
# Design Assignment 5

CPE 301 Fall 2016

Luis Ruiz

## PART 0

The assignment took me about 3 hours to do. I had to read the datasheet of the LCD and debug a few times.

## 1. PART A: Description

My design consists of a C program that receives a pin number from a Keypad and outputs a string to the user using the LCD depending on pin entered. The keypad being used is a 4x4 keypad with 8 ports, 4 which are inputs and 4 outputs to be read from the avr. The code determines the pin being pressed, whenever a column read in is low, this character is then stored into a buffer; a message will be displayed until 3 digits are received. At this point a buffer holding the digits will be read and will determine if the system to be unlocked or locked. If locked the user will be asked to enter the pin again, if unlocked the user will receive a message saying “Access Granted”.

## PART B: Code

**LCD & KEYPAD:**

**lcd.h**

```
/*
 * lcd.h
*/

#ifndef LDC_H
#define LDC_H

#define F_CPU 16000000L
#include <avr/io.h>
#include <util/delay.h>

// Connect LCD data pins to PORTB and control pins to PORTC
// RS = PC.0
// RW = PC.1
// EN = PC.2
#define LCD_CPRT PORTC //LCD COMMANDS PORT
#define LCD_CDDR DDRC  //LCD COMMANDS DDR
#define LCD_CPIN PINC  //LCD COMMANDS PIN
```

```
#define LCD_RS  0      //LCD RS   (PC.0)
#define LCD_RW  1      //LCD RW   (PC.1)
#define LCD_EN  2      //LCD EN   (PC.2)

//***************************************************************
//Function Declarations

/*
 * Send a command to the LCD, make pin RS = 1 and R/W = 0;
 * Send a H-to-L pulse to the E pin to enable the internal
 * latch. Place command in command Reg (output AVR: PortB.0...7)
*/
void lcdCommand (unsigned char cmd){
       sendData(cmd);                          //send cmd to data port
       LCD_CPRT &= ~(1<<LCD_RS);   //RS = 0 for command
       LCD_CPRT &= ~(1<<LCD_RW);   //RW = 0 for write
       LCD_CPRT |= (1<<LCD_EN);    //EN = 1 for H-to-L pulse
       _delay_us(1);                           //wait to make enable wide
       LCD_CPRT &= ~(1<<LCD_EN);   //EN = 0 for H-to_L pulse
       _delay_us(100);                         //wait to make enable wide
}

/*
 * Send data to the LCD, make pin RS = 0 and R/W = 0;
 * Send a H-to-L pulse to the E pin to enable the internal
 * latch. Place data in Data Reg (output AVR: PortB.0...7)
*/
void lcdData(unsigned char data){
              sendData(data);                                  //send data to data port
              LCD_CPRT |= (1<<LCD_RS);          //RS = 1 for data
              LCD_CPRT &= ~(1<<LCD_RW);   //RW = 0 for write
              LCD_CPRT |= (1<<LCD_EN);          //EN = 1 for H-to-L pulse
