3線式LCD制御ライブラリ

概要

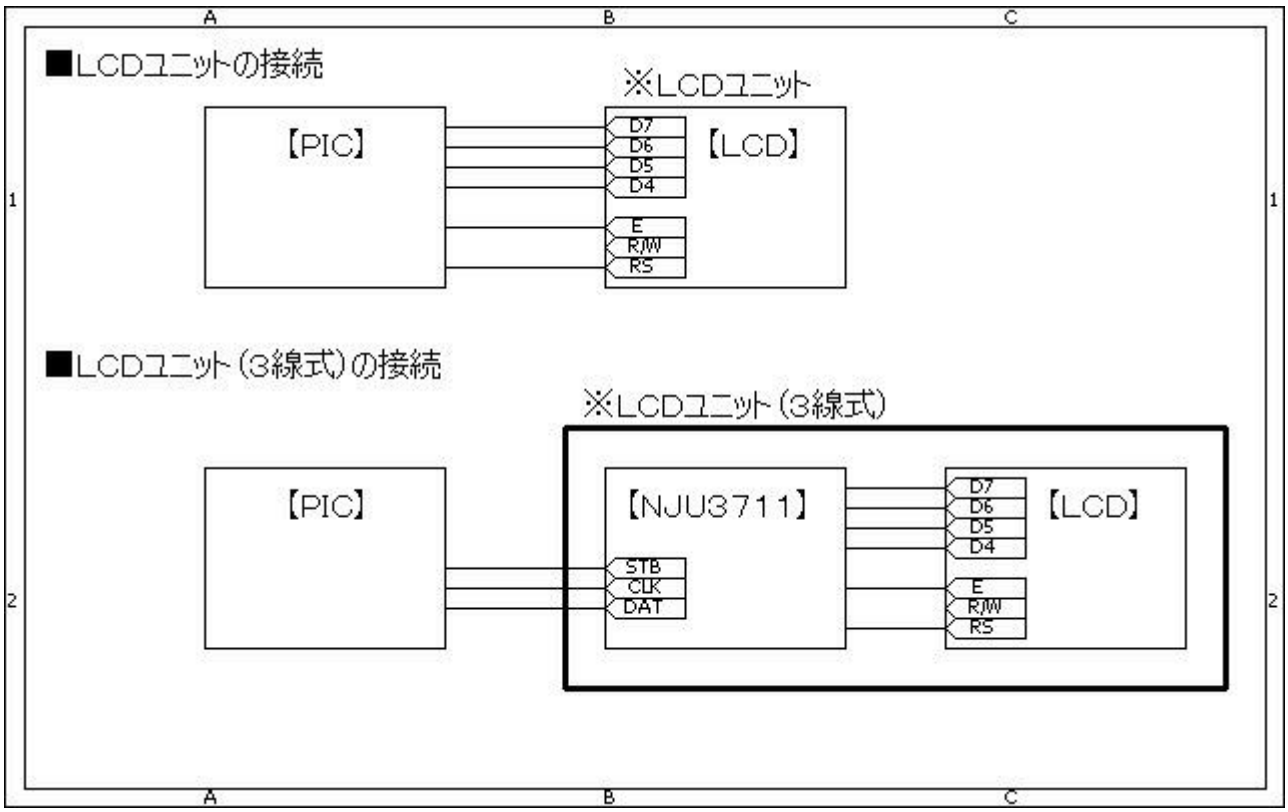
PICにLCDを接続して使う場合には、少なくとも6本のポートが必要となります。 これでは、ポート数の少ないPIC(PIC12F683など)ではLCDを接続することが出来ないことになります。

そこで、今回は、シリアル→パラレル変換ICを利用してPICとの接続には、3本のポートを使用するだけでLCDを制御することが出来るようにしました。

<仕様>

- 制御信号は、3線式とします(STB,CLK,DAT)
- LCDおよび拡張ポートを制御するための制御ライブラリを提供します。
- 3線式LCDライブラリの仕様は、mikroCコンパチブルとします。

