CPE301 – SPRING 2019

Design Assignment 4B

Student Name: Tanner Tindall

Student #: 8000733733

Student Email: tindat1@unlv.nevada.edu

Primary Github address: <https://github.com/TannerTindall51>

Directory:

<https://github.com/TannerTindall51/tindalltannerm_submission/tree/master/Design_Assignments/DA4B>

Design Assignment 4B: The goal of the assignment is to develop the above code to do the following:

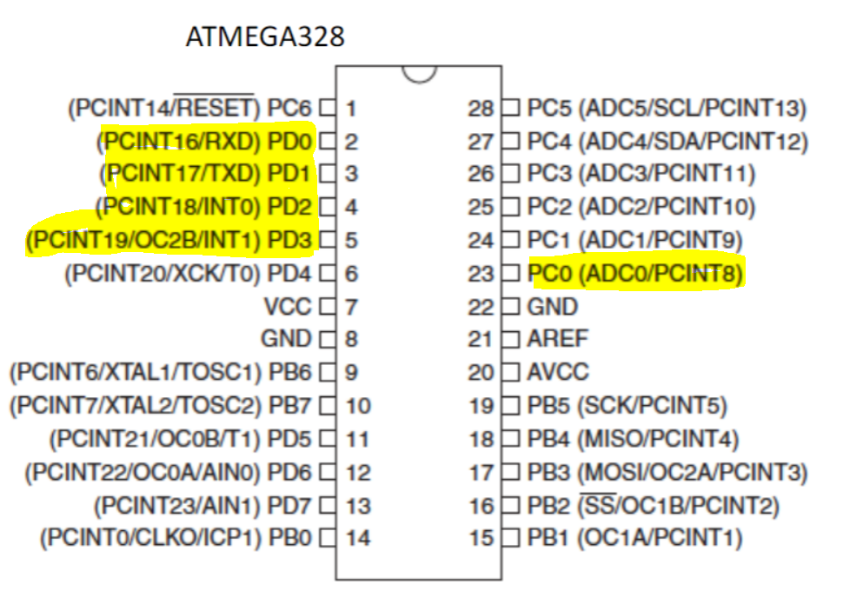
1. Write an AVR C program to control the speed of the Stepper Motor using a potentiometer connected to PC0. Use a timer in CTC mode to control the delay.

2. 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.

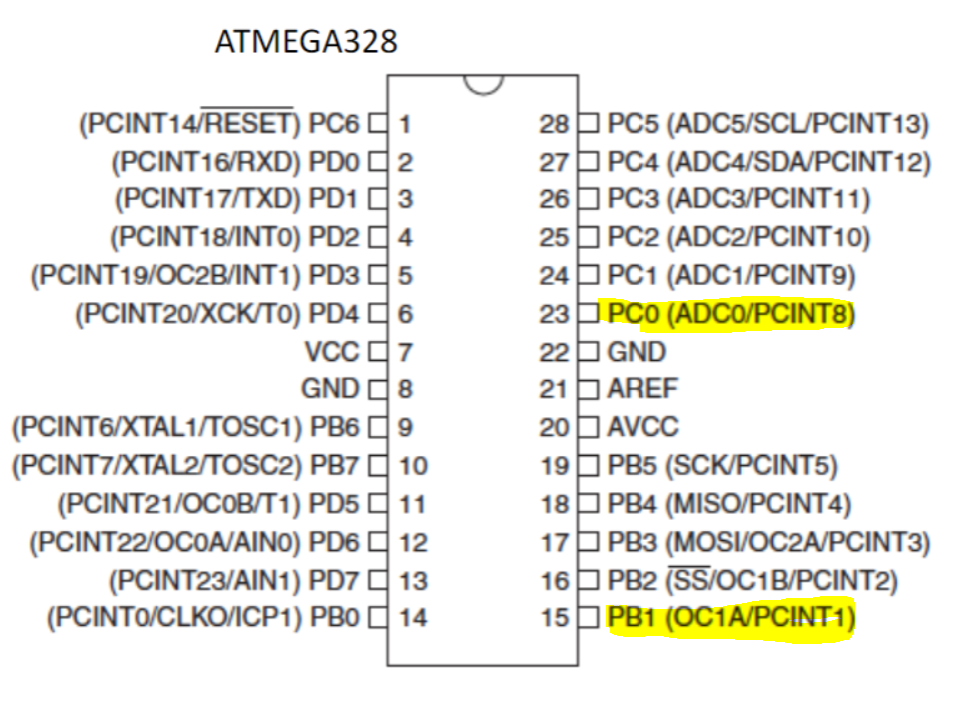
1. **COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS**

