```
/*
 *      LCD インターフェイスプログラム （HI-TECH社の原本参考）
 *        _delay_ms(), _delay_us()  を使用する。１２．８MHzXCO使
 *      うときはDELAYルーチンが別に必要となる。
 *        LCD制御用ICとして世の中標準である日立製HD44780コンパチ
 *        制御LSIインターフェイスに対応している。4ビットモード接続
 *        を使う。LCDの接続コネクタは標準14ピンを使う。
 *
 *      RC2-5をLCDデータビット4ビットに接続
 *      RA0はLCDのRS入力(レジスタ選択)に接続
 *      RA1はLCDのEN入力(イネーブル)に接続
 *      W:Low書き込み専用で使う
 *
 *      このプログラムを使用するためにまずはポート(TRISA,
 *      TRISB、TRISAC)を初期化しておく。その後このプログラムを
 *      呼ぶことができるようになる。
 *
 *      2011.09.29 LCD_STRBとLCD_writeタイミングを短めにした
 *      PIC12F1822のMSSPマスターで動作確認
 *              2011.10.21  PIC16F1823対応に変更しました
 *      2012.8.30   PIC16F1827対応に変更しました
 *              2014.8.09   PIC16F1827で16MHz対応タイミング  DATA/RS/ENに２０μS
 *              2014.10.03  8-14-20基板対応のPIC16F1829用に修正
 */

#define _XTAL_FREQ 8000000
#define PIC_CLOCK 800000

#ifndef _XTAL_FREQ
 // XTAL_FREQの指定ない場合はシステムクロックは4MHzだと思う
 #define _XTAL_FREQ 4000000
#endif

#include <htc.h>
#include "lcd.h"

#define  LCD_RS LATA0
#define LCD_EN LATA1

unsigned char lcd_rs,lcd_data;                  // RS command:0,RS data:0x80

void    LCD_STROBE(void){
        _delay_us(10);                          // タイミング
        LATC=((lcd_data&0xF)<<2) | (PORTC&0xC3);        // PORTC中位データ維持
        LCD_RS = lcd_rs;                        // コマンドかデータか
        _delay_us(20);                          // タイミング
        LCD_EN=1;
        _delay_us(20);                          // タイミング
        LCD_EN=0;
        _delay_us(20);                          // タイミング
        }

/*
 * 1バイトを2回の4ビットモードでLCDに書く関数
```

```
 * 事前にLCD_RSを1:データ、0:コマンドに設定必要
 */
void
lcd_write(unsigned char c)
{
        lcd_data = ( ( c >> 4 ) & 0x0F ) ;          // 4ビットシフトして上位4ビット
        _delay_us(5);                               // タイミング
        LCD_STROBE();                               // EN線をOn,OFF
        _delay_us(5);                               // タイミング
        lcd_data = ( c & 0x0F ) ;                   // 下位4ビットを出力
        LCD_STROBE();
}

/*
 *      LCDクリアして、カーソルはホームへ
 */
void
lcd_clear(void)
{
        lcd_rs = 0;                                 // RSをコマンドモードRS:0に
        lcd_write(0x1);                             // クリアコマンド1を書く
        _delay_ms(5);                               // クリア処理には2mSぐらい掛かる
}

/*
 * LCDにバイト列を書きこむ
 */
void
lcd_puts(const char * s)
{
        lcd_rs = 0x01;                              // データ転送モード  RS:1
        while(*s)                                   // バイト列の最後にゼロ
        lcd_write(*s++);                            // 1バイト書いては次のアドレス指定
}

/*
 * 1文字分のコード(ニブルを2回)書き込みます
 */
void
lcd_putch(char c)
{
        lcd_rs = 0x1;                               // 表示データ書き込み設定
        lcd_write( c );                             // 1バイト書きます(2ニブル)
}

/*
 * カーソル位置を設定します  HD44780ルールに従ったメモリ位置です
 */
void
lcd_goto(unsigned char pos)
{
        lcd_rs = 0;                                 // コマンド書き込み設定です
        lcd_write(0x80+pos);                        // カーソル位置指定は＃7をにします
}
```

```
/*
 * RSを変数でもらってWRITE（2ニブル）します
 */
void
lcd_write_rs(unsigned char c, unsigned char rs)
{
        lcd_rs = rs;                            // 引数のrs値を引き渡す
        lcd_write( c );                         // 1バイト書きます（2ニブル）
}

/*
 * PORTAを初期化、LCDコントローラを初期化するコマンド書き込み
 */
void
lcd_init()
{
        char init_value;

        init_value = 0x3;                       // LCDコントローラの初期化コマンド

        LCD_RS = 0;                             // RS信号（PB0)はコマンドモード：0
        LCD_EN = 0;                             // エネーブルビットを0

        _delay_ms(200);                         // 電源投入後最低でも15mS待ってから
        lcd_data = init_value;                  // 0x03を設定
        LCD_STROBE();
        _delay_ms(20);
        LCD_STROBE();
        _delay_us(200);
        LCD_STROBE();
        _delay_us(200);
        lcd_data  = 2;                          // 4ビットモード指定：0x02を設定
        LCD_STROBE();

        lcd_write(0x28);                        // インターフェイス長さコマンド
        _delay_us(10);                          // タイミング
        lcd_write(0xF);                         // 表示オン、カーソル表示オンでブリンクする
        _delay_us(10);                          // タイミング
        lcd_clear();                            // 画面の消去
        _delay_ms(200);
        lcd_write(0x6);                         // 入力 モードに設定
        _delay_us(10);                          // タイミング
}
```