Phillip Sortomme

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

experiment 1:
ATMega328P
push button
1k resistor
L293D motor driver
DC motor
Potentiometer

experiment 2:
ATMega328P
ULN2003
stepper motor
Potentiometer

experiment 3:
ATMega328P
servo motor
Potentiometer

2. DEVELOPED CODE OF TASK 1

```
#define F_CPU 1000000UL
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>

void adc_int(void);
volatile unsigned int speed;
volatile unsigned int stop = 0;

ISR(INT0_vect){
   stop ^= 1;          // toggles stop on interrupt
}

int main()
{
   DDRD = 0xFB;        // set motor outputs to PD0, PD1. leave PD2 as input for push button.
   EIMSK = 0x01;       // enable INT0
   EIFR = 0x01;        // enable interrupt flag 0
   EICRA = 0X03;       // set interrupt on rising edge
   sei();
   adc_int();
   TCCR0B=3;           // set prescaler to 1024
   TCCR0A=0x83;        // set fast PWM and clear OCR0A on match
   while (1)
   {
      while((ADCSRA&(1<<ADIF))==0);
      speed = ADC*95/400;         // speed equals the conversion for ADC to PWM = adc/4 *.95
      OCR0A = speed;
      if(stop == 0){
         PORTD = 0x01;   // make motor rotate clockwise
```

```
      }
      else
         PORTD = 0X00;
   }
}

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
   (0<<ADIE)|//    ADC Interrupt   Enable
   (1<<ADPS2)|//   ADC PrescalerSelect Bits
   (1<<ADPS1)|
   (1<<ADPS0);
}
```

3. DEVELOPED CODE OF TASK 2

```
#define F_CPU 1000000UL
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>

void adc_int(void);
void timer_init(void);
volatile unsigned int speed; // variable used to control delay
volatile int stop = 0; // if set to 1, the motor is turned off

int main(void)
{
   DDRD = 0xFF;          //Enable output on all of the B pins
   PORTD = 0x00;           // Set them all to 0v
   adc_int();
   TCCR1B = 0x0D;
   while(1){
// Convert the ADC value to a speed to control the motor. motor stops if ADC value is greater then 1015
      if     (ADC <= 4)       {stop = 0; speed = 1;}
      else if (ADC <= 85)  {stop = 0; speed = 2;}
      else if (ADC <= 170) {stop = 0; speed = 3;}
      else if (ADC <= 255) {stop = 0; speed = 4;}
      else if (ADC <= 340) {stop = 0; speed = 5;}
      else if (ADC <= 425) {stop = 0; speed = 6;}
      else if (ADC <= 510) {stop = 0; speed = 7;}
      else if (ADC <= 595) {stop = 0; speed = 8;}
      else if (ADC <= 680) {stop = 0; speed = 9;}
      else if (ADC <= 765) {stop = 0; speed = 10;}
      else if (ADC <= 850) {stop = 0; speed = 11;}
      else if (ADC <= 935) {stop = 0; speed = 12;}
      else if (ADC <= 1015) {stop = 0; speed = 13;}
      else               {stop = 1;}

      OCR1A = speed;     // set OCR1A to the determined speed
      TCNT1 = 0x00;      // reset the clock
```

