CPE301 – SPRING 2019

Design Assignment 4B

Student Name: Moriah Wingrove

Student #: 5000156247

Student Email: Wingrove@unlv.nevada.edu

Primary Github address: <https://github.com/windew/Tiny_Dragons.git>

Directory:

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**

Atmega 328pb

pushbutton

HiLetgo Multi-nction Shield ProtoShield Multi-functional Expansion Board Sensor Shield Module witn

stepper Motor

servo

**INITIAL/MODIFIED/DEVELOPED CODE OF TASK 4B/1**

**C code**

/\*

\* design\_assignment\_4B.c

\*

\* Created: 11/11/2019 10:21:04 AM

\* Author : Moriah Wingrove

\*/

//Task one: Write an AVR C program to control the speed of the stepper motor using a potentiometer connected to PC0. Use a timer in CTC

// mode to control the delay

#define *F\_CPU* 1000000UL

#include <avr/io.h>

#include <util/delay.h>

#include <avr/interrupt.h>

void init\_adc(void); // ADC functions

void timer0\_init(void); // ADC function for Timer 0 CTC

int control = 0; // converted ADC value to control motor speed

int main(void)

{

DDRB = 0xFF; // Set Port B for outputs

int wait = 0; // Wait is the delay

init\_adc(); // Initialize ADC conversions

timer0\_init(); // Initialize timer0 CTC function

while(1)

{

while((ADCSRA&(1<<ADIF))==0);

control = ADC\*80/100; // ADC Conversion

// Stepper Motor function in half stepper mode (7 commands vs 4)

if (control < 1) // when control < 1 potentiometer is lowest voltage

{

wait = 0; // wait is the delay that controls the

PORTB = 0x0C;

*\_delay\_ms* (wait);

PORTB = 0x04;

*\_delay\_ms* (wait);

PORTB = 0x06;

*\_delay\_ms* (wait);

PORTB = 0x02;

*\_delay\_ms* (wait);

PORTB = 0x01;

*\_delay\_ms* (wait);

PORTB = 0x09;

*\_delay\_ms* (wait);

PORTB = 0x08;

*\_delay\_ms* (wait);

}

else if (control < 3)

{

wait = 100;

PORTB = 0x0C;

*\_delay\_ms* (wait);

PORTB = 0x04;

*\_delay\_ms* (wait);

PORTB = 0x06;

*\_delay\_ms* (wait);

PORTB = 0x02;

*\_delay\_ms* (wait);

PORTB = 0x01;

*\_delay\_ms* (wait);

PORTB = 0x09;

*\_delay\_ms* (wait);

PORTB = 0x08;

*\_delay\_ms* (wait);

}

else if (control < 4)

{

wait = 50;

PORTB = 0x0C;

*\_delay\_ms* (wait);

PORTB = 0x04;

*\_delay\_ms* (wait);

PORTB = 0x06;

*\_delay\_ms* (wait);

PORTB = 0x02;

*\_delay\_ms* (wait);

PORTB = 0x01;

*\_delay\_ms* (wait);

PORTB = 0x09;

*\_delay\_ms* (wait);

PORTB = 0x08;

*\_delay\_ms* (wait);

}

else if (control > 4) // when control > 4 potentiometer at max value

{

wait = 10;

PORTB = 0x0C;

*\_delay\_ms* (wait);

PORTB = 0x04;

*\_delay\_ms* (wait);

PORTB = 0x06;

*\_delay\_ms* (wait);

PORTB = 0x02;

*\_delay\_ms* (wait);

PORTB = 0x01;

*\_delay\_ms* (wait);

PORTB = 0x09;

*\_delay\_ms* (wait);

PORTB = 0x08;

*\_delay\_ms* (wait);

}

}

}

void init\_adc(void) // Initiate ADC function

{

ADMUX = (1<<REFS0); // Reference voltage at Aref

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

// from ADCRSA we Enable ADC, Start Conversion, Set prescalar as 128

}

// Using a timer in CTC mode to control the delay

void timer0\_init()

{

TCCR0B |= (1 << WGM12)|(1 << CS11)|(1 << CS10); // Prescalar 64 set CTC mode

TCCR0A |= (1 << COM1A0); // Set timer in OCOA Pin Toggle Mode

TCNT0 = 0; // Initialize Counter

OCR0A = control; // OCROA reading ADC converted value

}

**INITIAL/MODIFIED/DEVELOPED CODE OF TASK 4B/2**

/\*

\* Design\_Assignment\_4B\_task2.c

\*

\* Created: 11/13/2019 9:46:51 AM

\* Author : Moriah Wingrove

\*/

//Task two: Write an AVR C program to control the position of the Servo Motor using a potentiometer