* Atmel Studio 7.0 (Assembler, Simulator, & Debugger)
* Atmega328PB-Xmini
* Micro USB
* ULN2003 Motor Driver
* 3001HB Analog Servo
* Potentiometer
* External 5V Power Source

Task 1:



Task 2:



1. **INITIAL/MODIFIED/DEVELOPED CODE OF TASK 1/A**

|  |
| --- |
|  |

//

//CPE301 - DA4B - Task 1

//Tanner Tindall

//

#define *F\_CPU* 16000000UL

#include <avr/io.h>

#include <avr/interrupt.h>

volatile float Value;

void startADC()

{

ADMUX = (1<<REFS0);

ADCSRA = (1<<ADEN) | (1<<ADPS2) | (1<<ADPS1) | (1<<ADPS0); //enable ADC, set prescalar to 128

}

float adcRead()

{

ADCSRA |= (1<<6); //enable ADC conversion

while (!(ADCSRA & (1<<4))); //waits for conversion to complete by ADIF flag

ADCSRA |= (1<<4); //reset when complete

return ADC; //return value once complete

}

void timer()

{

TCCR1A |= 0x00;

TCCR1B |= (1<<WGM12)| (1<<CS12) | (1<<CS10);

//set timer to CTC mode and set prescaler to 1024

TIFR1 |= (1<<OCF1A); //clear interrupt flag

TCNT1 = 0x00; //initialize timer

}

*uint16\_t* adcVal()

{

Value = adcRead()/1023;

return Value;

}

int main()

{

DDRD = 0x0F; //set PD0:3 as output to driver

DDRC &= ~(1<<PINC0); //set PC0 to input for ADC values

timer();

startADC();

while(1)

{

adcVal(); //collect a value from the ADC

//setting delay times based on value received from ADC

if (Value <= 205) //the delays available are the following:

OCR1A = 77; //5ms

else if (Value <= 410)

OCR1A = 155; //10ms

else if (Value <= 615)

OCR1A = 389; //25ms

else if (Value <= 820)

OCR1A = 780; //50ms

else if (Value <= 1024)

OCR1A = 1562; //100ms

//the following while statements loop until the flag is set, then reset & continue

while(!(TIFR1 & (1 << OCF1A)));

PORTD = 0x05;

TIFR1 |= (1 << OCF1A);

while(!(TIFR1 & (1 << OCF1A)));

PORTB = 0x09;

TIFR1 |= (1 << OCF1A);

while(!(TIFR1 & (1 << OCF1A)));

PORTD = 0x0A;

TIFR1 |= (1 << OCF1A);

while(!(TIFR1 & (1 << OCF1A)));

PORTD = 0x06;

TIFR1 |= (1 << OCF1A);

}

}

1. **DEVELOPED MODIFIED CODE OF TASK 2/A**

//

//CPE301 DA4B - Task 2

//Tanner Tindall

//

#define *F\_CPU* 16000000UL

#include <avr/io.h>

#include <avr/interrupt.h>

#include <util/delay.h>

volatile float Value;

void startADC()

{

ADMUX = (1<<REFS0);

ADCSRA = (1<<ADEN) | (1<<ADPS2) | (1<<ADPS1) | (1<<ADPS0); //enable ADC, set prescalar to 128

}

float adcRead()

{

ADCSRA |= (1<<6); //enable ADC conversion

while (!(ADCSRA & (1<<4))); //waits for conversion to complete by ADIF flag

ADCSRA |= (1<<4); //reset when complete

return ADC; //return value once complete

}

void timer()

