CPE301 – SPRING 2019

Design Assignment 4B

Student Name: Ricky Perez
Student #: 5002297620
Student Email: perezr1@unlv.nevada.edu
Primary Github address: https://github.com/RickyPerez79/submission_da.git
Directory: DA4B

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

- Xplained Mini
- Bread board
- Stepper Motor
- Servo Motor
- ULN2003
- Power Supply

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

```
/*
 * DA4B_Task_1.c
 *
 * Created: 4/19/2019 11:19:47 PM
 * Author : perezr1
 */

/*////////////////////////// Include Library ////////////////////////////////////////*/
#define F_CPU 16000000UL
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
/*////////////////////////////////////////////////////////////////////////////////////*/

/*////////////////////////////////Function Prototype//////////////////////////////////*/
void adc_int(void);
/*////////////////////////////////////////////////////////////////////////////////////*/

/*////////////////////////////////// Variables //////////////////////////////////////*/
volatile int status_control = 0; // if set to 1, the motor is turned off
volatile unsigned int speed_control; // variable used to control delay
/*////////////////////////////////////////////////////////////////////////////////////*/


int main(void)
{
/*////////////////////////////// Initializations ////////////////////////////////////*/
	DDRB = 0x0F; //Enable output on all of the B pins
	PORTB = 0x00; // initialize to 0v
	TCCR1B = (1 << WGM13) | (1 << WGM12) | (1 << CS11);//set CTC mode & prescaler of 8
	ICR1 = 62258;
/*////////////////////////////////////////////////////////////////////////////////////*/
	adc_int(); // call function to initialize ADC

	while(1){
		while((ADCSRA&(1<<ADIF))==0); // wait for conversion
		if (ADC >= 820)
		{
			status_control = 1;
		}
		if ((ADC < 820) && (ADC >= 617))
		{
			status_control = 0;
```

```
                speed_control = 0x1869;
            }
            if ((ADC < 617) && (ADC >= 414))
            {
                status_control = 0;
                speed_control = 0x124F;
            }
            if ((ADC < 414) && (ADC >= 211))
            {
                status_control = 0;
                speed_control = 0x0C34;
            }
            if ((ADC < 211) && (ADC >= 000))
            {
                status_control = 0;
                speed_control = 0x061A;
            }

            OCR1A = speed_control; // set OCR1A to the determined speed
            TCNT1 = 0x00; // reset timer
            // if the status is 1, then the motor will be OFF. If the status is 0 then
the motor will be on and will run a step with a delay.
            if(status_control == 0)
            {
                while((TIFR1 & 0x2) != 0x2);
                PORTB = 0x09;
                TIFR1 |= (1<<OCF1A);
                while((TIFR1 & 0x2) != 0x2);
                PORTB = 0x03;
                TIFR1 |= (1<<OCF1A);
                while((TIFR1 & 0x2) != 0x2);
                PORTB = 0X06;
                TIFR1 |= (1<<OCF1A);
                while((TIFR1 & 0x2) != 0x2);
                PORTB = 0X0C;
                TIFR1 |= (1<<OCF1A);
            }
        }
}
/*////////////////////////////// Function ////////////////////////////////////////*/
void adc_int(void)
{
        ADMUX = (0<<REFS1)| // Reference Selection Bits
        (1<<REFS0)| // AVcc-external cap at       AREF
        (0<<ADLAR)| // ADC Left Adjust Result
        (0<<MUX3)|
        (0<<MUX2)| // ANalogChannel Selection Bits
        (0<<MUX1)| // ADC0 (PC0)
        (0<<MUX0);

        ADCSRA = (1<<ADEN)| //ADC Enable
        (1<<ADSC)| // ADC Start Conversion
        (1<<ADATE)| // ADC Auto Trigger Enable
        (0<<ADIF)| // ADC Interrupt Flag
        (1<<ADIE)| // ADC Interrupt Enable
        (1<<ADPS2)| // ADC Prescaler Select Bits
        (1<<ADPS1)|
        (1<<ADPS0);
```

```
}
/*////////////////////////////////////////////////////////////////////////////////*/
```