<PICとの接続>



動作原理

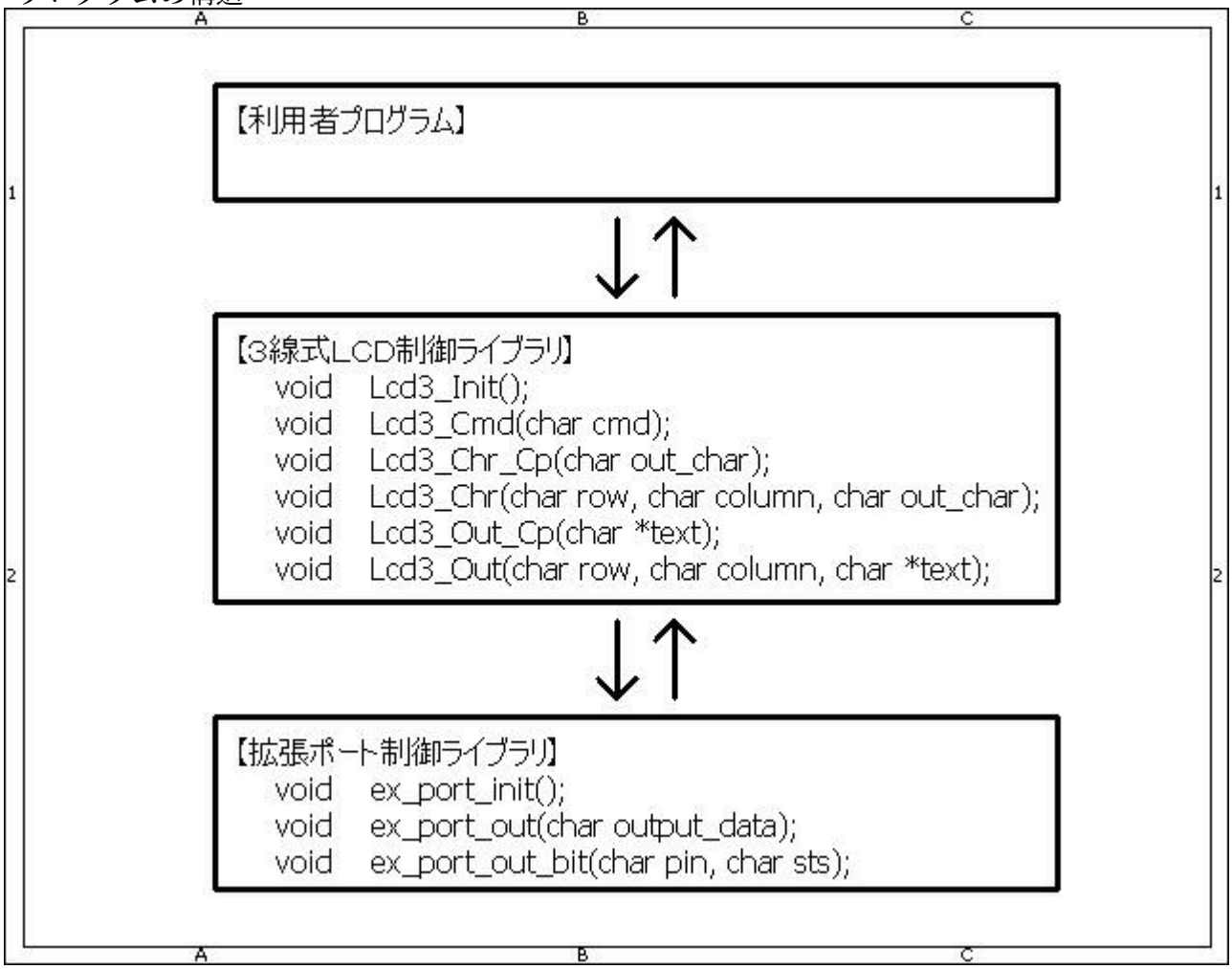
仕組み的にはLCD制御と拡張ポート(NJU3711)制御の組み合わせになります。

※LCD制御は、“LCD制御ライブラリ(mikroCコンパチブル)“ を参照してください。※拡張ポート制御は、“ 拡張ポート制御ライブラリ(8ビット出力) “ を参照してください。