{

TCCR1A |= (1<<COM1A1)| (1<<COM1B1)| (1<<WGM11); //clear OC1 and OCRA on match

TCCR1B |= (1<<WGM13) | (1<<WGM12) | (1<<CS11) |(1<<CS10); //CTC mode with ICR1 as top, set prescalar to 64

ICR1 = 4999;

}

*uint16\_t* adcVal()

{

Value = adcRead()/1023;

return Value;

}

int main()

{

DDRB &= (1<<PINB1); //set PB1 as output to servo

DDRC &= ~(1<<PINC0); //set PC0 to input for ADC values

timer();

startADC();

while(1)

{

adcVal(); //collect a value from the ADC

if (Value == 0) //min value

{

OCR1A = 0; //0

*\_delay\_ms*(20); //time needed for proper servo instruction

}

else if (Value == 255) //max value (already prescaled)

{

OCR1A = 535; //180

*\_delay\_ms*(20);

}

}

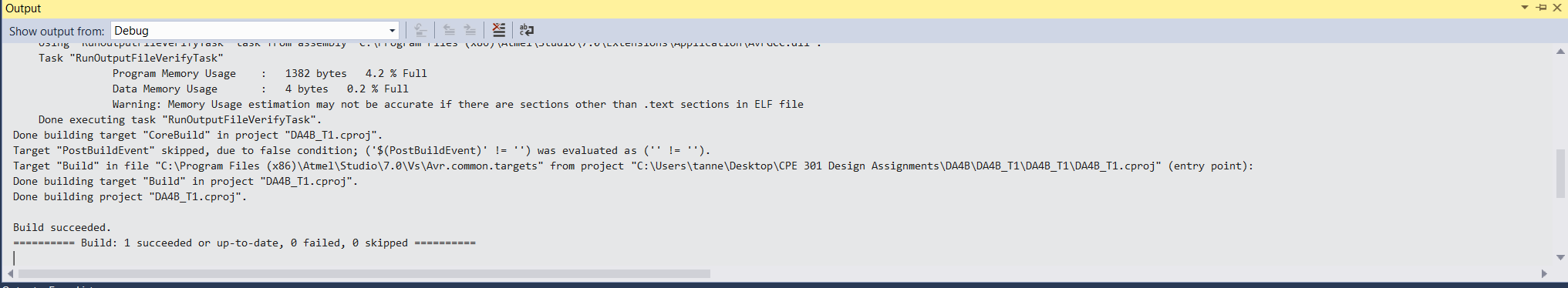
}

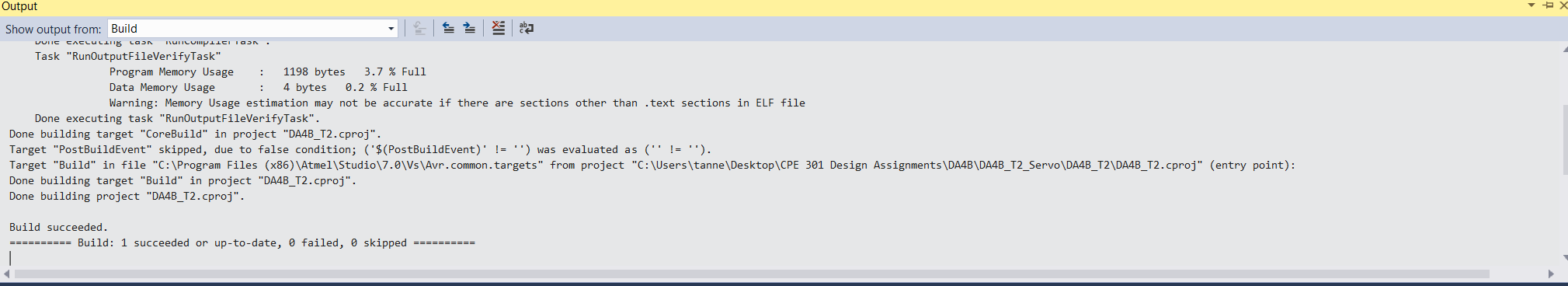
