CPE301 – SPRING 2019

Design Assignment 4B

Student Name: Ron Joshua Recrio
Student #: 5003825419
Student Email: recrio@unlv.nevada.edu
Primary Github address: https://github.com/recrio/submissions
Directory: /DesignAssignments/DA4A

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).

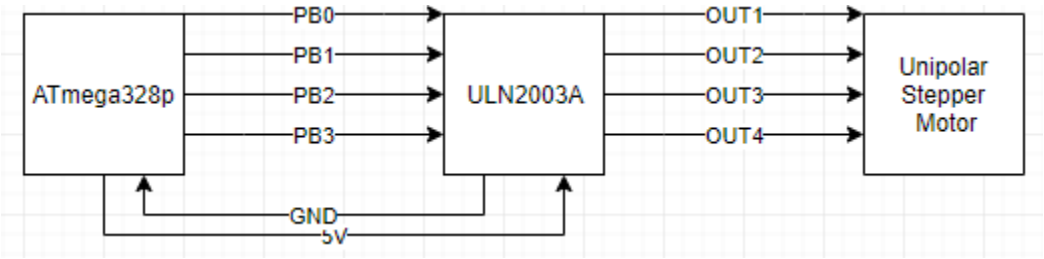
1. COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS

List of Components used:

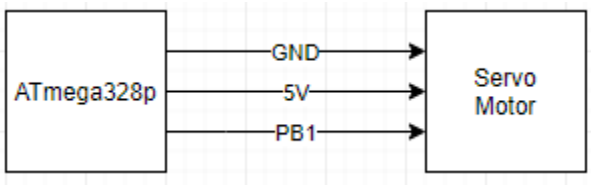
- ATmega328p Xplained Mini
- Multifunction Shield
- 28BYJ-48 (Stepper Motor)
- ULN2003A (Unipolar Stepper Motor Driver)

Block diagram with pins used in the Atmega328P

Task 1:



Task 2:



2. INITIAL CODE OF TASK 1/B

```
#define F_CPU 16000000UL
#define CONVERSION 10

#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>

void TIMER_init(void);
void ADC_init(void);

volatile int ADCvalue; // holds the value of ADC
int main(void)
{
	DDRB = 0x0F; // PORTB[3:0] set to output
	PORTB = 0; // Initially set output to 0
	ADC_init(); // initializes ADC
	TIMER_init(); // initializes TIMER

	while (1)
	{
		ADCSRA |= (1<<ADSC); // Start conversion
		while((ADCSRA&(1<<ADIF))==0); // while not done converting do nothing
		ADCSRA |= (1<<ADIF); // reset converter
		ADCvalue = ADC; // store ADC to ADCvalue for debugging
		OCR1A = CONVERSION*ADCvalue; // 224/1024 is the ratio of the speed so that
max speed is 95% of the pwm

		// Sequence
		PORTB = 0x09;
		while(!(TIFR1 & (1<<OCF1A)));   // Delays using CTC
		TIFR1 |= (1 << OCF1A);                    // Resets
		PORTB = 0x03;
		while(!(TIFR1 & (1<<OCF1A)));
		TIFR1 |= (1 << OCF1A);
		PORTB = 0x06;
		while(!(TIFR1 & (1<<OCF1A)));
		TIFR1 |= (1 << OCF1A);
		PORTB = 0x0C;
		while(!(TIFR1 & (1<<OCF1A)));
		TIFR1 |= (1 << OCF1A);

	}
}

void TIMER_init(void) {
	TCCR1B = (1<<WGM12) | // CTC mode
			(1<<CS11); // Prescaler 8
}

void ADC_init(void) {
	DDRC &= (0<<PORTC0); // PC0 as input
	PORTC |= (1<<PORTC1); // Pull up resistor
	ADMUX |= (1<<REFS0); // REFERENCE VOLTAGE AT AREF
	ADCSRA |=
	(1<<ADEN)| // ADC enable
```

