

**Computer Systems Engineering Technology**

**CST 120 – Embedded C Programming**

|  |  |
| --- | --- |
| Lab 07\_LCD Thermometer | Name\_\_Chris Thomas\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Spring 2022 | Due Date: 11:55 PM, May 13 |
| Instructor: George Drouant | |
| Possible Points: 100 | |

# Introduction

In this Lab we will use the LCD in the Kit to display the temperature measured by the thermistor. Using the LCD to display the temperature frees us from the need to have a computer on which to display the data. It allows us to create handheld, battery operated instruments.

|  |
| --- |
| **PARTS LIST** |
| UNO |
| Protoboard |
| LCD Module |
| wires |
| R1 – 10K Pot |
| R2 -10K |
| R3 -10K Thermistor |
|  |
| Battery with Snap On Connector |

# LCD – Liquid Crystal Display

The picture below shows the LCD that is supplied in the Kit. Note that the LCD’s pin 1 and pin 16 are indicated by the pcb’s stencil. The function of each pin is also indicated on the pcb next to the pin. This LCD has an LED backlight which must be on for the LCD characters to display correctly.

Build the circuit per the schematic and reference material.

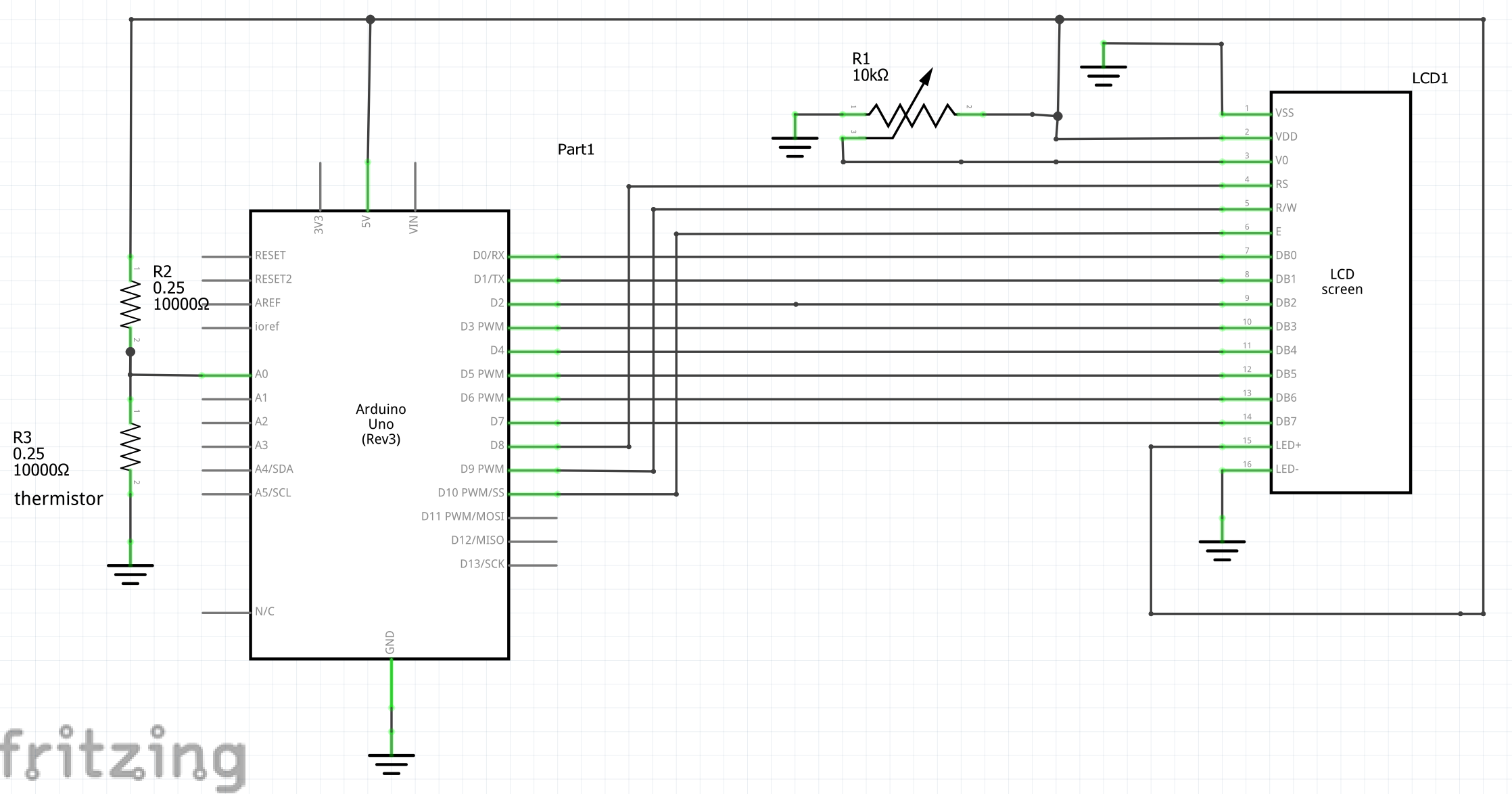
R1 is used to set the contrast for the LCD. R1 is a potentiometer (three terminal variable resistor know as a “pot”).The pots in our kits are a bit flakey so be careful using them – they don’t plug into the protoboards that well. If the LCD screen is not clear after power up try adjusting the pot until you can see characters on the LCD’s screen.