1. **SCHEMATICS**

N/A

1. **SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)**

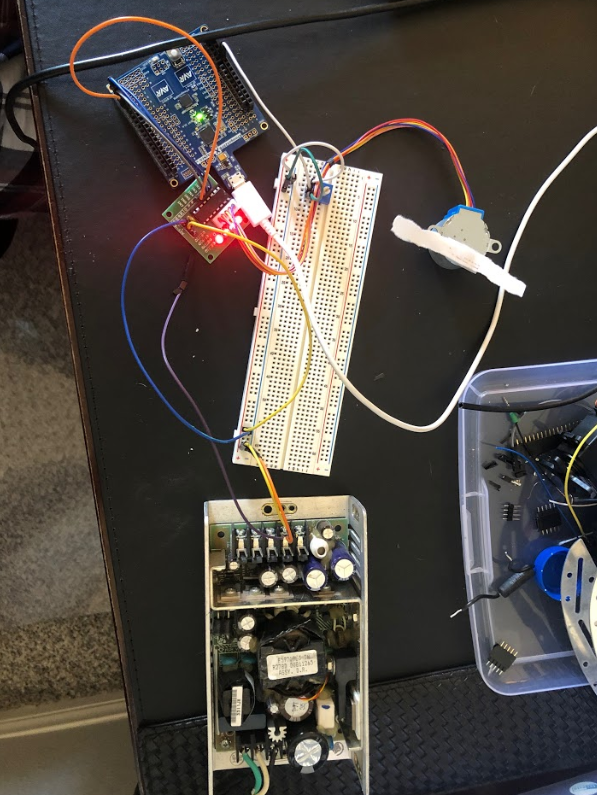
Task 1:



Task 2:

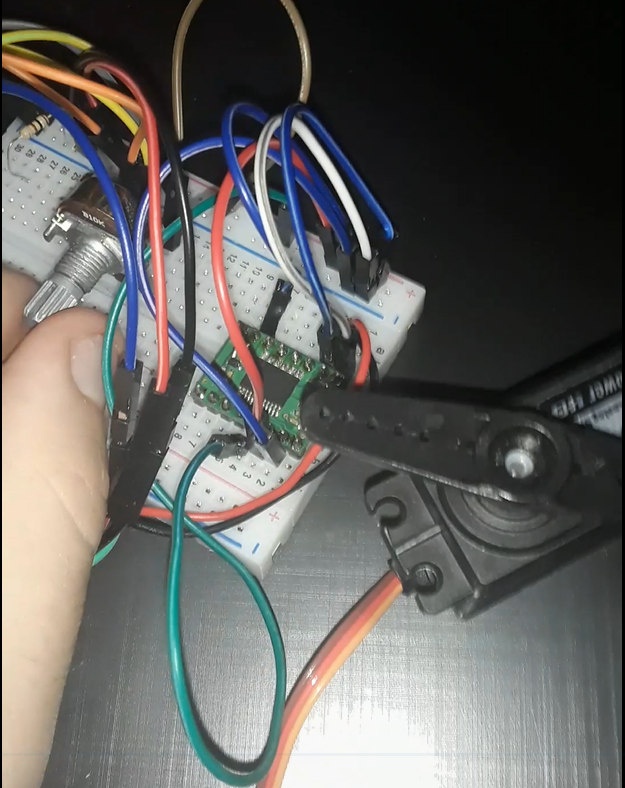
1. **SCREENSHOT OF EACH DEMO (BOARD SETUP)**

Task 1:



Note: Paper was attached to the motor for better visualization of the motion.

Task 2:



1. **VIDEO LINKS OF EACH DEMO**

Task 1: https://youtu.be/s4XKtgjm3aI

Task 2: https://youtu.be/oTwDx1Kscnc

1. **GITHUB LINK OF THIS DA**

<https://github.com/TannerTindall51/tindalltannerm_submission/tree/master/Design_Assignments/DA4B>

**Student Academic Misconduct Policy**

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

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

Tanner Tindall