

GPSロガー機能強化(エレキジャックNo.13)(PIC18F2550)

概要

エレキジャックNo.13「特集GPSロガーを作ろう!」で、製作したGPSロガーに対して「LCD表示の要望を多数頂きましたので「LCDを取り付け、次のデータを表示させて見ました。

<LCDへの表示内容>

- 緯度
- 経度
- 受信クオリティ
- 受信衛星数
- 時刻
- 標高

動作原理

GPSロガーの基本的な機能については、エレキジャックNo.13「特集GPSロガーを作ろう!」を参照してください。

この基本機能に対して「LCDへの表示機能を追加しました。

GPSモジュール(GT-720F)から送られてくるデータ(NMEA-0183フォーマット)の中には、緯度経度データ以外にも、時刻や標高が含まれています。詳細については、簡易緯度/経度表示ユニット(GPS-52D)を参照してください。

NMEA-0183フォーマットに含まれるデータから、「緯度、経度、受信クオリティ、受信衛星数、時刻、標高」データを、抽出しLCDに表示させます。

尚「LCDには、16文字×2行を使用したので、スイッチ(SW4)で、表示切替を行っています。

- SW4=OFF「緯度、経度、受信クオリティ、受信衛星数
- SW4=ON「時刻、標高

本GPSロガーは、3.3Vの電圧で動作しています。しかし「LCDは、5Vでなければ動作しません。そこで「PIC18F2550のPWM出力で、負電圧(-2V)を生成し、LCDのVo(輝度調整用)端子に供給します。詳細な、仕組みについては「LCD(液晶)モジュールの低電圧動作を参照してください。

