CPE301 – SPRING 2019

Design Assignment 3B

Student Name: Geovanni Portillo

Student #: 8000603824

Student Email: [portig1@unlv.nevada.edu](mailto:portig1@unlv.nevada.edu)

Primary Github address: <https://github.com/portig1/submissions_E>

Directory: submissions\_E/DA/LAB3B/

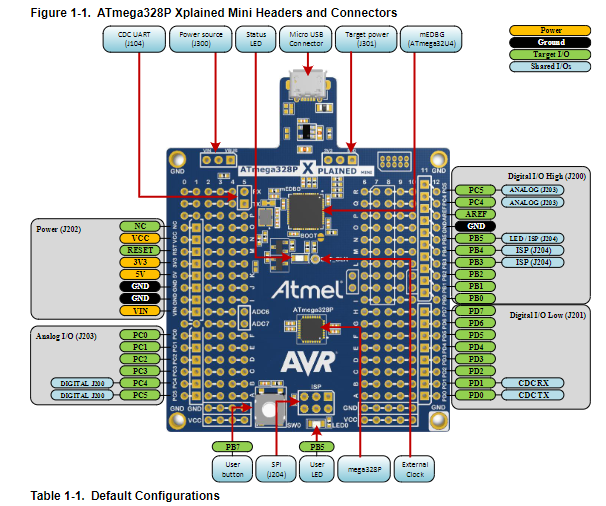
Submit the following for all Labs:

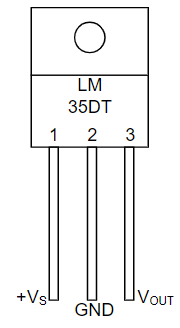
1. In the document, for each task submit the modified or included code (only) with highlights and justifications of the modifications. Also, include the comments.
2. Use the previously create a Github repository with a random name (no CPE/301, Lastname, Firstname). Place all labs under the root folder ESD301/DA, sub-folder named LABXX, with one document and one video link file for each lab, place modified asm/c files named as LabXX-TYY.asm/c.
3. If multiple asm/c files or other libraries are used, create a folder LabXX-TYY and place these files inside the folder.
4. The folder should have a) Word document (see template), b) source code file(s) and other include files, c) text file with youtube video links (see template).

1. **COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS**

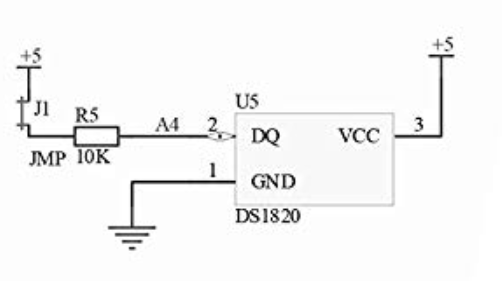
Atmel Studio 7

ATmega328PB Xplained mini





LM35 Schematic



Arduino Shield U5 Schematic

1. **INITIAL CODE OF TASK 1 (Module8B\_3 from website files)**

#define F\_CPU 16000000UL

#define BAUD\_RATE 9600

#include <avr/io.h>

#include <util/delay.h>

void usart\_init ();

void usart\_send (unsigned char ch);

int main (void)

{

usart\_init ();

/\*\* Setup and enable ADC \*\*/

ADMUX = (0<<REFS1)| // Reference Selection Bits

(1<<REFS0)| // AVcc - external cap at AREF

(0<<ADLAR)| // ADC Left Adjust Result

(1<<MUX2)| // Analog Channel Selection Bits

(0<<MUX1)| // ADC4 (PC4 PIN27)

(0<<MUX0);

ADCSRA = (1<<ADEN)| // ADC ENable

(0<<ADSC)| // ADC Start Conversion

(0<<ADATE)| // ADC Auto Trigger Enable

(0<<ADIF)| // ADC Interrupt Flag

(0<<ADIE)| // ADC Interrupt Enable

(1<<ADPS2)| // ADC Prescaler Select Bits

(0<<ADPS1)|

(1<<ADPS0);

while (1)