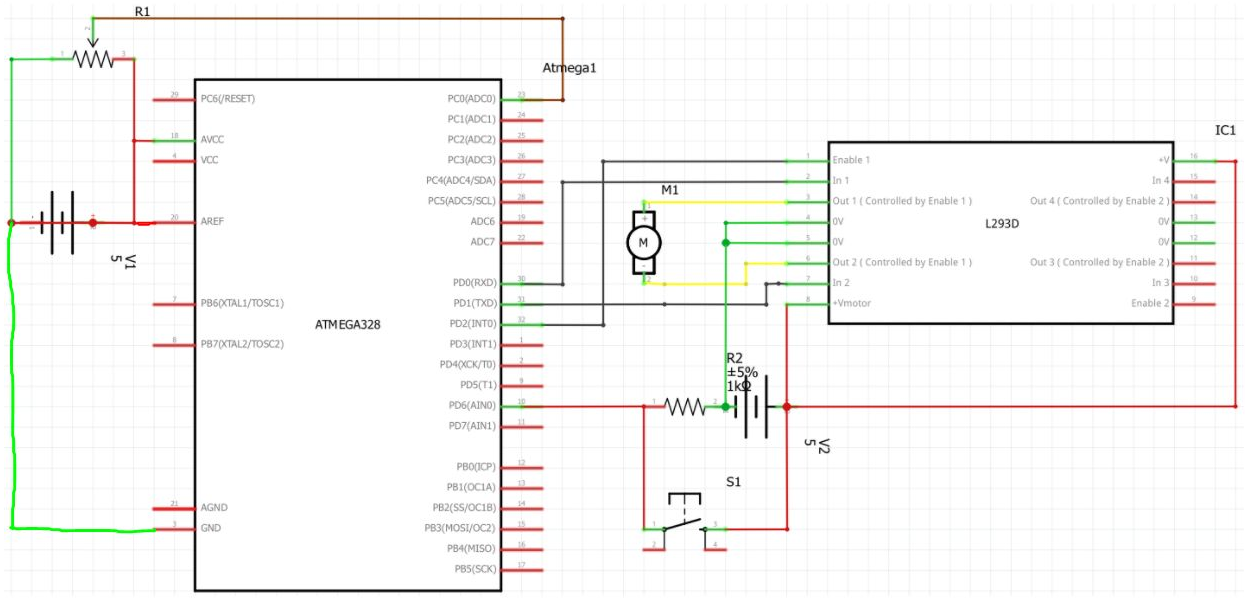
```
      if(stop == 0){
         // if the motor is not to be halted, run a step with the designated lenght delay
         while((TIFR1 & 0x2) != 0x2);
            PORTD = 0x06;
            TIFR1 |= (1<<OCF1A);
         while((TIFR1 & 0x2) != 0x2);
            PORTD = 0x0C ;
            TIFR1 |= (1<<OCF1A);
         while((TIFR1 & 0x2) != 0x2);
            PORTD = 0X09;
            TIFR1 |= (1<<OCF1A);
         while((TIFR1 & 0x2) != 0x2);
            PORTD = 0X03;
            TIFR1 |= (1<<OCF1A);
      }
   }
}

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 PrescalerSelect Bits
   (1<<ADPS1)|
   (1<<ADPS0);
}
```