<3線式LCD制御ライブラリで提供する関数>

- Lcd3_Init();
  LCDを初期化します。(拡張ポート(NJU3711)の初期化も含みます)
  PICの使用ポート(STB、CLK、DAT)を指定します。
- Lcd3_Out(char row, char column, char *text);
  行(row)と列(column)を指定して、LCDに文字列(text)を表示します。
- Lcd3_Out_Cp(char *text);
  カレントカーソルに、LCDに文字列(text)を表示します。
- Lcd3_Chr(char row, char column, char out_char);
  行(row)と列(column)を指定して、LCDに文字(out_char)を表示します。
- Lcd3_Chr_Cp(char out_char);
  カレントカーソルに、LCDに文字(out_char)を表示します。
- Lcd3_Cmd(char out_char);
  LCDにコマンド(out_char)を送信します。

<プログラムの構造>

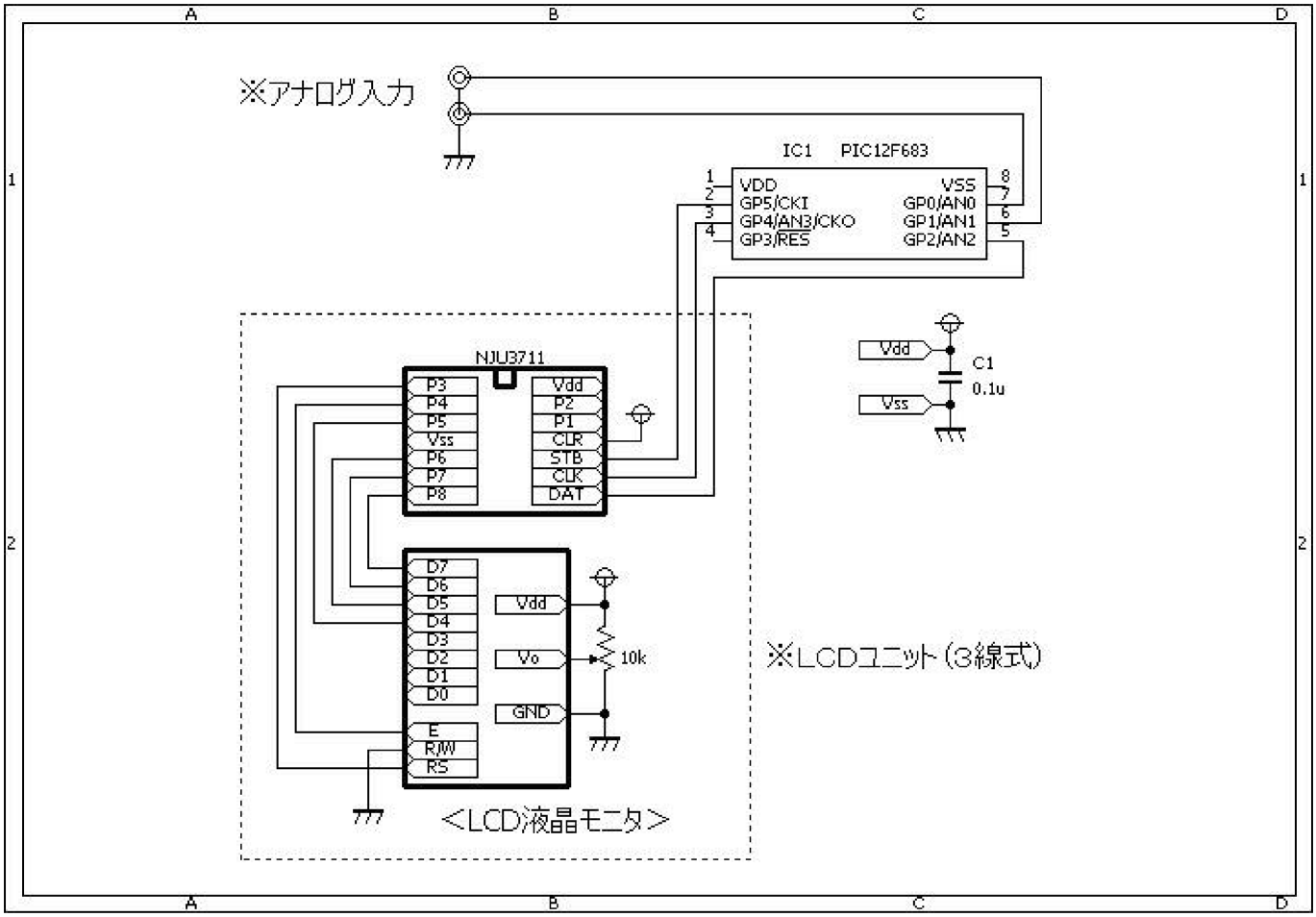


<動作確認用のプログラムの処理>

- LCDを初期化(NJU3711の初期化も含む)します。
- オープニングデモを行います。
  - Lcd3_Chr関数を使用して、文字を表示します。
  - Lcd3_Chr_Cp関数を使用して、文字を表示します。
  - Lcd3_Out関数を使用して、文字を表示します。
  - Lcd3_Out_Cp関数を使用して、文字を表示します。
  - Lcd3_Cmd関数を使用して、画面をクリアします。
  - Lcd3_Chr関数を使用して、画面の各行各列に ■ を、順次表示します。

    - Lcd3_Cmd関数を使用して、画面のオン/オフを繰り返します。
    - Lcd3_Cmd関数を使用して、カーソルをホームに戻します。
    - Lcd3_Chr関数を使用して、画面の各行各列にスペースを、順次表示します。
- アナログデータ(CH1~CH2)を取り込み、値を表示します。

回路図



ソースコード

動作確認用のプログラム

lcd_display_nju3711.c

```
//*****************************************************************************
*
/*
    ＜３線式LCD制御ライブラリ（mikroCコンパチブル）＞

  ※拡張ポート用のICには、NJU3711を使用します。
*/
//*****************************************************************************
*
//■■■関数宣言■■■
```

```
extern  void    main();
extern  void    opening_demonstration();
//*********************************************************************
*
//■■■インクルード■■■
#include "lcd_lib_4bit_nju3711.h"
//*********************************************************************
*
//■■■マクロ定義■■■
//LCD
const   short   LCD_RS = 2;         //NJU3711(P3)
const   short   LCD_EN = 3;         //NJU3711(P4)
const   short   LCD_D7 = 7;         //NJU3711(P8)
const   short   LCD_D6 = 6;         //NJU3711(P7)
const   short   LCD_D5 = 5;         //NJU3711(P6)
const   short   LCD_D4 = 4;         //NJU3711(P5)
//NJU3711
sbit    STB     at    GP5_bit;
sbit    CLK     at    GP4_bit;
sbit    DAT     at    GP2_bit;
sbit    STB_Direction     at    TRISIO5_bit;
sbit    CLK_Direction     at    TRISIO4_bit;
sbit    DAT_Direction     at    TRISIO2_bit;
//*********************************************************************
*
//■■■メイン関数■■■
void    main()
{
        short   cnt;
        char    buf[16];
        double  ad;
        //
        OSCCON = 0b01110000;
        CMCON0 = 0b00000111;