              _delay_us(1);                             //wait to make enable wide
              LCD_CPRT &= ~(1<<LCD_EN);   //EN = 0 for H-to_L pulse
              _delay_us(100);                           //wait to make enable wide
}

/*
 * Initiate the LCD
*/
void lcd_init()
{
       DDRB = 0xFF;
       LCD_CDDR = 0xFF;
       LCD_CPRT &=~(1<<LCD_EN);   //LCD_EN = 0
       _delay_us(2000);
       //wait for init
       lcdCommand(0x38);   //initialize LCD 2 line, 5x7
       lcdCommand(0x0E);   //display on, cursor on
       _delay_us(2000);            //wait
       lcdCommand(0x06);   //shift cursor right
}
/*
 * Print on to the LCD the 8-bit char value
 * lcdData is used to send the data
*/
void lcd_print(char * str){
       unsigned char i = 0;
```

```
	while (str[i]!=0)  {
		lcdData(str[i]); i++;
	}
}

/*
 * go to specific LCD locations
*/
void lcd_gotoxy(unsigned char x, unsigned char y){
	unsigned char firstCharAdr[] =  {0x80, 0xC0, 0x94, 0xD4};// locations of the first
character of each line

	lcdCommand(firstCharAdr[y-1] + x-1);
	_delay_us(100);
}

/*
 * Clear the LCD and reset the cursor to the home position
*/
void lcd_reset(){
	lcdCommand(0x01);   //clear LCD
	lcdCommand(0x02);   // return home: returns the cursor to the home position
	_delay_ms(500);
}

//send data using portb and portc
void sendData(unsigned char str){

	PORTB = 0X0;
	PORTC &= ~(1<<PC4); PORTC &= ~(1<<PC5);

	if((str & 0x01) == 0)
		PORTB|= (0<<PB0);
	else
		PORTB|= (1<<PB0);

	if((str & 0x02) == 0)
		PORTB|= (0<<PB1);
	else
		PORTB|= (1<<PB1);

	if((str & 0x04) == 0)
		PORTB|= (0<<PB2);
	else
		PORTB|= (1<<PB2);

	if((str & 0x08) == 0)
		PORTB|= (0<<PB3);
	else
		PORTB|= (1<<PB3);

	if((str & 0x10) == 0)
		PORTB|= (0<<PB4);
	else
		PORTB|= (1<<PB4);

	if((str & 0x20) == 0)
```

```
		PORTB|= (0<<PB5);
	else
		PORTB|= (1<<PB5);

	if((str & 0x40) == 0)
		PORTC|= (0<<PC4);
	else
		PORTC|= (1<<PD4);

	if((str & 0x80) == 0)
		PORTC|= (0<<PC5);
	else
		PORTC|= (1<<PD5);
}

//Test the LCD Functionality
int lcd_test(){
	lcd_init();
	lcd_print("Demo code");		// print some sample code on LCD
	_delay_ms(1000);

	// clear the LCD and print some more text
	lcdCommand(0x01);	//clear LCD
	lcdCommand(0x02);	// return home: returns the cursor to the home position
	lcd_print("Here is some more text that overruns a single line...");		//
print text that overruns a single line
	_delay_ms(1000);

	// print text at a specific location on LCD (1st line, then 2nd line)
	lcdCommand(0x01);	//clear LCD
	lcdCommand(0x02);	// return home: returns the cursor to the home position
	_delay_ms(1000);
	lcd_gotoxy(1,1);
	lcd_print("Text for line 1.");
	lcd_gotoxy(1,2);
	lcd_print("Text for line 2.");
	while(1);			//stay here forever
	return 0;
}

#endif
```

