Trace Stewart

CPE301 – SPRING 2018

Design Assignment 4

DO NOT REMOVE THIS PAGE DURING SUBMISSION:

The student understands that all required components should be submitted in complete for grading of this assignment.

NO	SUBMISSION ITEM	COMPLETED (Y/N)	MARKS (/MAX)
1	COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS		
2.	INITIAL CODE OF TASK 1/A		
3.	INCREMENTAL / DIFFERENTIAL CODE OF TASK 2/B		
3.	INCREMENTAL / DIFFERENTIAL CODE OF TASK 3/C		
3.	INCREMENTAL / DIFFERENTIAL CODE OF TASK 4/D		
3.	INCREMENTAL / DIFFERENTIAL CODE OF TASK 5/E		
4.	SCHEMATICS		
5.	SCREENSHOTS OF EACH TASK OUTPUT		
5.	SCREENSHOT OF EACH DEMO		
6.	VIDEO LINKS OF EACH DEMO		
7.	GOOGLECODE LINK OF THE DA		

1. COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS

The components used in this lab were 3 different motors: DC motor, Stepper Motor, and Servo Motor. The DC motor will be connected to the l293D chip and the Stepper motor will be connected to the ULN2003 chip.

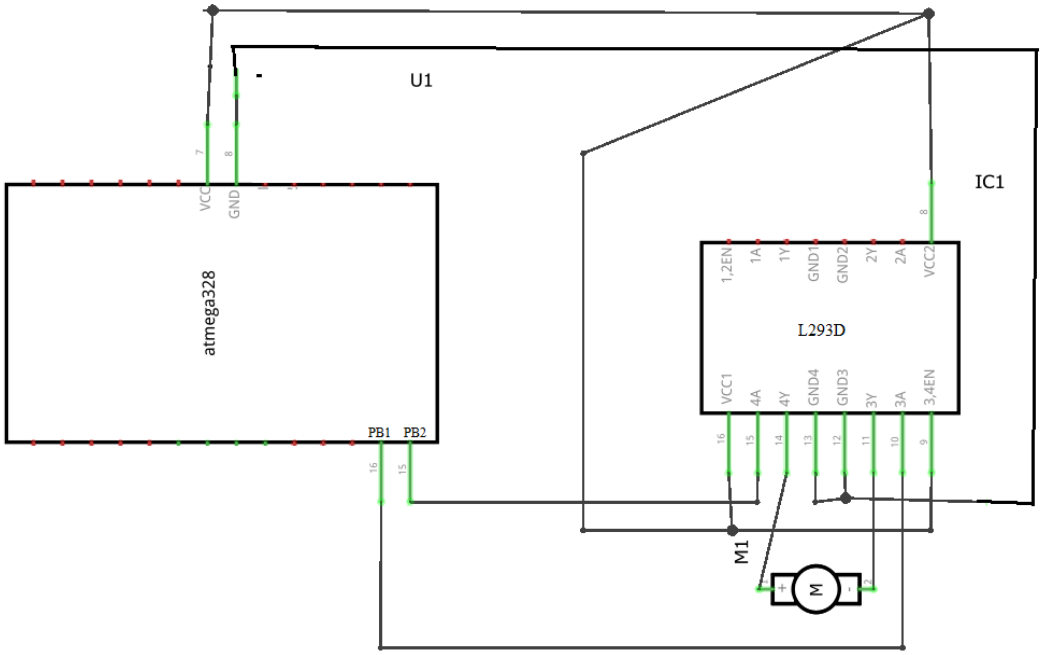

Figure 1: DC Motor

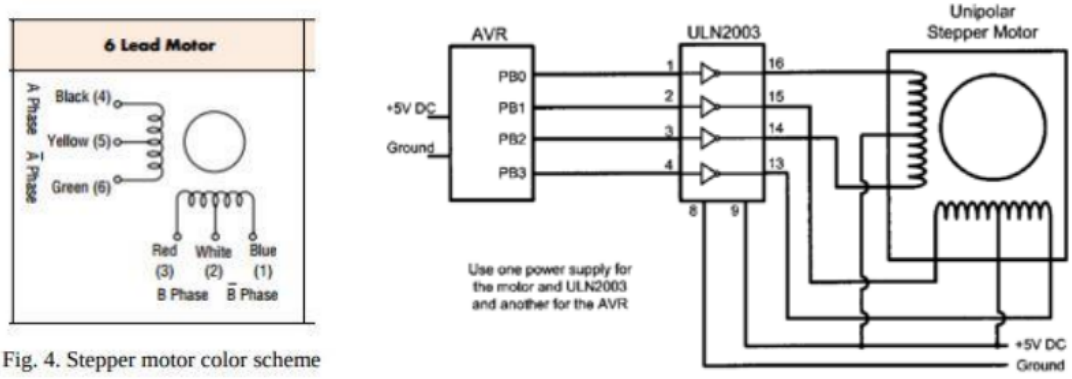

Figure 2: Stepper Motor

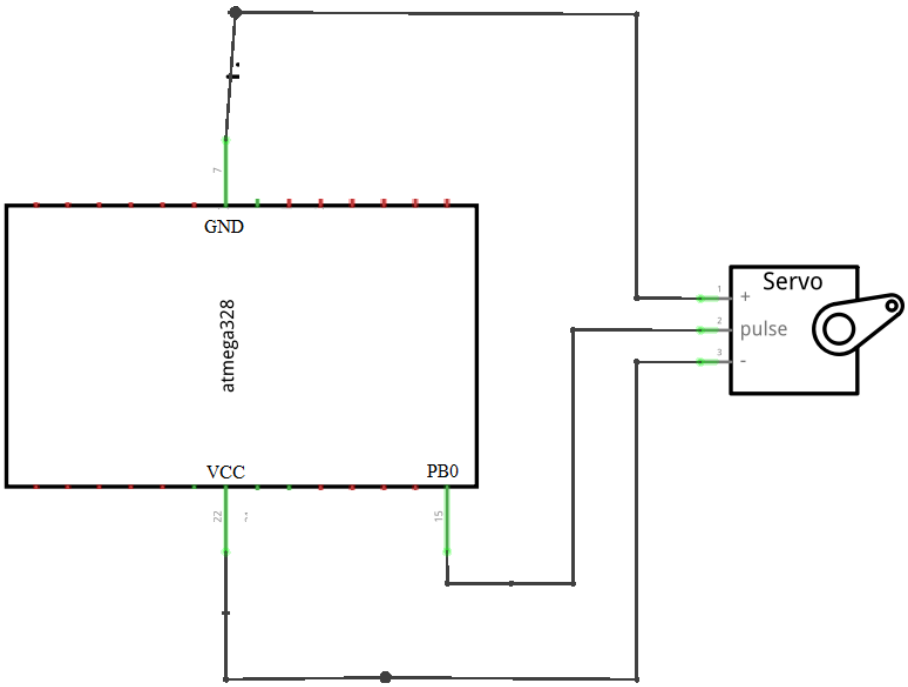

Figure 3: Servo Motor

2. CODE OF TASK 1

```
/*
 * DC Motor.c
 *
 * Created: 3/23/2018 7:28:10 AM
 * Author : trace
 */

#include <avr/io.h>
#define F_CPU 8000000UL
#include <util/delay.h>
#include <avr/interrupt.h>

// Variable Declaration
volatile unsigned int adcVal;                    // holds value of ADC
volatile unsigned int pressed;

int main(void){
        // port initialization
        DDRB = (1<<1)|(1<<2);  // PB.1-2 (OC1A & OC1B) as output to generate PWM