        ANSEL  = 0b00000011;
        TRISIO = 0b00001011;
        //
        Lcd3_Init();
        Lcd3_Cmd(_LCD_CLEAR);
        Lcd3_Cmd(_LCD_CURSOR_OFF);
        //
        opening_demonstration();
        //
        Lcd3_Out(1, 6, "mV");
        Lcd3_Out(2, 6, "mV");
        //
        while (1) {
                ad = Adc_Read(0);
                ad *= 4.8828125;
                WordToStr(ad, buf);
                Lcd3_Out(1, 1, buf);
```

```
                //
                ad = Adc_Read(1);
                ad *= 4.8828125;
                WordToStr(ad, buf);
                Lcd3_Out(2, 1, buf);
                //
                Delay_ms(500);
        }
}
//*********************************************************************
*
//■■■オープニングデモ関数■■■
void    opening_demonstration()
{
        short   cnt;
        //
        Lcd3_Chr(1, 1, 'j');
        Lcd3_Chr(1, 2, 'f');
        Lcd3_Chr(1, 3, '3');
        Lcd3_Chr(1, 4, 's');
        Lcd3_Chr(1, 5, 'f');
        Lcd3_Chr(1, 6, 'b');
        Delay_ms(1000);
        //
        Lcd3_Chr_Cp(' ');
        Lcd3_Chr_Cp('J');
        Lcd3_Chr_Cp('F');
        Lcd3_Chr_Cp('3');
        Lcd3_Chr_Cp('S');
        Lcd3_Chr_Cp('F');
        Lcd3_Chr_Cp('B');
        Delay_ms(1000);
        //
        Lcd3_Out(2, 1, "JF3SFB");
        Delay_ms(1000);
        //
        Lcd3_Out_Cp(" jf3sfb");
        Delay_ms(1000);
        //
        Lcd3_Cmd(_LCD_CLEAR);
        for (cnt = 0; cnt < 16; cnt++) {
                Lcd3_Chr(1, cnt + 1, 0xFF);
                Delay_ms(100);
        }
        for (cnt = 0; cnt < 16; cnt++) {
                Lcd3_Chr(2, cnt + 1, 0xFF);
                Delay_ms(100);
        }
        Delay_ms(1000);
        //
        for (cnt = 0; cnt < 10; cnt++) {
```

```
                Lcd3_Cmd(_LCD_TURN_OFF);
                Delay_ms(100);
                Lcd3_Cmd(_LCD_TURN_ON);
                Delay_ms(100);
        }
        //
        Lcd3_Cmd(_LCD_RETURN_HOME);
        for (cnt = 0; cnt < 16; cnt++) {
                Lcd3_Chr(1, cnt + 1, ' ');
                Delay_ms(100);
        }
        for (cnt = 0; cnt < 16; cnt++) {
                Lcd3_Chr(2, cnt + 1, ' ');
                Delay_ms(100);
        }
        Delay_ms(1000);
}
//*********************************************************************
*
```

