Medical Electronics - Laboratory 1

Part 1

Aim: implement the simplest task with Arduino Uno board: blink on-board LED.

Hardware Required

- Arduino Uno Board

Circuit

This example uses the built-in LED that most Arduino boards have. This LED is connected to a digital pin D13 in Arduino Uno. The constant *LED_BUILTIN* is specified in every board descriptor file and allows you to control the built-in LED easily.

Code

Plug your Arduino board into your computer, start the Arduino Software (IDE) and enter the code below.

The first thing you do is to initialize LED_BUILTIN pin as an output pin with the line

```
pinMode(LED_BUILTIN, OUTPUT);
```

In the main loop, you turn the LED on with the line:

```
digitalWrite(LED_BUILTIN, HIGH);
```

This supplies 5 volts to the LED anode. That creates a voltage difference across the pins of the LED, and lights it up. Then you turn it off with the line:

```
digitalWrite(LED_BUILTIN, LOW);
```

That takes the LED_BUILTIN pin back to 0 volts, and turns the LED off. In between the on and the off, you want enough time for a person to see the change, so the `delay()` commands tell the board to do nothing for 1000 milliseconds, or one second. When you use the `delay()` command, nothing else happens for that amount of time.

```
/*
  Blink

  Turns an LED on for one second, then off for one second, repeatedly.

  Most Arduinos have an on-board LED you can control. On the UNO, MEGA and ZERO
  it is attached to digital pin 13, on MKR1000 on pin 6. LED_BUILTIN is set to
  the correct LED pin independent of which board is used.
  If you want to know what pin the on-board LED is connected to on your Arduino
  model, check the Technical Specs of your board at:
  https://www.arduino.cc/en/Main/Products

  modified 8 May 2014
  by Scott Fitzgerald
  modified 2 Sep 2016
  by Arturo Guadalupi
  modified 8 Sep 2016
  by Colby Newman

  This example code is in the public domain.

  http://www.arduino.cc/en/Tutorial/Blink
*/

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

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

Part 2

Aim: build a circuit with external LED and resistor and blink the LED.

Hardware Required

- Arduino Uno Board
- LED
- 220 ohm resistor
- hook-up wires
- breadboard

Circuit

If you want to lit an external LED, you need to build the circuit from Figure 1, where you connect one end of the resistor to the digital pin correspondent to the *LED_BUILTIN* constant. Connect the long leg of the LED (the positive leg, called the anode) to the other end of the resistor. Connect the short leg of the LED (the negative leg, called the cathode) to the GND. The value of the resistor in series with the LED may be of a different value than 220 ohm; the LED will lit up also with values up to 1K ohm.

Schematic

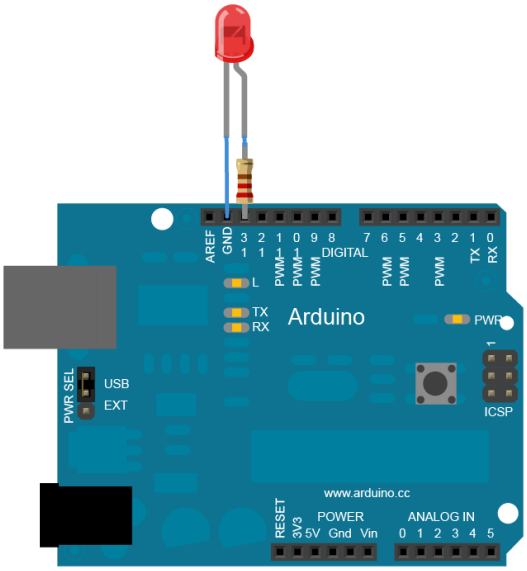

Figure 1 External LED and resistor circuit

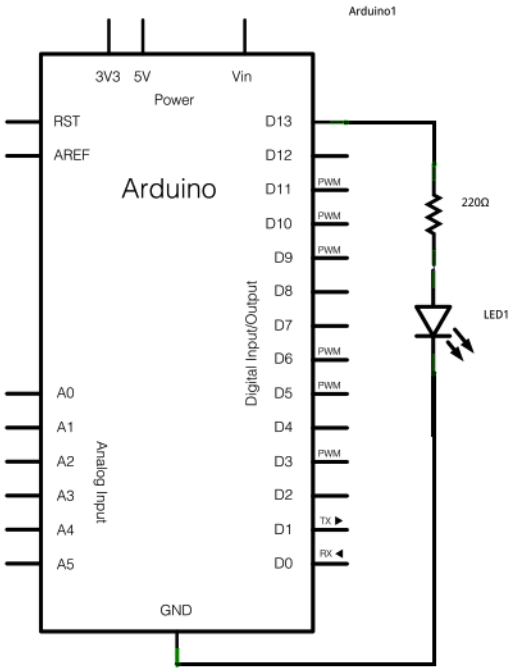

Figure 2 External LED and resistor circuit

