Lab 2: Blinkie Lab

Objectives

This lab introduces the Arduino Uno as students will need to use the Arduino to control their final robot. Students will build a basic circuit on their prototyping board and wire the board to the Arduino. Students will learn the basic programming structure for the Arduino. The final project for the lab will have students develop a program to blink an LED.

Materials

1) Arduino Uno
2) Prototyping Circuit Board
3) 220 Ω Resistor
4) Wires for Building Circuits
5) Wire Cutters
6) Wire Strippers

Theory

Introduction to Circuits

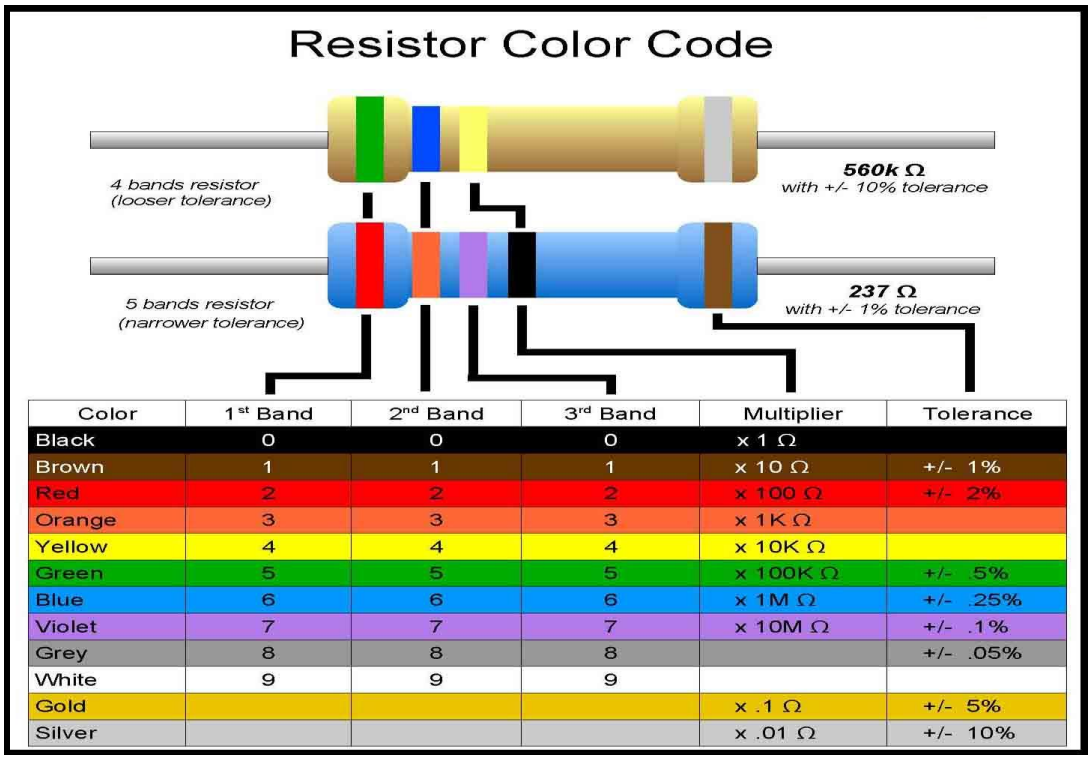
An electrical circuit contains a closed loop of wire that connects any number of electrical components, such as batteries, light bulbs, switches, resistors, motors, etc. Electrical charge, measured in **coulombs**, **C**, flows through a circuit in the same way that water would flow though a pipe. Electrons are the tiny particles that carry the electric charge through the circuit. Each electron has a value of -1.60×10^{-19} coulombs (making one coulomb of charge a very large number of electrons, $\sim 10^{20}$).

Lifting a mass to some height, h, above the ground is an example of potential energy. Potential energy is converted to kinetic energy when the mass is released and falls to the ground. Similarly, a battery contains electrons waiting to be used. Like the mass, electrons will flow from a high potential to a lower potential, a potential which is called **voltage**, **V**, and measured in *volts* $[V]$. Think of voltage as the force that pushes the electrons around the circuit. A battery creates a potential drop so that the electrical charges can flow through the wires in electrical components and bring power to the devices we want to use.