        DDRC = 0;                                // PORTC as input
```

```
	DDRD |= 0xFF; //PD2 Input
	//PORTD |= (1 << 2);

	// timer 1 initialization - generate 50Hz PWM
	TCCR1A |= (1<<COM1A1)|(1<<COM1B1)|(1<<WGM11); // enable PWM for OC1A & OC1B
	// Fast PWM, Non-inverted mode
	TCCR1B |= (1<<WGM13)|(1<<WGM12)|(1<<CS10);		// 8 prescaling
	ICR1 = 65535;                                                   // top value for
timer1

	OCR0A = 122;	//OCR0A is set compare register to 128
	TCCR0B=(1 << CS02) | (1 << CS00);	//TCCR0B sets prescaler to None
	TCCR0A=0x83; //TCCR0A sets WGM00 and WGM01 to 1 which sets Fast PWM as well as
COM0A1 and COM0B1 to 1 which clears OCR0A when compare match.
	TIMSK0 |= (1 << TOIE0);

	// initialize ADC
	DDRC = 0;			// Set PORTC as input for adc
	DIDR0 = 0x1;		// Disable digital input on ADC0 pin
	ADMUX = 0;			// ADC0 (PC.0) used as analog input
	ADMUX |= (1 << REFS0);		// use AVcc as the reference
	ADMUX |= (1 << ADLAR);		// Right adjust for 8 bit resolution


	ADCSRA = 0x87;		// Enable ADC, system clock, 10000111
	ADCSRB = 0x0;		// Free running mode

	EIMSK = 1<<INT0;                                        // Enable INT0
	EICRA = 1<<ISC01 | 1<<ISC00;	// Trigger INT0 on rising edge

	sei();


	while (1){
			ADCSRA |= (1 << ADSC);		// start conversion
			while((ADCSRA&(1<<ADIF))==0);		// wait for conversion to finish
			adcVal = ADCH;			// extract right 10-bits of ADC register

		if(pressed) {
		OCR1A = 257*adcVal;	// OCR1A value for duty cycle
		}
		if(!pressed) {
		OCR1A = 0;	// OCR1A value for duty cycle
