Class:	CPE301L Digital Systems Architecture and Design 1001		Semester:	Spring 2025
Points		Document author:	Narek Kalikian	
		Author's email:	kalikn1@unlv.nevada.edu	
		Document topic:	Postlab 8	
Instructor's comments:				

1. Introduction / Theory of Operation

The objective of this lab is to implement a Liquid Crystal Display (LCD) and keypad for interfacing with our ATmega328p microcontroller. LCDs are good to use when more specific information is required. Unlike LEDs, LCDs are far more complex because they do not emit light directly but instead manipulate the external light that is reflected off them. The important components of an LCD are DDRAM, CGRAM, cursor, and command register. The keypad, on the other hand, essentially functions as a matrix. We will be learning about these different components and using them to successfully display information on the LCD during the experiments.

2. Prelab Content

LCD: A liquid crystal display (LCD) is a flat panel display using liquid crystals that can replace light-emitting diodes (LEDs). LCDs are beneficial because they are much thinner and consume less power because they block out light instead of emitting it. LCDs light up using a backlight and their pixels are switched either on or off by using the liquid crystals to rotate polarized light. LCDs use a glass filter in front and behind all pixels.LCDs also use a matrix grid that has a grid of conductors at each matrix intersection. A current is sent across two conductors on the grid to control the light of any given pixel.

Keypad: The keypad is a 4x4 matrix with eight terminals (pins). Four terminals are rows of the matrix while the other four are columns of the matrix. The eight pins are driven by sixteen corresponding alphanumeric buttons. The keypad, across each button, has a maximum voltage of 24 volts and a maximum current of 30 milliamps. The rows of the keypad matrix will be used as outputs set at five volts and the columns will be used as inputs to sense HIGH logic. Then, the roles of the rows and columns are reversed. These types of keypads are commonly used in vending machines, security systems, industrial machines, and much more.