{

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

while((ADCSRA&(1<<ADIF))==0);//wait for conversion to finish

ADCSRA |= (1<<ADIF);

int a = ADCL;

a = a | (ADCH<<8);

a = (a/1024.0) \* 5000/10;

usart\_send((a/100)+'0');

a = a % 100;

usart\_send((a/10)+'0');

a = a % 10;

usart\_send((a)+'0');

usart\_send('\r');

\_delay\_ms(100);

}

return 0;

}

void usart\_init (void)

{

UCSR0B = (1<<TXEN0);

UCSR0C = (1<< UCSZ01)|(1<<UCSZ00);

UBRR0L = F\_CPU/16/BAUD\_RATE-1;

}

void usart\_send (unsigned char ch)

{

while (! (UCSR0A & (1<<UDRE0))); //wait until UDR0 is empty

UDR0 = ch; //transmit ch

}

void usart\_print(char\* str)

{

int i = 0;

while(str[i] != 0)

usart\_send(str[i]);

}

1. **MODIFIED CODE OF TASK1**

#define F\_CPU 16000000UL

#define BAUD\_RATE 9600

#include <avr/io.h>

#include <avr/interrupt.h>

#include <stdio.h>

void usart\_init ();

void USART\_send( unsigned char data);

void USART\_putstring(char\* StringPtr);

int main (void)

{

usart\_init ();

sei();

TIMSK1 = (1 << OCIE1A);

OCR1A = 62499; //Using TCNT = clk\*delay/prescaler - 1 to find OCR1A given clk = 16MHz, OCR1A was calculated to 62,499

TCCR1A = 0; // COM1A/B Normal Operation, OC1A/B Disconnected

TCCR1B = (1 << WGM12) | (1 << CS12); //WGM CTC Mode, Prescaler = 256

/\*\* Setup and enable ADC \*\*/

ADMUX = (0<<REFS1)| // Reference Selection Bits

(1<<REFS0)| // AVcc - external cap at AREF

(0<<ADLAR)| // ADC Left Adjust Result

(1<<MUX2)| // Analog Channel Selection Bits

(0<<MUX1)| // ADC4 (PC4 PIN27)

(0<<MUX0);

ADCSRA = (1<<ADEN)| // ADC Enable

(0<<ADSC)| // ADC Start Conversion

(0<<ADATE)| // ADC Auto Trigger Enable

(0<<ADIF)| // ADC Interrupt Flag

(0<<ADIE)| // ADC Interrupt Enable

(1<<ADPS2)| // ADC Prescaler Select Bits

(0<<ADPS1)|

(1<<ADPS0);

while (1)

{

}

return 0;

}

ISR(TIMER1\_COMPA\_vect)

{

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

while((ADCSRA&(1<<ADIF))==0);//wait for conversion to finish

ADCSRA |= (1<<ADIF);

char output[20];

int tempC = ADCL;

tempC = tempC | (ADCH<<8);

tempC = (tempC/1024.0) \* 5000/10;

snprintf(output, sizeof(output), "%d\r\n", tempC);

USART\_putstring(output);

}

void usart\_init (void)

{

UCSR0B = (1<<TXEN0);

UCSR0C = (1<< UCSZ01)|(1<<UCSZ00);

UBRR0L = F\_CPU/16/BAUD\_RATE-1;