**Main**

```
/*
 * DA5.c
 *
 * Created: 3/1/2017 10:19:00 PM
 * Author : Luis
 */

#include "lcd.h"
#include <avr/io.h>
```

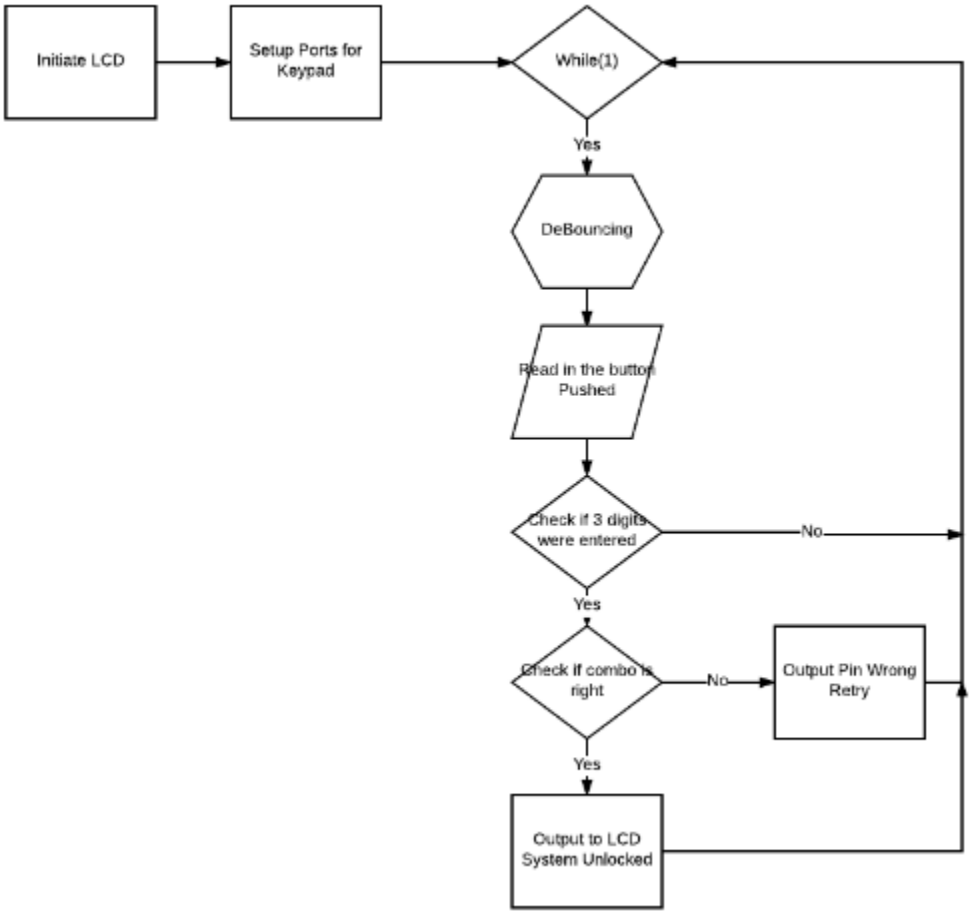
```
//KEYPAD PINS
#define KEYPORT PORTD
#define KEYDDR DDRD
#define KEYPIN PIND

//SET OUTUPT signals

/* Had to map it out differently based on the
// the data sheet

//How it looks on the keypad interface
{
	{'1','2','3','A'},
	{'4','5','6','B'},
	{'7','8','9','C'},
	{'*','0','#','D'}
}
*/

unsigned char keypad[4][4] ={
                                                {'1','4','7','*'},
                                                {'2','5','8','0'},
                                                {'3','6','9','#'},
                                                {'A','B','C','D'}
                                                };

int main(void)
{
		lcd_init();
		lcd_reset();
		lcd_print("Input Pin");

		unsigned char col,row = 0;
		unsigned char count = 0;
		unsigned char pin[4] = {0};

		while(1)
		{
			start:
				if(count == 0)
				{

					lcd_init();
					lcd_reset();
					lcd_print("Input Pin");
					lcd_gotoxy(1,2);
					col = row = 0;
					pin[4] = pin[3] = pin[2]= pin[1]= 0;
				}

				//Establish the Inputs and outputs
				//to the keypad
				KEYDDR =  0xF0;
				KEYPORT = 0xFF;


			/*
			 * Go in at least once,
```

```
                     * Then PORT7...4 are set low and wait until the pad pressed
                     * released.

                    */
                    do
                    {
                           KEYPORT &= 0x0F;
                           col = (KEYPIN & 0x0F);

                    }while(col != 0x0F);


                    /*
                     * De-bouncing
                    */
                     do
                     {
                            do
                            {
                                   _delay_ms(20);                                  //call
delay
                                   col = (KEYPIN & 0x0F);                  //see if any
key is pressed
                            } while (col == 0x0F);                         //keep
checking for key press

                            _delay_ms(20);                                         //call
delay for de-bounce
                            col = (KEYPIN & 0x0F);                         //read
columns
                     } while (col == 0x0F);                                //wait for
key press

                    while(1)
                    {
                            KEYPORT = 0xEF;             //ground row 0
                            _delay_ms(20);
                            col = (KEYPIN & 0x0F);      //read the columns
                            if (col != 0x0F)            //column detected
                            {
                                   row = 0;                    //save row location
                                   break;                             //exit while loop
                            }
                            KEYPORT = 0xDF;             //ground row 1
                            _delay_ms(20);
                            col = (KEYPIN & 0x0F);  //read the columns
                            if(col != 0x0F)             //column detected
                            {
                                   row = 1;                    //save row location
                                   break;                             //exit while loop
                            }
                            KEYPORT = 0xBF;            //ground row 2
                            _delay_ms(20);
                            col = (KEYPIN & 0x0F); //read the columns
                            if(col != 0x0F)            //column detected
                            {
                                   row = 2;            //save row location
                                   break;                     //exit while loop
```