3線式LCD制御ライブラリ

lcd_lib_4bit_nju3711.c

```
//*********************************************************************
*
/*
    ＜3線式□□□制御ライブラリ□□□□□□□コンパチブル）＞

  ※拡張ポート用の□□には□□□□□□□□を使用します。
*/
//*********************************************************************
*
//■■■インクルード■■■
#include "lcd_lib_4bit_nju3711.h"
#include "ex_port_lib_nju3711.h"
//*********************************************************************
*
//■■■□□□文字出力（カレントカーソル位置）関数■■■
void    Lcd3_Chr_Cp(char out_char)
{
        ex_port_out_bit(LCD_D7, out_char.B7);
        ex_port_out_bit(LCD_D6, out_char.B6);
        ex_port_out_bit(LCD_D5, out_char.B5);
        ex_port_out_bit(LCD_D4, out_char.B4);
        ex_port_out_bit(LCD_RS, 1);
        ex_port_out_bit(LCD_EN, 1);
        asm         nop;
        ex_port_out_bit(LCD_EN, 0);
        //
```

```
        ex_port_out_bit(LCD_D7, out_char.B3);
        ex_port_out_bit(LCD_D6, out_char.B2);
        ex_port_out_bit(LCD_D5, out_char.B1);
        ex_port_out_bit(LCD_D4, out_char.B0);
        ex_port_out_bit(LCD_RS, 1);
        ex_port_out_bit(LCD_EN, 1);
        asm         nop;
        ex_port_out_bit(LCD_EN, 0);
        //
        Delay_us(100);
}
//*********************************************************************
*
//■■■□□□文字出力（行列指定）関数■■■
void    Lcd3_Chr(char row, char column, char out_char)
{
        Lcd3_Cmd_8bit(0x80 + (column - 1) + ((row - 1) * 0x40));
        Lcd3_Chr_Cp(out_char);
}
//*********************************************************************
*
//■■■□□□文字列出力（カレントカーソル位置）関数■■■
void    Lcd3_Out_Cp(char *text)
{
        while (*text != 0x00) {
                Lcd3_Chr_Cp(*text);
                text++;
        }
}
//*********************************************************************
*
//■■■□□□文字列出力（行列指定）関数■■■
void    Lcd3_Out(char row, char column, char *text)
{
        Lcd3_Cmd_8bit(0x80 + (column - 1) + ((row - 1) * 0x40));
        Lcd3_Out_Cp(text);
}
//*********************************************************************
*
//■■■□□□コマンド出力関数■■■
void    Lcd3_Cmd(char cmd)
{
        Lcd3_Cmd_8bit(cmd);
        Delay_ms(10);
}
//*********************************************************************
*
//■■■□□□初期化関数■■■
void    Lcd3_Init()
{
        ex_port_init();
```

```
        ex_port_out_bit(LCD_RS, 0);
        ex_port_out_bit(LCD_EN, 0);
        Delay_ms(20);
        Lcd3_Cmd_4bit(0b00110000);    //ファンクションセット(8bitモード)
        Delay_ms(10);
        Lcd3_Cmd_4bit(0b00110000);    //ファンクションセット(8bitモード)
        Delay_ms(10);
        Lcd3_Cmd_4bit(0b00110000);    //ファンクションセット(8bitモード)
        Delay_ms(10);
        Lcd3_Cmd_4bit(0b00100000);    //ファンクションセット(4bitモード)
        Delay_ms(10);
        Lcd3_Cmd_8bit(0b00101000);    //ファンクションセット(4bitモード,1/16
デューティ,5×7ドット)
        Delay_ms(10);
        Lcd3_Cmd_8bit(0b00001000);    //表示オフ
        Delay_ms(10);
        Lcd3_Cmd_8bit(0b00000001);    //表示クリア
        Delay_ms(10);
        Lcd3_Cmd_8bit(0b00000110);    //エントリーモードセット
        Delay_ms(10);
}
//*********************************************************************
*
//■■■□□□初期化用コマンド出力(4bit)関数■■■
void    Lcd3_Cmd_4bit(char cmd)
{
        ex_port_out_bit(LCD_D7, cmd.B7);
        ex_port_out_bit(LCD_D6, cmd.B6);
        ex_port_out_bit(LCD_D5, cmd.B5);
        ex_port_out_bit(LCD_D4, cmd.B4);
        ex_port_out_bit(LCD_RS, 0);
        ex_port_out_bit(LCD_EN, 1);
        asm         nop;
        ex_port_out_bit(LCD_EN, 0);
        //
        Delay_us(100);
}
//*********************************************************************
*
//■■■□□□初期化用コマンド出力(8bit)関数■■■
void    Lcd3_Cmd_8bit(char cmd)
{
        ex_port_out_bit(LCD_D7, cmd.B7);
        ex_port_out_bit(LCD_D6, cmd.B6);
        ex_port_out_bit(LCD_D5, cmd.B5);
        ex_port_out_bit(LCD_D4, cmd.B4);
        ex_port_out_bit(LCD_RS, 0);
        ex_port_out_bit(LCD_EN, 1);
        asm         nop;
        ex_port_out_bit(LCD_EN, 0);
        //
```

```
        ex_port_out_bit(LCD_D7, cmd.B3);
        ex_port_out_bit(LCD_D6, cmd.B2);
        ex_port_out_bit(LCD_D5, cmd.B1);
        ex_port_out_bit(LCD_D4, cmd.B0);
        ex_port_out_bit(LCD_RS, 0);
        ex_port_out_bit(LCD_EN, 1);
        asm         nop;
        ex_port_out_bit(LCD_EN, 0);
        //
        Delay_us(100);
}
//*********************************************************************
*
```