You will use the same circuit for both test programs and the program you will build.

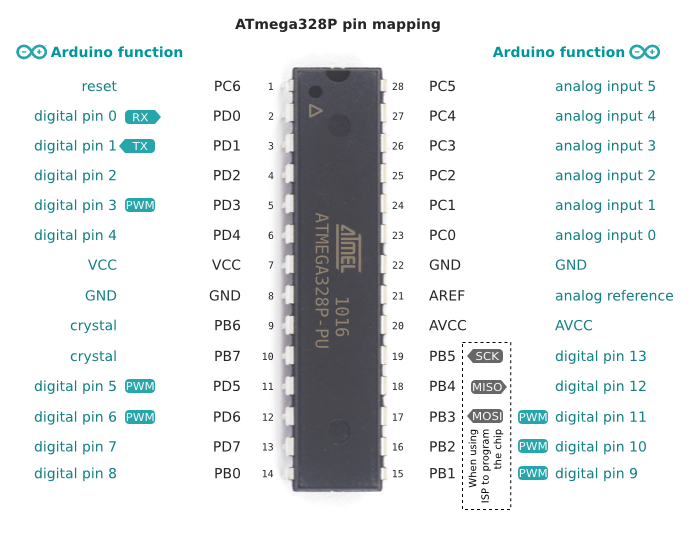


The LCD pin functions and circuit schematic follow:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
|  | Pin # | Pin Function |  | Pin # | Pin Function |  |
|  | 1 | Vss > Ground |  | 9 | D2 (DB2) |  |
|  | 2 | Vdd > +5V |  | 10 | D3 (DB3) |  |
|  | 3 | V0 > LCD contrast |  | 11 | D4 (DB4) |  |
|  | 4 | RS > register select |  | 12 | D5 (DB5) |  |
|  | 5 | R/W' > read/write' |  | 13 | D6 (DB6) |  |
|  | 6 | E > enable |  | 14 | D7 (DB7) |  |
|  | 7 | D0 (DB0) |  | 15 | A (LED+ > +5V) |  |
|  | 8 | D1 (DB1) |  | 16 | K (LED- > Ground) |  |
|  |  |  |  |  |  |  |



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | PORTB Pins | Connected to LCD Pins |  | PORTD Pins | Connected to LCD Pins |  |
|  | PB0 | RS |  | PD0 | DB0 |  |
|  | PB1 | R/W' |  | PD1 | DB1 |  |
|  | PB2 | E |  | PD2 | DB2 |  |
|  |  |  |  | PD3 | DB3 |  |
|  |  |  |  | PD4 | DB4 |  |
|  |  |  |  | PD5 | DB5 |  |
|  |  |  |  | PD6 | DB6 |  |
|  |  |  |  | PD7 | DB7 |  |
|  |  |  |  |  |  |  |