The flow of charge is measured as a variable called current. *Current*, I, is measured in *amperes* $[A]$ where 1 ampere is defined a 1 coulomb per second $(1A = 1C/s)$/ Since 1 C $\sim 10^{20}$ electrons, you can see that trillions of electrons are flowing through a circuit in any given second.
Resistance, **R**, is the measure of how difficult it is to push electrons through a substance or device. Resistance has the units of *Ohms* $[\Omega]$, which is defined as the unit of volts/ampere.

Figure 1: Resistor

Another way to think of resistance is the ratio of voltage drop to the current for a given circuit. For example, a high resistance means that a large voltage drop is required to achieve a given current. When a voltage is applied across a circuit a current that depends on the equivalent resistance of the circuit is generated.



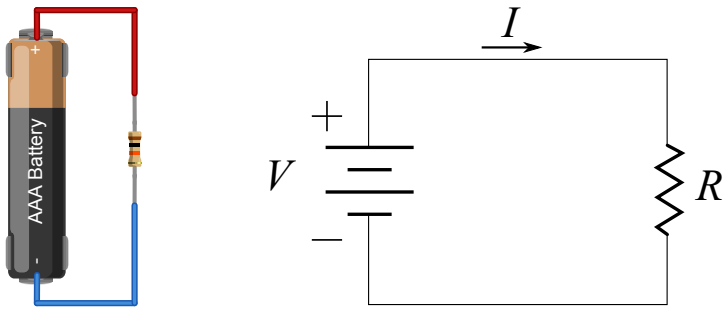
Color	1st Band	2nd Band	3rd Band	Multiplier	Tolerance
Black	0	0	0	x 1 Ω	
Brown	1	1	1	x 10 Ω	+/- 1%
Red	2	2	2	x 100 Ω	+/- 2%
Orange	3	3	3	x 1K Ω	
Yellow	4	4	4	x 10K Ω	
Green	5	5	5	x 100K Ω	+/- .5%
Blue	6	6	6	x 1M Ω	+/- .25%
Violet	7	7	7	x 10M Ω	+/- .1%
Grey	8	8	8		+/- .05%
White	9	9	9		
Gold				x .1 Ω	+/- 5%
Silver				x .01 Ω	+/- 10%

Figure 2

The relationship between voltage (V), current (I), and resistance (R) can be defined by **Ohm's Law**. The voltage drop needed to keep the charges moving through the wire is determined by the resistance. The resistance, or the ratio of voltage drop to current remains constant for all applied voltage drops. It is given by Equation (1):

$$R = \frac{V}{I} \tag{1}$$

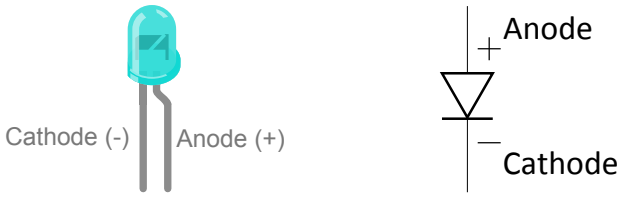
More commonly, this relationship is written $V = IR$. A schematic of this circuit is shown in Figure 3:



(a) (b)

Figure 3: (a) Breadboard. (b) Connectivity Diagram.

On a more advanced side note, Benjamin Franklin established the convention that current flows from the positive to negative side of the voltage as shown in Figure 3. It was discovered many years later (unfortunately) that the current carriers are electrons and that they actually travel in the opposite direction of the established current convention (negative to positive).



(a) (b)

Figure 4: (a) LED. (b) LED Circuit Symbol

A light emitting diode, or **LED**, is a two-lead semiconductor light source. An LED emits light when an appropriate current flows. The LED's brightness is dependent upon the current. Placing a resistor in series with an LED determines the circuit current which helps prevent burning out the LED, as connected below.