}

void USART\_send( unsigned char data) {

while (!(UCSR0A & (1 << UDRE0))); //wait until UDR0 is empty

UDR0 = data; //transmit ch

}

void USART\_putstring(char\* StringPtr) {

while (\*StringPtr != 0x00) {

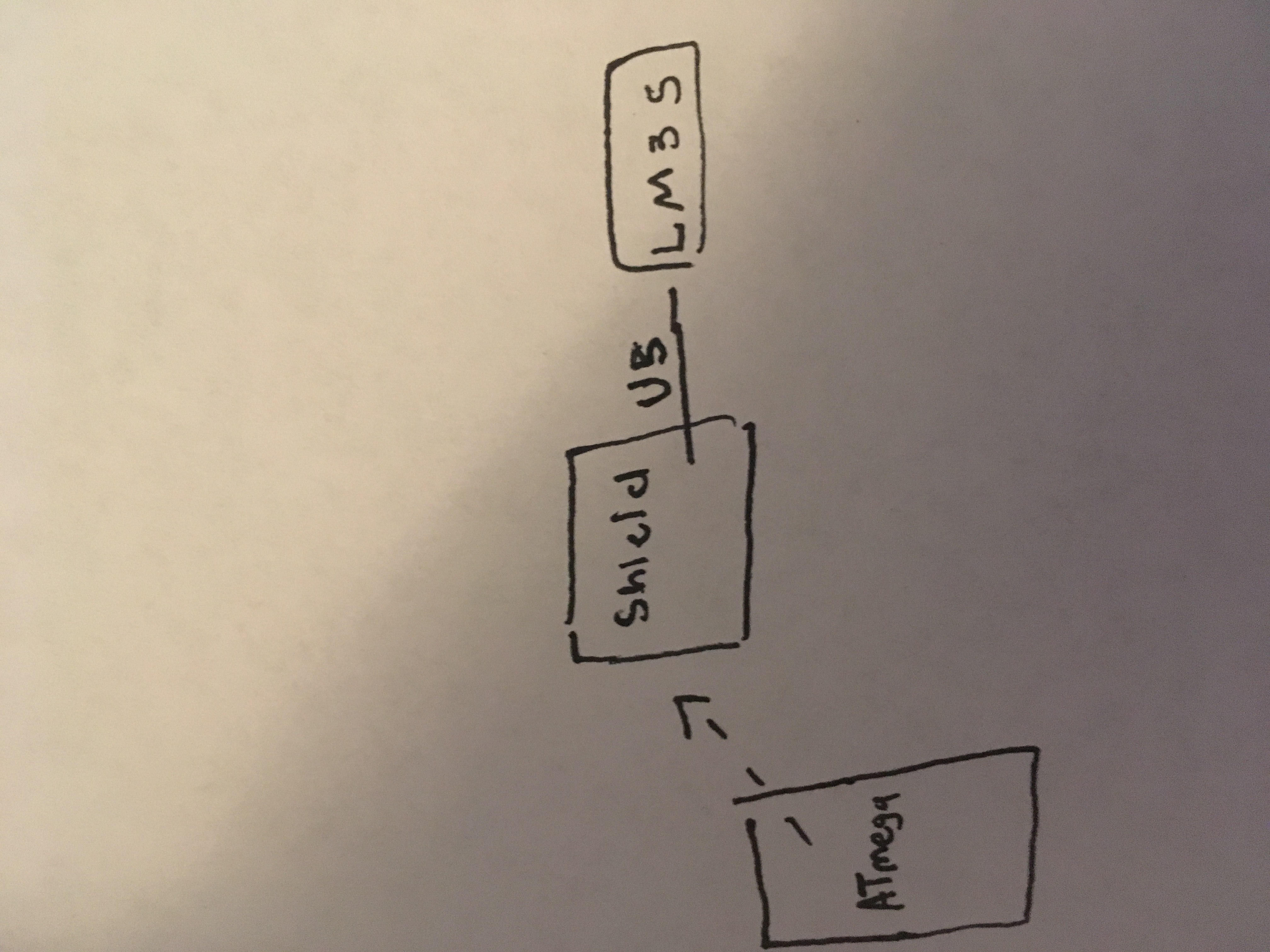
USART\_send(\*StringPtr);

StringPtr++;