TEST Code #1

Copy and paste the code below into Atmel Studio. It is the code we discussed in class. Use it to test your LCD circuit. Download the code to the UNO and press the Reset Button (red button by the USB connector). If the circuit and code are correct the words “Embedded System” should be printed on the top line of the LCD and the word “Laboratory” on the second line. This code shows how to send string constants to the LCD for display.

Caution:

1. Make sure that you disconnect the wires from the UNO’s pins PD0 and PD1 to the LCD’s pins DB0 and DB1 before you download the Test code to the UNO. You must do this while downloading since UNO pins PD0 and PD1 are used by the UNO’s UART to “talk with” the host computer. If they are connected to the LCD’s pins DB0 and DB1 the UART will not function properly. Reconnect the wires after you are finished downloading and before you press the reset button.

/\*

\*

\*

\* Created: 4/25/2020 10:05:04 AM

\* Author : Deyr & Drouantg

\*/

#define *F\_CPU* 16000000UL

#include <avr/io.h>

#include <util/delay.h>

#define LCD\_data\_dir DDRD

#define LCD\_command\_dir DDRB

#define LCD\_data\_port PORTD //PORTD defined as DATA Port -> sends characters to LCD

#define LCD\_command\_port PORTB //PORTB defined as Port used to send Commands to LCD

#define RS PB0 //Port B bit 0 defined as register select bit

#define RW PB1 //Port B bit 1 defined as Read/Write' bit

#define EN PB2 //Port B bit 2 is defined as Enable bit

void LCD\_Init(void)

{

/\* Initialize LCD \*/

LCD\_command\_dir = 0b00000111; //PB0, PB1, PB2 set as output pins

LCD\_data\_dir = 0b11111111; //all bits in Port D defined as output pins

*\_delay\_ms*(45); // wait for internal initialization of LCD

LCD\_command(0x38);//sets 8-bit interface with LCD, 2 Line Display,5x8 dot character font

*\_delay\_us*(40);

LCD\_command(0x0C); //turns Display On with Cursor Off

*\_delay\_us*(40);

LCD\_command(0x06); //sets increment character position by one

*\_delay\_us*(40);

LCD\_command(0x01); //clears entire Display

*\_delay\_us*(40);

LCD\_command(0x80); //Sets Display to start at first character of first Line

*\_delay\_us*(40);

}

void LCD\_command (char cmnd)

{

/\*write a command to LCD \*/

LCD\_data\_port = cmnd; //outputs hex command code to PORTD and the LCD’s data bus

LCD\_command\_port &= ~(1<<RS); //sets RS bit Low -> Instruction Register

LCD\_command\_port &= ~(1<<RW); //sets R/W’ bit Low -> write

LCD\_command\_port |= (1<<EN); //sets EN bit High

*\_delay\_us*(1);

LCD\_command\_port &= ~(1<<EN);//Low going transition of EN bit Low writes data to LCD

*\_delay\_ms*(2); //give LCD time to complete write

}

void LCD\_char(char\_data)

{

/\* write a character to LCD \*/

LCD\_data\_port = char\_data; //outputs char to PORTD and the data bus of the LCD

LCD\_command\_port |= (1<<RS); //sets RS bit High -> Data Register

LCD\_command\_port &= ~(1<<RW); //sets R/W’ bit Low -> write

LCD\_command\_port |= (1<<EN); //sets EN bit High

*\_delay\_us*(1);

LCD\_command\_port &= ~(1<<EN);//Low going transition of EN bit Low writes data to LCD

*\_delay\_ms*(2); //give LCD time to complete write

}

void LCD\_string (char str[])

