CPE301 – SPRING 2019

Design Assignment 4, Part 2

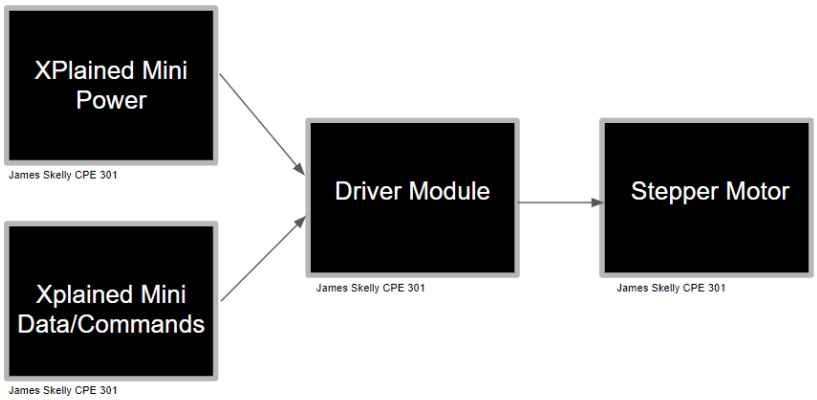
Student Name: James Skelly
Student #: 2000945485
Student Email: skellj1@unlv.nevada.edu
Primary Github address: https://github.com/skellj1/submission_da
Directory: skellj1/submission_da

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

Components used include Atmel Studio 7, stepper motor, servo motor, jumper wire, atmega328p, screwdriver, multifunction shield, iphone camera, driver module for stepper.



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

```
/*
 * DA4A_JS.c
 * Created: 4/11/2019 10:54:25 AM
 * Author : James Skelly
 */

// define clock for delay function, include headers
#define F_CPU 16000000UL
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
int adcVal;                     // holds value of ADC
char toggle = 0;                // initialize toggle variable

int main(void)
{
        DDRB = 0xFF;                    // PORTB as output
        DDRC = 0x00;                    // PORTC as input
        PORTB |= 0xFF;                  // Set all of PORTB high initially

        // set timer1 operation mode and prescaler
        TCCR1A |= (1<<COM1A1)|(1<<COM1B1)|(1<<WGM11);    // use PWM for output compare pins

        // OC1A/B, Fast PWM, Non-inverted mode
        TCCR1B |= (1<<WGM13)|(1<<WGM12)|(1<<CS11);              // timer1 pre-scaler set to 8
        ICR1 = 10000;                                           // set TOP value

        PORTC |= (1<<PORTC1);   // set portc initially high to wait for button press
        PCICR = (1<<PCIE1);     // set portc input capture enable bit
        PCMSK1 = (1<<PCINT9);   // enable pin change interrupt

        // Initialize ADC
        DIDR0 = 0x1;            // disable the digital input on ADC0 pin
        ADMUX = (1<<REFS0);     // set ADC reference pin to PC0 (Potentiometer)
        ADCSRA |= (1<<ADEN) | (1<<ADPS2) | (1<<ADPS1) | (1<<ADPS0);  // enable ADC using system clock
        ADCSRB = 0x0;           // free running mode

        sei();                          // globally enable interrupts

        while(1)        // wait for pin change interrupt request
        {
        }
}
ISR(PCINT1_vect){
        if(!(PINC & (1 << PINC2)))                              // if button is pressed...
        {
                _delay_ms(200);                                         // debounce button pressed
                while(!(PINC & (1 << PINC2)));

                if(toggle == 1){
                        PORTB |= (1 << PORTB1);         // set PB1 high
                        PORTB &= ~(1 << PORTB5);        // turn on LED D1
                        ADCSRA |= (1 << ADSC);          // start conversion
                        while((ADCSRA&(1<<ADIF))==0);   // wait for conversion to finish
                        adcVal = ADC & 0x03FF;          // extract right 10-bits of ADC register
                        OCR1A = 10*adcVal;              // OCR1A value for duty cycle
                }
                else if(toggle == 0){
                        OCR1A = 0;                      // set output compare value to 0
                        PORTB &= ~(1 << PORTB1);        // set PB1 low
                        PORTB |= (1 << PORTB5);         // turn of LED D1
                }

                toggle ^= 1;                            // toggle the variable used to toggle
        }
}
```

