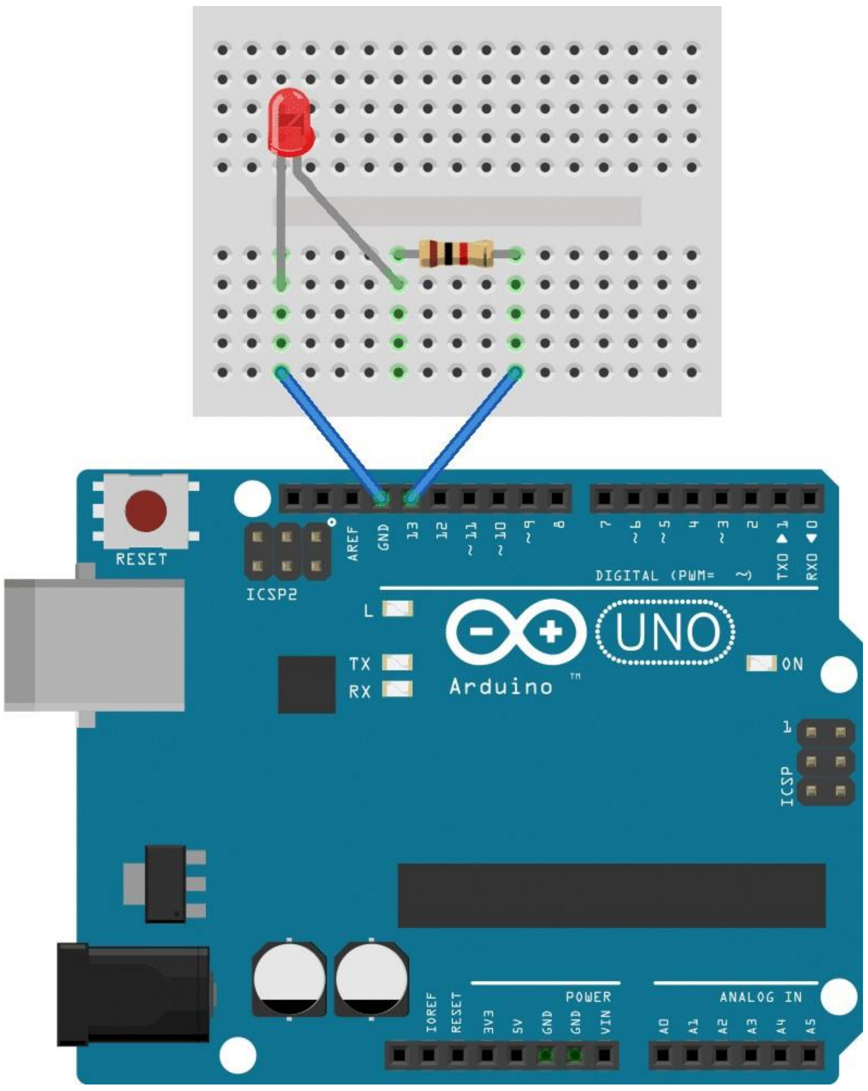
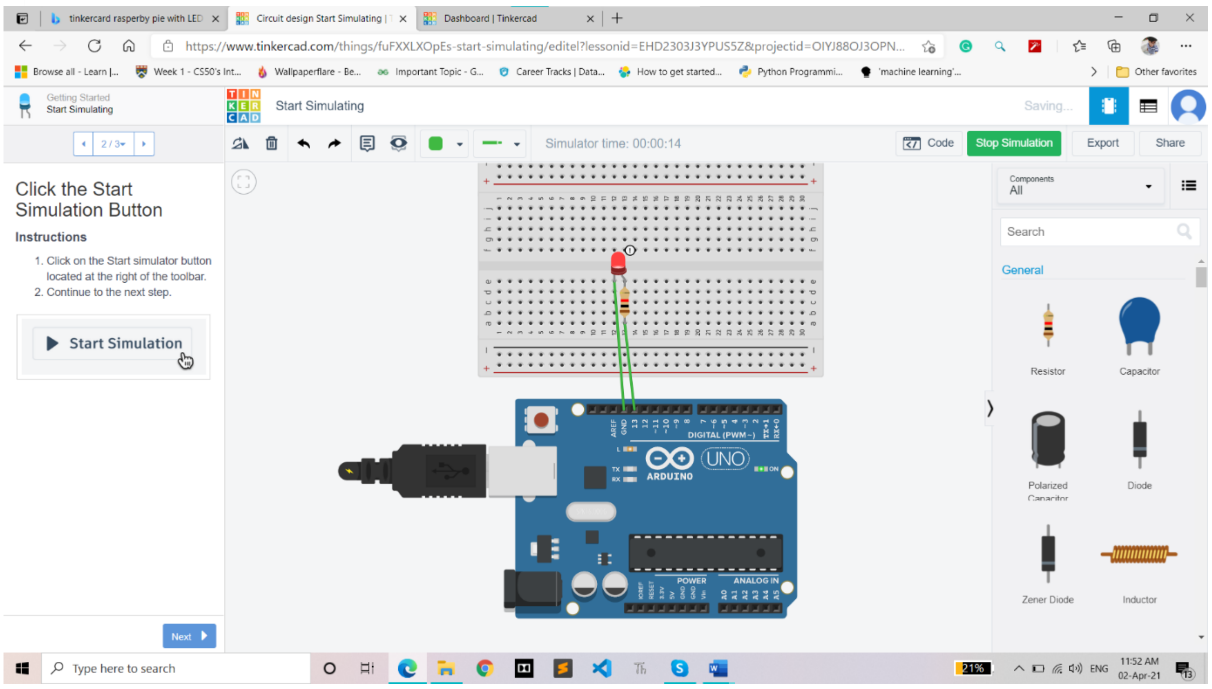
Internet of Things