		}
	}
	return 0;
}
```

```
ISR(TIMER0_OVF_vect) {

	if((PIND & (1<< PIND0))!=0){
		pressed = !pressed;

	}

}
```

3. CODE OF TASK 2

```
#include <avr/io.h>
#include <avr/interrupt.h>
#define F_CPU 8000000L
#include <util/delay.h>
#include <stdlib.h>

volatile int ovrflw;		// global variable for keeping track of # of times Timer0 overflows
volatile uint8_t ADCvalue; // Global variable, set to volatile if used with ISR

int main(void)
{
	DDRB = 0xFF;

	// initialize ADC
	DDRC = 0;				// Set PORTC as input for adc
	DIDR0 = 0x1;			// Disable digital input on ADC0 pin
	ADMUX = 0;				// ADC0 (PC.0) used as analog input
	ADMUX |= (1 << REFS0);		// use AVcc as the reference
	ADMUX |= (1 << ADLAR);		// Right adjust for 8 bit resolution


	ADCSRA = 0x87;			// Enable ADC, system clock, 10000111
	ADCSRB = 0x0;			// Free running mode

// set up timer with prescaler = 64 and CTC mode
   TCCR1B |= (1 << WGM12)|(1 << CS11)|(1 << CS10);

   // initialize counter
   TCNT1 = 0;

   // initialize compare value
   OCR1A = 0;

   // enable compare interrupt
   TIMSK1 |= (1 << OCIE1A);
```

```
	while(1)
	{
		ADCSRA |= (1 << ADSC);			// Start conversion
		while((ADCSRA&(1<<ADIF))==0);		// Wait for conversion to finish

		if(ADCH < 245){
		ADCvalue = ADCH + 10;			// Only need to read the high value for 8 bit
then equation for Fahrenheit
				PORTB = 0x0C;
				sei();
				PORTB = 0x09;
				sei();
				PORTB = 0x03;
				sei();
				PORTB = 0x06;
				sei();

}
	}
}

ISR (TIMER1_COMPA_vect) {

		for(int i=0;i<=ADCvalue;i++)
			{
				_delay_ms(1);
			}
}
```