// connected to PC0. When pot value is 0 the servo is at position

// 0 deg. and when pot value is max (approx. 5V) the servo is at position 180 deg

#define *F\_CPU* 16000000UL /\* Define CPU Frequency e.g. here its 8MHz \*/

#include <avr/io.h> /\* Include AVR std. library file \*/

#include <util/delay.h> /\* Include Delay header file \*/

void main()

{

//Configure TIMER1

TCCR1A|=(1<<COM1A1)|(1<<COM1B1)|(1<<WGM11); //NON Inverted PWM

TCCR1B|=(1<<WGM13)|(1<<WGM12)|(1<<CS11)|(1<<CS10); //PRESCALER=64 MODE 14(FAST PWM)

ICR1=4999; //fPWM=50Hz (Period = 20ms Standard).

DDRB|=(1<<PB1); //PWM Pins as Out

while(1)

{

OCR1A=97; //0 degree

*\_delay\_ms*(200);

OCR1A=316; //90 degree

*\_delay\_ms*(200);

OCR1A=425; //135 degree

*\_delay\_ms*(200);

OCR1A=535; //180 degree

*\_delay\_ms*(200);

}

}

1. **SCHEMATICS**

Use fritzing.org

1. **SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)**
2. **SCREENSHOT OF EACH DEMO (BOARD SETUP)**