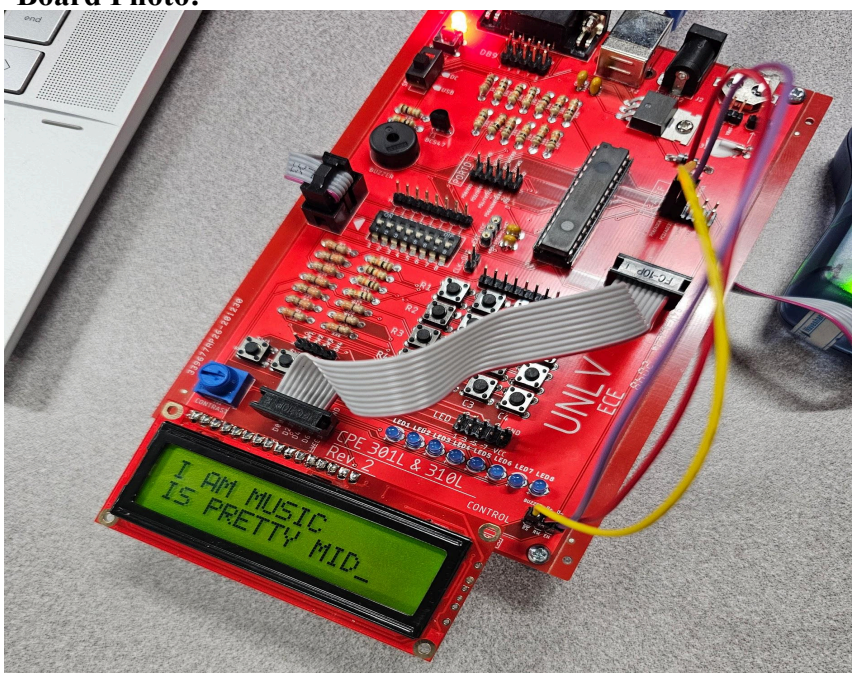
3. Description of Experiments

Experiment 1 - Code:

```
#define F_CPU 1000000UL
#include<avr/io.h>
#include<util/delay.h>
//definition for the wire connections
#define rs PC0
#define rw PC1
#define en PC2
#define data PORTD

//functions necessary
void lcd_init(); //to initialize LCD.
void lcd_cmd(char cmd_out); //to send command to LCD.
void lcd_data(char data_out); //to send data to LCD.
void lcd_str(char *str); //to send string, basically stripping each character and sending.
// Quick delay function for 10ms
void delay() {
    _delay_ms(10);
}

int main()
{
    //set DDR
    DDRD = 0xff; // Set all Port B to output
    DDRC |= (1 << rs) | (1 << rw) | (1 << en);
    lcd_init();
    lcd_str("I AM MUSIC");
    lcd_cmd(0xC0); // Command to go to next line
    lcd_str("IS PRETTY MID");
    while(1)
    {
    }
}

void lcd_init()
{
    //PORTC &= ~(1 << en);
    lcd_cmd(0x38); // Initializing to 2 lines & 5x7 font.
    delay();
    lcd_cmd(0x0E); // Display on, cursor on
    delay();
    lcd_cmd(0x01); // Clear LCD
    delay();
}

void lcd_cmd(char cmd_out)
{
    data = cmd_out; //send the cmd_out to data
    delay();
    PORTC &= ~(1 << rs) & ~(1 << rw); //set rs = 0 ,rw = 0
    PORTC |= (1 << en); // and en = 1
    delay(); //wait for small delay 10ms
    PORTC &= ~(1 << en); //set rs = 0 ,rw=0 and en =0
    delay(); //wait for small delay 10ms
}

void lcd_data(char data_out)
{    data = data_out; //send the data_out to data
    delay(); //wait for small delay 10ms
    PORTC &= ~(1 << rw); //set rs = 1, rw = 0 and en = 1
    PORTC |= (1 << rs) | (1 << en);
    delay(); //wait for small delay 10ms
    PORTC &= ~(1 << en);
    delay(); //wait for small delay 10ms
}

void lcd_str(char *str)
{
    unsigned int i=0;
    while(str[i]!='\0')
    {
        lcd_data(str[i]);
        i++;
    }
}
```

```
Output
Show output from: Build
Target "PostBuildEvent" skipped, due to false condition; ('$(PostBuildEvent)' != '') was evaluated as ('' != '').
Target "Build" in file "C:\Program Files (x86)\Atmel\Studio\7.0\Vs\Avr.common.targets" from project "C:\Users\Narek\Documents\Atmel Studio\7.0\Lab8\Lab8\Lab8.cproj" (entry point):
Done building target "Build" in project "Lab8.cproj".
Done building project "Lab8.cproj".

Build succeeded.
========== Build: 1 succeeded or up-to-date, 0 failed, 0 skipped ==========
```

Experiment 1 - Board Photo:



Experiment 2 - Code:

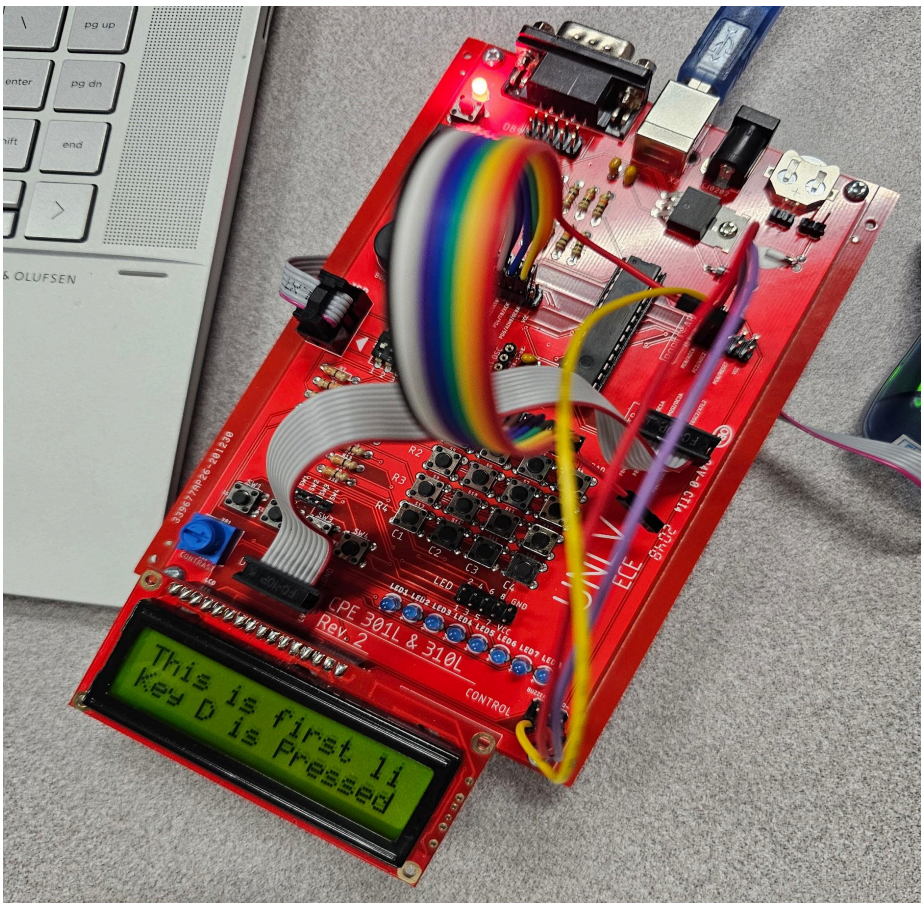
```c
#define F_CPU 1000000UL
#include<avr/io.h>
#include<util/delay.h>
//definition for the wire connections
#define rs PC0
#define rw PC1
#define en PC2
#define data PORTB

//functions necessary
void lcd_init(); //to initialize LCD.
void lcd_cmd(char cmd_out); //to send command to LCD.
void lcd_data(char data_out); //to send data to LCD.
void lcd_str(char *str); //to send string, basically stripping each character and sending.

// Keypad Functions
void checkCol1();
void checkCol2();
void checkCol3();
void checkCol4();
void keypadCheck();

// Quick delay function for 10ms
void delay() {
    _delay_ms(25);
}


int main()
{
    //set DDR
    DDRB = 0xff; // Set all Port B to output
    DDRC |= (1 << rs) | (1 << rw) | (1 << en);
    // DDRD = Row:0000 Col:1111
    DDRD = 0x0f; // Make all 4 rows as outputs and all 4 cols as inputs.
    PORTD = 0xf0; // Initializes Port D with pull-ups
    lcd_init();
    lcd_str("This is first line");
    lcd_cmd(0xC0); // Command to go to next line
    lcd_str("Key _ is Pressed");
    delay();
    // Checks for keypad inputs when not default
    while(1)
    {
        delay();
        keypadCheck();
    }
```

```
void lcd_init()
{
    lcd_cmd(0x38); // Initializing to 2 lines & 5x7 font.
    delay();
    lcd_cmd(0x0C); // Display on, cursor off
    delay();
    lcd_cmd(0x01); // Clear LCD
    delay();
    lcd_cmd(0x80); // Set cursor position to top row 0x80
    delay();
}

void lcd_cmd(char cmd_out)
{
    data = cmd_out; //send the cmd_out to data
    delay();
    PORTC &= ~(1 << rs) & ~(1 << rw); //set rs = 0 ,rw = 0
    PORTC |= (1 << en); // and en = 1
    delay(); //wait for small delay 10ms
    PORTC &= ~(1 << en); //set rs = 0 ,rw=0 and en =0
    delay(); //wait for small delay 10ms
}

 void lcd_data(char data_out)
{   data = data_out; //send the data_out to data
    delay(); //wait for small delay 10ms
    PORTC &= ~(1 << rw); //set rs = 1, rw = 0 and en = 1
    PORTC |= (1 << rs) | (1 << en);
    delay(); //wait for small delay 10ms
    PORTC &= ~(1 << en);
    delay(); //wait for small delay 10ms
}

void lcd_str(char *str)
{