Code

After you build the circuit plug your Arduino board to the computer and run the code from Part 1 of the exercise. Modify the delay values and test various blinking frequencies.

Part 3

Aim: build a circuit with external LED and a button to turn on and off LED.

Hardware

- Arduino Uno Board
- Momentary button or Switch
- 10K ohm resistor
- 220 ohm resistor
- LED
- hook-up wires
- breadboard

Circuit

Connect three wires to the board according to Figure 3. The first two, red and black, connect to the two long vertical rows on the side of the breadboard to provide access to the 5 volt supply and ground. The third wire goes from digital pin 2 to one leg of the pushbutton. That same leg of the button connects through a pull-down resistor (here 10K ohm) to ground. The other leg of the button connects to the 5 volt supply.

When the pushbutton is open (unpressed) there is no connection between the two legs of the pushbutton, so the pin is connected to ground (through the pull-down resistor) and we read a LOW. When the button is closed (pressed), it makes a connection between its two legs, connecting the pin to 5 volts, so that we read a HIGH.

You can also wire this circuit the opposite way, with a pullup resistor keeping the input HIGH, and going LOW when the button is pressed. If so, the behavior of the sketch will be reversed, with the LED normally on and turning off when you press the button.

If you disconnect the digital I/O pin from everything, the LED may blink erratically. This is because the input is "floating" - that is, it will randomly return either HIGH or LOW. That's why you need a pull-up or pull-down resistor in the circuit.

Schematic

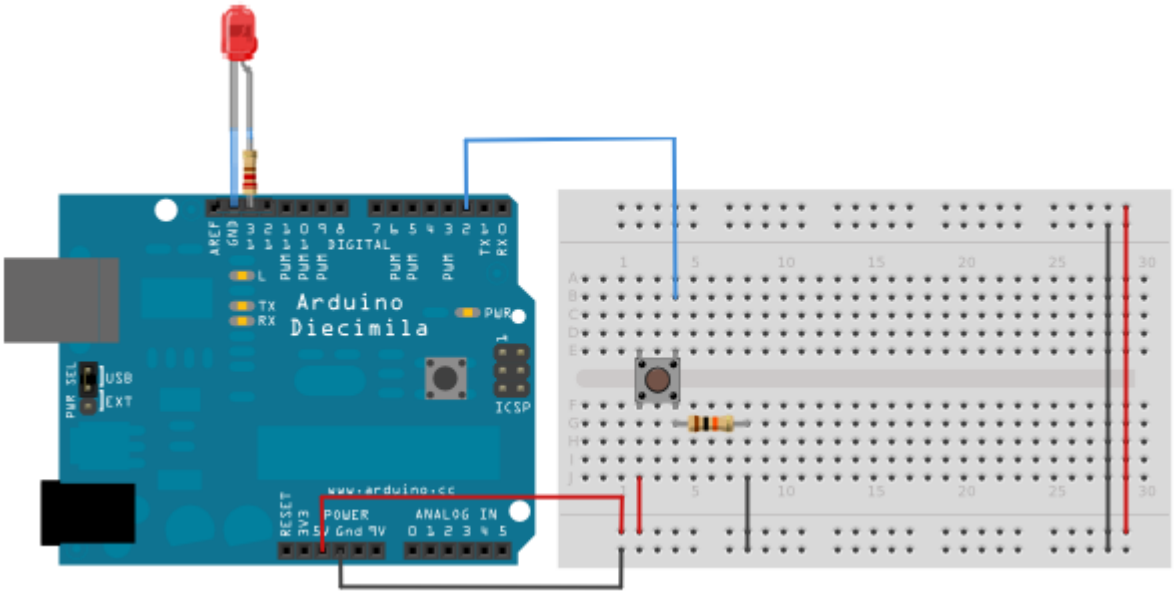

Figure 3 External LED and button circuit

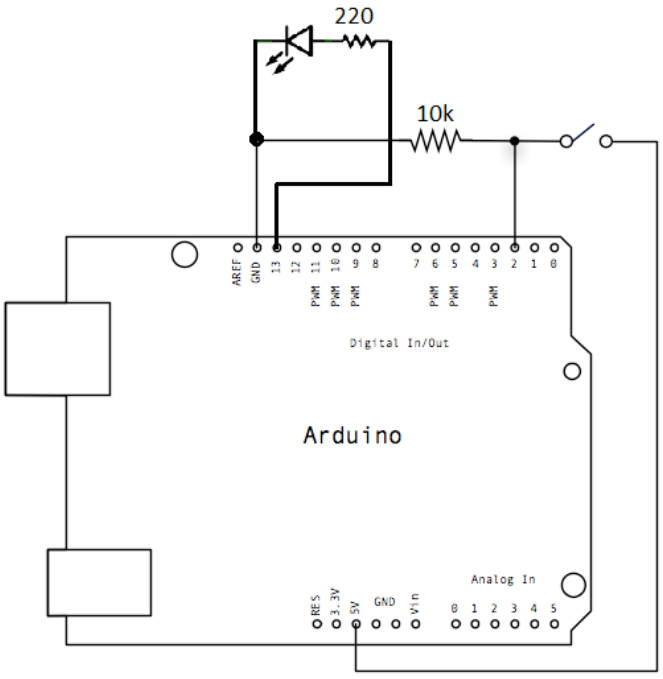

Figure 4 External LED and button circuit

Code

```
/*
  Button

  Turns on and off a light emitting diode(LED) connected to digital pin 13,
  when pressing a pushbutton attached to pin 2.

  created 2005
  by DojoDave <http://www.0j0.org>
  modified 30 Aug 2011
  by Tom Igoe

  This example code is in the public domain.

  http://www.arduino.cc/en/Tutorial/Button
*/

// constants won't change. They're used here to set pin numbers:
const int buttonPin = 2;    // the number of the pushbutton pin
const int ledPin =  13;     // the number of the LED pin

// variables will change:
int buttonState = 0;        // variable for reading the pushbutton status

void setup() {
  // initialize the LED pin as an output:
  pinMode(ledPin, OUTPUT);
  // initialize the pushbutton pin as an input:
  pinMode(buttonPin, INPUT);
}

void loop() {
  // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);

  // check if the pushbutton is pressed. If it is, the buttonState is HIGH:
  if (buttonState == HIGH) {
    // turn LED on:
    digitalWrite(ledPin, HIGH);
  } else {
    // turn LED off:
    digitalWrite(ledPin, LOW);
  }
}
```