```
/* ************************************************************************
 *  LCD interface                                                         *
 *  Uses routines from delay.c                                            *
 *  This code will interface to a standard LCD controller                 *
 *  like the Hitachi HD44780. It uses it in 4 bit mode, with              *
 *  the hardware connected as follows (the standard 14 pin                *
 *  LCD connector is used):                                               *
 *                                                                        *
 *  PORTB bits 0-3 are connected to the LCD data bits 4-7 (high nibble)   *
 *  PORTA bit 2 is connected to the LCD RS input (register select)        *
 *  PORTA bit 3 is connected to the LCD EN bit (enable)                   *
 *                                                                        *
 *  To use these routines, set up the port I/O (TRISA, TRISB) then        *
 *  call lcd_init(), then other routines as required.                     *
 *                                                                        *
 * R/W not used !  Also remember if using port A to set PORTA to digital  *
 * ---------------------------------------------------------------------- *
 * Copyright (C) 2004 KOCH Engineering                                    *
 * Henrik J. Koch                                                         *
 * email:   mail@koch-engineering.com                                     *
 * web:     http://www.koch-engineering.com                               *
 * (MPLAB-IDE 6.43)                                                       *
 * Version 1.0  Marts 14. 2004                                            *
 ************************************************************************ */
#include    <pic.h>         // PORTA
#include    "lcd.h"         // constants
#include    "delay.h"       // DelayUs, DelayMs
#include    "receiver.h"    // BYTE

static bit LCD_RS   @ ((unsigned)&PORTA*8+2);   // Register select
static bit LCD_EN   @ ((unsigned)&PORTA*8+3);   // Enable
#define LCD_STROBE  ((LCD_EN = 1),(LCD_EN=0))   // Falling edge latches data

/* ************************************************
 * write a byte to the LCD in 4 bit mode          *
 ************************************************ */
void lcd_write(unsigned char c)
{
    PORTB = (PORTB & 0xF0) |   (c >> 4);
    LCD_STROBE;
    PORTB = (PORTB & 0xF0) |   (c & 0x0F);
    LCD_STROBE;
    DelayUs(40);
}

/* ************************************************
 *  Clear and home the LCD                        *
 ************************************************ */
void lcd_clear(void)
{
    LCD_RS = 0;
    lcd_write(0x1);
    DelayMs(2);
}

/* ************************************************
 * write a string of chars to the LCD             *
 ************************************************ */
void lcd_puts(const char * s)
{
    LCD_RS = 1; // write characters
    while(*s)
        lcd_write(*s++);
}

/* ************************************************
 * write one character to the LCD                 *
 ************************************************ */
void lcd_putch(char c)
{
    LCD_RS = 1; // write characters
    PORTB = (PORTB & 0xF0) |   (c >> 4);
    LCD_STROBE;
```

```
        PORTB =
75
    /* ************************************************
     *  Entry Mode                                    *
     ************************************************ */
    void lcd_entryMode(BYTE entrymodevar)
80  {
        LCD_RS = 0;                 // write control bytes
        lcd_write((entrymodevar & 0x03) | 0x04);    // set display control
    }

85  /* ************************************************
     *  Function Set                                  *
     ************************************************ */
    void lcd_functionSet(BYTE functionsetvar)
    {
90      LCD_RS = 0;                 // write control bytes
        lcd_write((functionsetvar & 0x1C) | 0x20);  // set display control
    }

    /* ************************************************
95   *  Cursor or Display Shift                       *
     ************************************************ */
    void lcd_cursorDisplayShift(BYTE cursordisplayshiftvar)
    {
        LCD_RS = 0;                 // write control bytes
100     lcd_write((cursordisplayshiftvar & 0x0C) | 0x10);   // set display control
    }

    /* ************************************************
     * initialise the LCD - put into 4 bit mode       *
105  ************************************************ */
    void lcd_init(void)
    {
        DelayMs(15);        // power on delay
        PORTB = 0x3;        // attention!
110     LCD_STROBE;
        DelayMs(5);
        LCD_STROBE;
        DelayUs(100);
        LCD_STROBE;
115     DelayMs(5);
        PORTB = 0x2;        // set 4 bit mode
        LCD_STROBE;
        DelayUs(40);
        lcd_functionSet(LCD_4_BITS | LCD_2_LINES | LCD_5x7_DOTS);
120     lcd_entryMode(LCD_SHIFT_OFF | LCD_INCREMENT);
        lcd_cursorDisplayShift(LCD_CURSOR_MOVE | LCD_SHIFT_RIGHT);
        lcd_displayControl(LCD_DISPLAY_ON | LCD_CURSOR_OFF | LCD_BLINK_OFF);    // dis
    }
```