CGRAM活用(キャラクタ表示LCD)

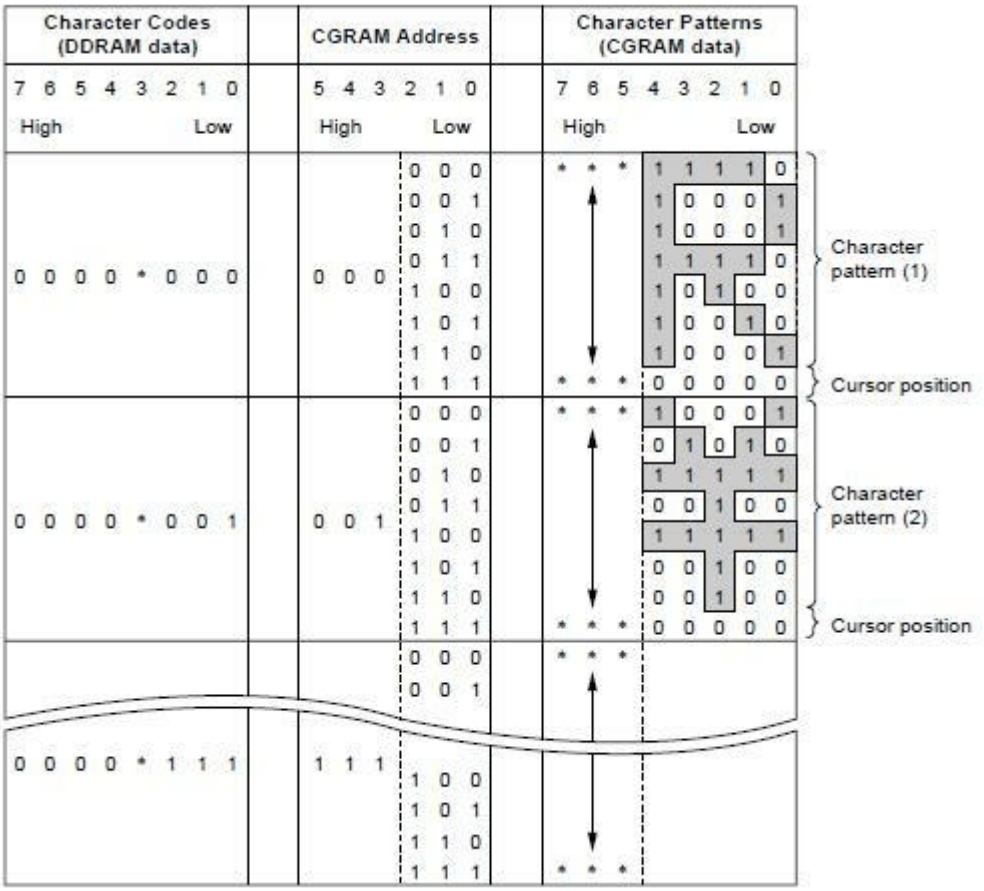
概要

LCD(Liquid Crystal Display:液晶ディスプレイ)には□CGRAM(Character Generator RAM)が用意されており、これを使うことによって、独自のキャラクタを表示させることが出来ます。 今回は、7チャンネル分のアナログデータを、LCDにバー表示させてみました。

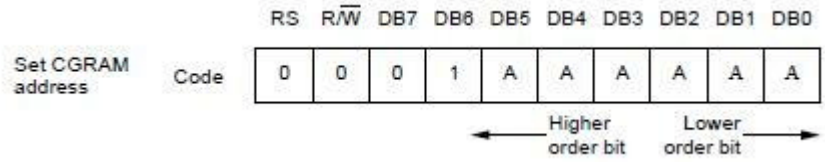
動作原理

<キャラクタコードとキャラクタパターンの関係> 独自のキャラクタは、キャラクタコード(0x00~0x07)に8個分割り当てられています。

<キャラクタコード(0x00~0x07)とCGRAMアドレスとCGRAMデータの関係> キャラクタデータは、8バイト×5個で一組となります。次の例では、キャラクタコード(0x00)に、“R“という文字、キャラクタコード(0x01)に、“ \ ”という 文字を登録した場合の内容です。