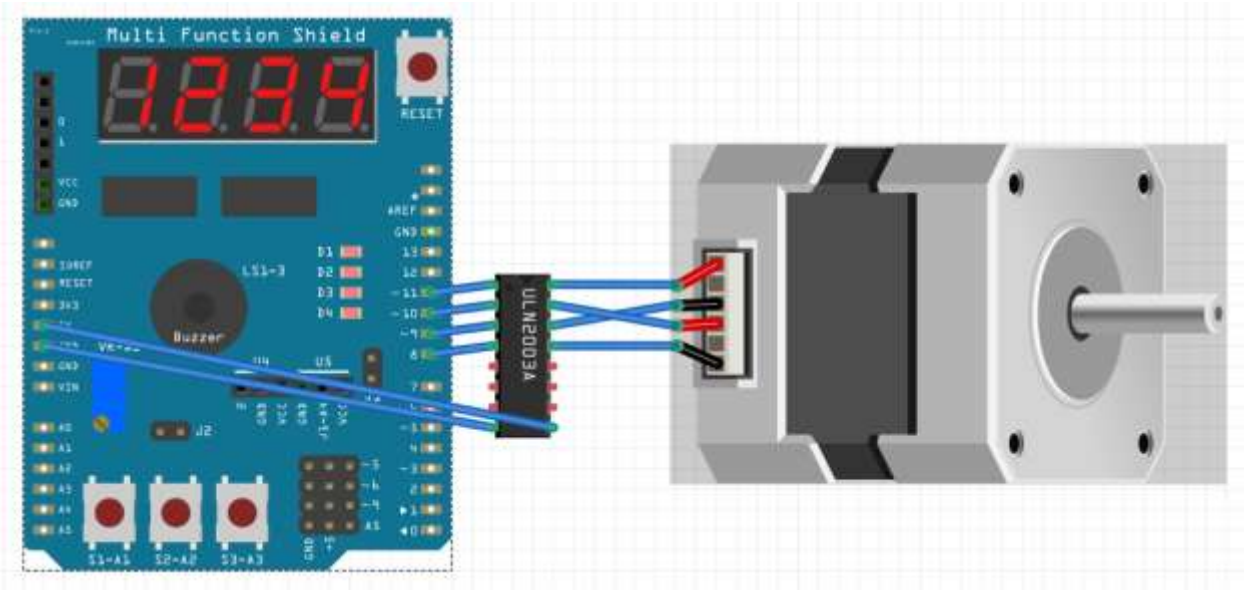
```
        (1<<ADPS2)| // ADC
        (1<<ADPS1)| // Prescaler
        (1<<ADPS0); // 128
        sei();
}
```

3. INITIAL CODE FOR TASK 2/B

```
#define F_CPU 16000000UL
#include <avr/io.h>
#include <util/delay.h>

void TIMER_init(void);
void ADC_init(void);

int main(void)
{
        ADC_init(); // initialize ADC settings
        TIMER_init(); // initialize TIMER1


        while(1)
        {
                ADCSRA |= (1 << ADSC);      // Starts conversion
                while((ADCSRA&(1<<ADIF))==0); // wait for conversion
                ADCSRA |= (1<<ADIF); // reset converter
                ICR1 = 4999;  // fPWM = 50Hz
                DDRB |= (1<<PB1);    // Set PB1 as output
                OCR1A = ADC;  // Adjusts pulse width range
                _delay_ms(100);      // Short Delay
        }
}

void TIMER_init(void)
{
        //Timer1
        TCCR1A = (1<<COM1A1) | // Clear OC1A on Compare non-inverting
                        (1<<COM1B1) | //
                        (1<<WGM11)  ; // Fast
        TCCR1B = (1<<WGM13) |  // PWM
                        (1<<WGM12) |  // TOP = ICR1
                        (1<<CS11) |  // Prescaler
                        (1<<CS10);     // 64
}

void ADC_init (void)
{
        ADMUX = (1<<REFS0);
        // Set Aref

        // right-justified data
        ADCSRA |= (1<<ADEN) |       // ADC enable
                        (1<<ADSC) | // AD start conversion
                        (1<<ADPS2)| // ADC
                        (1<<ADPS1)| // Prescaler
                        (1<<ADPS0); // 128
}
```

