```
//-----------------------------------------------------------------------------
// LCD.c
//-----------------------------------------------------------------------------
// Author: Baylor Electromechanical Systems
//
// Operates on an internal 2 MHz oscillator.
//
// Target: Cygnal Educational Development Board / C8051F001
// Tool chain: KEIL C51 6.03 / KEIL EVAL C51
//
// Utilizes the C8051F001 as a serial LCD driver at 9600 bps. See accompanying
// documentation for more details. Code has been left commented to run on
// external 18.432 MHz clock as well.
//

//-----------------------------------------------------------------------------
// Includes
//-----------------------------------------------------------------------------
#include <c8051f000.h>              // SFR declarations
#include <stdio.h>

//-----------------------------------------------------------------------------
// Global CONSTANTS
//-----------------------------------------------------------------------------

//#define SYSCLK 18432000            // approximate SYSCLK frequency in Hz
#define SYSCLK 2000000              // approximate SYSCLK frequency in Hz
#define BAUDRATE 9600               // baud rate for UART

sbit    RS  = P0^5;
sbit    RW  = P0^6;
sbit    E   = P0^7;

static int DSP_CLR = 0x01;
static int FN_SET = 0x38;
static int LCD_OFF = 0x0B;
static int LCD_ON = 0x0F;
static int ENT_MD = 0x06;


//-----------------------------------------------------------------------------
// Function PROTOTYPES
//-----------------------------------------------------------------------------
void SYSCLK_Init (void);
void PORT_Init (void);
void LCD_DAT (unsigned char dat);
void LCD_CMD (unsigned char cmd);
void wait (int ms);
void LCD_Init (void);
void UART0_Init(void);
void banner (void);

//-----------------------------------------------------------------------------
// Global VARIABLES
//-----------------------------------------------------------------------------
bit row = 0;                // row indicator

//-----------------------------------------------------------------------------
// MAIN Routine
//-----------------------------------------------------------------------------
void main (void)
{

    unsigned char c;            // temp character
    WDTCN = 0xde;               // disable watchdog timer
    WDTCN = 0xad;
    SYSCLK_Init ();
    wait (1000);                // wait 1 sec for LCD to cycle on
```

```
    PORT_Init ();
    LCD_Init ();                // intialize LCD
    UART0_Init();
    wait (10);
    banner ();                  // display boot up banner

    while (1)
    {
        c=_getkey();            // read in char
        switch (c)
          {
            case 0xfe : LCD_CMD (_getkey()); break;     // instruction
            case '\n' : break;                          // return and newline
            case '\r' :  {  if (row) LCD_CMD (0x80);    // go to 1st line
                            else LCD_CMD (0xc0);    }   // go to 2nd line
                         break;
            case '\0' : LCD_DAT (0);    break;          // null byte
            default   : LCD_DAT (c);                    // display character
          }                                             // end switch
    }                           // end while
}

//-------------------------------------------------------------------------
// Initialization Subroutines
//-------------------------------------------------------------------------


//-------------------------------------------------------------------------
// SYSCLK_Init
//-------------------------------------------------------------------------
//
// This routine initializes the system clock to use an 3.6864 MHz crystal
// as its clock source.
//
void SYSCLK_Init (void)
{
//  int i;                          // delay counter
//  OSCXCN = 0x66;                  // EXTERNAL Oscillator Control Register

//  for (i=0; i < 255; i++) ;       // XTLVLD blanking interval (>1ms)

//  while ( (OSCXCN & 0x80) == 0 ); // wait for xtal osc to start up

//  OSCICN = 0x88;                  // select external oscillator as SYSCLK
                                    // source and enable missing clock
                                    // detector
        OSCICN = 0x94;              // Internal Oscillator enabled
}

//-------------------------------------------------------------------------
// PORT_Init
//-------------------------------------------------------------------------
void PORT_Init (void)
{
    XBR0    = 0x04;             // Enable UART
    XBR1    = 0x00;
    XBR2    = 0x40;             // Enable crossbar and weak pull-ups
    PRT0CF |= 0xE0;             // Output configuration for P0^5 - P0^7
    PRT1CF  = 0xFF;             // Output configuration for P1
    P0      = 0x00;
    P1      = 0x00;

}

//-------------------------------------------------------------------------
// UART0_Init
//-------------------------------------------------------------------------
//
```

```
// Configure the UART0 using Timer1, for <baudrate> and 8-N-1.