4. CODE OF TASK 3

```
/*
 * Servo Motor.c
 *
 * Created: 3/23/2018 7:28:44 AM
 * Author : trace
 */

#define F_CPU 8000000L
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>


int main(void)
{
```

```
	// initialize ADC
	DDRC = 0;			// Set PORTC as input for adc
	DIDR0 = 0x1;		// Disable digital input on ADC0 pin
	ADMUX = 0;			// ADC0 (PC.0) used as analog input
	ADMUX |= (1 << REFS0);		// use AVcc as the reference
	ADMUX |= (1 << ADLAR);		// Right adjust for 8 bit resolution

	ADCSRA = 0x87;		// Enable ADC, system clock, 10000111
	ADCSRB = 0x0;		// Free running mode

	//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 Pin as Out

	while(1)
	{
		ADCSRA |= (1 << ADSC);			// Start conversion
		while((ADCSRA&(1<<ADIF))==0);		// Wait for conversion to finish

		//OCR1A=75;	//90 degrees
		//OCR1A=290;	//0 degrees
		OCR1A = (ADCH * 5 / 6) + 75;

	}

}
```

5. SCHEMATICS

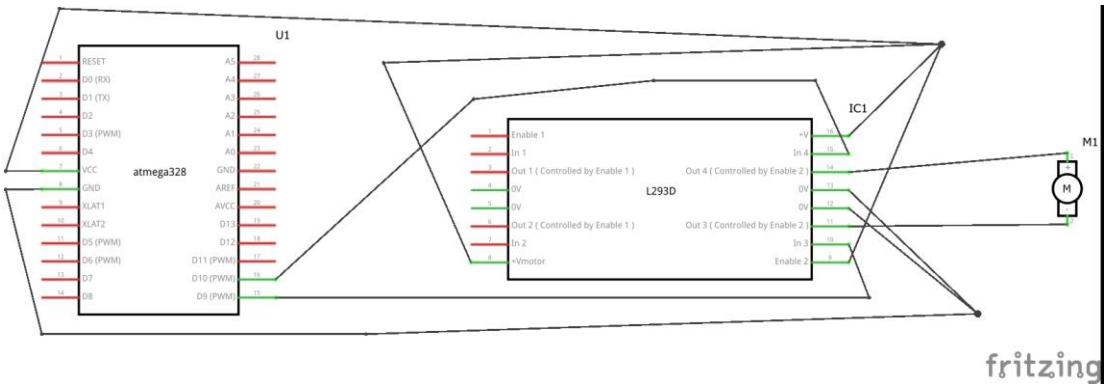

Figure 4:DC Motor

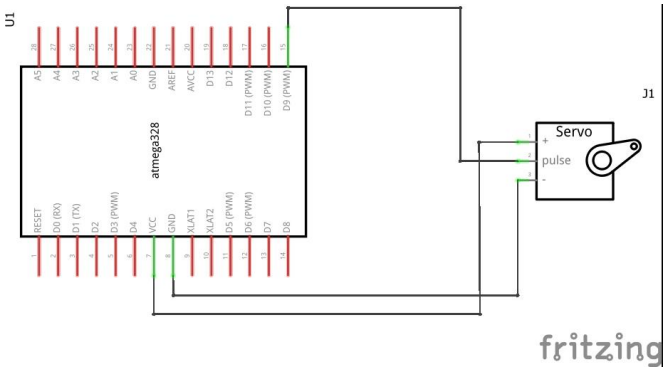

Figure 5:Servo Motor

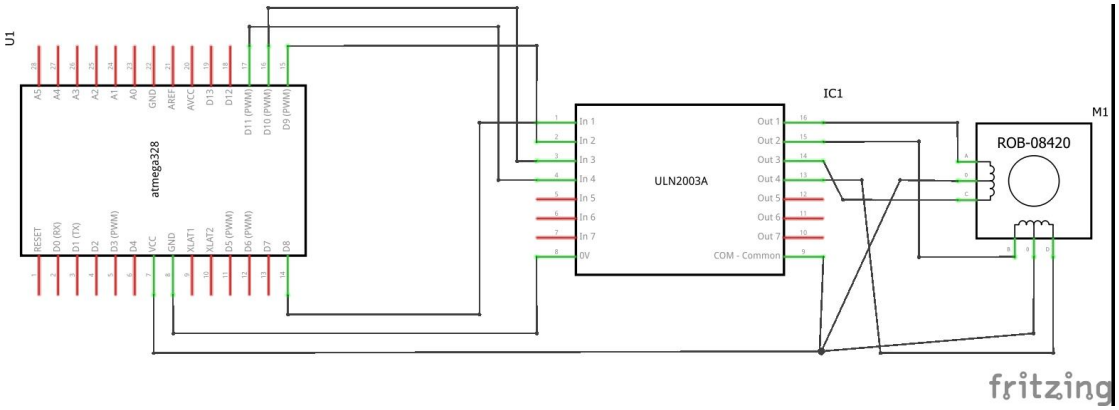

Figure 6: Stepper Motor

6. SCREENSHOT OF EACH DEMO (BOARD SETUP)

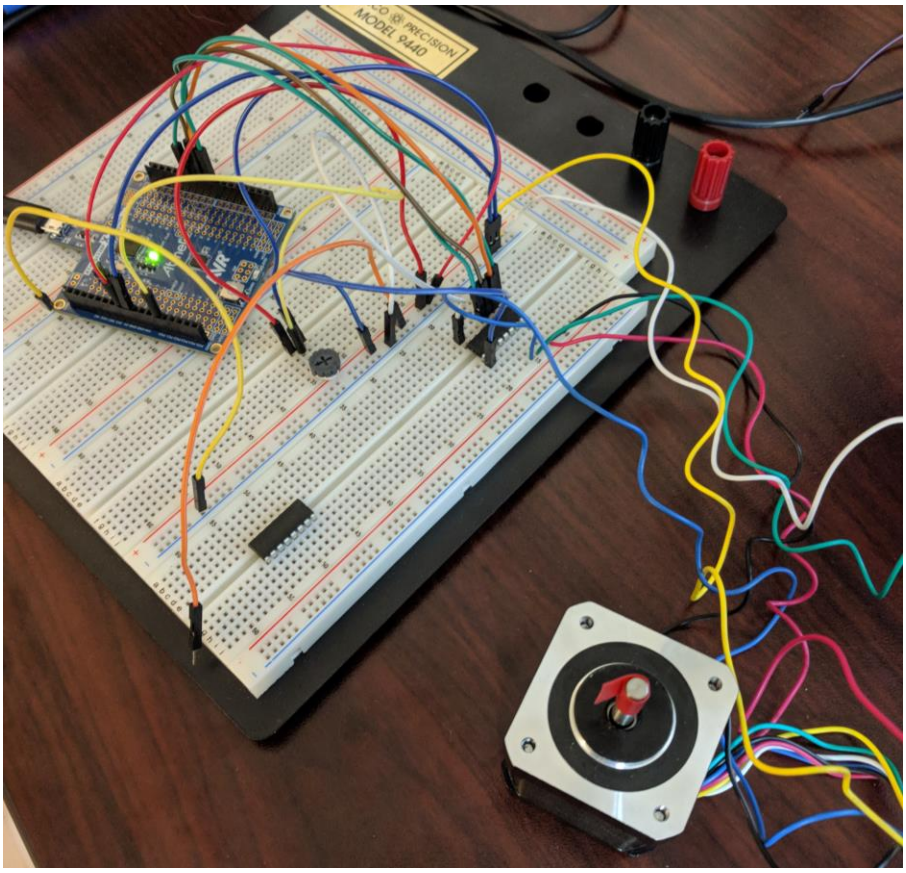

