CE 186: Design of Cyber Physical Systems
University of California, Berkeley

Fall 2016

Professor Scott Moura & TA Anushri Kumar
Original Content: Eric Burger

Lab 1: Introduction to Arduino Microcontrollers, Data Types, and Serial Communication

Images developed using Fritzing.

Part 1: Getting Started with Arduino

Exercise 1: Arduino Installation Guide

Items:

- 1x Arduino Uno Boards with USB Cables

Deliverables:

- None

Directions:

1. Download and install the Arduino IDE software (http://arduino.cc/en/Main/Software)
2. Follow the Getting Started Guide and complete the Blink Example (http://arduino.cc/en/Guide/HomePage).
    a. We are using Arduino UNO boards.
    b. Note: A dialog box may or may not appear when the Arduino Uno is first connected to the USB port.

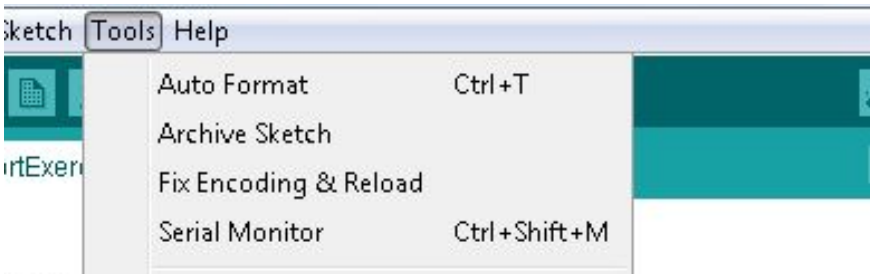
Exercise 2: The Serial Port Object

Deliverables:

- None

Directions:

1. Open the SerialPortExercise file (located in the Code folder of the Lab 1 Packet) and upload the program to your Arduino board.



2. In the Arduino IDE, open the Serial Monitor (ctrl+shift+M or shift-⌘-M). You should see a sequence of numbers printed to the screen.

Exercise 3: Expanding the Blink Sketch

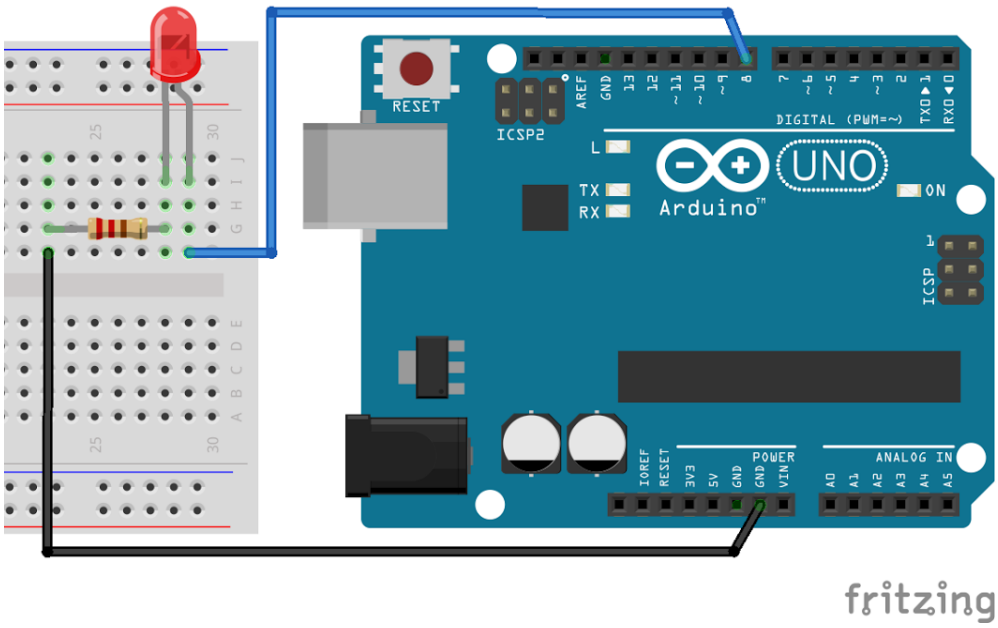
Additional Items:

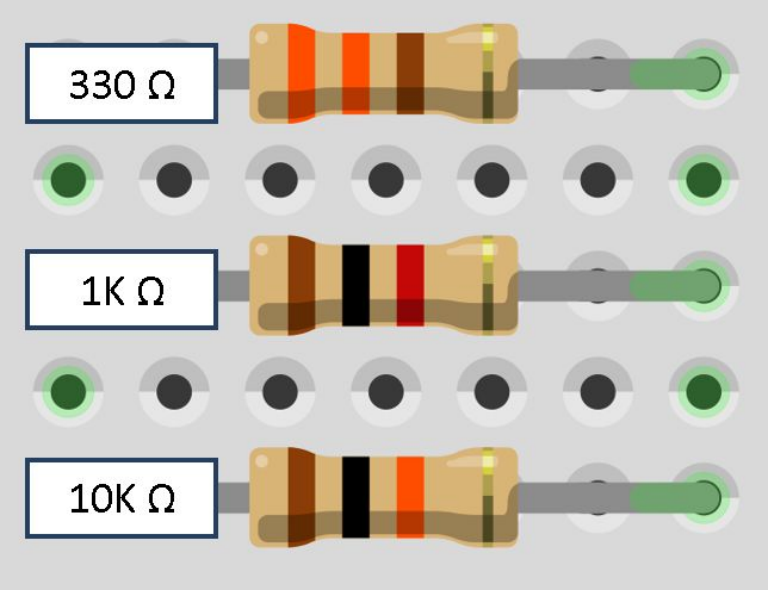
- 1x LED
- 1x Resistor (Between 200 and 1k Ohm)
- 1x Breadboard (A white plastic board with holes)
- Jumper Wires

Deliverables (0.5 pts):

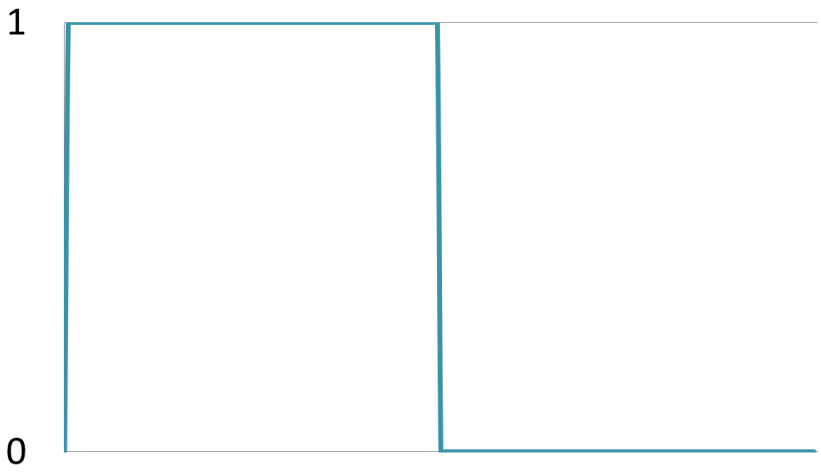
- Submit a file named BlinkLED.ino containing the program described below. Additionally, in your lab report, copy & paste your Arduino code into a code box.

Directions:





1. Add another LED to the Arduino, as shown in the image above. Connect the long pin (anode) of the LED to Pin 8 and the short pin (cathode) to one end of the resistor. Connect the other end of the resistor to ground (GND). If you are using an LED Breakout board, the + pin is the anode and the – pin is the cathode through a resistor. The image above shows the color codes for some of the most common resistors.
    a. Note: On a breadboard, each of the terminals in a numbered row are connected within the board. Thus, in the image above, the LED cathode (short pin) is connected to the resistor through the breadboard.
2. Open the Arduino Blink example (File->Examples->01.-Basics->Blink). Save the program to your computer with the name BlinkLED.
3. In the file, change the LED integer from 13 to 8.
4. Upload the program to the Arduino. The LED should begin blinking.