回路図



```
//*****
*
/*
『簡易□□□口ガー（基本機能）』
*/
//*****
*
```

```
#define SW1 PORTB.F4
#define SW2 PORTA.F4
#define LED1 PORTB.F7
#define LED2 PORTB.F6
#define LED3 PORTB.F5

#define CR 0x0d
#define LF 0x0a

//*****
*

static char buf[32], tmp[16];

static char *HD1 = "H,TRACK,255,0,0,2,1,,,,,\r\n";
static char *HD2 = "T,TRACK,          ,          ,\r\n";
,2000-01-01,00:00:00,WGS84,-9.9,-9.9,          \r\n";

static unsigned short flg, len;
static unsigned short hd[8], utc[12], latitude[12],
longitude[12];
static unsigned short quality[4], satellites[4], asl[8],
temperature[8];

//*****
*

void GPGGA_recv()
{
    static unsigned short rd, rec;
    //
    flg = 0;
    len = 0;
    //
    while (flg != 10) {
        rd = Soft_Uart_Read(&rec);
        if (rd == LF) {
            len = 0;
            continue;
        }
        //
        switch (flg) {
        case 0:
            hd[len] = rd;
            len++;
            if (len == 7) {
                len = 0;
                if (strncmp(hd, "$GPGGA,", 7) == 0) {
                    flg = 1;
                }
            }
        }
    }
}
```

```
        break;
    case 1:
        if (rd == ',') {
            utc[len] = 0x00;
            len = 0;
            flg = 2;
        } else {
            utc[len] = rd;
            len++;
        }
        break;
    case 2:
        if (rd == ',') {
            latitude[len] = 0x00;
            len = 0;
            flg = 3;
        } else {
            latitude[len] = rd;
            len++;
        }
        break;
    case 3:
        if (rd == ',') {
            len = 0;
            flg = 4;
        }
        break;
    case 4:
        if (rd == ',') {
            longitude[len] = 0x00;
            len = 0;
            flg = 5;
        } else {
            longitude[len] = rd;
            len++;
        }
        break;
    case 5:
        if (rd == ',') {
            len = 0;
            flg = 6;
        }
        break;
    case 6:
        if (rd == ',') {
            quality[len] = 0x00;
            len = 0;
            flg = 7;
        } else {
            quality[len] = rd;
            len++;
        }
    }
```

```
    }
    break;
case 7:
    if (rd == ',') {
        satellites[len] = 0x00;
        len = 0;
        flg = 8;
    } else {
        satellites[len] = rd;
        len++;
    }
    break;
case 8:
    if (rd == ',') {
        len = 0;
        flg = 9;
    }
    break;
case 9:
    if (rd == ',') {
        asl[len] = 0x00;
        len = 0;
        flg = 10;
    } else {
        asl[len] = rd;
        len++;
    }
    break;
}
}
}

//*****
*

void init_sdc()
{
    //□□□□□□□□の初期化
    Spi_Init_Advanced(MASTER_OSC_DIV64, DATA_SAMPLE_MIDDLE,
CLK_IDLE_LOW, LOW_2_HIGH);
    if (Mmc_Fat_Init(&PORTC, 6)) {
        while (1) {
            LED3 = 1;
            Delay_ms(100);
            LED3 = 0;
            Delay_ms(100);
        }
    }
    Spi_Init_Advanced(MASTER_OSC_DIV16, DATA_SAMPLE_MIDDLE,
CLK_IDLE_LOW, LOW_2_HIGH);
}
```

```
//*****
*
//UTC(Coordinated Universal Time 協定世界時)、日本標準時(Japan Standard
Time, JST)+9時間の時差

void    utc2jst(char* pUtc)
{
    static    short    i;
    //
    tmp[8] = 0x00;
    tmp[7] = pUtc[5];
    tmp[6] = pUtc[4];
    tmp[5] = ':';
    tmp[4] = pUtc[3];
    tmp[3] = pUtc[2];
    tmp[2] = ':';
    pUtc[2] = 0x00;
    i = atoi(pUtc) + 9;
    i = i < 24 ? i : i - 24;
    ByteToStr(i, buf);
    tmp[1] = buf[2];
    tmp[0] = (buf[1] != ' ') ? buf[1] : '0';
    memcpy(pUtc, tmp, 8);
    pUtc[8] = 0x00;
}

//*****
*
// dddmm.mmmm→ddd.mmssssやまおたく形式)

void    secConv(char* msg)
{
    char    *p;