Figure 7:Stepper Motor

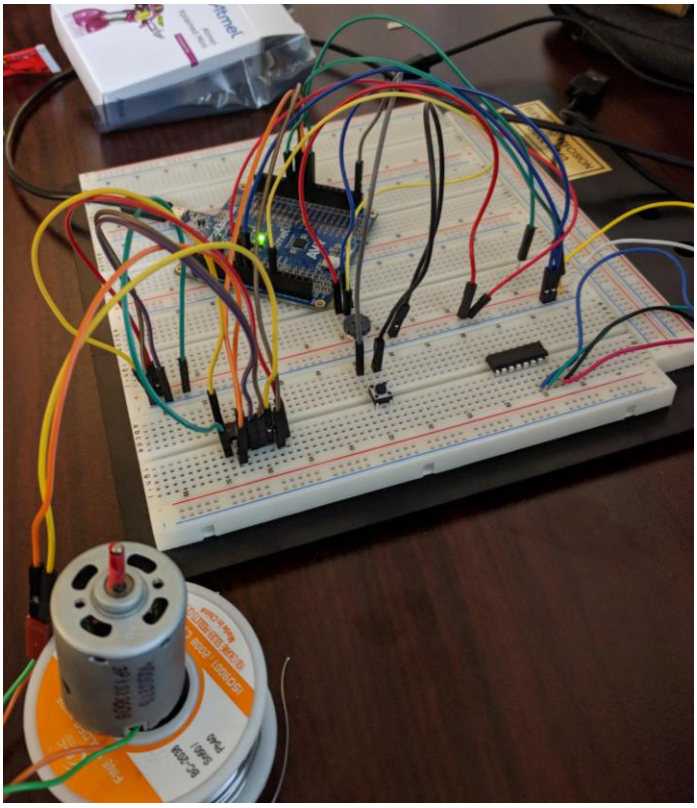

Figure 8: DC Motor

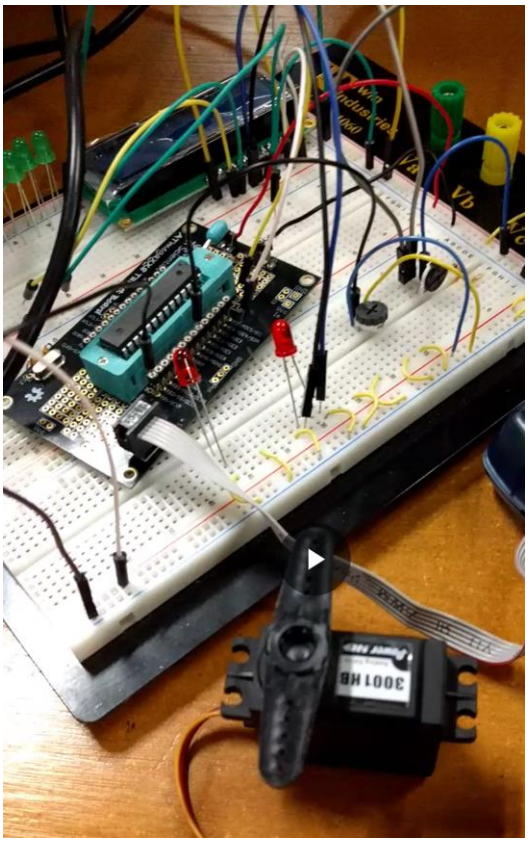

Figure 9: Servo Motor

7. VIDEO LINKS OF EACH DEMO

Task 1: https://www.youtube.com/watch?v=mORrXXEiRAg
Task 2: https://www.youtube.com/watch?v=3_fidSS8jNE
Task 3: https://www.youtube.com/watch?v=XMIAvHLSbeA

8. GITHUB LINK OF THIS DA

https://github.com/TraceStewart/epc103gnirps8102vlnu/tree/master/DA4

Student Academic Misconduct Policy
http://studentconduct.unlv.edu/misconduct/policy.html

"*This assignment submission is my own, original work*".
Trace Stewart