Practical No 6: Controlling an LED using Arduino.

- Let’s see how to turn on and off the Arduino’s on-board LED. Then will see that how to turn on and off an LED connected to one of the Arduino’s digital pins. Understand how to change the LEDs flashing rate, and how to change the pin that powers the LED. You should have the Arduino IDE software installed on your computer.





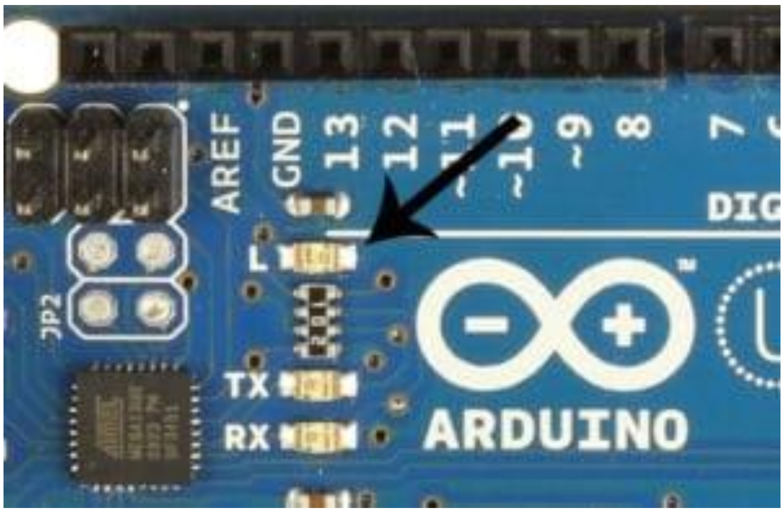
Click the Start Simulation Button

Instructions

1. Click on the Start simulator button located at the right of the toolbar.
2. Continue to the next step.

➢ **Controlling the Arduino's LED:**

- To turn on an LED, the Arduino needs to send a HIGH signal to one of its pins. To turn off the LED, it needs to send a LOW signal to the pin. You can make the LED flash by changing the length of the HIGH and LOW states.
- The Arduino has an on-board surface mount LED that's hard wired to digital pin 13. It's the one with an "L" next to it:



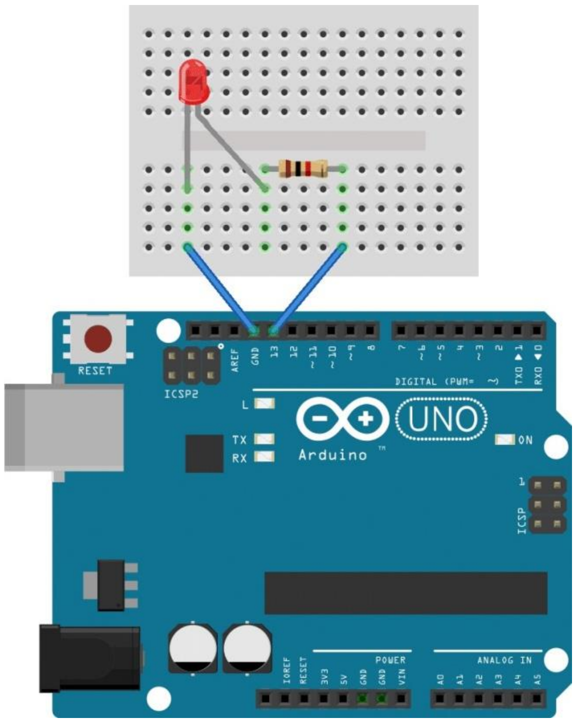
- To get this LED flashing, upload the "Blink" program to your Arduino:

```
void setup() {
        pinMode(13, OUTPUT);
}
void loop() {
        digitalWrite(13, HIGH);
        delay(1000);
        digitalWrite(13, LOW);
        delay(1000);
}
```