3. DEVELOPED MODIFIED CODE OF TASK 2/A from TASK 1/A

PART 1: Stepper Motor

```
/*
 * Stepper_PART1.c
 *
 * Created: 4/18/2019 6:22:27 PM
 * Author : James Skelly
 */
#define F_CPU 16000000UL
#include <avr/io.h>
#include <util/delay.h>

// Function Prototypes
void timer_init(void);
void adc_init(void);

int main(void)
{
	int period;    // integer used to set the period

	while (1)
	{
		timer_init();      // initialize and set up timer to be used
		adc_init();        // initialize and set up ADC
		DDRB=0x0F;   // Port B low bits to output
		OCR0A = ADC;  // set value of ADC data registers as compare value for
timer

// The following if statements each contain the same code with a different period,
// depending on the voltage read out of the potentiometer at PC0. The values that
// PORTB is set to (i.e. 0x09, 0x03, 0x06, and 0x0C) allow the motor to make steps
// and rotate at a speed specified by the potentiometer. Since all of this is inside
// the while(1) loop, the motor will keep spinning until OCR0A satisfies a different
// condition, in which case the motor will spin at a different speed.

		if ((OCR0A>=0)&&(OCR0A<=50))
		{
			period =2 ;
			PORTB = 0x09;
			_delay_ms(period);
			PORTB = 0x03;
			_delay_ms(period);
			PORTB = 0x06;
			_delay_ms(period);
			PORTB = 0x0c;
			_delay_ms(period);
			OCR0A = ADC;
		}
		else if ((OCR0A>50)&&(OCR0A<=120))
		{
			period=30;
			PORTB = 0x09;
```

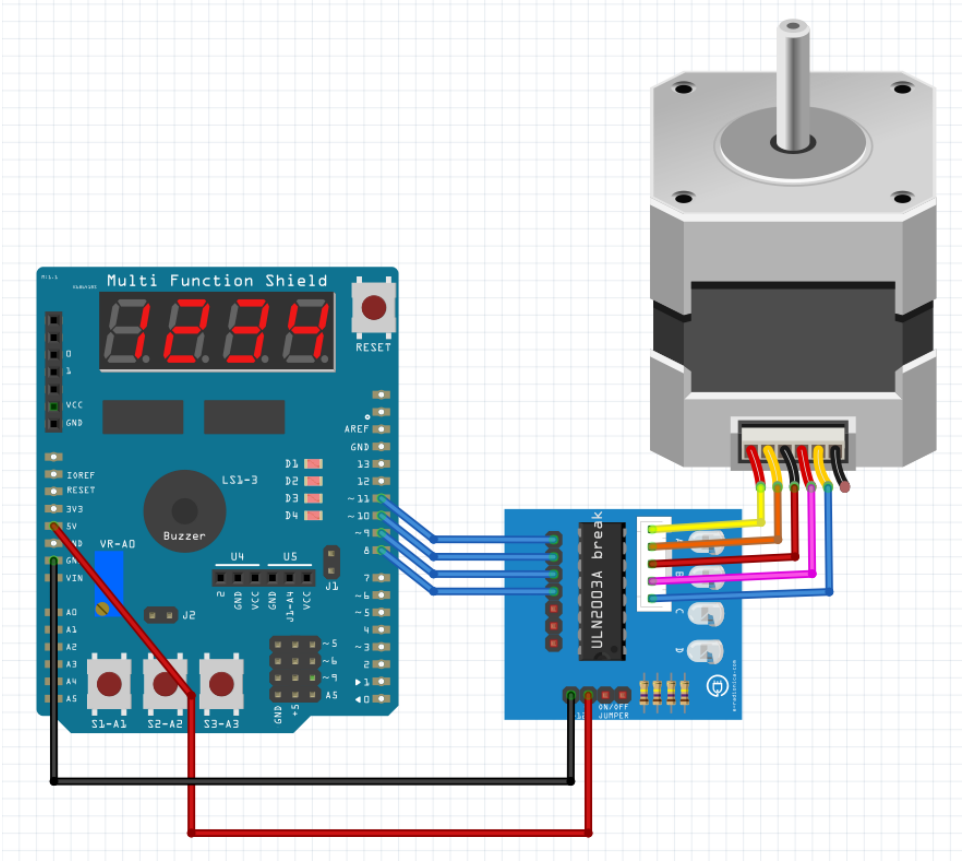
```
                _delay_ms(period);
                PORTB = 0x03;
                _delay_ms(period);
                PORTB = 0x06;
                _delay_ms(period);
                PORTB = 0x0c;
                _delay_ms(period);
                OCR0A = ADC;
            }
            else if ((OCR0A>120)&&(OCR0A<=150))
            {
                period=100;
                PORTB = 0x09;
                _delay_ms(period);
                PORTB = 0x03;
                _delay_ms(period);
                PORTB = 0x06;
                _delay_ms(period);
                PORTB = 0x0c;
                _delay_ms(period);
                OCR0A = ADC;
            }
            else if ((OCR0A>150)&&(OCR0A<=200))
            {
                period=200;
                PORTB = 0x09;
                _delay_ms(period);
                PORTB = 0x03;
                _delay_ms(period);
                PORTB = 0x06;
                _delay_ms(period);
                PORTB = 0x0c;
                _delay_ms(period);
                OCR0A = ADC;
            }
            else if (OCR0A>200)
            {
                PORTB = 0; // turn motor off when OCR0A approaches max value

                OCR0A = ADC;
            }
        }
}
void timer_init(void)
{
        // Clear OC0A on compare match, CTC mode, prescaler 256
        TCCR0A |= (1<<COM0A1)|(1<<WGM01);
        TCCR0B |= (1<<CS02);
}

void adc_init(void)
{
        ADCSRA |= ((1<<ADEN)|(1<<ADSC)|(1<<ADPS1)|(1<<ADPS0));
        ADMUX |= (1<<REFS0);
}
```