<制御**コマンド**> **セット**CGRAM**アドレスコマンドで、**CGRAM**のアドレスを**指定**して、**以降**は**順次**キャラ**

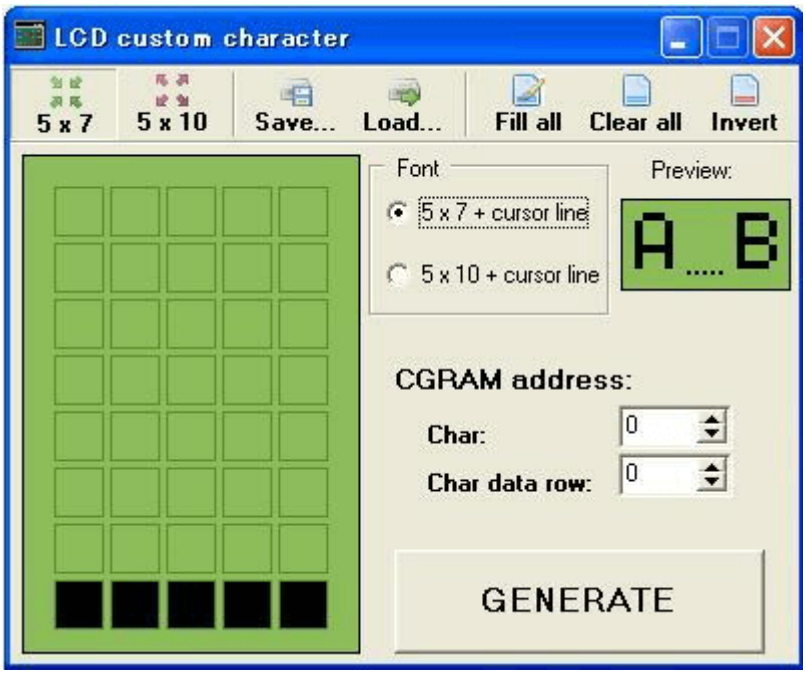


クタデータを書き込**んでいきます**。

<実際**の**表示> 通常**の**文字表示**と**同**じです**。

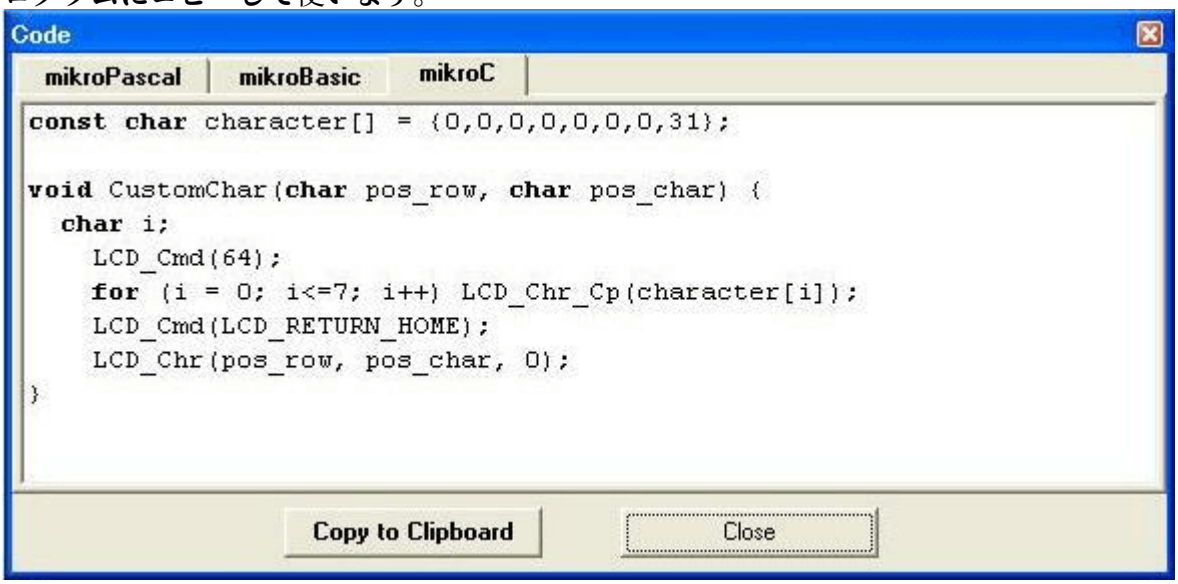
- 標準**キャラクタの**表示
  LCD_Chr(1, 1, 'A');
  LCD_Chr(1, 2, 0x30');
- 独自**キャラクタの**表示
  LCD_Chr(2, 1, 0x00);
  LCD_Chr(2, 2, 0x07);