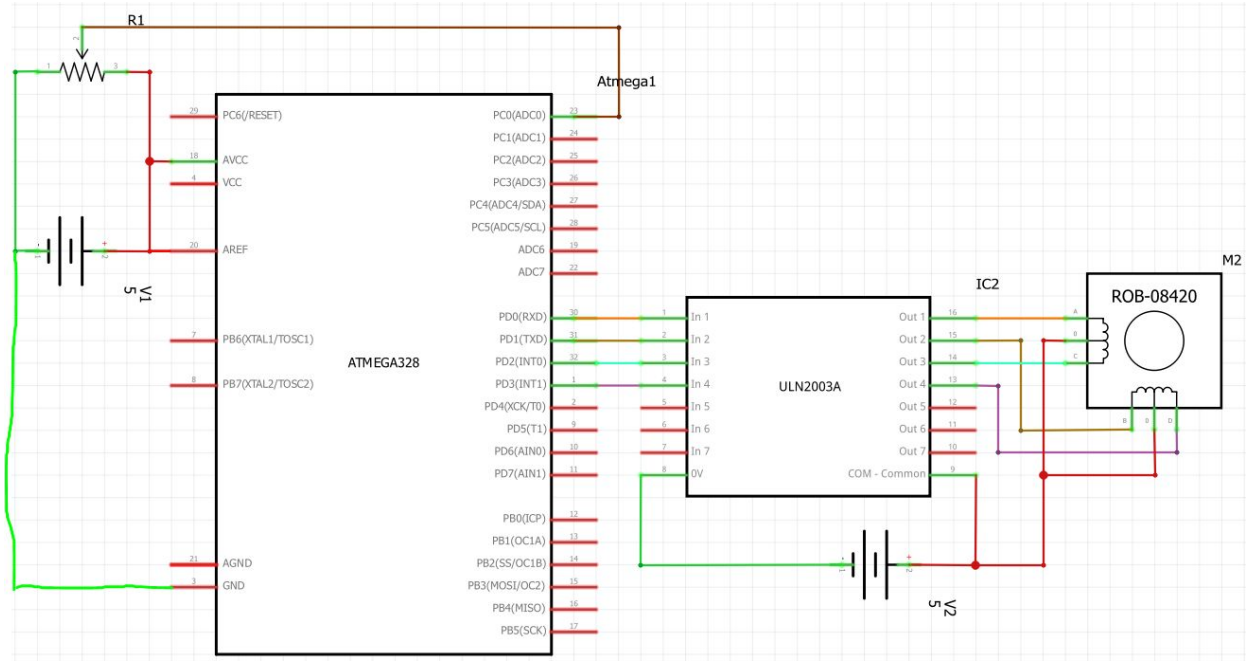
4. DEVELOPED CODE OF TASK 3

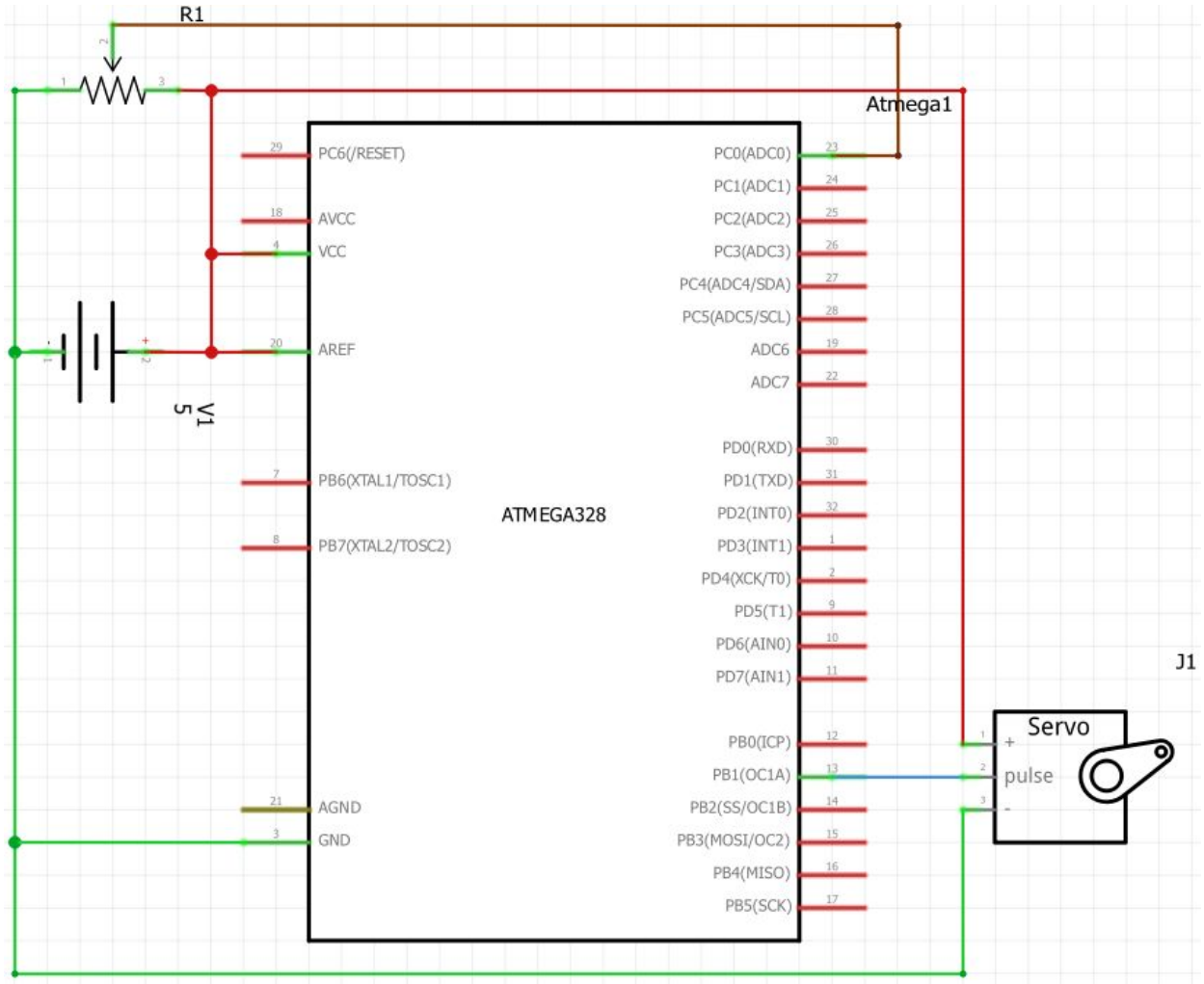
```
#define F_CPU 1000000UL
#include <avr/io.h>
#include <util/delay.h>

void adc_int(void);
volatile unsigned int rotate;
int main(void)
{
   DDRD = 0xFF;
   TCCR0B=3;
   TCCR0A=0x83;
   adc_int();
   while (1)
   {
      while((ADCSRA&(1<<ADIF))==0);
// for the Servo Motor used (3001HB) I found 0 degree mapped to 0,(I mapped it to 1 however due to it sometimes trying to move
too far)
// and 180 degree was approxamatly 32. To make the conversion,  divide ADC by 33 and add 1
      OCR0A = (ADC /33 +1);
   }
}

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 PrescalerSelect Bits
  (1<<ADPS1)|
  (1<<ADPS0);
}
```

