Lab 10: **30 points.**

Objective: Develop an understanding of the Digital to Analog Converters (DAC's), and the effect of the RC circuit and the period of the PWM on the output waveform.

Description of Lab 10: In this lab we will be trying to create a voltage, using the Pulse Width Modulation (PWM) output pins on the Arduino. As I stated in class, the analogWrite() function supplied by Arduino has a couple short comings. Thus we will be using the Timer1 system, that has been installed in the lab, to generate the PWM for our experiment.

For this lab the first thing to do is hook up the circuit shown in Figure 10-1. The connection to the Analog Discovery is similar to that done in Lab 9, except now we are watching the PWM signal and the output from the RC circuit. Also, note that no values were given for the R and C in the drawing. That is because we will be changing them through the course of the experiment. Initially they should be set to a 5 K Ω (Green, Black, Red color coding) and a 0.1 μF (104 marking) capacitor.

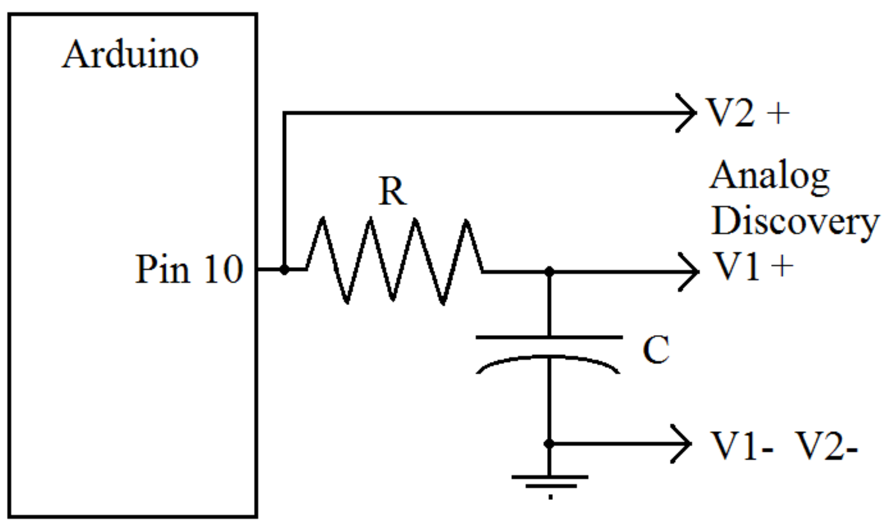

Figure 10-1. Analog Output.

The objective will be to generate a waveform that is close to a sine wave on V1. For this we will use the following equations

```
Time += 5e-3;  // Advance time by sample interval
if( Time > Period ) // if time goes past Period = 1/frequency
    Time -= Period;  // Wrap by Period, not allowing Time to grow to large.
// Using time and frequency generate next sine value
// and shifting it to go from approximately 1 to 1023
x = 511*sin( 6.2831853*Frequency*Time) + 512;
```

One last point, in the program described below it calls for the LED or pin 13 to be turned on before the sampling operation and off after it is complete. This signal from the Arduino allows us to monitor the sampling rate we are actually getting from the Arduino.

In the later part of the lab we will be driving an LED with the PWM signal. Thus the following circuit will be used.

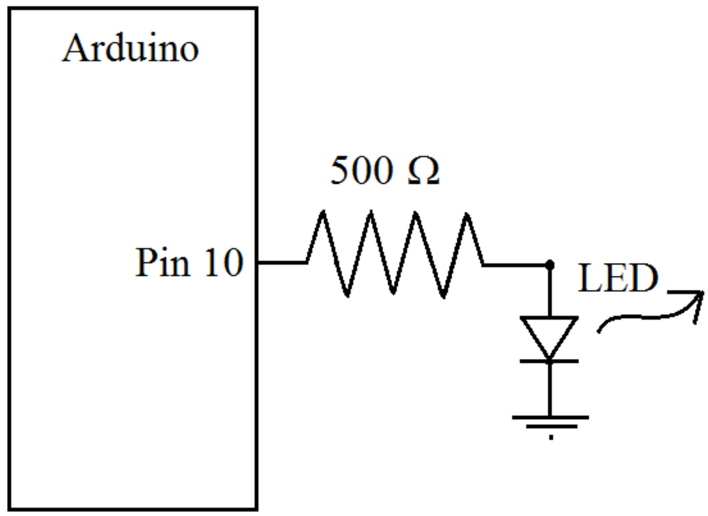

Figure 10-2. Circuit for LED Output.

Documenting the results of your experiments with screen shots will be a critical part of this lab!

Lab Assignment: The code for this lab is very simple. It is intended to generate a new value for the sine wave every 5 milliseconds and send it out as a PWM signal on pin 10. Note we will be using the Timer1 library to generate the PWM output and MsTimer2 to run an ISR ever 5 milliseconds. A demo of how to use Timer1 the can be found on the Files->CourseNotes->Hardware Documentation->I_PWM_Demo.pdf.

#include <math> // Needed for sin function.
#include <TimerOne.h>
#include <MsTimer2.h>

global variables of Time, Frequency and Period

ISR:
    Turn on the LED (pin 13)
    Generate next Time, and then next output value x
    Send x out as a PWM signal on pin 10.
    Turn off the LED

Setup:
    Set up Timer2 to run an ISR every 5 milliseconds
    Set Timer1 to have a 500 microseconds period
        and a PWM on pin 10.
Loop:
    Nothing is here.

Prelab: Write the program described above and upload it to the Prelab 10 location.

Lab 10:

1) Setup the circuit in Figure 10-1, using a 5 K Ω resistor and a 0.1 μF capacitor. Set frequency to 5 Hertz (cycles / second), which is commonly written as 5 Hz. Set up the Analog Discovery as shown in Figure 10-1, then capture and document the waveform generated. Demonstrate this to your lab instructor.

2) Change the capacitor to 0.2 μF's and capture and document the output.

3) Connect V2 of the Analog Discovery, to pin 13 (the LED) and measure the number of pulses per second (it should be close to 200) and the length of the average pulse. Be sure to document the results of these measurements.

   With the system running, let the Analog Discovery run and watch to see if the length of the ISR's run time (the time the LED pin is high) varies or is consistent. Record your observation.

4) Replace the RC circuit with the LED circuit shown in Figure 10-2. Then set the frequency to 0.5 Hz (that is one half of a Hertz). Then demonstrate it to a lab instructor.

Questions:

1) Which capacitor value do you think gave a "better" sine wave?

2) How close to the desired 200 samples per second or 5 millisecond sampling interval did you achieve?