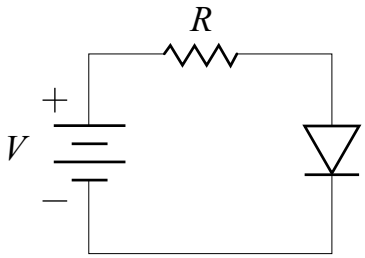
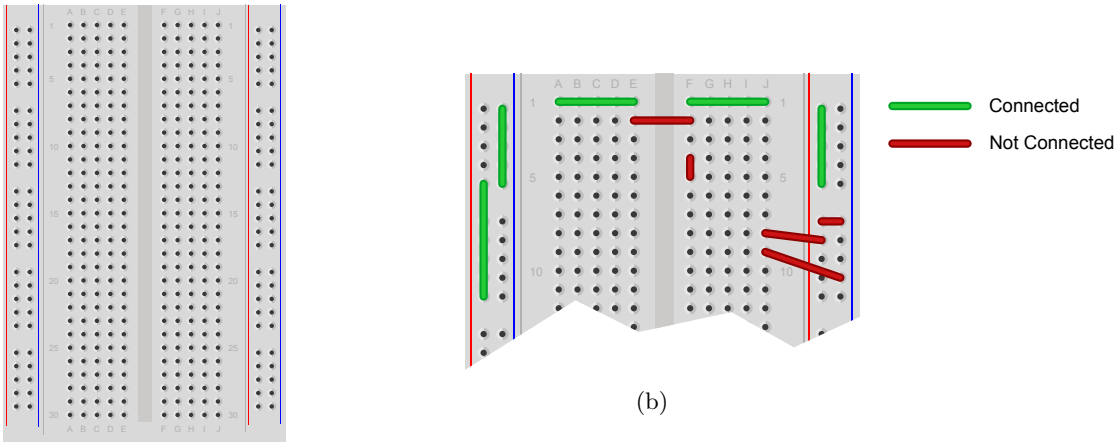


Figure 5: Basic LED Circuit

Breadboard Basics

Prototyping circuit boards (or breadboards) are a more convenient way to test circuits than soldering components together. Wires or components can be pushed directly into the breadboard. However, certain conventions on the breadboard must be followed in order for the

circuit to work properly.



(a) (b)

Figure 6: (a) Real Circuit. (b) Circuit Schematic Diagram.

A breadboard schematic is shown in Figure 6. Letters are used to identify vertical columns, and numbers are used to identify horizontal rows. The second image shows how the vertical columns are connected. Current flows only along these internal connections. An advantage of this setup is that you can supply power, say $+5V$, to and entire column on the breadboard or to a set of 5 horizontal sockets. By connecting multiple groups of the 5 horizontal sockets, you can carry the same voltage across greater proportions of the breadboard.

Arduino

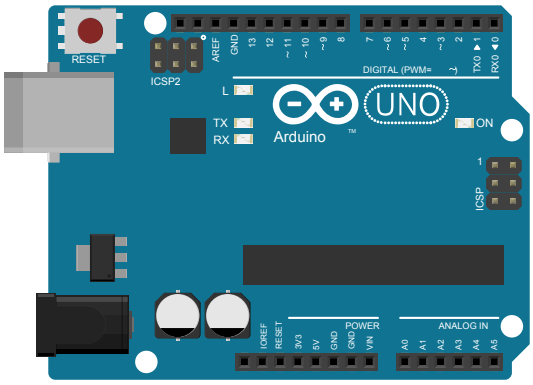


Figure 7: Arduino UNO

Arduino is a small computer (or microcontroller) that can easily interface with hardware. We can program the Arduino with another computer though the USB connection. Arduino can run with other computers or independently. It has a brain that called microcontroller which does all the computation. The microcontroller has different pins which communicate with other components.