    float    f;
    int      i;
    //
    p = strchr(msg, '.');
    f = atof(p);
    f = f * 6000.0;
    WordToStr(f, tmp);
    if (tmp[1] == ' ')
        tmp[1] = '0';
    if (tmp[2] == ' ')
        tmp[2] = '0';
    if (tmp[3] == ' ')
        tmp[3] = '0';
    *(p) = *(p - 1);
    *(p - 1) = *(p - 2);
    *(p - 2) = '.';
    *(p + 1) = 0x00;
}
```

```
    strcat(p + 1, &tmp[1]);
}

//*****
*

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

//*****
*

static char    file_name[] = "gps_log.txt";

void    main()
{
    //変数の定義
    static    short    mode, cnt, i, disp_mode;
    //コンパレータは使用しない。
    CMCON = 0b00000111;
    //変換の設定
    ADCON1.PCFG3 = 1;
    ADCON1.PCFG2 = 1;
    ADCON1.PCFG1 = 1;
    ADCON1.PCFG0 = 1;
    //
    OSCCON.IRCF2 = 1;
    OSCCON.IRCF1 = 1;
    OSCCON.IRCF0 = 1;
    //ポートの設定
    TRISA = 0b00010000;
    TRISB = 0b00011000;
    TRISC = 0b00000000;
    //変数の初期化
    mode = 0;
    disp_mode = PORTE.F3;
    //LEDの点灯確認
    for (cnt = 0; cnt < 3; cnt++) {
        LED1 = 1;
        LED2 = 1;
        LED3 = 1;
        Delay_ms(100);
        LED1 = 0;
        LED2 = 0;
        LED3 = 0;
        Delay_ms(100);
    }
}
```

```
//□□□の初期化□□□のマイナス電圧用)
Pwm_Init(5000);
Pwm_Change_DutyEx((PR2 * 4) / 2);
Pwm_Start();
Delay_ms(100);
//□□□の初期化
Lcd_Custom_Config(&PORTA,0,1,2,3,&PORTC,0,4,1);
Lcd_Custom_Cmd(LCD_CURSOR_OFF);
Lcd_Custom_Cmd(LCD_CLEAR);
Lcd_Custom_Out(1, 1, "GPS-Logger v2");
Delay_ms(1000);
Lcd_Custom_Cmd(LCD_CLEAR);
//□□□□□□□□の初期化
init_sdc();
for (cnt = 0; cnt < 3; cnt++) {
    LED2 = 1;
    Delay_ms(300);
    LED2 = 0;
    Delay_ms(300);
}
//□□□□□□の初期化
Soft_Uart_Init(PORTB,3, 2, 9600, 0);
Soft_Uart_Write(CR);
Soft_Uart_Write(LF);
//□□□データの受信(最初の1回のみ)
GPGBA_recv();
for (cnt = 0; cnt < 3; cnt++) {
    LED3 = 1;
    Delay_ms(300);
    LED3 = 0;
    Delay_ms(300);
}
//
while (1) {
    if ((mode == 0) && (SW1 == 0)) {
        while (SW1 == 0) {
            Delay_ms(100);
        }
        mode = 1;
        LED3 = 1;
        //ファイルの作成
        Mmc_Fat_Assign(file_name, 0xA0);
        Mmc_Fat_Rewrite();
        //カシミール□□の□□□形式のヘッダの書き込み
        Mmc_Fat_Write(HD1, strlen(HD1));
    }
    //
    if ((mode == 1) && (SW2 == 0)) {
        while (SW2 == 0) {
            Delay_ms(100);
        }
    }
}
```

```
        mode = 0;
        LED3 = 0;
    }
    //データを受信
    LED1 = 1;
    GPGBA_recv();
    LED1 = 0;
    if (mode == 0) {
        Delay_ms(100);
    }
    //受信衛星数の確認
    if ((satellites[0] != '0') || (satellites[1] != '0')) {
        LED2 = 1;
    } else {
        LED2 = 0;
    }
    //緯度データの形式変換 // dddmm.mmmm→ddd.mmssssやまおたく形式)
    secConv(latitude);
    //経度データの形式変換 // dddmm.mmmm→ddd.mmssssやまおたく形式)
    secConv(longitude);
    //UTCのJSTへの変換
    utc2jst(utc);
    //カシ米尔の形式のデータの書き込み
    if (mode == 1) {
        memcpy(HD2 + 8, latitude, strlen(latitude));
        memcpy(HD2 + 19, longitude, strlen(longitude));
        memcpy(HD2 + 52, utc, strlen(utc));
        //
        Mmc_Fat_Write(HD2, strlen(HD2));
    }
    // <緯度経度の表示>、<時刻、高度の表示>
    if (PORTE.F3 == 1) {
        if (disp_mode != 1) {
            Lcd_Custom_Cmd(LCD_CLEAR);
        }
        disp_mode = 1;
        //
        Lcd_Custom_Out(1, 1, "N:");
        Lcd_Custom_Out(2, 1, "E:");
        Lcd_Custom_Out(1, 3, latitude);
        Lcd_Custom_Out(2, 3, longitude);
        Lcd_Custom_Out(1, 15, quality);
        Lcd_Custom_Out(2, 15, satellites);
    } else {
        if (disp_mode != 0) {
            Lcd_Custom_Cmd(LCD_CLEAR);
        }
        disp_mode = 0;
        //
        Lcd_Custom_Out(1, 1, utc);
        Lcd_Custom_Out(2, 1, asl);
    }
}
```