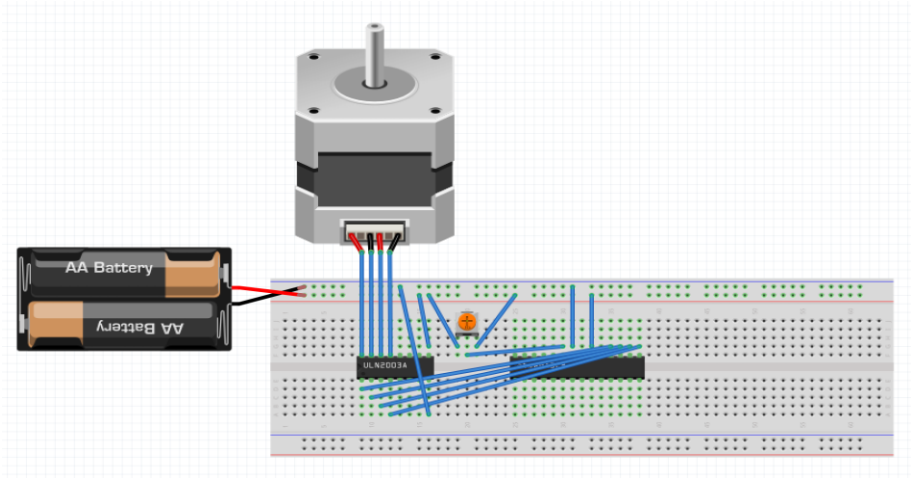
3. DEVELOPED MODIFIED CODE OF TASK 2/A

```
 /*
 * da_4b_task2.c
 *
 * Created: 4/19/2019 11:21:06 PM
 * Author : perezr1
 */

/*////////////////////////////// Include Library ////////////////////////////////////*/
#define F_CPU 16000000UL
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
/*////////////////////////////////////////////////////////////////////////////////*/

/*////////////////////////////////Function Prototype//////////////////////////////////*/
void adc_int(void);
/*////////////////////////////////////////////////////////////////////////////////*/

/*////////////////////////////// Variable ////////////////////////////////////////*/
int check_value = 0;
/*////////////////////////////////////////////////////////////////////////////////*/

int main(void)
{
/*////////////////////////////// Initializations ////////////////////////////////////*/
	DDRB = 0xFF; //DDRB as an output
	DDRD = 0xFF;
	TCCR1A|=(1<<COM1A1)|(1<<COM1B1)|(1<<WGM11); // Set OC1A/OC1B on Compare Match,
clear OC1A/OC1B at BOTTOM (inverting mode)
	TCCR1B|=(1<<WGM13)|(1<<WGM12)|(1<<CS11)|(1<<CS10); // set to Fast PWM mode & 64
prescaler
	ICR1=4999;
	adc_int(); // call function to initialize ADC
/*////////////////////////////////////////////////////////////////////////////////*/
	while (1)
	{
		ADCSRA |= (1 << ADSC); // start conversion
		while((ADCSRA & (1 << ADIF))== 0);
		check_value = ADCH; // temp value

		if(check_value == 0) // the minimum value
		{
			OCR1A = 0; //turn 0 degrees
			_delay_ms(1000);
		}
		if(check_value == 255) // the max value of the potentiometer
		{
			OCR1A = 535;  //servo motor will turn 180 degrees
			_delay_ms(1000);
		}
	}
}
```

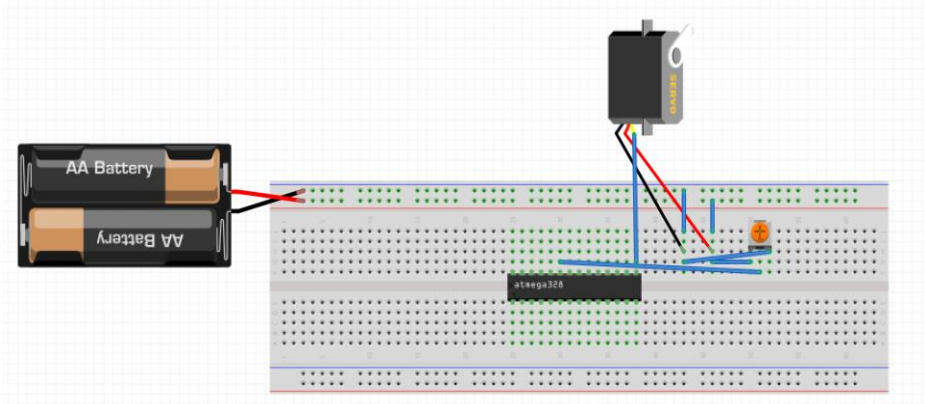
```
/*////////////////////////////////// Function ////////////////////////////////////*/
void adc_int(void)
{
	ADMUX = (0<<REFS1)| // Reference Selection Bits
	(1<<REFS0)| // AVcc-external cap at	AREF
	(1<<ADLAR)| // ADC Left Adjust Result
	(0<<MUX3)|
	(0<<MUX2)| // ANalogChannel Selection Bits
	(0<<MUX1)| // ADC0 (PC0)
	(0<<MUX0);

	ADCSRA = (1<<ADEN)| //ADC Enable
	(0<<ADSC)|
	(1<<ADATE)| // ADC Auto Trigger Enable
	(0<<ADIF)|
	(0<<ADIE)|
	(1<<ADPS2)| // ADC Prescaler Select Bits (64)
	(1<<ADPS1)|
	(0<<ADPS0);
}
/*////////////////////////////////////////////////////////////////////////////////*/
```

