Design Assignment 4

CpE 301 **Due: Monday, Feb. 27th, 2017 by 11:59pm** **Spring 2017 (Dr. Harris)**

Description

The goal of the assignment is to write two programs that do the following:

1. DA4_1: Drive a **DC motor** – and then extend the program to control the **speed and direction** of the DC motor using a potentiometer.
2. DA4_2: Drive a **servo motor** – and then extend the program to control the **position** of the servo motor using a potentiometer.

In further detail, the programs should work as follows:

1. **DA4_1: DC motor**
    a. If the value read from the potentiometer is above the mid-point (i.e., 2.5 V for a 5V system or 1.65 V for a 3.3 V system), the motor should rotate forward. The higher the voltage is above its mid-point, the faster the motor should rotate forward until its speed maxes out at the largest value (i.e., 3.3 or 5V, depending on your system).
    b. If the value read from the potentiometer is below the mid-point, the motor should rotate backward, with increasing speed as the value read moves closer to 0 V. At 0V, the motor should rotate backward at the maximum speed.
2. **DA4_2: Servo motor**
    The servo motor should rotate from 0 to 180 degrees corresponding to voltages read from the potentiometer of, respectively, 0 to 5 V (or 3.3 V if you're working with a 3.3 V system).

Additional hints and sample code are available at the end of this document.

What to Turn In

The following must be submitted via WebCampus by the due date/time to receive credit for the assignment. Messy, difficult to understand, or disorganized work will receive no credit.

Total points available: 100

0. **Time:** Indicate the amount of time this assignment took in hours. This will not affect your grade (unless omitted) but will help gauge the workload for this and future semesters. **[-5 points if omitted]**
1. **A pdf or word document** that contains details about your design. Be sure to submit them **in this order**:
    a. A 1-paragraph description of each of the two designs (only of the "extended" portion). **[20 pts]**
    b. Your two programs, which must have been built (i.e., compiled/assembled) and working. The code should be clear, well-formatted, and well-documented. **[40 pts]**

c. A flow chart of your programs (hand-drawn is fine as long as it's legible). **[15 pts]**
d. A schematic of your hardware (again, hand-drawn is fine as long as it's legible). **[15 pts]**
e. A link to a ~1-2 minute Youtube video showing your programs working in hardware. **[10 pts]**

Additional Hints and Sample Code

Servo wiring: The orange wire on the servo receives the control (PWM) signal. The red wire is power (5 V) and the brown wire is ground (GND). **Warning:** Do not turn your servo by hand, especially when power is applied; you may damage it. Before you run the code, visually inspect that the servo will be able to turn to the left and right freely, without bumping against wires. Each time you turn on the Arduino board, the servo may twitch and move to a new position.

Hint: you can use Timer1 to create a 50 Hz signal for your PWM. You can use the ICR1 register to set the TOP for Timer1.

A/D code: You can use the following A/D program as a starting point to write code that reads the potentiometer values. Don't forget to connect the following pins on the ATmega328p: AVCC and AREF to VCC and AGND to GND.

```
#include <avr/io.h>

int main(void) {
	DDRB = 0xFF;		// make Port B an output
	DDRD = 0xFF;		// make Port D an output
	DDRC = 0x0;		// make Port C an input (for ADC input)
	DIDR0 = 0x1;		// disable digital input on ADC0 pin
	ADMUX = 0x0;		// Reference = Aref, ADC0 (PC.0) used as analog input
				// data is right-justified
	ADCSRA = 0x87;		// enable ADC, system clock used for A/D conversion
	ADCSRB = 0x0;		// free running mode

	while (1) {
		ADCSRA |= (1 << ADSC);			// start conversion
		while ( (ADCSRA&(1<<ADIF)) == 0 ) ;	// wait for conversion to finish
		PORTD = ADCL;				// send low byte to PORTD
		PORTB = ADCH;				// send high byte to PORTB
	}
	return 0;
}
```