<mikroC**付属のキャラクタデータ**生成**ツール**> 手計算**でキャラクタデータを**設定**しても**良**いのですが、**結構面倒**くさい**作業**になります。 **このツールを**使**うことにより、**視覚的**にキャラクタデータを**設定**する**

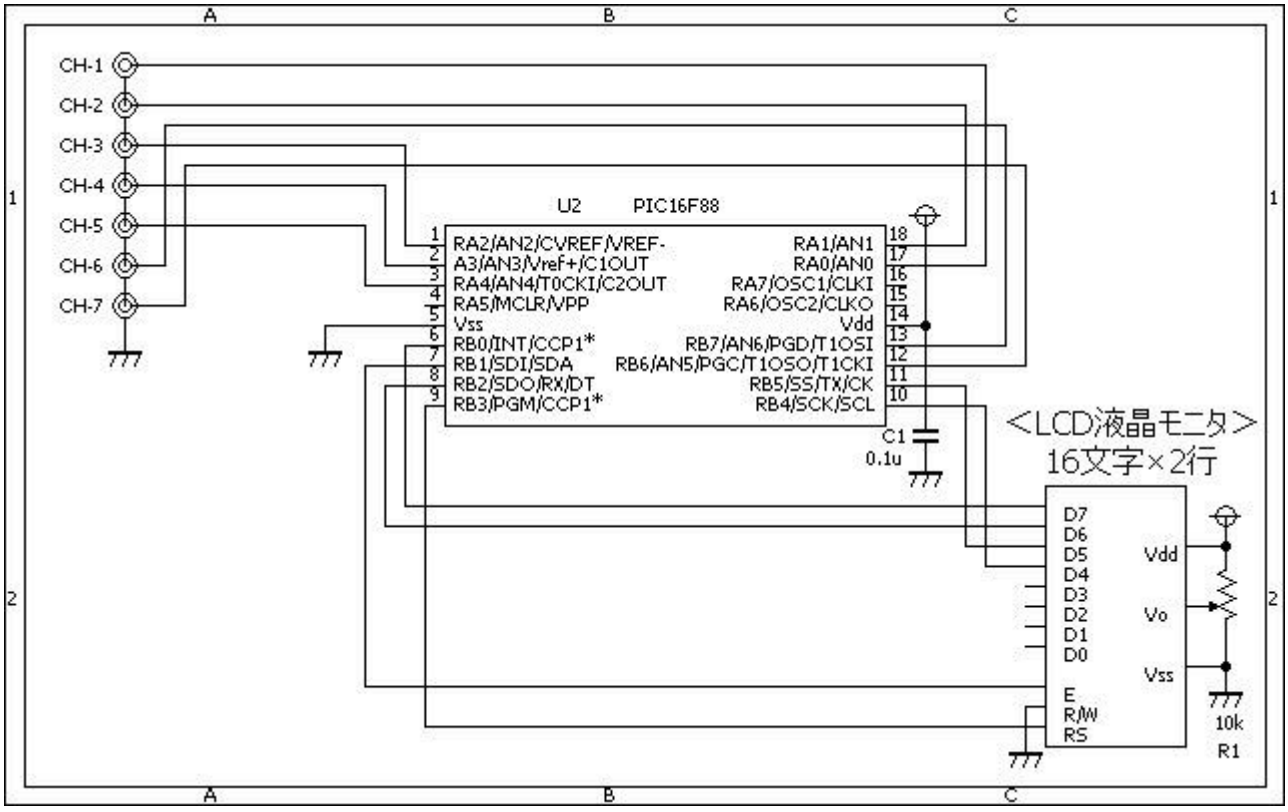


ことが出来るので便利です。

GENERATE(生成)ボタンを押すとキャラクタデータの登録&表示関数を自動的に生成します。 これをプログラムにコピーして使います。



回路図



ソースコード

cgramTest.c

```
//*****************************************************************************
*
/*
□□□□□□□□□□活用（キャラクタ表示□□□□□
*/
//*****************************************************************************
*

const char character1[] = {0,0,0,0,0,0,0,31};
const char character2[] = {0,0,0,0,0,0,31,31};
const char character3[] = {0,0,0,0,0,31,31,31};
const char character4[] = {0,0,0,0,31,31,31,31};
const char character5[] = {0,0,0,31,31,31,31,31};
const char character6[] = {0,0,31,31,31,31,31,31};
const char character7[] = {0,31,31,31,31,31,31,31};
const char character8[] = {31,31,31,31,31,31,31,31};

void RegistCustomChar()
{
    char i;
    //
    LCD_Cmd(64);
    for (i = 0; i<=7; i++) {
```