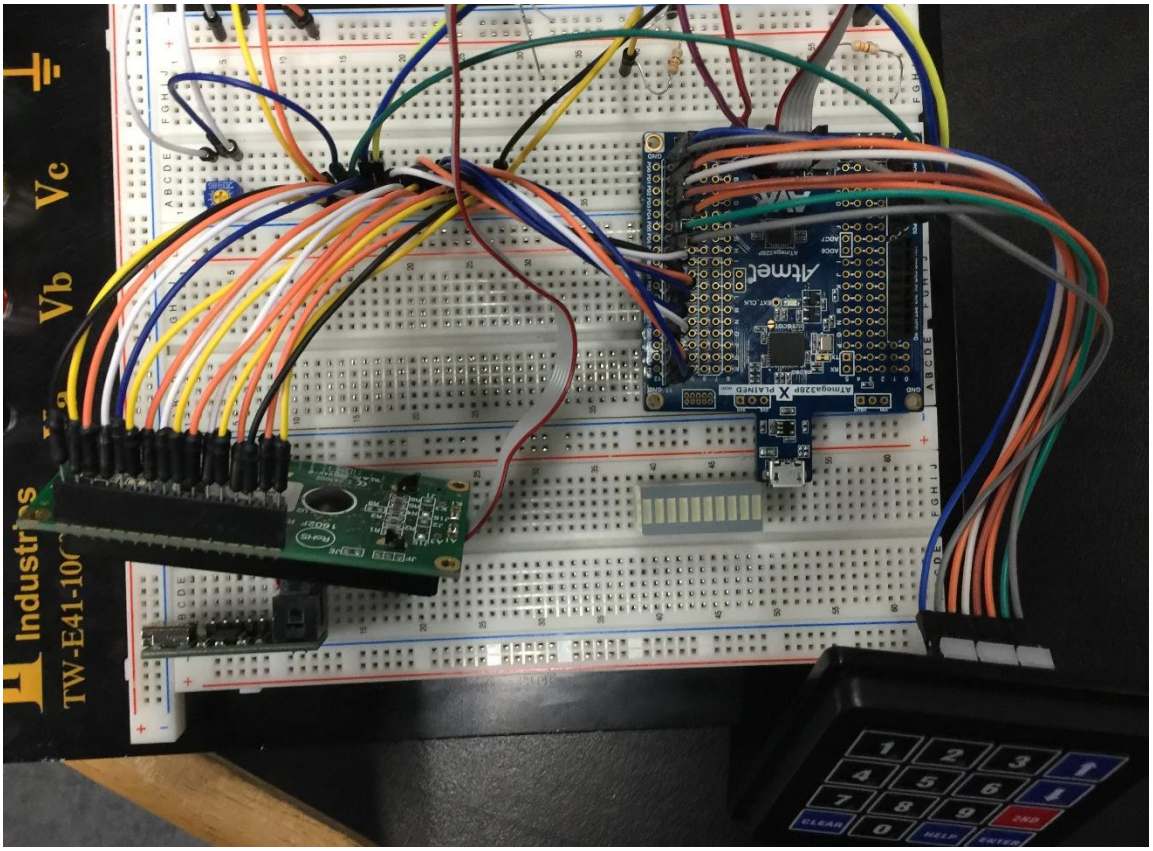
```
        }

        KEYPORT = 0x7F;            //ground row 3
        _delay_ms(20);
        col = (KEYPIN & 0x0F); //read the columns
       if(col!= 0x0F)
        {
              row = 3;                          //save row location
              break;                            //exit while loop
       }

}

if(count >= 3)
{
       for(int i = 0; i < 3; i = i +1)
       {
              switch(i)
              {
               case 0: if(pin[i] != '5'){
                                  lcdData(pin);
                                  _delay_ms(1000);
                                  printError(pin);
                                  count = 0;
                                  goto start;
                           }
                            break;
               case 1:if(pin[i] != '2'){
                                   lcdData(pin);
                                   _delay_ms(1000);
                                  printError(pin);
                                  count = 0;
                                  goto start;
                           }
                           break;

               case 2:if(pin[i] != '7'){
                                  lcdData(pin);
                                  _delay_ms(1000);
                                  printError(pin);
                                  count = 0;
                                  goto start;
                           }
                           else
                                  Correct(pin);
                           break;
              default:count = 0;goto start;break;

              }
       }

       count = 0;
       goto start;
}


 //check column and send result to Port D
 if(col == 0x0E)
```