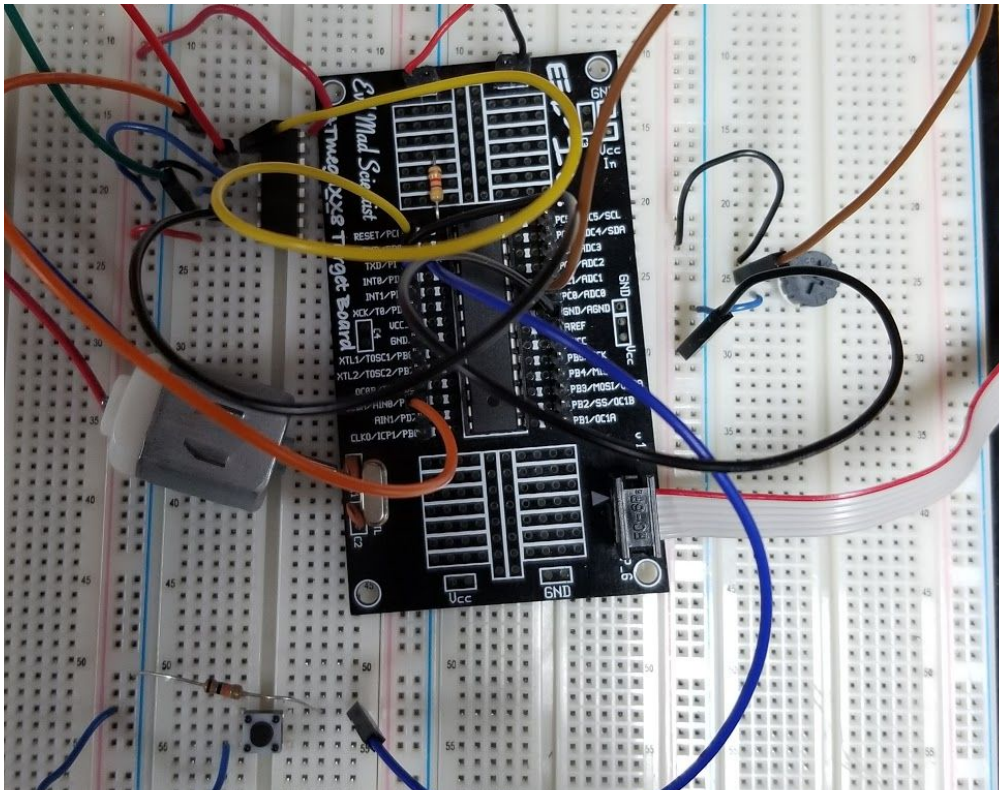
5. SCHEMATICS

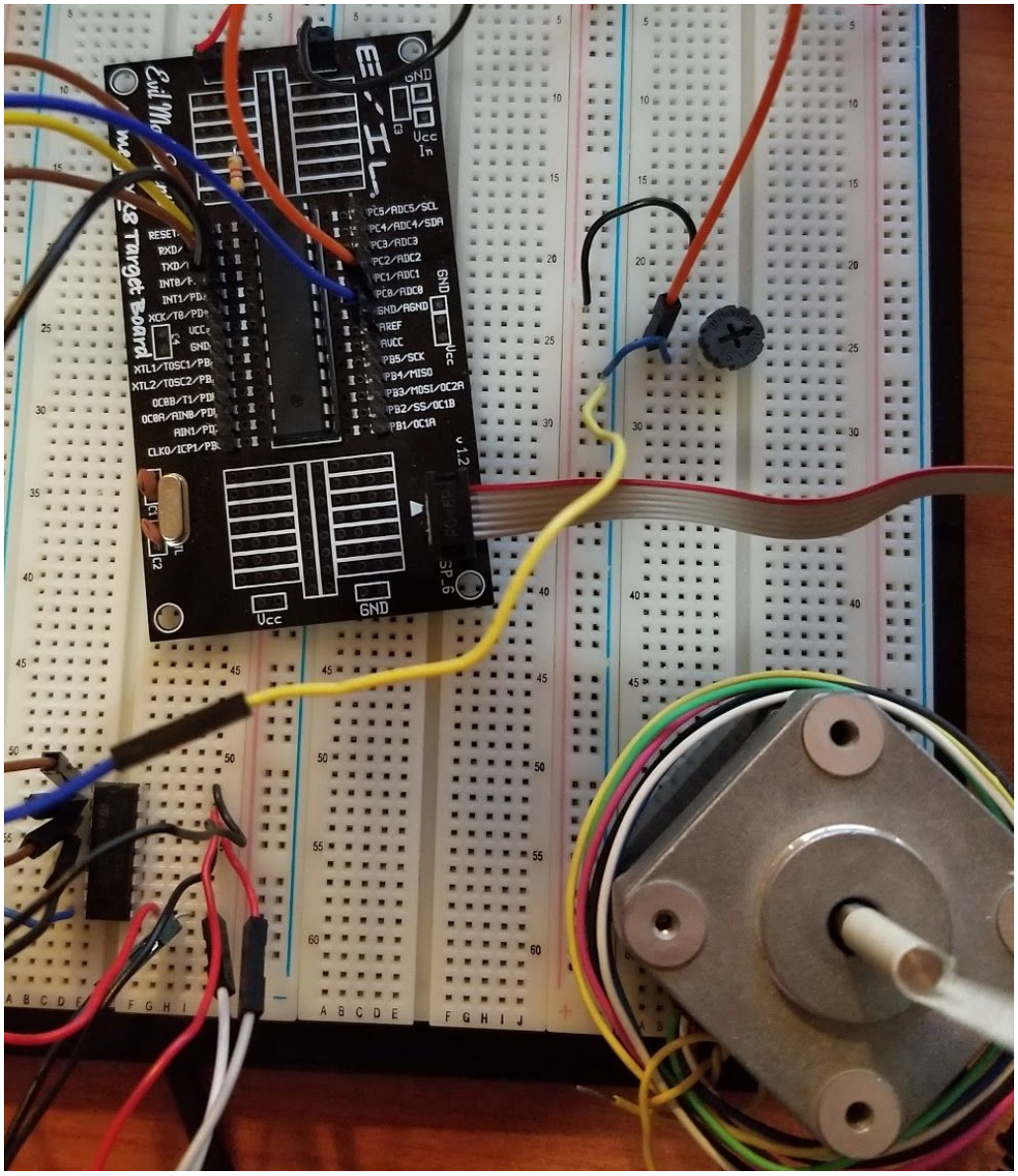


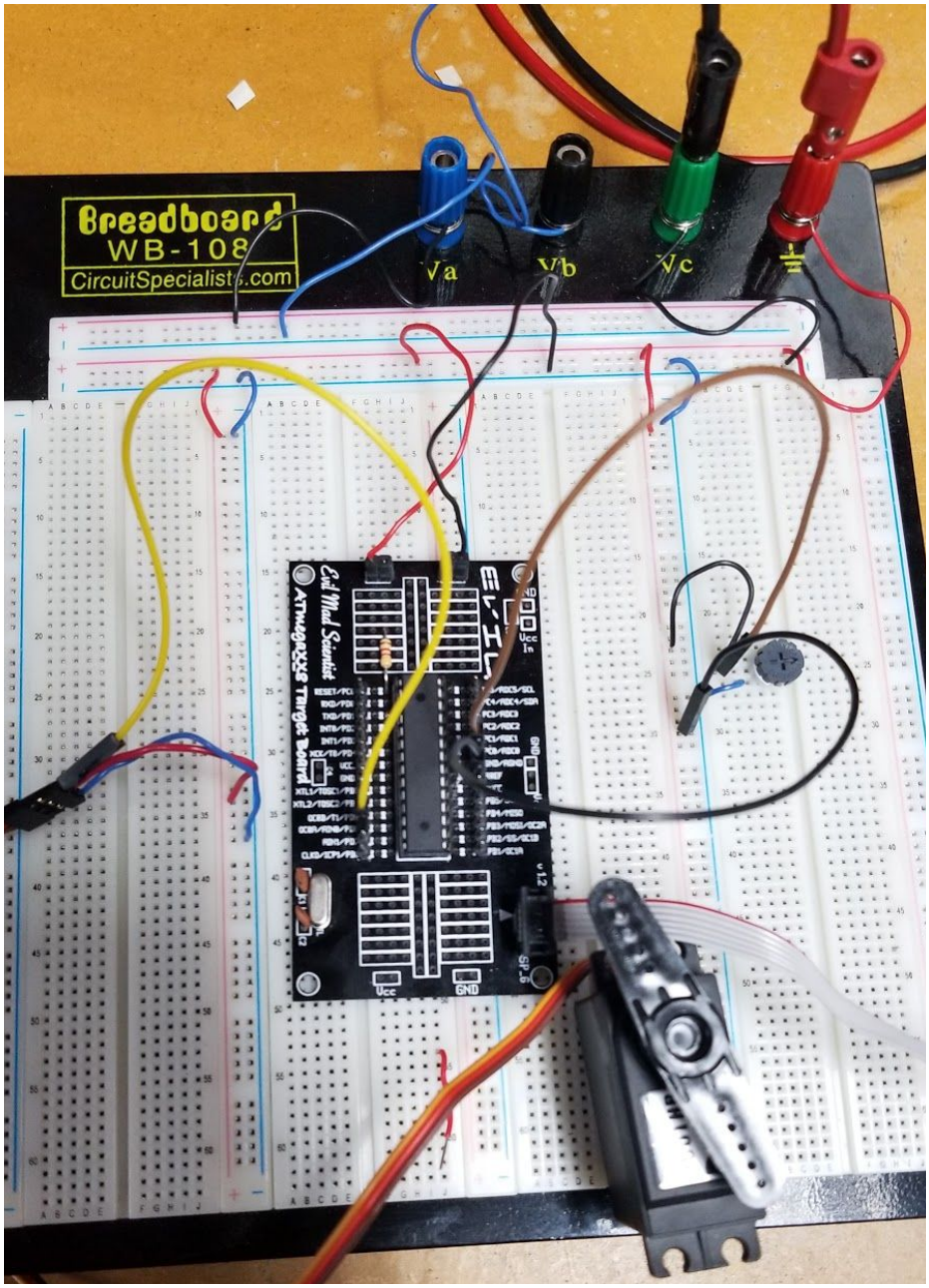

R1
Atmega1
PC6(/RESET)
AVCC
VCC
AREF
AGND
GND
PB6(XTAL1/TOSC1)
PB7(XTAL2/TOSC2)
PC0(ADC0)
PC1(ADC1)
PC2(ADC2)
PC3(ADC3)
PC4(ADC4/SDA)
PC5(ADC5/SCL)
ADC6
ADC7
PD0(RXD)
PD1(TXD)
PD2(INT0)
PD3(INT1)
PD4(XCK/T0)
PD5(T1)
PD6(AIN0)
PD7(AIN1)
PB0(ICP)
PB1(OC1A)
PB2(SS/OC1B)
PB3(MOSI/OC2)
PB4(MISO)
PB5(SCK)
ATMEGA328
V1
5
IC2
ULN2003A
In 1
In 2
In 3
In 4
In 5
In 6
In 7
0V
Out 1
Out 2