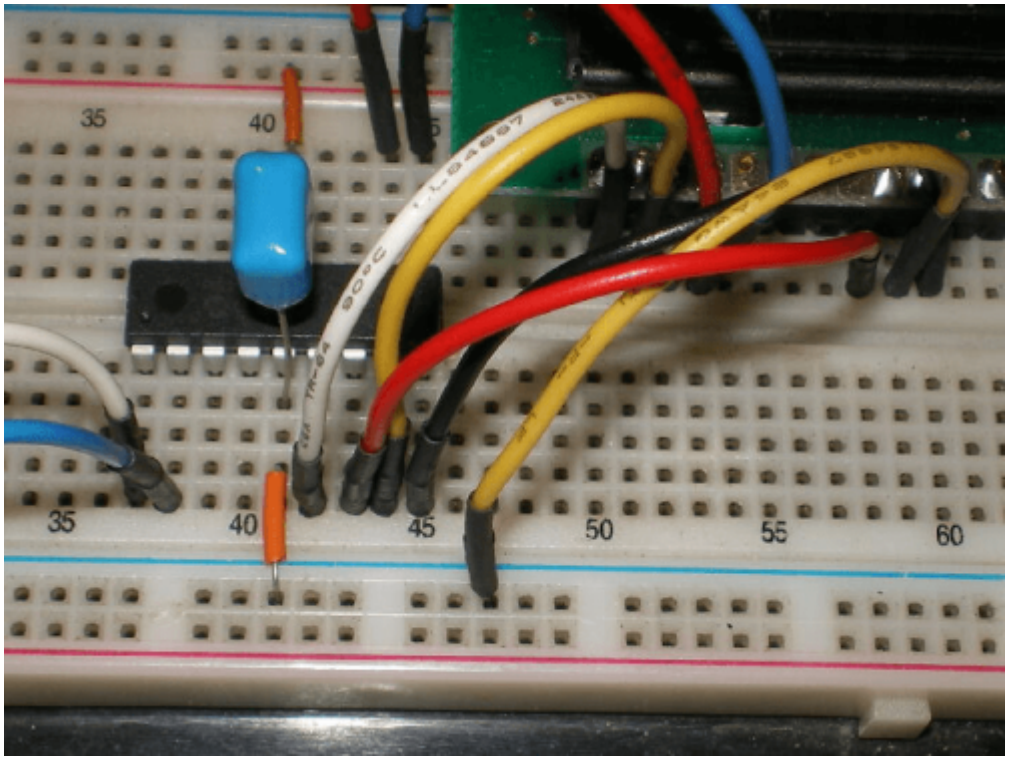
```
        LCD_Chr_Cp(character1[i]);
    }
    for (i = 0; i<=7; i++) {
        LCD_Chr_Cp(character2[i]);
    }
    for (i = 0; i<=7; i++) {
        LCD_Chr_Cp(character3[i]);
    }
    for (i = 0; i<=7; i++) {
        LCD_Chr_Cp(character4[i]);
    }
    for (i = 0; i<=7; i++) {
        LCD_Chr_Cp(character5[i]);
    }
    for (i = 0; i<=7; i++) {
        LCD_Chr_Cp(character6[i]);
    }
    for (i = 0; i<=7; i++) {
        LCD_Chr_Cp(character7[i]);
    }
    for (i = 0; i<=7; i++) {
        LCD_Chr_Cp(character8[i]);
    }
    LCD_Cmd(LCD_RETURN_HOME);
}

//*********************************************************************
*

void   BarDisp(char index, int dat)
{
    switch (dat / 64) {
    case 0:
        Lcd_Chr(1, index, ' ');
        Lcd_Chr(2, index, 0);
        break;
    case 1:
        Lcd_Chr(1, index, ' ');
        Lcd_Chr(2, index, 1);
        break;
    case 2:
        Lcd_Chr(1, index, ' ');
        Lcd_Chr(2, index, 2);
        break;
    case 3:
        Lcd_Chr(1, index, ' ');
        Lcd_Chr(2, index, 3);
        break;
    case 4:
        Lcd_Chr(1, index, ' ');
        Lcd_Chr(2, index, 4);
```

```
        break;
    case 5:
        Lcd_Chr(1, index, ' ');
        Lcd_Chr(2, index, 5);
        break;
    case 6:
        Lcd_Chr(1, index, ' ');
        Lcd_Chr(2, index, 6);
        break;
    case 7:
        Lcd_Chr(1, index, ' ');
        Lcd_Chr(2, index, 7);
        break;
    case 8:
        Lcd_Chr(1, index, 0);
        Lcd_Chr(2, index, 7);
        break;
    case 9:
        Lcd_Chr(1, index, 1);
        Lcd_Chr(2, index, 7);
        break;
    case 10:
        Lcd_Chr(1, index, 2);
        Lcd_Chr(2, index, 7);
        break;
    case 11:
        Lcd_Chr(1, index, 3);
        Lcd_Chr(2, index, 7);
        break;
    case 12:
        Lcd_Chr(1, index, 4);
        Lcd_Chr(2, index, 7);
        break;
    case 13:
        Lcd_Chr(1, index, 5);
        Lcd_Chr(2, index, 7);
        break;
    case 14:
        Lcd_Chr(1, index, 6);
        Lcd_Chr(2, index, 7);
        break;
    case 15:
        Lcd_Chr(1, index, 7);
        Lcd_Chr(2, index, 7);
        break;
    }
}

//*****************************************************************************
*
```