PART 2: Servo Motor

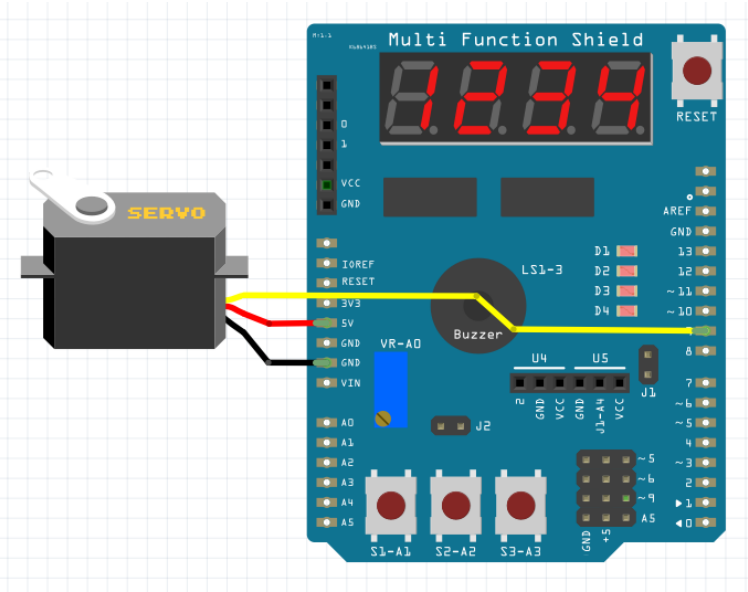
```
/*
 * Servo.c
 *
 * Created: 4/18/2019 9:13:12 PM
 * Author : James Skelly
 */
#define F_CPU 16000000UL
#include <avr/io.h>
#include <util/delay.h>

// Function Prototypes
void timer_init(void);   // timer1 setup function
void adc_init(void);  // ADC setup function

int main(void)
{
	while (1)
	{
		timer_init();		// initialize timer1
		adc_init();		// initialize ADC
		ICR1=5000;		// ICR is TOP for FAST PWM mode 14; set top to 5000
		DDRB |= (1<<PB1);	// PB1 output mode, because OC1A is internally tied to
PB1
		OCR1A=ADC;		// set ADC output value as compare register value
		_delay_ms(500);		// call for delay as motor moves from adjusting pot
value
	}
}

// Clear OC1A & OC1B on compare match, FAST PWM mode, prescaler of 64
void timer_init(void)
{
	TCCR1A |= (1<<COM1A1)|(1<<COM1B1)|(1<<WGM11);
	TCCR1B |= (1<<WGM13)|(1<<WGM12)|(1<<CS11)|(1<<CS10);
}

void adc_init(void)
{
	ADCSRA |= ((1<<ADEN)|(1<<ADSC)|(1<<ADPS1)|(1<<ADPS0));
	ADMUX |= (1<<REFS0);
}
```