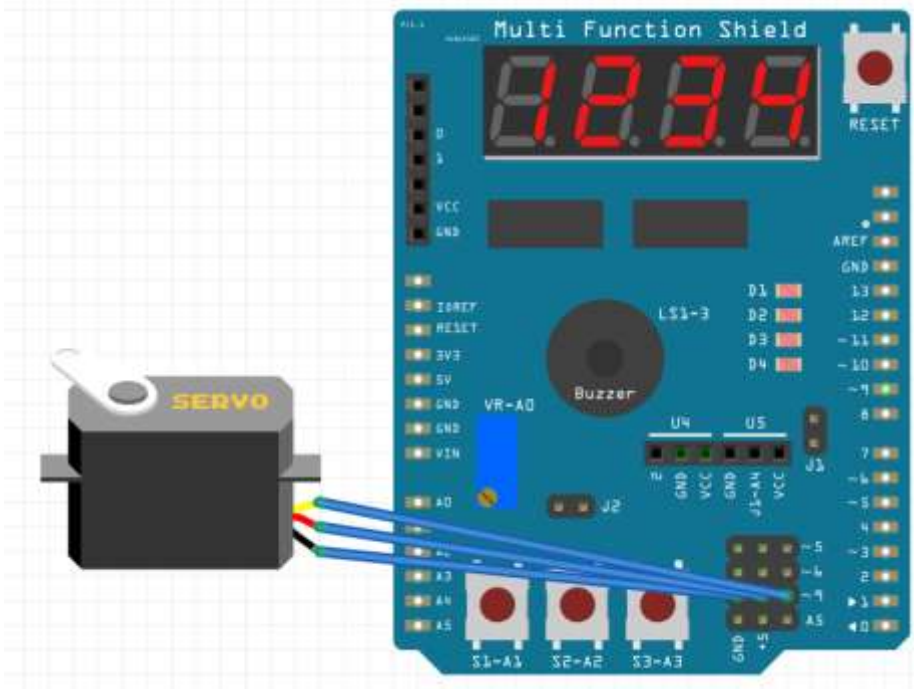
4. SCHEMATICS

Task 1:



*not exact models

Task 2:



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

Task1:

Before

OCR1A	0x127a
ADCvalue	0x01d9

After

OCR1A	0x0de8
ADCvalue	0x0164

Task2:

Before

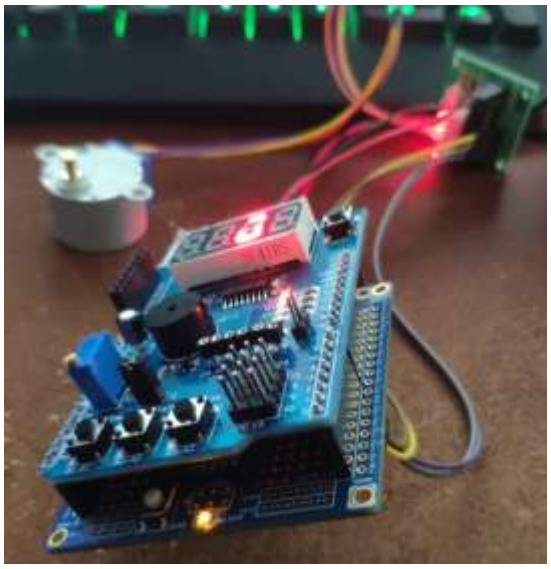
OCR1A	0x0176
ADC	0x0176

After

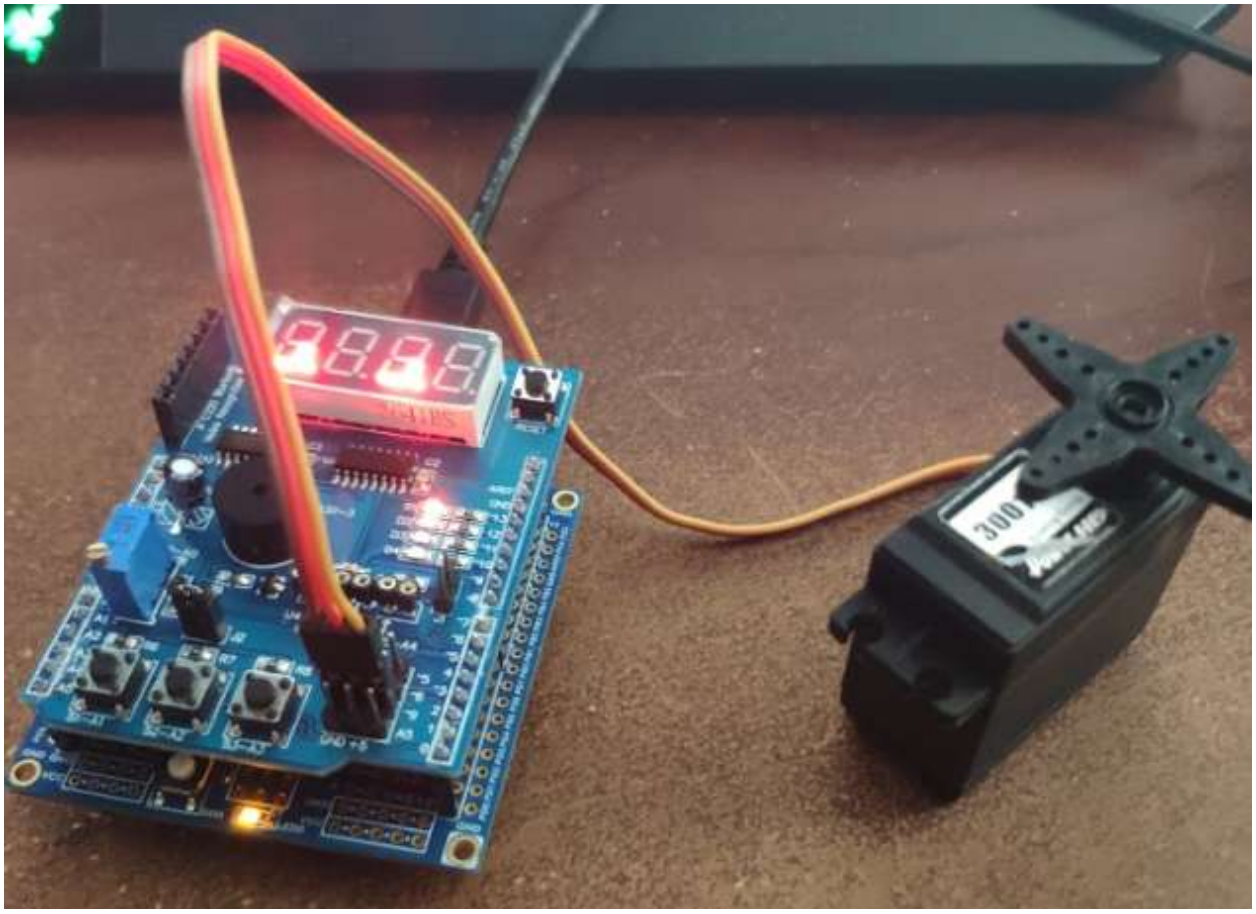
OCR1A	0x01d9
ADC	0x01d9

6. SCREENSHOT OF EACH DEMO (BOARD SETUP)

Task 1:



Task 2:



7. VIDEO LINKS OF EACH DEMO

Task 1:
https://youtu.be/ouQdYaP1QoE

Task 2:
https://youtu.be/pdVJZY0RRas

8. GITHUB LINK OF THIS DA

https://github.com/recrio/submissions/tree/master/DesignAssignments/DA4B

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

"*This assignment submission is my own, original work*".
Ron Joshua Recrio