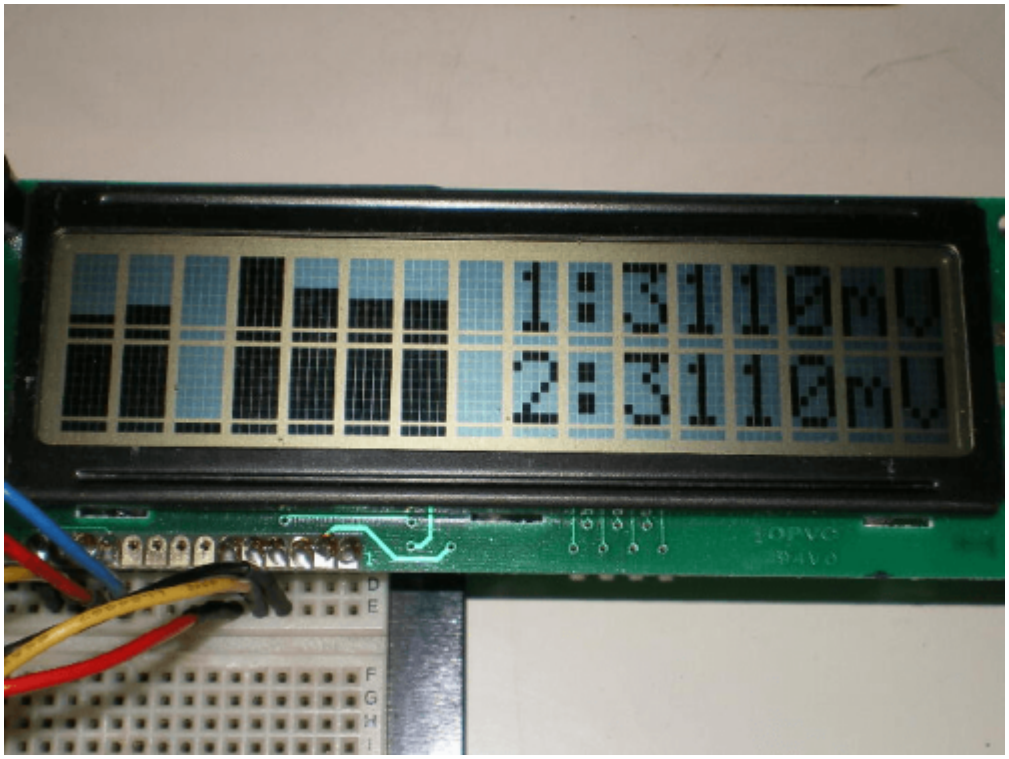
```
void   main()
{
    int        dat0, dat1, dat2, dat3, dat4, dat5, dat6;
    char    buf[6];
    //
    OSCCON = 0b01110000;         //クロックは8Mhz
    CMCON  = 0b00000111;         //コンパレータは使用しない。
    ANSEL  = 0b01111111;         // □□□変換を使用する。
    TRISA  = 0b11111111;
    TRISB  = 0b11000000;
    //□□□の初期化
    Lcd_Config(&PORTB, 3, 1, 2, 0, 2, 5, 4);
    RegistCustomChar();
    Lcd_Cmd(LCD_CURSOR_OFF);
    Lcd_Cmd(LCD_CLEAR);
    //
    while (1) {
        BarDisp(1, dat0 = Adc_Read(0));
        BarDisp(2, dat1 = Adc_Read(1));
        BarDisp(3, dat2 = Adc_Read(2));
        BarDisp(4, dat3 = Adc_Read(3));
        BarDisp(5, dat4 = Adc_Read(4));
        BarDisp(6, dat5 = Adc_Read(5));
        BarDisp(7, dat6 = Adc_Read(6));
        Lcd_Out(1, 9, "1:");
        WordToStr(dat0 * 4.8828125, buf);
        Lcd_Out(1, 11, &buf[1]);
        Lcd_Out(1, 15, "mV");
        Lcd_Out(2, 9, "2:");
        WordToStr(dat1 * 4.8828125, buf);
        Lcd_Out(2, 11, &buf[1]);
        Lcd_Out(2, 15, "mV");
        Delay_ms(500);
        //
        BarDisp(1, dat0 = Adc_Read(0));
        BarDisp(2, dat1 = Adc_Read(1));
        BarDisp(3, dat2 = Adc_Read(2));
        BarDisp(4, dat3 = Adc_Read(3));
        BarDisp(5, dat4 = Adc_Read(4));
        BarDisp(6, dat5 = Adc_Read(5));
        BarDisp(7, dat6 = Adc_Read(6));
        Lcd_Out(1, 9, "3:");
        WordToStr(dat2 * 4.8828125, buf);