5. Edit the program such that the LEDs on Pins 8 and 13 both blink.
6. Define a function with the name "blinkLEDs". blinkLEDs() should:
    a. Return nothing.
    b. Accept one input: an integer specifying the duration of each blink in seconds.
    c. Turn on and off the LEDs attached to Pins 8 and 13.
    d. Write the function such that both LEDs are not on at the same time.
    e. HINT: Refer to https://www.arduino.cc/en/Reference/FunctionDeclaration
7. Edit the loop() function so that it calls the blinkLEDs() function and nothing else.

Exercise 4: The Serial Port Object

Deliverables (0.5 pts):

- Submit a file named SerialPortForLoop.ino containing the program described below. Additionally, in your lab report, copy & paste your Arduino code into a code box.

Directions:

1. Create a program with the file name SerialPortForLoop.ino.
2. Use the code from SerialPortExercise.ino (located in the Code folder of the Lab 1 Packet) and write a program that prints the integers between 0 and 9 to the Serial port during each loop (i.e. add a "for loop" inside the loop() function).
    a. HINT: https://www.arduino.cc/en/Tutorial/ForLoop
3. After the new "for loop", print an empty line to the Serial port (i.e. Serial.println();).
4. Before the end of the loop() function, add a 1 second (1000 millisecond) delay to the program.
5. The output of your program should look similar to...

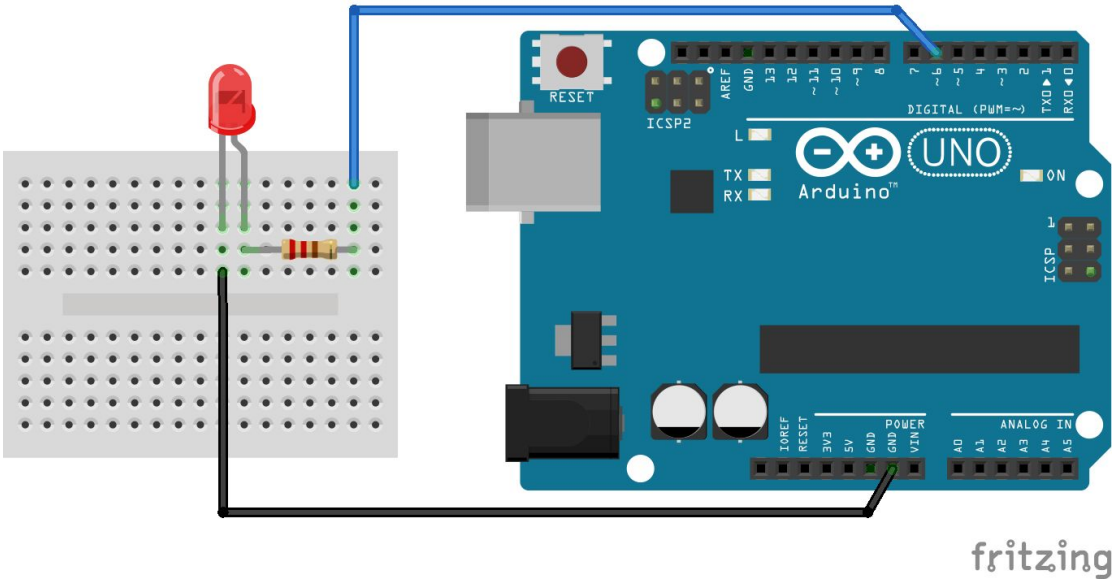
```
0
1
2
3
4
5
6
7
8
9

0
1
```

Exercise 5: Fade the LED with Pulse Width Modulation (PWM)

Deliverables (1.5 pt):

