Overview

This exercise introduces

- the Arduino prototyping platform
- the structure of the AVR ATMEGA168 microcontroller
- the Arduino programming language (based on C but with important extensions and limitations)
- the Arduino Integrated Development Environment (IDE)
- analog and digital input and output with the Arduino Duemilanove
  - Pulse Width Modulation (PWM) analog output
- serial communication with ASCII characters.

Before coming to lab, you should, to the extent possible, review the following brief references, which are linked on the Physics 116B web site:

- Arduino home page: http://www.arduino.cc/
In the lab, you will work with an Arduino Duemilanova board containing an Atmel ATMEGA168 microcontroller. Figure 1 shows an Arduino Duemilanove attached to a computer by a USB connection. See the first link above for a summary of its features. It has 6 ADC channels hooked to analog input pins and 6 pins which can provide PWM output. The USB cable is used to provide power and load programs to the board from a PC (Linux, Mac or Windows). The microcontroller can also communicate with the computer over the USB connection during operation and display output in a computer window. Once programmed, the Arduino can run without the computer attached if power is provided separately.

We will use the Arduino development system and software for this lab. The language is based on C. The Wikipedia article gives a good overview of the programming environment. The Arduino links in the order given will help you to understand how to program the Arduino. They also provide simple examples to build and try out in the lab.

Figure 1: Arduino Duemilanove prototyping platform connected to a laptop computer over USB.

After you have looked at the development environment reference above, you should look briefly at three example programs (Arduino programs are called “sketches”) on the Tutorial home page (the page heading is “Examples”):

- **Blink** (the “Hello World” equivalent)
- **Fading** (PWM control of LED brightness)
- **SerialCallResponseASCII** (Serial communication in ASCII; readout of measurements on analog pins and state of digital pin).

These will give you an idea of what a typical Arduino sketch looks like and the special roles of the `setup()` and `loop()` functions.

Then you should carefully read all the links on the Foundations page except the last one on making your own libraries. This forms the heart of the introduction to the Arduino software and should ready you for digging into the examples in lab.
The online Arduino documentation has more detailed descriptions of its version of C code and function libraries in the Reference link on the Arduino home page.

In the last lab (Lab 18), we will also examine a more complex Arduino example featuring 62.5 kHz waveform sampling and playback. This will be described in a separate note.

**Lab Exercises for Lab 17**

In the lab, you should learn about

- the Arduino development environment
- Arduino programs and how they relate to the microcontroller hardware and the AVR Libc environment
- Arduino hardware operation and digital I/O
- Analog output and PWM waveforms
- serial communication with the host computer (use Serial Call Response ASCII sketch)
  - Analog input with microcontroller ADC

Specifically, you should continue to keep a record of what you do in your lab notebook.

1. To start, you should follow the directions in the Arduino on-line tutorial and run the Blink application. There is an LED with series resistor on the expansion breadboard which can be connected using a wire from Digital Pin 13 to the eyelet in the PC board next to the LED.

2. You should also do the Fading example and use the oscilloscope to observe the PWM waveform. *Be sure to use a 270 \( \Omega \) or greater resistor in series with the LED*. Sketch the PWM waveform in your notebook and explain how it could be used to provide a continuously variable analog signal in the range 0-5 V.

3. Measure the speed of a short Loop() function. Make a program to toggle a digital output pin on and off in successive iterations of the Loop() function and measure the loop period with an oscilloscope. Is the period stable?

4. Run SerialCallResponseASCII. Before a character is sent from the computer to the Arduino, the Arduino sends an ASCII stream 0, 0, 0 CR LF on the Tx (serial transmit) output pin (digital pin 1 is dedicated to this purpose; digital pin 0 is the serial receive pin). Observe the waveform. Can you decode the characters? What is meant by the Baud rate?

5. Continue with example programs or your own variations as desired, keeping the goals in mind. You might try writing a program to read an analog input from a potentiometer or other source (limited to 0-5 V) and use the value read to set a PWM analog output to control the brightness of a LED.

Turn in your notebook log at the end of the lab.