```
/***********************************************
 * File: lcd.c          Project by: David Hodgdon *
 * Class: ECE476        Instructor: Bruce Land    *
 ***********************************************/
unsigned char lcd_buffer[60];                   // Buffer for strings
#ifdef ATMEL
/*************************************************************
 * Based on code from previous semesters.
 * I think the first code base was 2001 graphing calculator
 * http://instruct1.cit.cornell.edu/courses/ee476/FinalProjects/s2001/yl131/
 * but more recently Cornopoly in 2002
 * http://instruct1.cit.cornell.edu/courses/ee476/FinalProjects/s2002/jq13/webpa
ge/476monopoly.html
 * and The Russian Bloc Game also 2002
 * http://instruct1.cit.cornell.edu/courses/ee476/FinalProjects/s2002/dkl22/
 ************************************************************************
*****/
#pragma warn-
// LCD drivers
// PORT B is the control port
#define CONTROLPORT PORTA
// PORT C is the data port
#define DATAPORT        PORTC
// NOTE: you must also set DDRX registers in init function


#include <Mega32.h>
#include <delay.h>
#include <stdlib.h>
#include "lcd.h"


// LCD control definitions
#define LcdReset                0x00            // LCD reset
#define DataWrite               0x03            // Data write
#define DataSetup               0x07            // Data setup
#define CommandWrite            0x43            // Command write
#define DataRead                0x45            // Data receive
#define LcdNop                  0x47            // Do nothing
#define CommandSetup            0x47            // Command setup

// LCD command definitions
#define SystemSet                       0x40            // Initialize LCD
#define MemoryWrite             0x42            // Write to display memory
#define MemoryRead              0x43            // Read display memory
#define Scroll                          0x44            // Set start address, re
gions
#define SetCursor               0x46            // Set cursor address
#define ReadCursor              0x47            // Read cursor address
#define CursorRight             0x4C            // Cursor right
#define CursorLeft              0x4D            // Cursor left
#define CursorUp                0x4E            // Cursor up
#define CursorDown              0x4F            // Cursor down
#define Standby                         0x53            // Go to standby
#define DisplayOff              0x58            // Disable display
#define DisplayOn               0x59            // Enable display
#define HorizontalScroll        0x5A            // Set horizontal scroll positio
n
#define Overlay                         0x5B            // Set display overlay
#define CharAddress             0x5C            // Start address of character RA
M
#define CursorForm              0x5D            // Set cursor type


flash unsigned char SystemCommand[8]={0x30,0x87,0x07,0x27,0x2F,0xC7,0x28,0x00};
//#define NEWSCROLL 1
#ifdef NEWSCROLL
//                                  SAD1L 1H  SL1  SAD2L 2H  SL2
flash unsigned char ScrollCommand[6]={0x00,0x00,0xC7,0x00,0x05,0xC7};
```



```
#else
flash unsigned char ScrollCommand[6]={0x00,0x00,0xC8,0xE8,0x03,0xC8};
#endif

static unsigned char lineNum;
static unsigned char colNum;

// Declare LCD functions
void WriteCommand(unsigned char Command);       // Write command to LCD
void WriteData(unsigned char Data);             // Write data to LCD
void MoveCursor(unsigned int Address);          // Move the cursor
void InitLCD(void);                             // Initialize the LCD
void ClearLCD(void);                            // Clear the LCD
void PrintBuffer(void);                         // Print contents in LCD buffer
string


// LCD functions
static void WriteCommand(unsigned char Command) // Write command sequence to LCD
{
        DATAPORT = Command;
        CONTROLPORT = CommandSetup;
        CONTROLPORT = CommandWrite;
        CONTROLPORT = CommandSetup;
}

static void WriteData(unsigned char Data)               // Write data sequence t
o LCD
{
        DATAPORT = Data;
        CONTROLPORT = DataSetup;
        CONTROLPORT = DataWrite;
        CONTROLPORT = DataSetup;
}

static void MoveCursor(unsigned int Address)            // Move the cursor
{
        WriteCommand(SetCursor);
        WriteData(Address);
        WriteData(Address>>8);
}