    unsigned int i=0;
    while(str[i]!='\0')
    {
        lcd_data(str[i]);
        i++;
    }
}

void checkCol1()
{
    PORTD = 0b00000001; //Checking buttons 1,2,3,and D
    if((PIND & 0b11110000) == 0b00010000)
    lcd_data('A');
    else if((PIND & 0b11110000) == 0b00100000)
    lcd_data('4');
    else if((PIND & 0b11110000) == 0b01000000)
    lcd_data('7');
    else if((PIND & 0b11110000) == 0b10000000)
    lcd_data('*');
    else
    lcd_cmd(0xC4); // Setting cursor in bottom left
}

void checkCol2()
{
    PORTD = 0b00000010; // Checking buttons 4,5,6, and A
    if((PIND & 0b11110000) == 0b00010000)
    lcd_data('1');
    else if((PIND & 0b11110000) == 0b00100000)
    lcd_data('5');
    else if((PIND & 0b11110000) == 0b01000000)
    lcd_data('8');
    else if((PIND & 0b11110000) == 0b10000000)
    lcd_data('0');
    else
    lcd_cmd(0xC4);
}

void checkCol3()
{
    PORTD = 0b00000100; // Checking buttons 7,8,9, and B
    if((PIND & 0b11110000) == 0b00010000)
    lcd_data('2');
    else if((PIND & 0b11110000) == 0b00100000)
    lcd_data('6');
    else if((PIND & 0b11110000) == 0b01000000)
    lcd_data('9');
    else if((PIND & 0b11110000) == 0b10000000)
    lcd_data('#');
    else
    lcd_cmd(0xC4);
}
```

```
void checkCol4()
{
    PORTD = 0b00001000; // Checking buttons *,0,#, and C
    if((PIND & 0b11110000) == 0b00010000)
    lcd_data('3');
    else if((PIND & 0b11110000) == 0b00100000)
    lcd_data('B');
    else if((PIND & 0b11110000) == 0b01000000)
    lcd_data('C');
    else if((PIND & 0b11110000) == 0b10000000)
    lcd_data('D');
    else
    lcd_cmd(0xC4);
}

void keypadCheck() {
    checkCol1();
    delay();
    checkCol2();
    delay();
    checkCol3();
    delay();
    checkCol4();
    delay();
}
```

```
Output
Show output from: Build
Target "Build" in file "C:\Program Files (x86)\Atmel\Studio\7.0\Vs\Avr.common.targets" from project "C:\Users\Narek\Documents\Atmel Studio\7.0\Lab8\Lab8\Lab8.cproj" (entry point):
Done building target "Build" in project "Lab8.cproj".
Done building project "Lab8.cproj".

Build succeeded.
========== Build: 1 succeeded or up-to-date, 0 failed, 0 skipped ==========
```