{

/\* Walks through a string a character at a time until

finding a null terminator at the end of the string. Sends a character

at a time to to function LCD\_char \*/

int i;

for(i=0; str[i]!= 0; i++)

{

LCD\_char (str[i]);

}

}

void LCD\_clear()//clears display and goes to position of 1st char on 1st line

{

LCD\_command (0x01);//clears entire Display

*\_delay\_us*(40);

}

int main(void)

{

LCD\_Init();//clears display and sets display to first char position, first row

LCD\_string("Embedded System");//print string constant to LCD

LCD\_command(0xC0);//go to first char of the 2nd line of the display

LCD\_string("Laboratory");//print string constant to LCD

while (1)

{

}

return 0;

}

Sign OFF \_\_\_\_\_7\_\_\_\_\_\_\_

TEST Code #2

Copy and paste the code below into Atmel Studio. It is the code we discussed in class. Use it to test your LCD/Thermistor Thermometer circuit. Download the code to the UNO and press the Reset Button (red button by the USB connector). If the circuit and code are correct the Temperature expressed in Degrees Centigrade will be displayed on the LCD. This code shows how to send floating point values to the LCD for display.

Caution:

1) Make sure that you disconnect the wires from the UNO’s pins PD0 and PD1 to the LCD’s pins DB0 and DB1 before downloading the Test code to the UNO. You must do this while downloading since UNO pins PD0 and PD1 are used by the UNO’s UART to “talk with” the host computer. If they are connected to the LCD’s pins DB0 and DB1 the UART will not function properly. Reconnect the wires after you are finished downloading and before you press the reset button.

1. Make sure you set the flags in the AVR/GNU Linker dropdown box to the values used in Lab 6.

/\*

\*

\*

\* Created: 4/25/2020 10:05:04 AM

\* Author : Deyr & Drouantg

\*Modified: 5/8/2022

\*/

#define *F\_CPU* 16000000UL

#include <avr/io.h>

#include <util/delay.h>

#include <math.h>

#include <stdio.h>

#define LCD\_data\_dir DDRD

#define LCD\_command\_dir DDRB

#define LCD\_data\_port PORTD //PORTD defined as DATA Port -> sends characters to LCD

#define LCD\_command\_port PORTB //PORTB defined as Port used to send Commands to LCD

#define RS PB0 //Port B bit 0 defined as register select bit

#define RW PB1 //Port B bit 1 defined as Read/Write' bit

#define EN PB2 //Port B bit 2 is defined as Enable bit

void LCD\_Init(void)

{

/\* Initialize LCD \*/

LCD\_command\_dir = 0b00000111; //PB0, PB1, PB2 set as output pins

LCD\_data\_dir = 0b11111111; //all bits in Port D defined as output pins

*\_delay\_ms*(45); // wait for internal initialization of LCD

LCD\_command(0x38);//sets 8-bit interface with LCD, 2 Line Display,5x8 dot character font

*\_delay\_us*(40);

LCD\_command(0x0C); //turns Display On with Cursor Off

*\_delay\_us*(40);

LCD\_command(0x06); //sets increment character position by one

*\_delay\_us*(40);

LCD\_command(0x01); //clears entire Display

*\_delay\_us*(40);

LCD\_command(0x80); //Sets Display to start at first character of first Line

*\_delay\_us*(40);

}

void LCD\_command (char cmnd)

{

/\*write a command to LCD \*/

LCD\_data\_port = cmnd;

LCD\_command\_port &= ~(1<<RS); //sets RS bit Low -> Instruction Register

LCD\_command\_port &= ~(1<<RW); //sets R/W’ bit Low -> write

LCD\_command\_port |= (1<<EN); //sets EN bit High

*\_delay\_us*(1);

LCD\_command\_port &= ~(1<<EN); //Low going transition of EN bit Low writes data to LCD

*\_delay\_ms*(2); //give LCD time to complete write

}

void LCD\_char(char char\_data)

{

/\* write a character to LCD \*/

LCD\_data\_port = char\_data;