void InitLCD(void)                              // Initialization routine (see d
atasheet, Section 9)
{
        unsigned char i;

        DDRA = 0xff;                            // Set control port to output
        DDRC = 0xff;                            // Set data port to output
        CONTROLPORT = LcdNop;
        delay_ms(5);                            // Delay 2 ms
        CONTROLPORT = LcdReset;                 // Assert reset
        delay_ms(3);
        CONTROLPORT = LcdNop;
        delay_ms(2);

        WriteCommand(SystemSet);
        for(i=0;i<8;i++)
        {
                WriteData(SystemCommand[i]);    // System set commands
        }

        WriteCommand(Scroll);                   // Scrolling, addressing
        for(i=0;i<6;i++)
        {
                WriteData(ScrollCommand[i]);    // Scrolling commands
        }

        WriteCommand(HorizontalScroll);
```

```
        WriteData(0x00);                        // No scrolling

        WriteCommand(Overlay);                  // Overlay
        WriteData(0x01);                        // Set to 2-layer text/graphics,
 XOR composition
        WriteCommand(DisplayOff);
        WriteData(0x14);                        // No cursor
        ClearLCD();                             // Clear the LCD (blank characte
r and graphic spaces)

//      MoveCursor(0x0000);                     // Move to lcd init
        WriteCommand(CursorForm);               // Choose cursor form
        WriteData(0x04);                        // Cursor horizontal
        WriteData(0x86);                        // Cursor vertical, block vertic
al

        WriteCommand(DisplayOn);                // Turn on the display
#if 0
//      lineNum=0;
//      sprintf(lcd_buffer, "X");
//      LCDPrintString(lcd_buffer);
/*      for(i=0;i<30;i++) {
                MoveCursor((unsigned int)i*40);
                sprintf(lcd_buffer, "%d", (unsigned int)i*40);
                LCDPrintString(lcd_buffer);
        }       */
        for(i=0;i<60;i++) {
//              sprintf(lcd_buffer, "ABCDEFGHIJKLMNOPQRSTUVWXYZABCDEFGHIJKLMNOPQ
RSTUVWXYZ");
                sprintf(lcd_buffer, "%d", i);
                LCDPrintString(lcd_buffer);
                delay_ms(100);
                LCDNewLine();
        }
#endif
}

void ClearLCD(void)                             // LCD clearing function
{
        unsigned int i;

        WriteCommand(CursorRight);
        MoveCursor(0x0000);
        WriteCommand(MemoryWrite);

        for(i=0;i<1000;i++)
        {
                WriteData(0x20);                // Clear the character space
        }

        MoveCursor(0x03E8);
        WriteCommand(MemoryWrite);

        for(i=0;i<8000;i++)
        {
                WriteData(0x00);                // Clear graphics space
        }
        colNum=0;lineNum=0;
        MoveCursor(0x0000);
}

void LCDPrintString(char * str) {
        char i;
        WriteCommand(MemoryWrite);
     for(i=0;lcd_buffer[i]!='\0';i++){
                WriteData(lcd_buffer[i]);
                colNum++;
        }
        sprintf(lcd_buffer, "*");
```



```
        lcd_buffer[0]='\0';
//      WriteData(lcd_buffer[0]);
        lineNum+=(colNum/40);
        colNum=colNum%40;
}
#if 0
void LCDPrintBuffer(void)                               // Print the LCD
 buffer
{
        unsigned char i = 0;

        WriteCommand(MemoryWrite);
        while(lcd_buffer[i] != 0)
        {
                WriteData(lcd_buffer[i++]);
        }
        delay_ms(100);
}
#endif
void LCDBackspace() {
        char rt,ct;
        if(colNum>0) colNum--;
        else {
                colNum=39;
                if(lineNum==0) lineNum=22;
                else lineNum--;
        }
        rt=lineNum;
        ct=colNum;
        MoveCursor((unsigned int)40*(rt)+ct);
        sprintf(lcd_buffer, " ");
        LCDPrintString(lcd_buffer);
        lineNum=rt;
        colNum=ct;
        MoveCursor((unsigned int)40*(rt)+ct);
 }

void LCDNewLine() {
        unsigned int l;
        char i;
        delay_ms(100);
        if(lineNum<22) lineNum++;
        else lineNum=0;
        l=(unsigned int)40*(lineNum);
        colNum=0;
        MoveCursor(l);
        WriteCommand(MemoryWrite);
    for(i=0;i<80;i++){
                WriteData(0x20);
        }
        MoveCursor(l);
}

#else
#include <stdio.h>
void LCDPrintString(char * str) {
        printf(str);
}
#endif
```