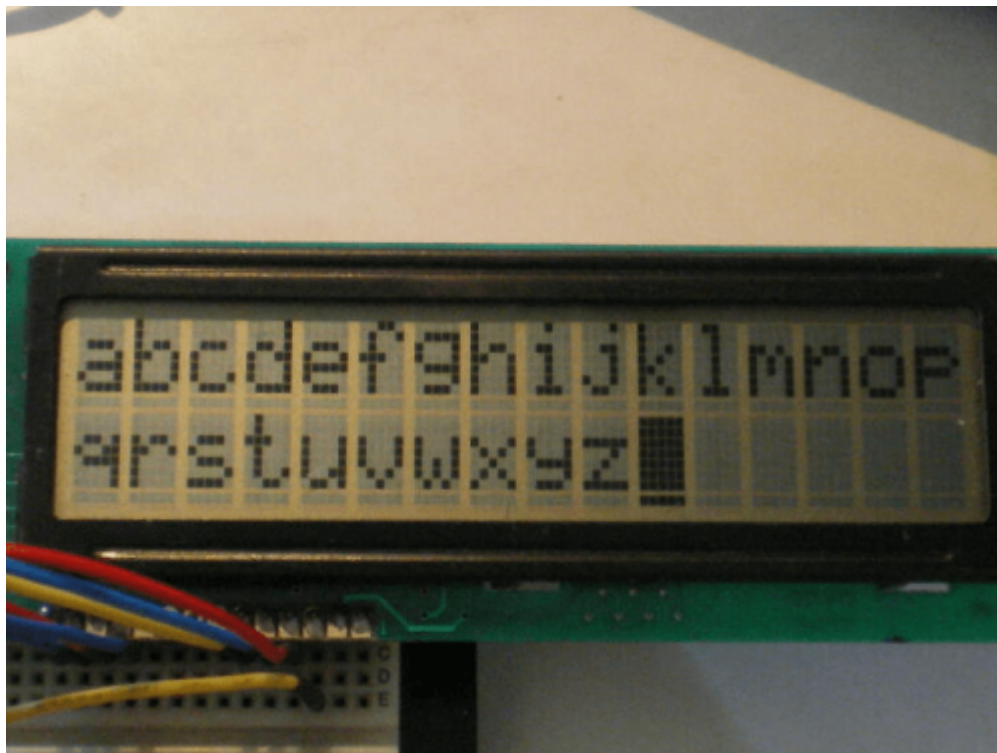
オードを使用して、負電圧を生成しています。



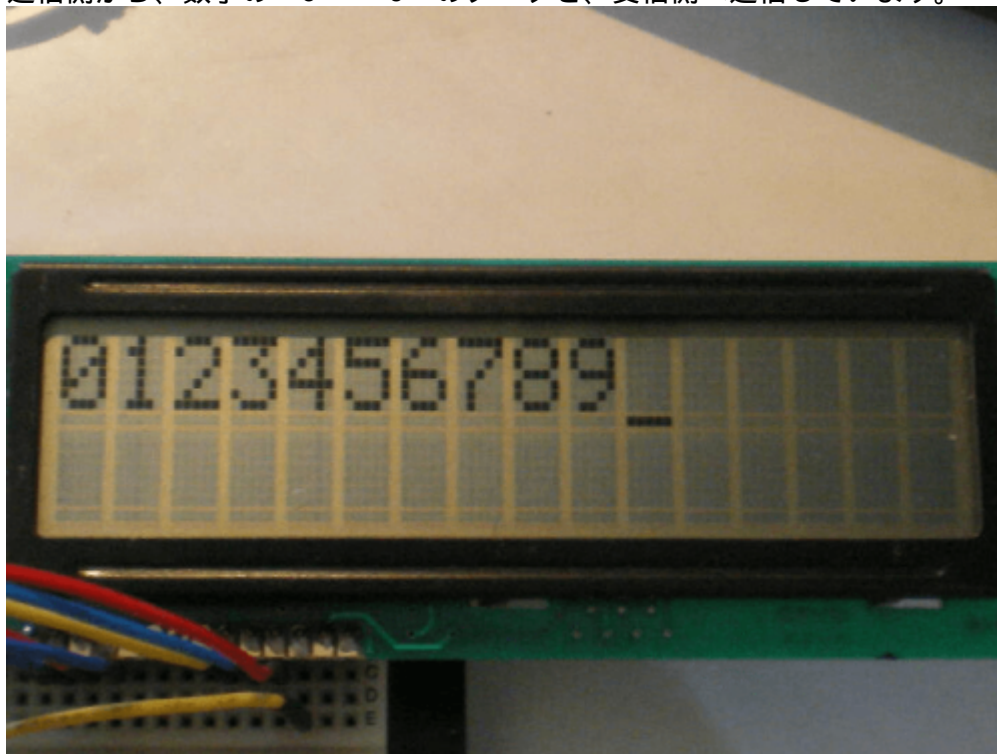
送信側から、英字の “ A”~“Z”のデータを、受信側へ送信しています。



送信側から、英字の “ a”~“z”のデータを、受信側へ送信しています。



送信側から、数字の“0”～“9”のデータを、受信側へ送信しています。



著作権表示 copyright notice

このページは稲崎様の閉鎖したHPのコピーで、著作権は稲崎様にあります。[詳細](#) This page is a copy of Mr. Inasaki's closed website, and the copyright is held by him.[Details](#)

From:

<http://deepsky.jp/wiki/> - うごくといいな

Permanent link:

<http://deepsky.jp/wiki/doku.php?id=elechobby:picdic:pic16f88:129>

Last update: **2020/05/01 17:46**