Out 3
Out 4
Out 5
Out 6
Out 7
COM - Common
V2
5
M2
ROB-08420


R1
Atmega1
PC6(/RESET)
AVCC
VCC
AREF
AGND
GND
PB6(XTAL1/TOSC1)
PB7(XTAL2/TOSC2)
PC0(ADC0)
PC1(ADC1)
PC2(ADC2)
PC3(ADC3)
PC4(ADC4/SDA)
PC5(ADC5/SCL)
ADC6
ADC7
PD0(RXD)
PD1(TXD)
PD2(INT0)
PD3(INT1)
PD4(XCK/T0)
PD5(T1)
PD6(AIN0)
PD7(AIN1)
PB0(ICP)
PB1(OC1A)
PB2(SS/OC1B)
PB3(MOSI/OC2)
PB4(MISO)
PB5(SCK)
ATMEGA328
V1
5
J1
Servo
+
pulse
-

6. SCREENSHOT OF EACH DEMO (BOARD SETUP)


Vcc
GND
RESET/PC
TXD/PD
INT0/PD
INT1/P
XTL1/TOSC1/PB
XTL2/TOSC2/PB
AIN1/PD
CLKO/ICP1/PB
PC5/SCL
PC4/SDA
PC3/ADC3
PC2/ADC2
PC1/ADC1
PC0/ADC0
GND/AGND
AREF
PB3/MOSI/
PB2/SS/OC1B
PB1/OC1A
Vcc
GND


GND
Vcc
In
PC5/ADC5/SCL
PC4/ADC4/SDA
PC3/ADC3
PC2/ADC2
PC1/ADC1
PC0/ADC0
GND/AGND
AREF
AVCC
PB5/SCK
PB4/MISO
PB3/MOSI/OC2A
PB2/SS/OC1B
PB1/OC1A
v.12
Vcc
GND



7. GITHUB LINK OF THIS DA

https://github.com/Pogoptomus/CPE301/tree/master/DA4

Student Academic Misconduct Policy

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

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

Phillip SortommeT