LCD\_command\_port |= (1<<RS); //sets RS bit High -> Data Register

LCD\_command\_port &= ~(1<<RW); //sets R/W’ bit Low -> write

LCD\_command\_port |= (1<<EN); //sets EN bit High

*\_delay\_us*(1);

LCD\_command\_port &= ~(1<<EN); //Low going transition of EN bit Low writes data to LCD

*\_delay\_ms*(2); //give LCD time to complete write

}

void LCD\_string (char str[])

{

/\* Walks through a string a character at a time until

finding a null terminator at the end of the string. Sends a character

at a time to to function LCD\_char \*/

int i;

for(i=0; str[i]!= 0; i++)

{

LCD\_char (str[i]);

}

}

void LCD\_clear()

{

LCD\_command (0x01);

*\_delay\_us*(40);

}

int main(void)

{

char result[10];

unsigned int ADC\_data = 0;

float ADC\_voltage = 0.0;

float Temp = 0.0;

ADMUX = (1<<REFS0);//5V supply used for Vref; Analog Channel 0 (ADC0) used as input; ADC result is right justified

DIDR0 = (1 <<ADC0D);//Disables digital input buffer circuit of the ADC0 pin to reduce power consumption

//Enable ADC; Start ADC; Prescaler Value = 128; ADC Clock = 125 KHz

ADCSRA = (1 << ADEN) | (1 << ADSC) | (1<<ADPS2) | (1<<ADPS1) |(1<<ADPS0);

LCD\_Init();//initializes LCD, clears display and points to position of first character in display's first line

while (1)

{

if (!(ADCSRA & (1<< ADSC)))//ADSC in register ADCSRA will be 1 as long as conversion in progress

{

ADC\_data = ADC;//read ADC\_Output Code

ADC\_voltage = (5.0\*ADC\_data)/1024.0;//calculate Analog Voltage into ADC;type conversion of ADC\_data from int to float

Temp = ((-3.5549\*(*pow*(ADC\_voltage,3)))+(27.625\*(*pow*(ADC\_voltage,2)))+((-89.697)\*((ADC\_voltage)))+132.3);

LCD\_clear(); //clears display and points to position of first character in display's first line

*sprintf*(result,"%.2f",Temp);//stores float "Temp" in str array "result"

LCD\_string(result);//prints contents of "result" to screen

*\_delay\_ms*(1000);

}

ADCSRA |= (1 << ADSC);//start another ADC conversion

*\_delay\_ms*(1000);

}

while (1)

{

}

return 0;

}

MODIFIED CODE

Modify TEST Code #2 by building a “library” (LCD.c) with all of the functions used by the LCD in TEST Code #2 and by building a “header file” (LCD.h) with the prototypes of all of the LCD functions placed in the library.