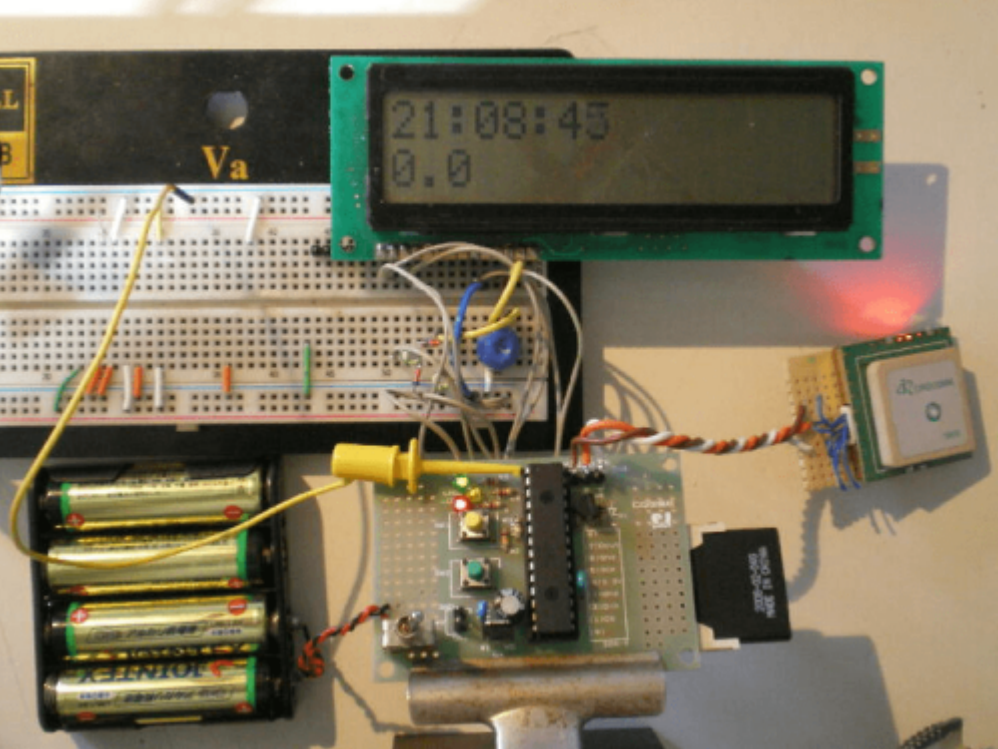
```

}
}
}

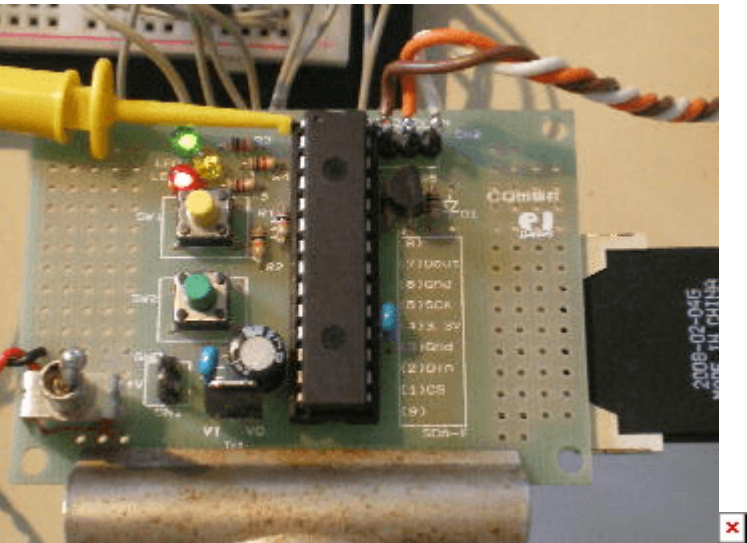
// *****
*
```