Experiment 2 - Board Photo:



Experiment 3 - Code:

```
#define F_CPU 1000000UL
#include<avr/io.h>
#include<util/delay.h>
//definition for the wire connections
#define rs PC0
#define rw PC1
#define en PC2
#define data PORTB

//functions necessary
void lcd_init(); //to initialize LCD.
void lcd_cmd(char cmd_out); //to send command to LCD.
void lcd_data(char data_out); //to send data to LCD.
void lcd_str(char *str); //to send string, basically stripping each character and sending.

// Keypad Functions
void checkCol1();
void checkCol2();
void checkCol3();
void checkCol4();
void keypadCheck();

// Quick delay function for 10ms
void delay() {
    _delay_ms(65);
}

// Global Variables
uint8_t inputCount = 0; // Counter for Keypad inputs
char guess[6] = "----?"; // Extra space for Null Char
char password[6] = "5317D"; // new password to check for
int correct = 0; // Flag for Correct Password

int main()
{
    //set DDR
    DDRB = 0xff; // Set all Port B to output
    DDRC |= (1 << rs) | (1 << rw) | (1 << en);
    // DDRD = Row:0000 Col:1111
    DDRD = 0x0f; // Make all 4 rows as outputs and all 4 cols as inputs.
    PORTD = 0xf0; // Initializes Port D with pull-ups
    // Checks for keypad inputs until correct password
    while(!correct) {
        lcd_init();
        lcd_str("Enter Password: ");
        lcd_cmd(0xC0); // Command to go to next line
        lcd_str("____");
        delay();
        while(inputCount < 5) {
            delay();
            keypadCheck();
        }
        if (guess[0] == password[0]) {
            if (guess[1] == password[1]) {
                if (guess[2] == password[2]) {
                    if (guess[3] == password[3]) {
                        if (guess[4] == password[4]) {
                            if (guess[5] == password[5]) {
                                correct = 1;
                            }
                        }
                    }
                }
            }
        }
        else {
            lcd_init();
            lcd_str("Wrong Sequence");
            inputCount = 0; // resets counter
            _delay_ms(3000);
        }

    }
    lcd_init(); // Resets LCD
    lcd_str("Lock Opened!");
}
```

```
void lcd_init()
{
    lcd_cmd(0x38); // Initializing to 2 lines & 5x7 font.
    delay();
    lcd_cmd(0x0C); // Display on, cursor off
    delay();
    lcd_cmd(0x01); // Clear LCD
    delay();
    lcd_cmd(0x80); // Set cursor position to top row 0x80
    delay();
}

void lcd_cmd(char cmd_out)
{
    data = cmd_out; //send the cmd_out to data
    delay();
    PORTC &= ~(1 << rs) & ~(1 << rw); //set rs = 0 ,rw = 0
    PORTC |= (1 << en); // and en = 1
    delay(); //wait for small delay 10ms
    PORTC &= ~(1 << en); //set rs = 0 ,rw=0 and en =0
    delay(); //wait for small delay 10ms
}

 void lcd_data(char data_out)
{   data = data_out; //send the data_out to data
    delay(); //wait for small delay 10ms
    PORTC &= ~(1 << rw); //set rs = 1, rw = 0 and en = 1
    PORTC |= (1 << rs) | (1 << en);
    delay(); //wait for small delay 10ms
    PORTC &= ~(1 << en);
    delay(); //wait for small delay 10ms
}

void lcd_str(char *str)
{
    unsigned int i=0;
    while(str[i]!='\0')
    {
        lcd_data(str[i]);
        i++;
    }
}

void checkCol1() {
    PORTD = 0b00000001; //Checking buttons 1,2,3,and D
    if((PIND & 0b11110000) == 0b00010000) {
        lcd_data('A');
        guess[inputCount] = 'A';
        inputCount++;
        lcd_cmd(14); // Shifts Cursor right by 1
    }
    else if((PIND & 0b11110000) == 0b00100000) {
        lcd_data('4');
        guess[inputCount] = '4';
        inputCount++;
        lcd_cmd(14); // Shifts Cursor right by 1
    }
    else if((PIND & 0b11110000) == 0b01000000) {
        lcd_data('7');
        guess[inputCount] = '7';
        inputCount++;
        lcd_cmd(14); // Shifts Cursor right by 1
    }
    else if((PIND & 0b11110000) == 0b10000000) {
        lcd_data('*');
        guess[inputCount] = '*';
        inputCount++;
        lcd_cmd(14); // Shifts Cursor right by 1
    }
    else {
        lcd_cmd(0xC0 + inputCount); // Setting cursor to last input
    }
}

void checkCol2() {
    PORTD = 0b00000010; // Checking buttons 4,5,6, and A
    if((PIND & 0b11110000) == 0b00010000) {
        lcd_data('1');
        guess[inputCount] = '1';
        inputCount++;
        lcd_cmd(14); // Shifts Cursor right by 1
    }
    else if((PIND & 0b11110000) == 0b00100000) {
        lcd_data('5');
        guess[inputCount] = '5';
        inputCount++;
        lcd_cmd(14); // Shifts Cursor right by 1
    }
    else if((PIND & 0b11110000) == 0b01000000) {
        lcd_data('8');
        guess[inputCount] = '8';
        inputCount++;
        lcd_cmd(14); // Shifts Cursor right by 1
    }
```

```
    else if((PIND & 0b11110000) == 0b10000000) {
        lcd_data('0');
        guess[inputCount] = '0';
        inputCount++;
        lcd_cmd(14); // Shifts Cursor right by 1
    }
    else {
        lcd_cmd(0xC0 + inputCount); // Setting cursor to last input
    }
}

void checkCol3() {
    PORTD = 0b00000100; // Checking buttons 7,8,9, and B
    if((PIND & 0b11110000) == 0b00010000) {
        lcd_data('2');
        guess[inputCount] = '2';
        inputCount++;
        lcd_cmd(14); // Shifts Cursor right by 1
    }
    else if((PIND & 0b11110000) == 0b00100000) {
        lcd_data('6');
        guess[inputCount] = '6';
        inputCount++;
        lcd_cmd(14); // Shifts Cursor right by 1
    }
    else if((PIND & 0b11110000) == 0b01000000) {
        lcd_data('9');
        guess[inputCount] = '9';
        inputCount++;
        lcd_cmd(14); // Shifts Cursor right by 1
    }
    else if((PIND & 0b11110000) == 0b10000000) {
        lcd_data('#');
        guess[inputCount] = '#';
        inputCount++;
        lcd_cmd(14); // Shifts Cursor right by 1
    }
    else {
        lcd_cmd(0xC0 + inputCount); // Setting cursor to last input
    }
}

void checkCol4() {
    PORTD = 0b00001000; // Checking buttons *,0,#, and C
    if((PIND & 0b11110000) == 0b00010000) {
        lcd_data('3');
        guess[inputCount] = '3';
```