4. SCHEMATICS

STEPPER MOTOR PART 1



SERVO MOTOR PART 2



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

```
/*
 * Servo.c
 *
 * Created: 4/18/2019 9:13:12 PM
 * Author : James Skelly
 */
#define F_CPU 16000000UL
#include <avr/io.h>
#include <util/delay.h>

// Function Prototypes
void timer_init(void);   // timer1 setup function
void adc_init(void);     // ADC setup function

int main(void)
{
    while (1)
    {
        timer_init();       // initialize timer1
        adc_init();         // initialize ADC
        ICR1=5000;          // ICR is TOP for FAST PWM mode 14; set top to 5000
        DDRB |= (1<<PB1);   // PB1 output mode, because OC1A is internally tied to PB1
        OCR1A=ADC;          // set ADC output value as compare register value
        _delay_ms(500);     // call for delay as motor moves from adjusting pot value
    }
}

// Clear OC1A & OC1B on compare match, FAST PWM mode, prescaler of 64
void timer_init(void)
{
```

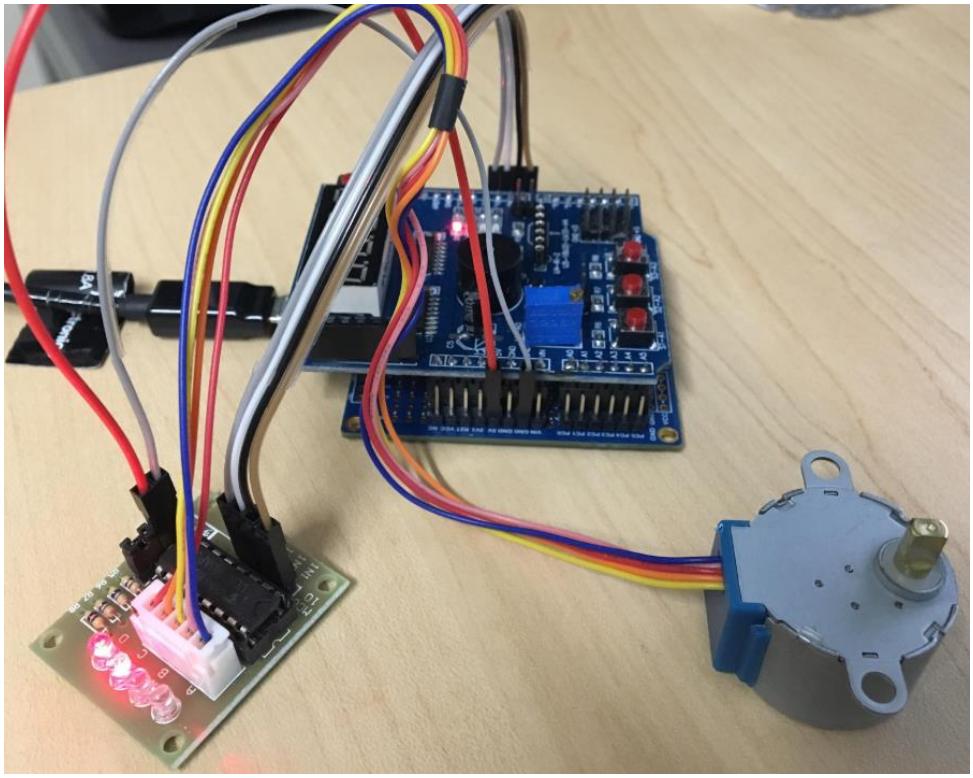
```
utput
how output from: Build
Target "PostBuildEvent" skipped, due to false condition; ('$(PostBuildEvent)' != '') was evaluated as ('' != '').
Target "Build" in file "C:\Program Files (x86)\Atmel\Studio\7.0\Vs\Avr.common.targets" from project "C:\Users\james\Documents\
Done building target "Build" in project "Servo.cproj".
Done building project "Servo.cproj".

Build succeeded.
========== Build: 2 succeeded or up-to-date, 0 failed, 0 skipped ==========
```

```
/*
 * Stepper.c
 *
 * Created: 4/20/2019 6:22:27 PM
 * Author : James Skelly
 */
#define F_CPU 16000000UL
#include <avr/io.h>
#include <util/delay.h>

// Function Prototypes
void timer_init(void);
void adc_init(void);

int main(void)
{
    int period;    // integer used to set the period

    while (1)
    {
        timer_init();      // initialize and set up timer to be used
        adc_init();        // initialize and set up ADC
        DDRB=0x0F;         // Port B low bits to output
        OCR0A = ADC;       // set value of ADC data registers as compare value for timer

        // The following if statements each contain the same code with a different period,
        // depending on the voltage read out of the potentiometer at PC0. The values that
        // PORTB is set to (i.e. 0x09, 0x03, 0x06, and 0x0C) allow the motor to make steps
        // and rotate at a speed specified by the potentiometer. Since all of this is inside
        // the while(1) loop, the motor will keep spinning until OCR0A satisfies a different
```