        Lcd_Out(1, 11, &buf[1]);
        Lcd_Out(1, 15, "mV");
        Lcd_Out(2, 9, "4:");
        WordToStr(dat3 * 4.8828125, buf);
        Lcd_Out(2, 11, &buf[1]);
        Lcd_Out(2, 15, "mV");
        Delay_ms(500);
        //
```

```
        BarDisp(1, dat0 = Adc_Read(0));
        BarDisp(2, dat1 = Adc_Read(1));
        BarDisp(3, dat2 = Adc_Read(2));
        BarDisp(4, dat3 = Adc_Read(3));
        BarDisp(5, dat4 = Adc_Read(4));
        BarDisp(6, dat5 = Adc_Read(5));
        BarDisp(7, dat6 = Adc_Read(6));
        Lcd_Out(1, 9, "5:");
        WordToStr(dat4 * 4.8828125, buf);
        Lcd_Out(1, 11, &buf[1]);
        Lcd_Out(1, 15, "mV");
        Lcd_Out(2, 9, "6:");
        WordToStr(dat5 * 4.8828125, buf);
        Lcd_Out(2, 11, &buf[1]);
        Lcd_Out(2, 15, "mV");
        Delay_ms(500);
        //
        BarDisp(1, dat0 = Adc_Read(0));
        BarDisp(2, dat1 = Adc_Read(1));
        BarDisp(3, dat2 = Adc_Read(2));
        BarDisp(4, dat3 = Adc_Read(3));
        BarDisp(5, dat4 = Adc_Read(4));
        BarDisp(6, dat5 = Adc_Read(5));
        BarDisp(7, dat6 = Adc_Read(6));
        Lcd_Out(1, 9, "7:");
        WordToStr(dat6 * 4.8828125, buf);
        Lcd_Out(1, 11, &buf[1]);
        Lcd_Out(1, 15, "mV");
        Lcd_Out(2, 9, "        ");
        Delay_ms(500);
        //
    }
}//~!

//*********************************************************************
*
```