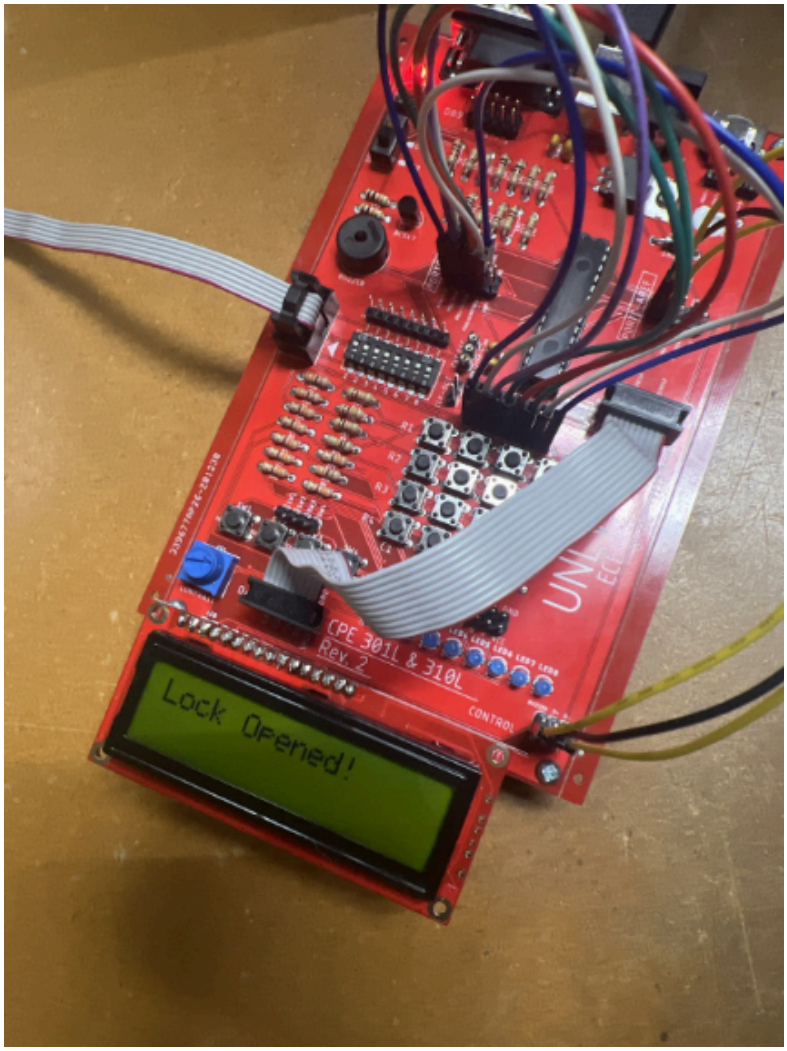
```
        inputCount++;
        lcd_cmd(14); // Shifts Cursor right by 1
    }
    else if((PIND & 0b11110000) == 0b00100000) {
        lcd_data('B');
        guess[inputCount] = 'B';
        inputCount++;
        lcd_cmd(14); // Shifts Cursor right by 1
    }
    else if((PIND & 0b11110000) == 0b01000000) {
        lcd_data('C');
        guess[inputCount] = 'C';
        inputCount++;
        lcd_cmd(14); // Shifts Cursor right by 1
    }
    else if((PIND & 0b11110000) == 0b10000000) {
        lcd_data('D');
        guess[inputCount] = 'D';
        inputCount++;
        lcd_cmd(14); // Shifts Cursor right by 1
    }
    else {
        lcd_cmd(0xC0 + inputCount); // Setting cursor to last input
    }
}

void keypadCheck() {
    checkCol1();
    delay();
    checkCol2();
    delay();
    checkCol3();
    delay();
    checkCol4();
    delay();
}
```