拡張ポート制御ライブラリ

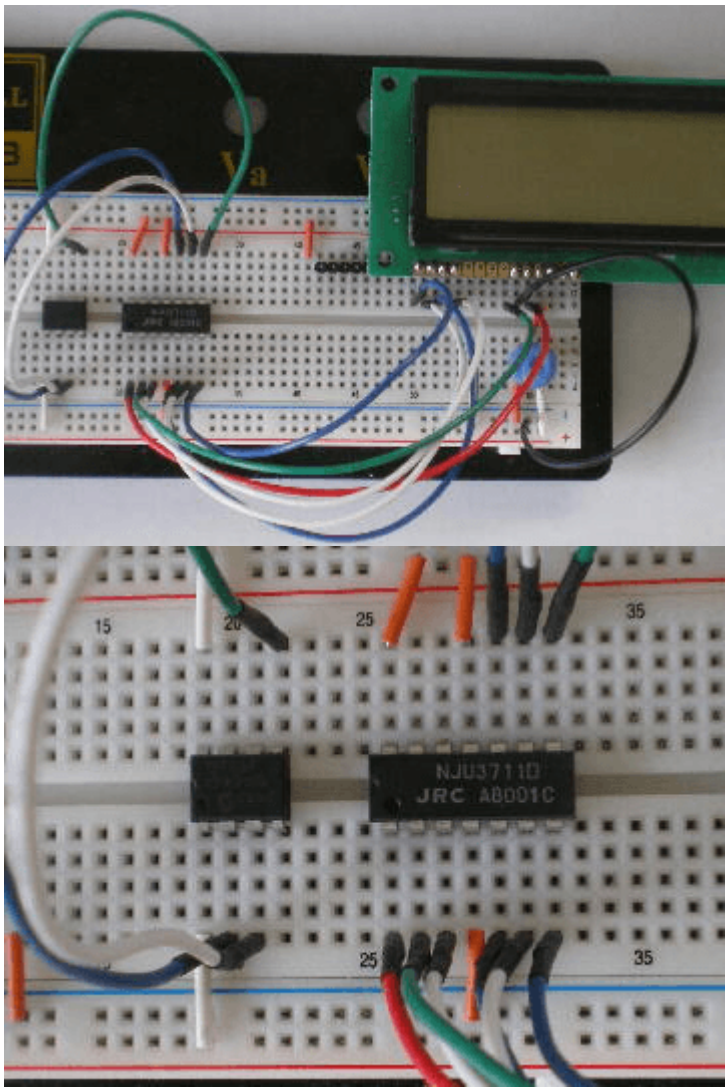
ex_port_lib_nju3711.c

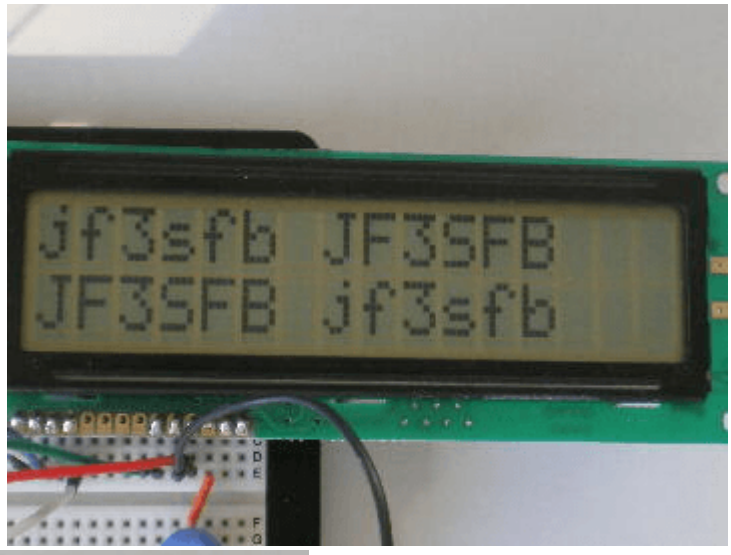
```
//*********************************************************************
*
/*
    ＜拡張ポート（出力専用）制御ライブラリ＞

  ※拡張ポート用の□□には□□□□□□□□を使用します。
*/
//*********************************************************************
*
//■■■インクルード■■■
#include "ex_port_lib_nju3711.h"
//*********************************************************************
*
//■■■拡張ポート初期化関数■■■
char    output_data_register;
void    ex_port_init()
{
        short   cnt;
        //
        STB_Direction = 0;
        CLK_Direction = 0;
        DAT_Direction = 0;
        //
        STB = 1;
        DAT = 0;
        CLK = 0;
        for (cnt = 0; cnt < 8; cnt++) {
                CLK = 1;
                CLK = 0;
        }
        STB = 0;
        STB = 1;
```

```
        output_data_register = 0;
}
//*********************************************************************
*
//■■■拡張ポート出力関数■■■
void    ex_port_out(char output_data)
{
        short   cnt;
        //
        output_data_register = output_data;
        for (cnt = 0; cnt < 8; cnt++) {
                if ((output_data & 0b10000000) != 0) {
                        DAT = 1;
                } else {
                        DAT = 0;
                }
                CLK = 1;
                CLK = 0;
                output_data = output_data << 1;
        }
        STB = 0;
        STB = 1;
}
//*********************************************************************
*
//■■■拡張ポートビット出力関数■■■
void    ex_port_out_bit(char pin, char sts)
{
        switch (pin) {
        case 0:
                output_data_register.B0 = sts;
                break;
        case 1:
                output_data_register.B1 = sts;
                break;
        case 2:
                output_data_register.B2 = sts;
                break;
        case 3:
                output_data_register.B3 = sts;
                break;
        case 4:
                output_data_register.B4 = sts;
                break;
        case 5:
                output_data_register.B5 = sts;
                break;
        case 6:
                output_data_register.B6 = sts;
                break;
        case 7:
```