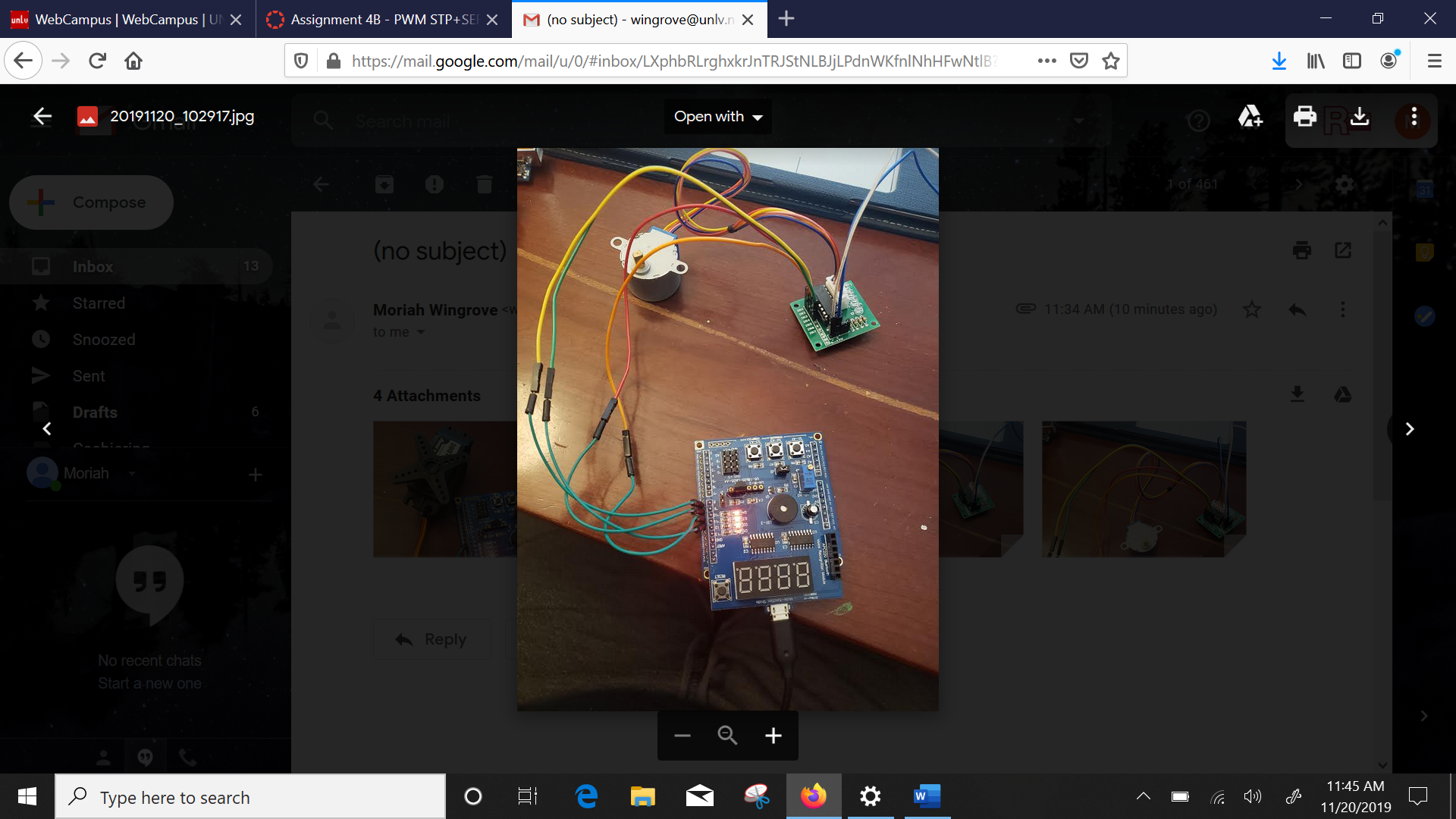


Figure :Stepper board setup

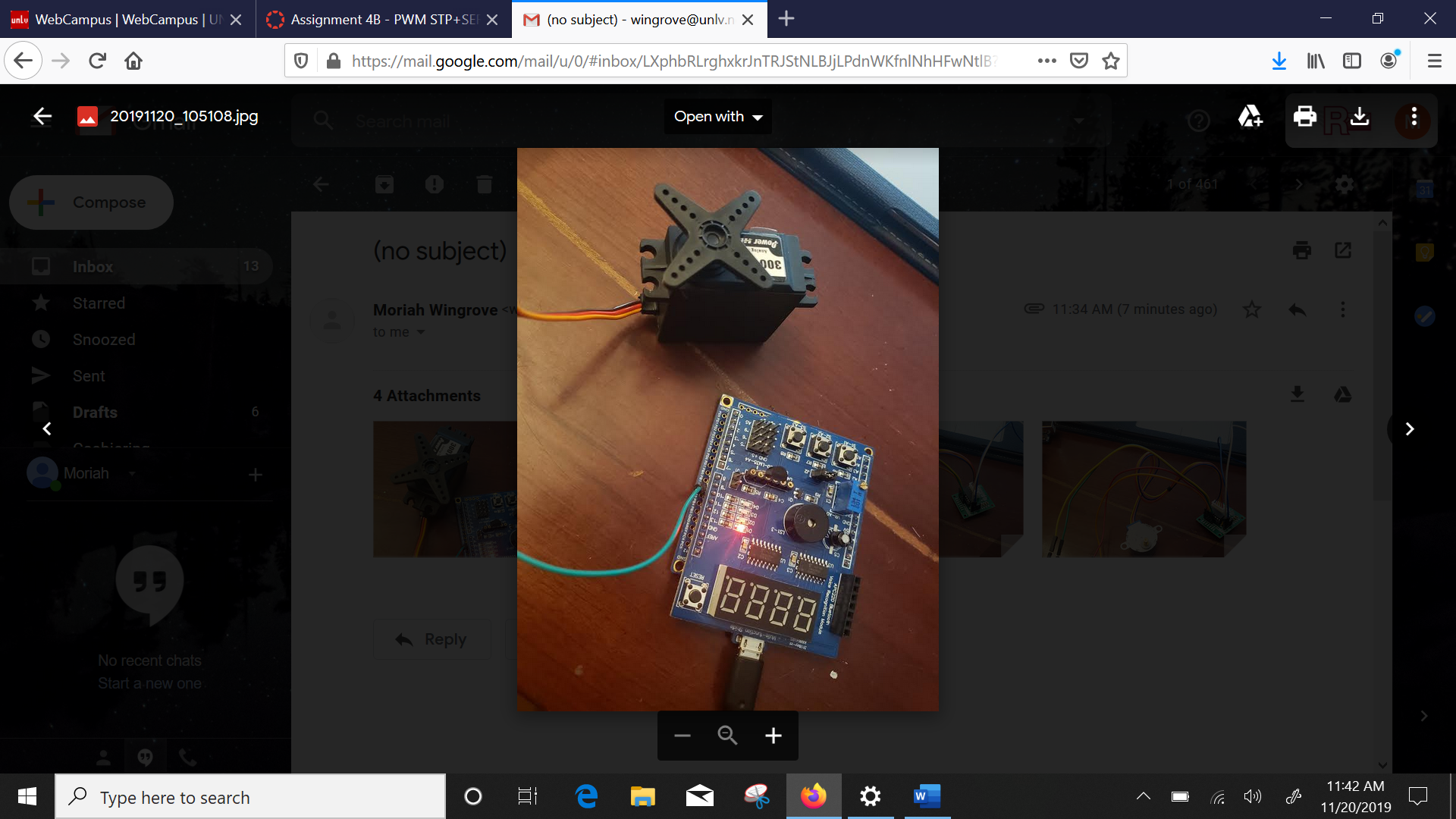


Figure : Servo setup

1. **VIDEO LINKS OF EACH DEMO**
2. **GITHUB LINK OF THIS DA**

<https://github.com/windew/Tiny_Dragons>

**Student Academic Misconduct Policy**

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

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

NAME OF THE STUDENT