In electronics there are two types of signals, analog and digital. An analog signal consists real voltage within a specific range. It can be 0, 2.5, -104.2443 or other values depends on the range. A digital signal consists of a series of binary values (0s and 1s). A 0 refers to off or low voltage (LOW) and 1 refers to on or high voltage (HIGH). Arduino has 3 types of pins. Digital in and out (I/O), analog out, and analog in. The Arduino's pins can be used in many different situations and combinations. Arduino can change the value of output pins with respect to the computer's commands or the value of input pins. More than pins, there are some preassembled circuits that can easily attach to the Arduino that are called *shields*. With shields, Arduino can connect to different instruments directly.

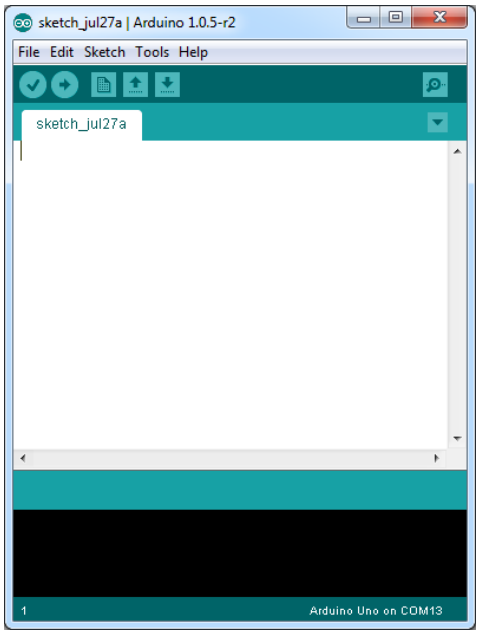


Figure 8: Arduino Sketch Pad

Arduino comes with a software to write codes (or algorithms) named *sketch pad*. Codes in sketch pad are compiled (or collected) and then sent (or flashed) onto the Arduino's microcontroller with communication through the USB cable. There are several examples and tutorials in the sketch pad. More than regular codes there are several prewritten codes (libraries) which are available in sketch pad. There are different commands in codes. There are several libraries that provide access to prewritten commands. Sketch pad has some libraries and we can add more in case we need more commands.

Code Structure

Commenting is necessary for good programming practices. Comments are not executed in the program and allow a programmer explain how the code works for future reference. For commenting a single line, use two forward slashes (//) and the write the comment.

```
1 // Blink LED code, created 8/21/14
```

To add a comment that spans multiple lines, enter /* to start the comment section and */ to end the comment.

```
/*
Arduino Blink LED
Robotics LLC Lab, created 8/21/14
*/
```

The **setup function** is one of two main loops in any Arduino code. This function contains instructions that are used only once when the Arduino is turned or reset.

```
void setup()
{

}
```

The setup function is useful for setting up which pins are inputs or outputs and other functions. Pins set to be inputs receive signals and output pins control physical hardware (like lighting an LED). Example code is shown below:

```
void setup()
{
  pinMode(12, OUTPUT); // Set digital pin 12 as output
  pinMode(11, INPUT); // Set digital pin 11 as input
}
```

The **main loop** contains the main code that is executed repeatedly. Example code is shown below:

```
/*
Arduino Blink LED
Robotics LLC Lab, created 8/21/14
*/

void setup()
{

}

void loop()
{
  // Main code is placed here
}
```

Laboratory Exercises

1. Wait for the TAs to explain how the pins are connected to each other in the breadboard, properties of LEDs, and the Arduino. Try to find input and output pins.
2. Use Figure 2 to read the resistor value. After the TAs explained the multimeter, use the multimeter to measure the resistor value. Be sure you have the right resistor (220 Ω). For future reference, DO NOT MEASURE CURRENT IN PARALLEL.

3. Use jumper wires and make the circuit that you have in the figure below. (DO NOT CONNECT ANY THING TO THE USB PORT OR POWER SOURCE BEFORE CHECKING WITH TAs). The wrong connection in Arduino can damage the main board.