動作確認

LCD表示部とPWMパルス 負電圧変換部は、ブレッドボードに実装しました GPSロガーの基板の半田面から、直接リード線を半田付けし、ブレッドボードに差し込んでいます。

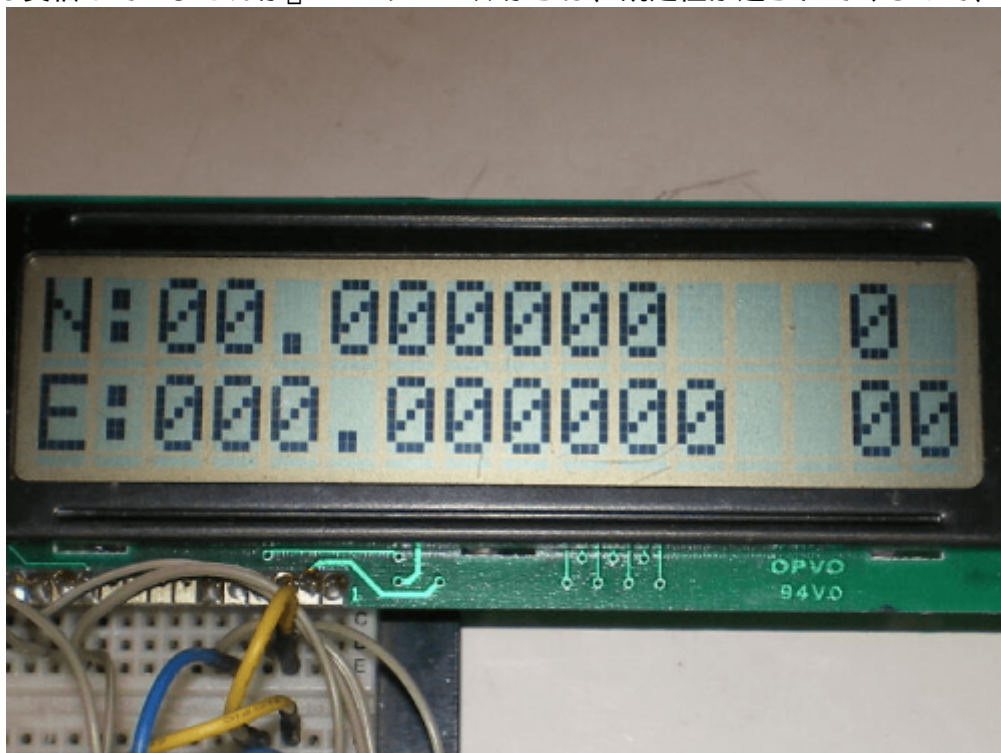


左側:GPSロガー基板です。 右側:PWMパルス 負電圧変換部です。



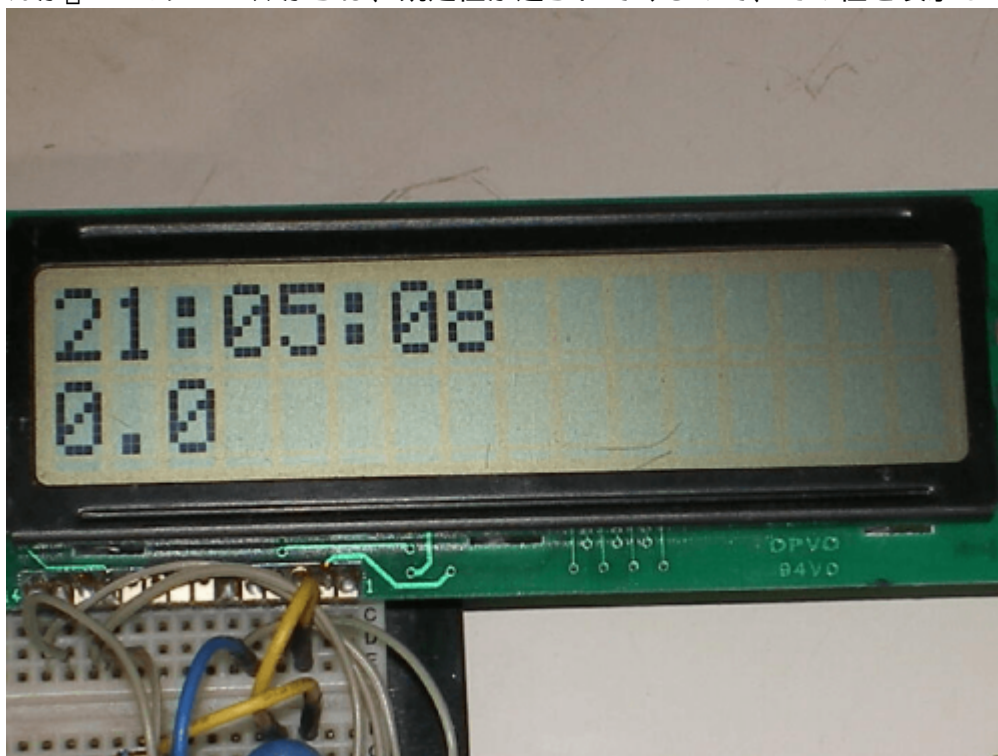
左上=緯度 左下=経度 右上=受信クオリティ 右下=受信衛星数 これらのデータは、屋内で確認したため

GPS衛星からの電波は受信していませんがGPSモジュールからは、既定値が送られてくるので、その値



を表示しています。

上側=時刻 下側=標高 これらのデータは、屋内で確認したためGPS衛星からの電波は受信していませんがGPSモジュールからは、既定値が送られてくるので、その値を表示しています。



如何ですか? これで、ハイキングなどで、本GPSロガーを持参すれば、リアルで現在地や標高が分かりますね! 😊

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:otherpic:182>

Last update: **2020/05/01 19:12**