//
void UART0_Init (void)
{
   SCON   = 0x50;                       // SCON0: mode 1, 8-bit UART, enable RX
   SCON &= 0xFC;                        // clear interrupt pending flags
   TMOD    = 0x20;                      // TMOD: timer 1, mode 2, 8-bit reload
   TCON = 0x40;                         // Timer Control Register
   TH1    = -(SYSCLK/BAUDRATE/16);      // set Timer1 reload value for baudrate
   TR1    = 1;                          // start Timer1
   CKCON |= 0x10;                       // Timer1 uses SYSCLK as time base
   PCON  |= 0x80;                       // SMOD00 = 1
   TI    = 1;                           // Indicate TX0 ready
}

//-----------------------------------------------------------------------------
// LCD_Init
//-----------------------------------------------------------------------------
void LCD_Init (void)
{
    E = 1;              wait (50);
    LCD_CMD (0x30);     wait (50);
    LCD_CMD (0x30);     wait (50);
    LCD_CMD (0x30);     wait (50);
    LCD_CMD (FN_SET);   wait (2);
    LCD_CMD (LCD_OFF);  wait (2);
    LCD_CMD (DSP_CLR);  wait (2);
    LCD_CMD (ENT_MD);   wait (2);
    LCD_CMD (LCD_ON);   wait (2);

}

//-----------------------------------------------------------------------------
// Local Functions
//-----------------------------------------------------------------------------

//-----------------------------------------------------------------------------
// LCD_DAT
//-----------------------------------------------------------------------------
void LCD_DAT (unsigned char dat)
{

    P1 = 0x00;
    RS = 1;
    E = 1; P1 = dat;
    E = 0;
    E = 1; P1 = 0x00;

}

//-----------------------------------------------------------------------------
// LCD_CMD
//-----------------------------------------------------------------------------
void LCD_CMD (unsigned char cmd)
{
    if ((cmd == 0x01) || (cmd == 0x02)) // clear and home indicate row 0
        row = 0;
    else
        if ((cmd >= 0x80) && (cmd <= 0x8f))
            row = 0;                    // first row addresses indicate row 0
        else
            if ((cmd >= 0xc0) && (cmd <= 0xcf))
                row = 1;                // second row addresses indicate row 1

    P1 = 0x00;
    RS = 0;
    E = 1; P1 = cmd;
    E = 0;
```

```
    E = 1; P1 = 0x00;

}

//-----------------------------------------------------------------------------
// Wait
//-----------------------------------------------------------------------------
//
// This is an approximate X ms delay routine.
//
void wait (int ms)
{
    int i;
    int j;
    for (j = 1; j <= 50; j++) for (i = 1; i <= ms; i++);

}



//-----------------------------------------------------------------------------
// Banner
//-----------------------------------------------------------------------------
//
// This routine displays a scrolling banner on the LCD. This function is
// called at boot up.
//
void banner (void)
{
    LCD_CMD (DSP_CLR);              // clear LCD display
    wait (10);
    LCD_CMD (0x85); wait (1);   LCD_DAT ('C');  wait (150);
    LCD_DAT ('Y');  wait (150); LCD_CMD (0x85); wait (1);
    LCD_DAT (' ');  wait (1);   LCD_CMD (0x87); wait (1);
    LCD_DAT ('G');  wait (150); LCD_CMD (0x86); wait (1);
    LCD_DAT (' ');  wait (1);   LCD_CMD (0x88); wait (1);
    LCD_DAT ('N');  wait (150); LCD_CMD (0x87); wait (1);
    LCD_DAT (' ');  wait (1);   LCD_CMD (0x89); wait (1);
    LCD_DAT ('A');  wait (150); LCD_CMD (0x88); wait (1);
    LCD_DAT (' ');  wait (1);   LCD_CMD (0x8a); wait (1);
    LCD_DAT ('L');  wait (150); LCD_CMD (0x89); wait (1);
    LCD_DAT (' ');  wait (1);   LCD_CMD (0x8b); wait (1);
    wait (150); LCD_CMD (0x8a); wait (1);   LCD_DAT (' ');
    wait (200); LCD_CMD (0x85); wait (1);   LCD_DAT ('C');
    wait (1);   LCD_DAT ('Y');  wait (1);   LCD_DAT ('G');
    wait (1);   LCD_DAT ('N');  wait (1);   LCD_DAT ('A');
    wait (1);   LCD_DAT ('L');
    wait (3000);                    // wait 3 seconds at end
    LCD_CMD (DSP_CLR);              // clear LCD display
    wait (10);
}
```