Output

Show output from: Build

```
    Done executing task "RunOutputFileVerifyTask".
Done building target "CoreBuild" in project "Lab8.cproj".
Target "PostBuildEvent" skipped, due to false condition; ('$(PostBuildEvent)' != '') was evaluated as ('' != '').
Target "Build" in file "C:\Program Files (x86)\Atmel\Studio\7.0\Vs\Avr.common.targets" from project "C:\Users\Narek\Documents\Atmel Studio\7.0\Lab8\Lab8\Lab8.cproj" (entry point):
Done building target "Build" in project "Lab8.cproj".
Done building project "Lab8.cproj".

Build succeeded.
========== Build: 1 succeeded or up-to-date, 0 failed, 0 skipped ==========
```

Experiment 3 - Board Photo:



4. Questions and Answers

1. LCD Wiring: The LCD is wired to the ATmega328p microcontroller by connecting female-female cables to control and data pins. RS, RW, and EN control pins are connected to Port C pins PC0, PC1, and PC2. D0-D7 data pins are connected to PORTB.

2. Keypad Internal Organization: The keypad component is internally organized with a matrix. The connections between the rows and columns of the matrix are used to detect the pressed key. Low signals are sent through the rows, and columns are read to determine what the active connecting point is between the matrix's rows and columns.

3. 16x2 vs 40x2 Displays: The names of the two different display types describe how each works. A 16x2 display allows 2 lines and up to 16 characters per line for displaying messages. A 40x2 display allows 2 lines and up to 40 characters per line for displaying messages. Each display also has a different address range for rows and columns in the keypad matrix.

5. Conclusions

During this lab, we learned about LCDs and keypads and how to integrate them with our ATmega328p microcontroller. We learned how each component works based on their datasheets, and from this information, we were able to code functionality for each in Microchip Studio. The LCD was connected to our ATmega328p board, and coded messages were sent from the microcontroller and displayed on the LCD. The keypad was also connected to the microcontroller, and then key presses from the keypad were detected and also displayed on the LCD. We didn't run into any real problems during this lab. However, understanding and using the keypad was initially a bit complicated.