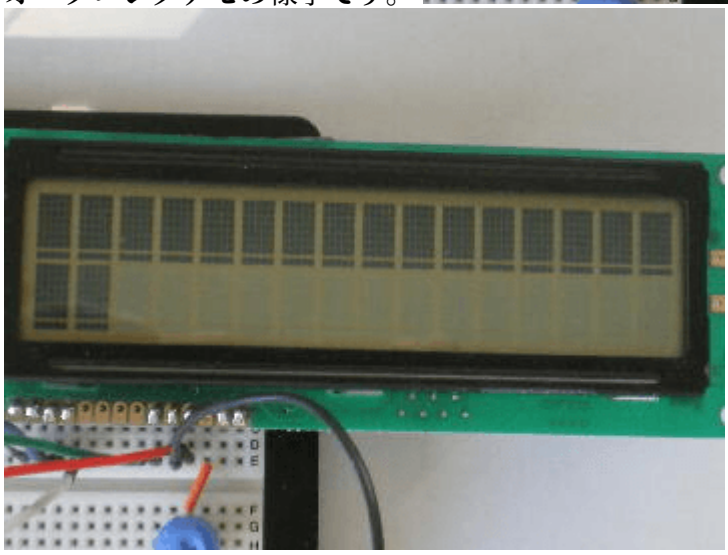
```
                output_data_register.B7 = sts;
                break;
        }
        ex_port_out(output_data_register);
}
//*****************************************************************************
*
```

動作確認

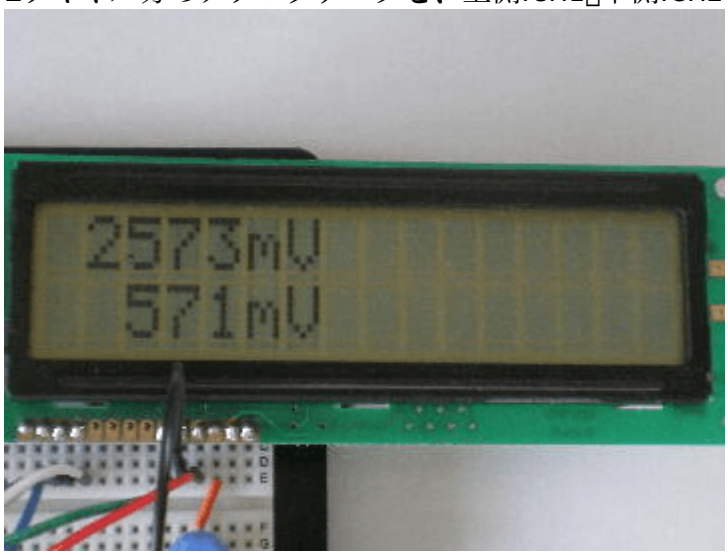


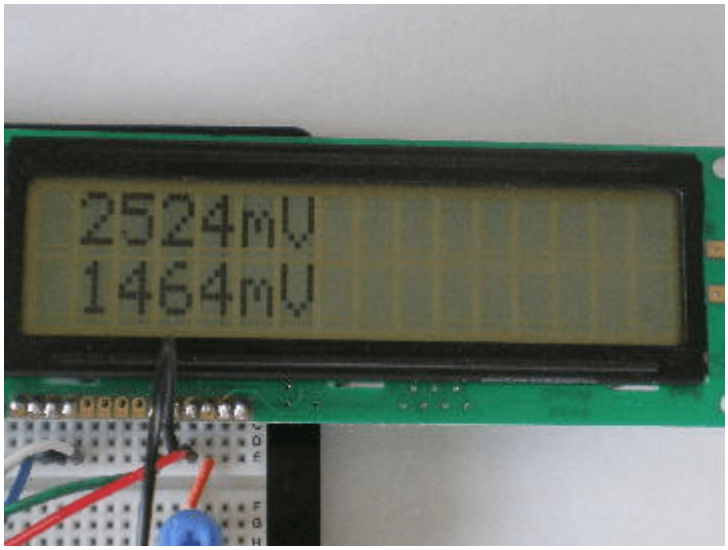


オープニングデモの様子です。



2チャネル分のアナログデータを、上側:CH1、下側:CH2の順番に表示します。





※3線式LCD制御ライブラリの動作確認には□PIC12F683を使用しましたが、他のPICでも同様に動作します。