4. SCHEMATICS

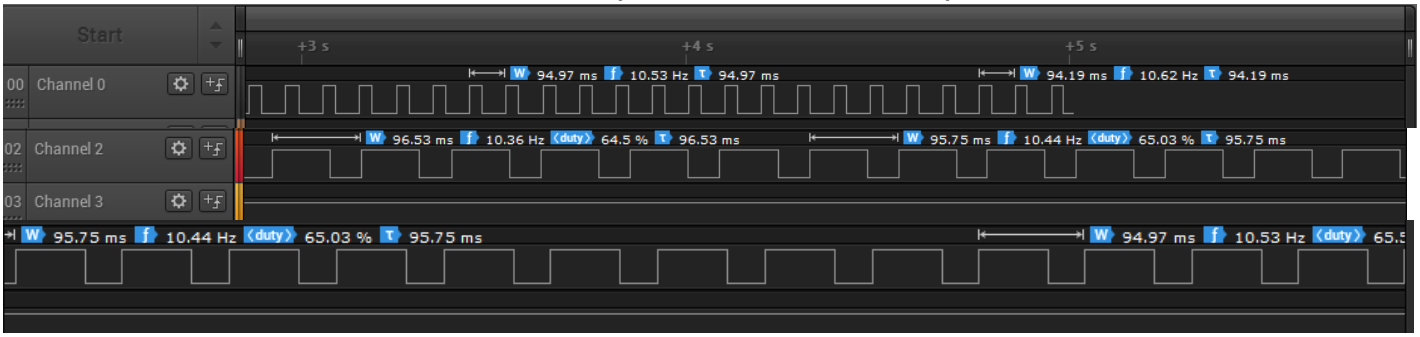
Stepper Motor



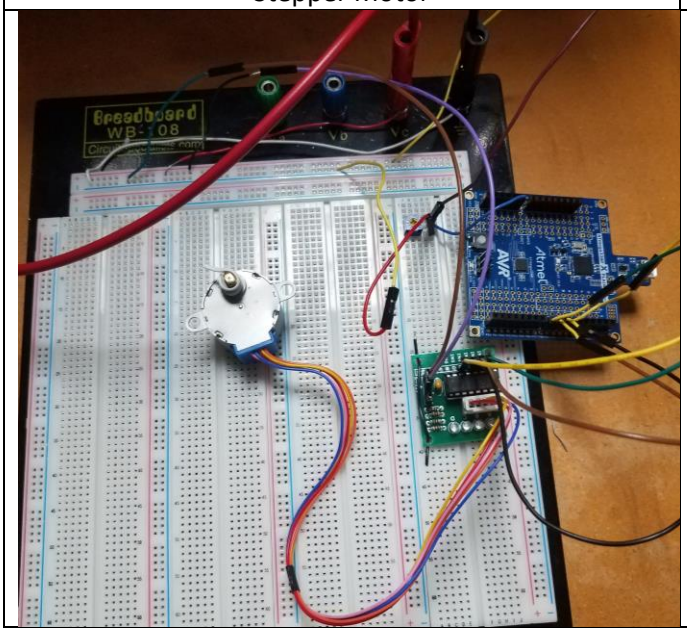
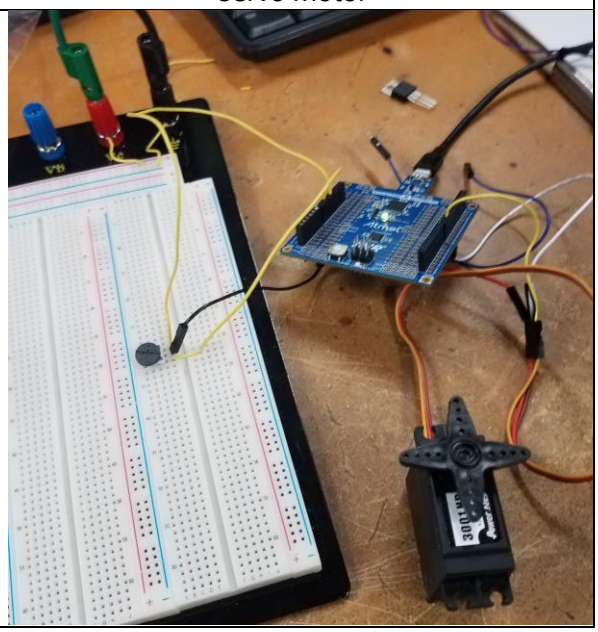
Servo Motor



5. SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)



6. SCREENSHOT OF EACH DEMO (BOARD SETUP)

Stepper Motor	Servo Motor
	

7. VIDEO LINKS OF EACH DEMO

Task 1: https://youtu.be/IctroFKpYUM
Task 2: https://youtu.be/V-Hrd6JQ5kI

8. GITHUB LINK OF THIS DA

https://github.com/RickyPerez79/submission_da.git

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

"This assignment submission is my own, original work".
RICKY PEREZ