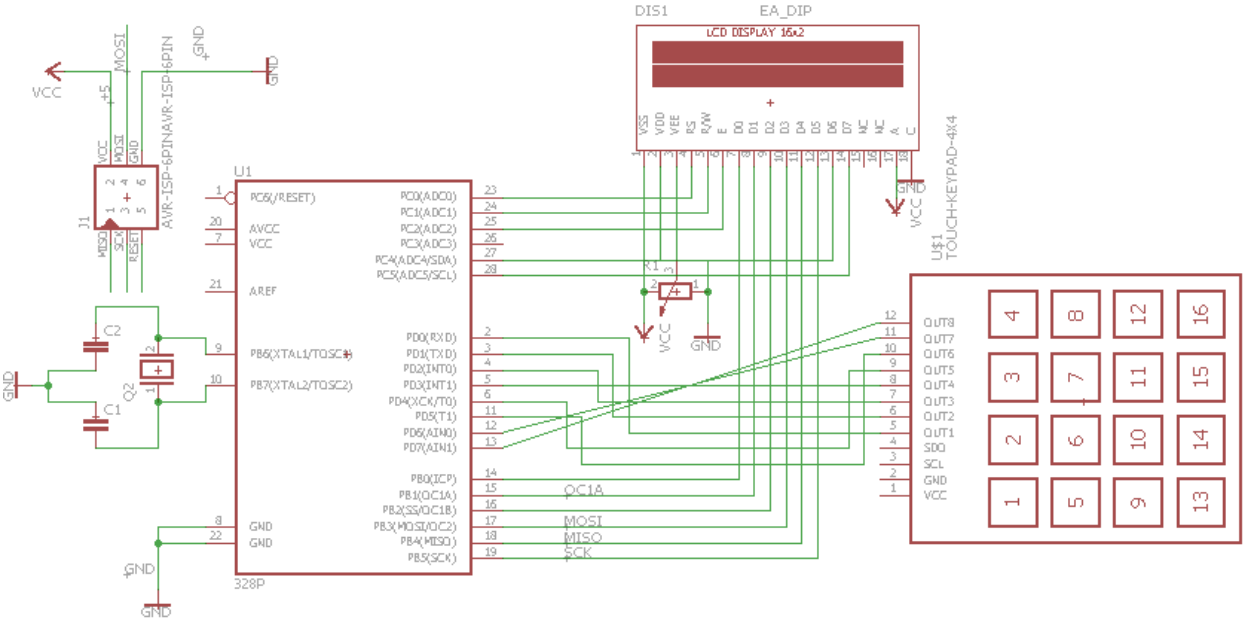
```
                        pin[count] = (keypad[row][0]);
                    else if(col == 0x0D)
                        pin[count] = (keypad[row][1]);
                    else if(col == 0x0B)
                        pin[count] = (keypad[row][2]);
                    else
                        pin[count] = (keypad[row][3]);

                    if(count < 3)
                    {
                        lcdData(pin[count]);
                        _delay_ms(500);
                    }
                    ++count;
                }
                return 0;
}

void printError(char *str)
{
        lcd_init();
        lcd_reset();
        lcd_print(str);
        lcd_gotoxy(1,2);
        lcd_print("Wrong Pin!!");
        _delay_ms(2500);
}

void Correct(char *str){

        lcd_init();
        lcd_reset();
        lcd_print(str);
        lcd_gotoxy(1,2);
        //“System unlocked”
        lcd_print("Access Granted!!");
        //_delay_ms(2500);
        while(1);
}
```

## PART C:Flow Chart



## PART D: Schematic





## PART E: Video

URL Video of Design Assignment 5: https://youtu.be/XtZO00H5-B0