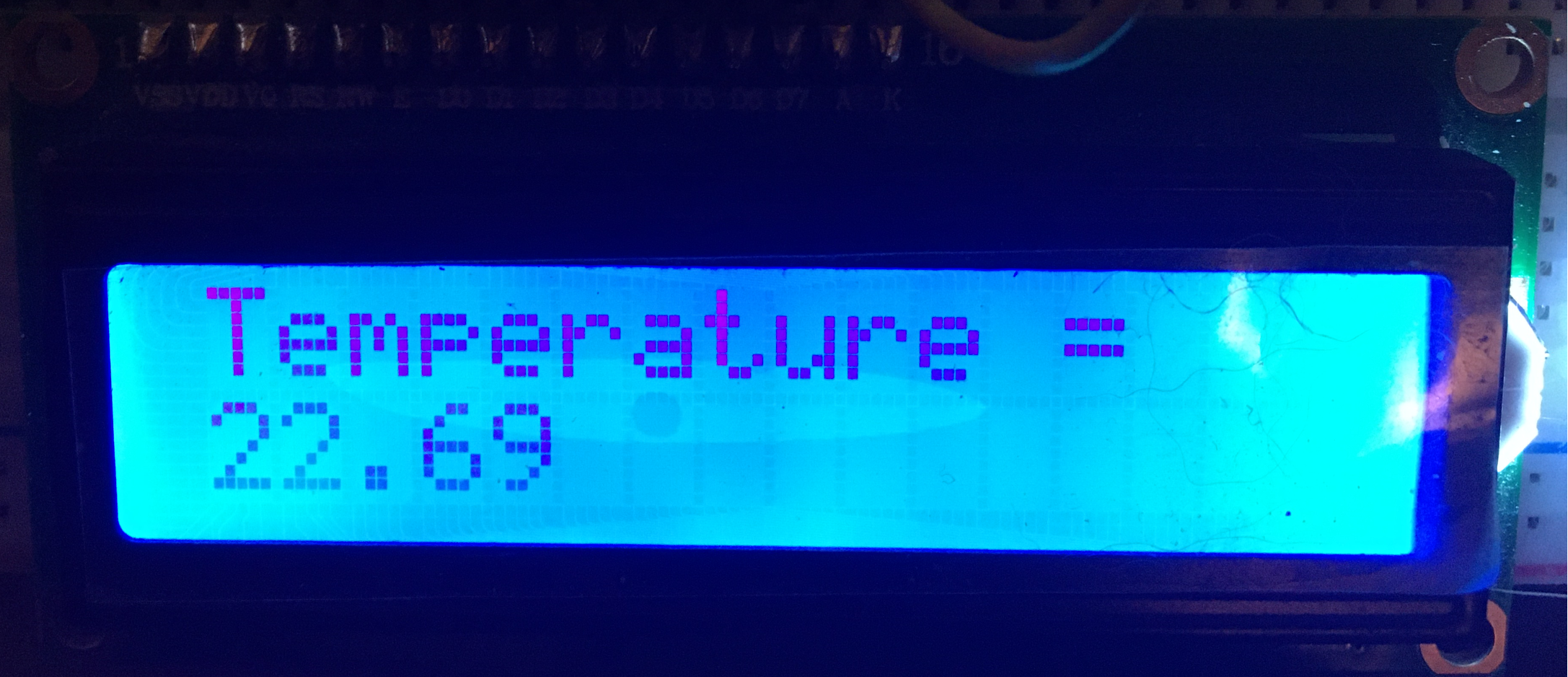
To build LCD.c right click on the project name. Choose ADD -> New Item. You will be brought to another screen where you will choose the option “C File” and name the file “LCD.c”. Click the ADD button. A file named “LCD.c” will appear in the project window of Atmel Studio. The blank file will give you a place to put your code for your library. **Copy** the #include statements needed by the library functions from main.c and paste them into LCD.c. **Cut** the #define statements used by the LCD from main.c and paste them into LCD.c below the #include statements. You should now **cut** the functions used by the LCD from main.c and paste the into LCD.c.

To build LCD.h right click on the project name. Choose ADD -> New Item. You will be brought to another screen where you will choose the option “Include File” and name the file “LCD.h”. Click the ADD button. The blank file will give you a place to put the prototypes for all of the functions in LCD.c. Don’t forget to put a “;” at the end of each prototype **copied** to LCD.h. Place a **copy** of the function prototypes from LCD.c into LCD.h between the “#define LCD\_H\_” and the “#endif /\* LCD\_H\_\*/” statements.

Add “#include "LCD.h" to main.c.

Modify your code such that the first line of your LCD displays “Temperature = “.

The second line of your LCD will display the temperature in degrees Centigrade. This is shown in the photo below.



Download and run your modified code. The temperature expressed in degrees Centigrade should appear on the LCD. Disconnect your UNO board from the USB cable to cut power to the UNO. Plug the power connector from the 9V battery’s clip into the UNO’s barrel connector for power. The system should power up and display the temperature. You now have a portable thermometer!

Sign OFF\_\_\_\_7\_\_\_\_\_\_\_

Deliverables

Send in your main.c, LCD.c, and LCD.h of your Modified Code as text files (.txt) to the Canvas assignments section.