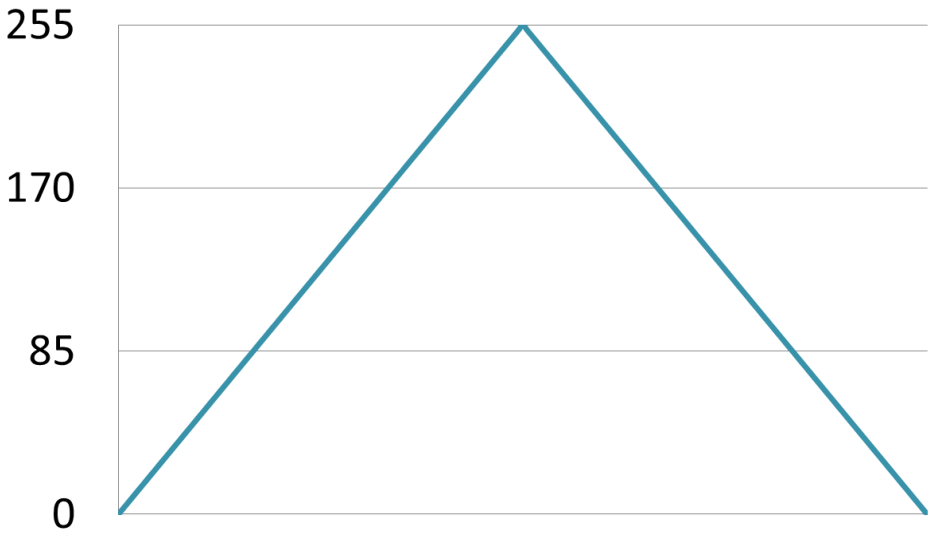
- In your lab report, respond to Questions.

Directions:



1. The Arduino UNO has six digital pins capable of pulse width modulation (indicated with the tilde symbol, ~). With PWM, a digital pin can produce an analog signal rather than simply on (HIGH) or off (LOW). This allows the Arduino to output a specific voltage or a wave function. More explicitly, you can give the PWM pin an integer between 0 and 255 and the pin will output a voltage between (about) 0 and 5V. In this exercise, we will use a digital pin with PWM to incrementally fade an LED.
2. Connect the LED through the resistor to pin 6 (instead of pin 8).

3. Open the FadeLEDExercise file and upload it to your Arduino board. The LED should begin to blink with the intensity (roughly) represented by the function below.



4. Note the integer algebra in the FadeLEDExercise program. Microcontrollers are capable of floating point algebra, but it is generally slow and imprecise. Integer algebra is faster but may not produce the results you expect. If you are doing algebra with integers and floats, pay attention to where integer rounding is (or is not) taking place.

```
// Calculate the LED brightness
// as a percentage (0 to 100)
// Note: int cannot have decimals
int percent = i*100/timesteps;
// Calculate the LED brightness
// as PWM output (0 to 255)
// Note: percent*255/100 does not = percent/100*255
int brightness = percent*255/100;
```

5. In the Arduino IDE, open the Serial Monitor (ctrl+shift+M or shift-⌘-M). You will see a printout of each LED brightness value.
6. **Questions**:
    a. What happens if you increase or decrease the value of "timesteps"? Explain what "timesteps" represents.
    Hint: Try timesteps=1, timesteps=5, timesteps=100, and timesteps=200.
    b. Why do we multiple percent by 255 and why could one argue that 256 is the maximum value of "timesteps"?
    Hint: See the Arduino documentation for analogWrite().

c. There is a problem with how the FadeLEDExercise program is implemented. For example, if timesteps is 256, how many possible values of brightness would we expect? With the current implementation, how many different values of brightness do we observe? What is the problem with the code? Be specific.
Hint: Check the variable types.
Hint: How many possible values of percent do we expect? How many do we observe?

Exercise 6: Arduino Arithmetic

Deliverables:

- None

Directions:

1. Open the ArduinoArithmetic program and upload it to your Arduino board.
2. The ArduinoArithmetic program includes 4 demonstrations, each showing some of the issues with using microcontrollers to perform basic arithmetic. Read through the code and make sure you understand what is being done.
3. In the Arduino IDE, open the Serial Monitor (ctrl+shift+M or shift-⌘-M). Send the numbers/characters '0', '1', '2', and '3' to run each demo.