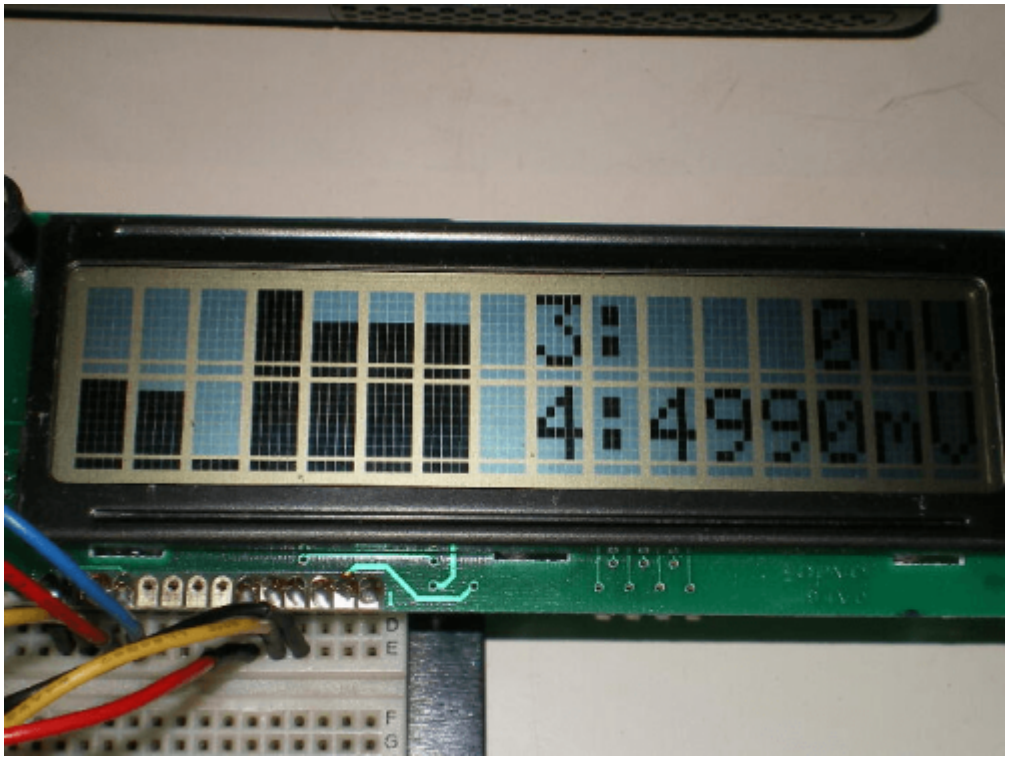
動作確認



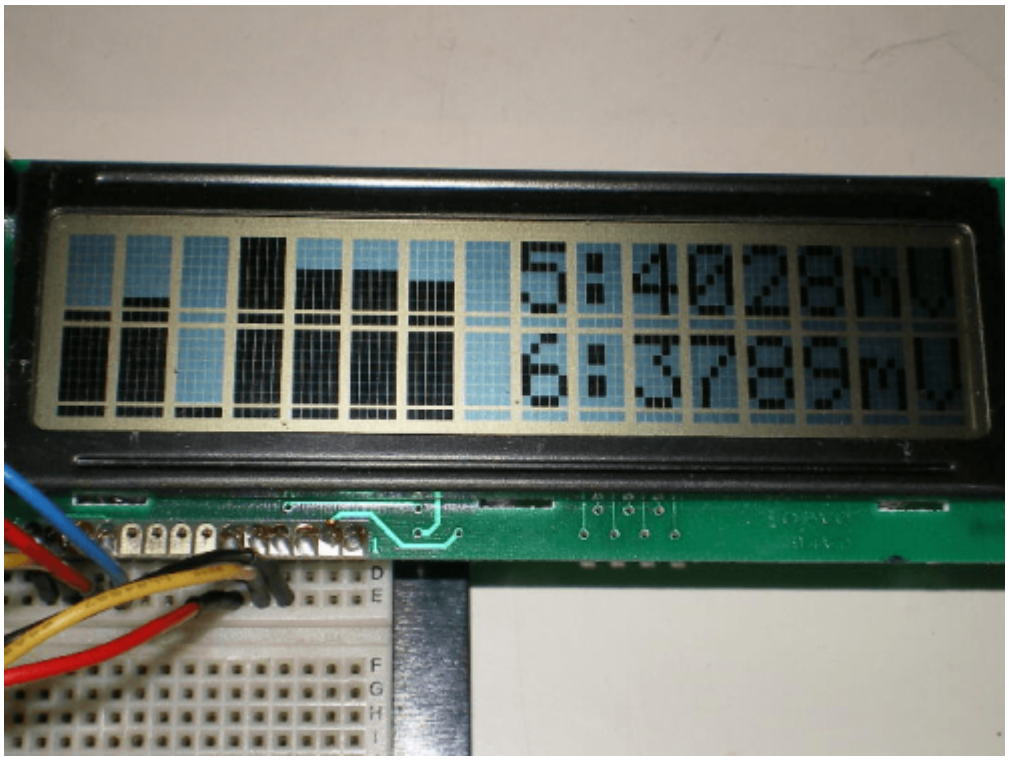
左側:**アナログデータのバー**表示(CH-1~CH-7) 右側:**アナログデータの**数値表示(CH-1とCH-2)



左側:**アナログデータのバー**表示(CH-1~CH-7) 右側:**アナログデータの**数値表示(CH-3とCH-4)



左側:**アナログデータのバー**表示(CH-1~CH-7) 右側:**アナログデータの**数値表示(CH-5とCH-6)



左側:**アナログデータのバー**表示(CH-1~CH-7) 右側:**アナログデータの**数値表示(CH-7)