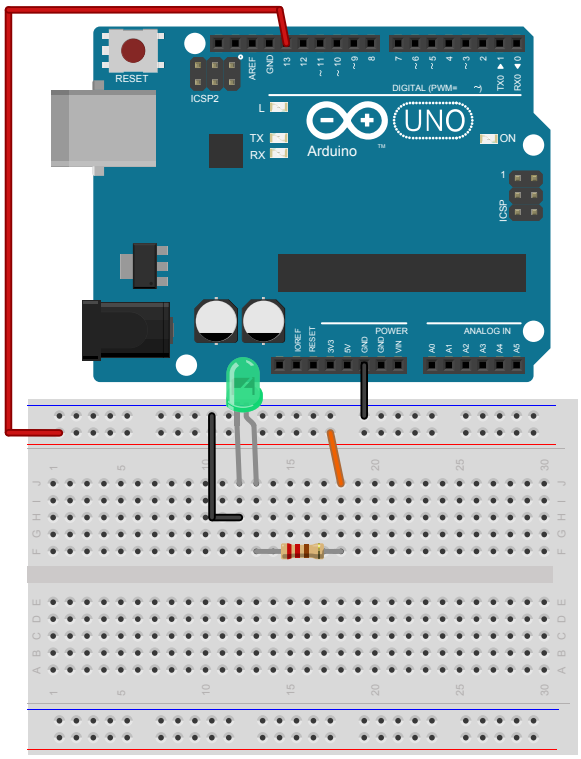


Figure 9: Connection Diagram

4. Now open the sketchpad and wait for TAs to show you how to find the right port number and settings.

5. In sketchpad, follow the file path File, Examples, 01.Basics, and click on *Blink*. Copy the whole code and paste the code into a new sketch. Wait for the TAs to explain how the program functions. There are some extra spaces in handout for your notes. Be sure you take notes for your later references. Ask TAs if you do not understand something in the code.

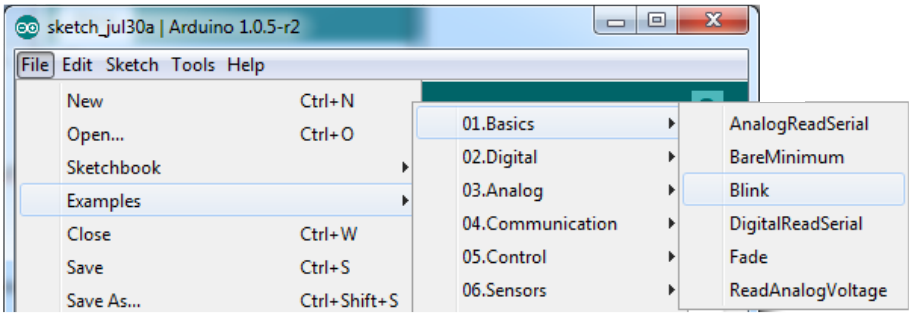


Figure 10: File Path

```
/*
  Blink
  Turns on an LED on for one second, then off for one second, repeatedly.

  This example code is in the public domain.
 */

// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(led, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);               // wait for a second
  digitalWrite(led, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);               // wait for a second
}
```

6. After you copied the code and TAs explained them to you, you can send your code to the Arduino. Connect your USB cable to the computer and the larger end to the Arduino. Now click the Upload (or Arrow) button to send the code to the Arduino.

Changing the Blinking parameters

1. Now we want to change some parameters to have the LED on for 0.5 second and off for 0.5 second (eg: *delay(1000)* = delay for 1 second.
2. With your knowledge try to change the blinking pattern to have the LED on for 0.5 sec and off for 0.5 sec. (Hint: you do not have to change any thing in the hardware. It is just some small changes in the code). Try to find the parameters that can change the pattern. Ask a TA if you have any questions.
3. To save the file for latter, ask a TA on how to save the code to your U-drive. You can also email the code to yourself, save the code on a flash drive, or other methods.
4. Unassemble the circuit and put every thing back in the boxes.
5. Please do not leave the lab if you have questions about the lab exercises.