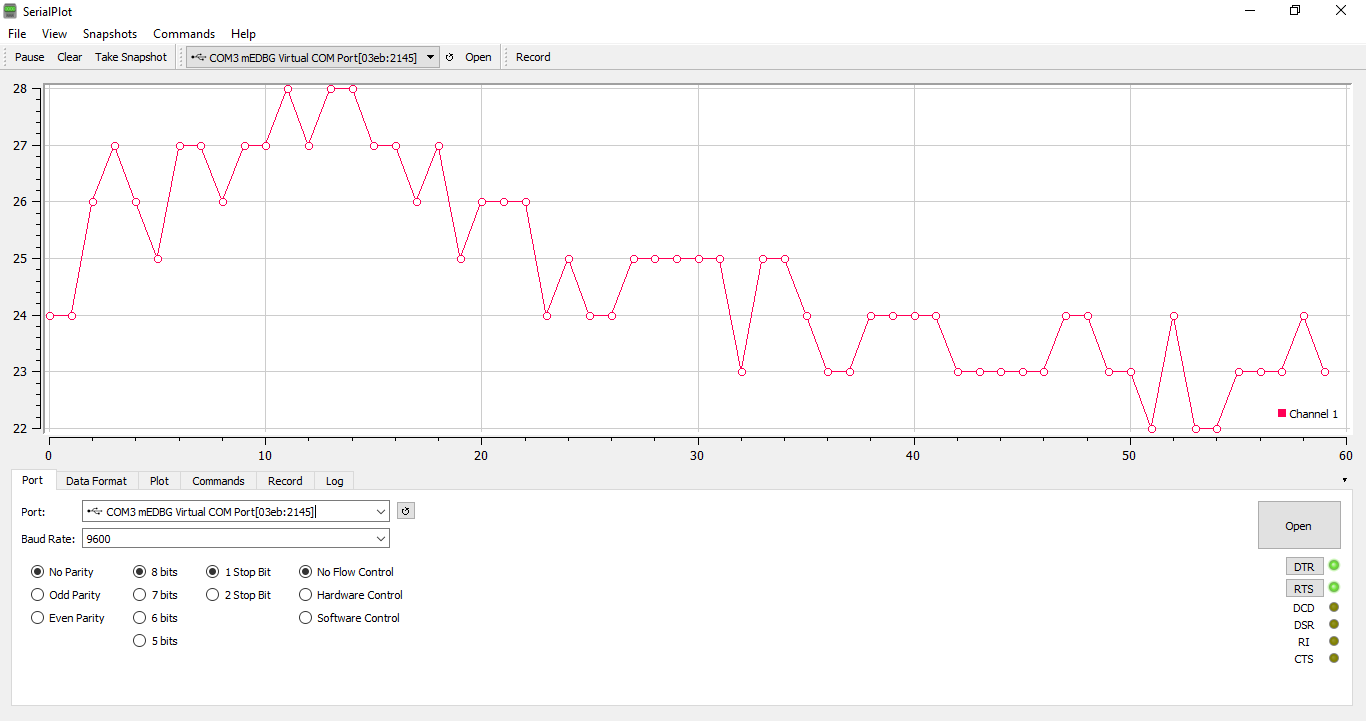
}

}

1. **SCHEMATICS**

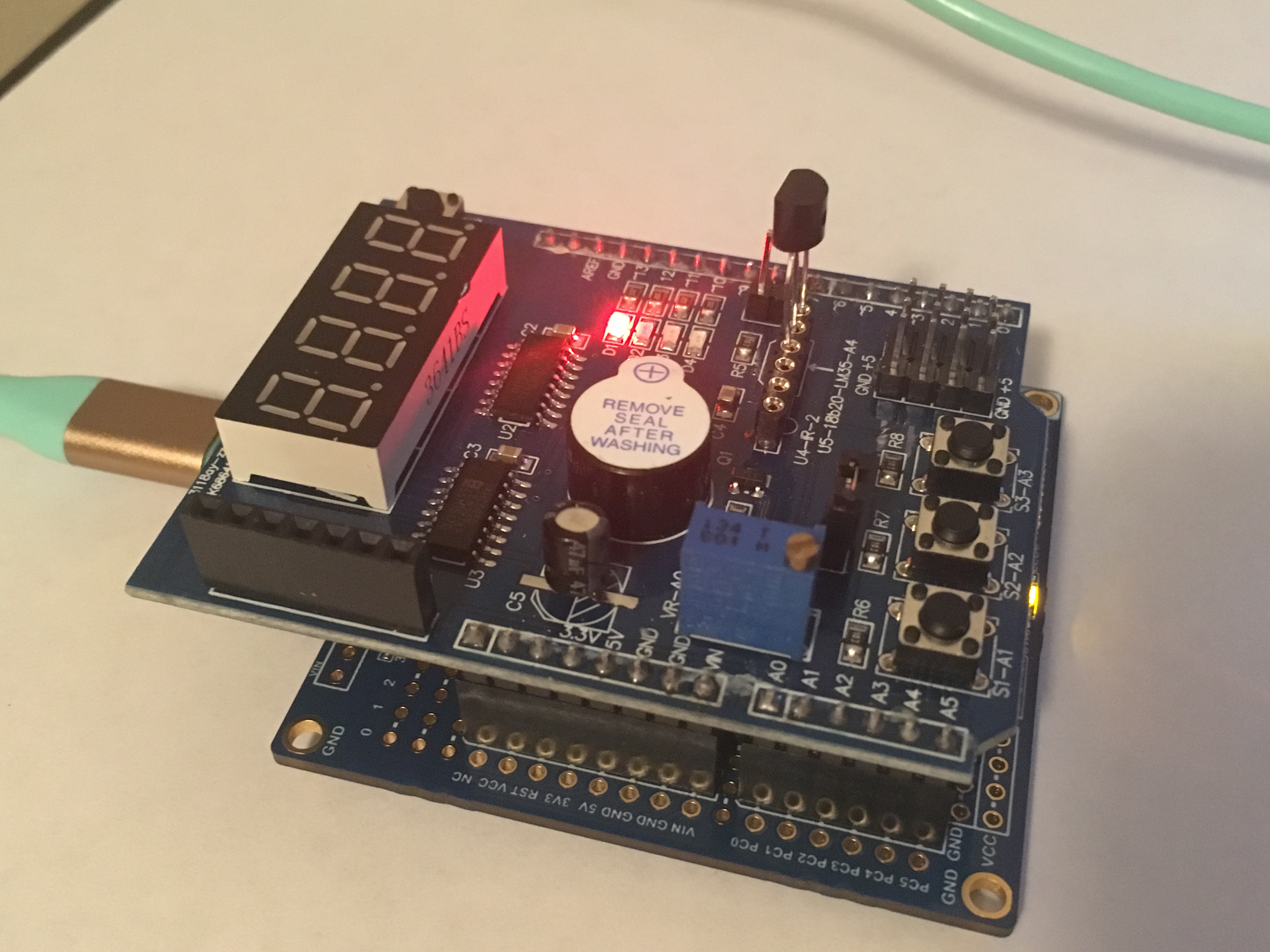


1. **SCREENSHOTS OF EACH TASK OUTPUT**



Task 1/Task 2 output in Serial Plot

1. **SCREENSHOT OF EACH DEMO (BOARD SETUP)**

****

Board setup for Task 1/Task 2

1. **VIDEO LINKS OF EACH DEMO**

<https://youtu.be/TBz9H7icXFQ>

1. **GITHUB LINK OF THIS DA**

https://github.com/portig1/submissions\_E/tree/master/DA/LAB3B

**Student Academic Misconduct Policy**

<http://studentconduct.unlv.edu/misconduct/policy.html>

“This assignment submission is my own, original work”.

Geovanni Portillo