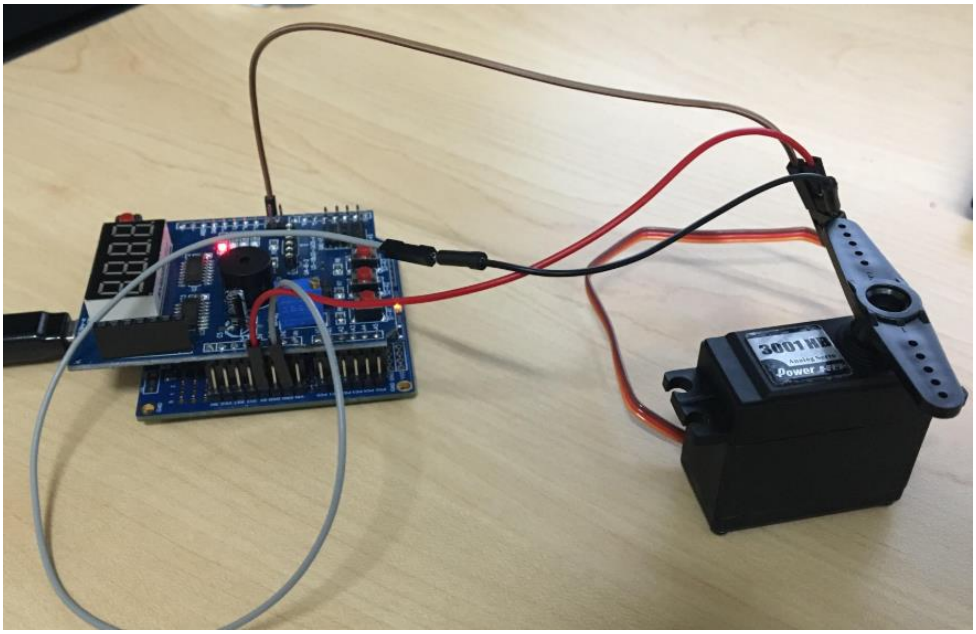
```
utput
how output from: Build
   Done executing task "RunOutputFileVerifyTask".
Done building target "CoreBuild" in project "Servo.cproj".
Target "PostBuildEvent" skipped, due to false condition; ('$(PostBuildEvent)' != '') was evaluated as ('' != '').
Target "Build" in file "C:\Program Files (x86)\Atmel\Studio\7.0\Vs\Avr.common.targets" from project "C:\Users\james\Documents
Done building target "Build" in project "Servo.cproj".
Done building project "Servo.cproj".

Build succeeded.
========== Build: 2 succeeded or up-to-date, 0 failed, 0 skipped ==========
```

6. SCREENSHOT OF EACH DEMO (BOARD SETUP)

STEPPER SETUP



SERVO SETUP



7. VIDEO LINKS OF EACH DEMO

Stepper: https://www.youtube.com/watch?v=3mt-dW8fx8I

Servo: https://www.youtube.com/watch?v=HFwg0y40PsQ

8. GITHUB LINK OF THIS DA

https://github.com/skellj1/submission_da

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

"*This assignment submission is my own, original work*".
James W. Skelly