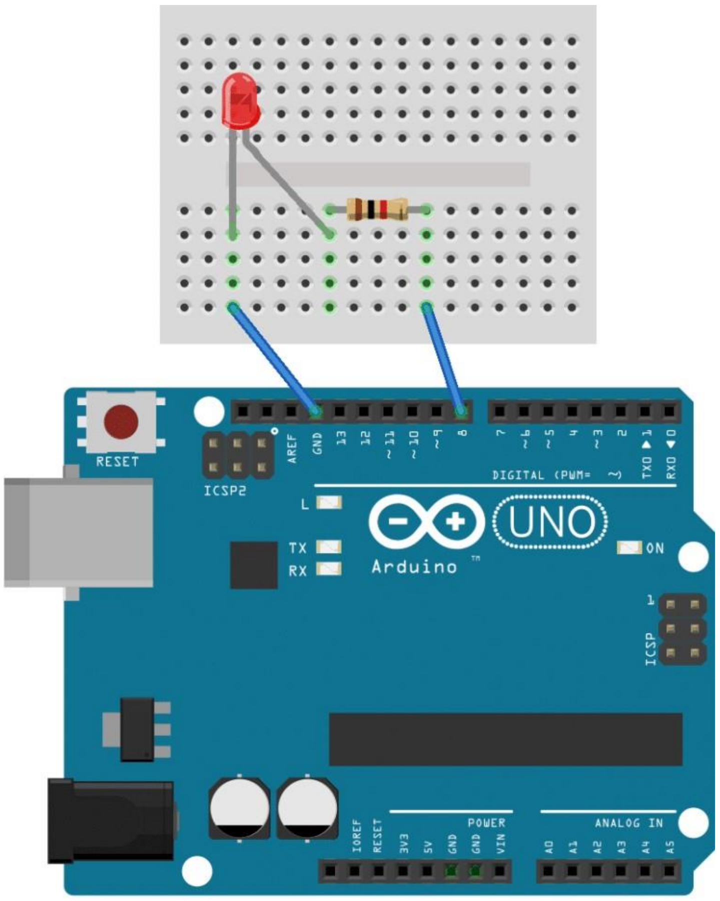
- The LED should now be blinking on and off at a rate of 1000 milliseconds (1 second).
- The delay () function on line 6 tells the Arduino to hold the HIGH signal at pin 13 for 1000 ms. The delay () function on line 8 tells it to hold the LOW signal at pin 13 for 1000 ms. You can change the blinking speed by changing the number inside the parentheses of the delay () functions.

➢ **Controlling an external LED:**

- LEDs need to have a resistor placed *in series* (in-line) with it. Otherwise, the unrestricted current will quickly burn out the LED. The resistor can be any value between 100 Ohms and about 10K Ohms. Lower value resistors will allow more current to flow, which makes the LED brighter. Higher value resistors will restrict the current flow, which makes the LED dimmer.
- Also, most LED’s have polarity, which means that they need to be connected the right way around. Usually, the LED’s shortest lead connects to the ground side.
- If you connect the LED to pin 13 as shown in the image below, you can use the same code we used above to make the LED flash on and off.



- If you want to use a different pin to power the LED, it’s easy to change it. For example, say you want to use pin 8 instead of pin 13. First move the signal wire from pin 13 over to pin 8:



- Now you'll need to edit a line of code in the program so the Arduino knows which pins to use as output pins. That's done on line 2 of the code above, where it says:

```
    pinMode (13, OUTPUT);
```

- To use pin 8, you just have to change the 13 to an 8:

```
    pinMode (8, OUTPUT);
```

- Next, change the code that tells the Arduino which pins will get the HIGH and LOW output signals. That's done everywhere there's a digitalWrite() function. In the program above, there's one on line 5 and one on line 7:

```
    digitalWrite(13, HIGH);
    digitalWrite(13, LOW);
```

- To specify that pin 8 should get the HIGH and LOW signals, you just need to change the 13's to 8's:

```
    digitalWrite(8, HIGH);
    digitalWrite(8, LOW);
```

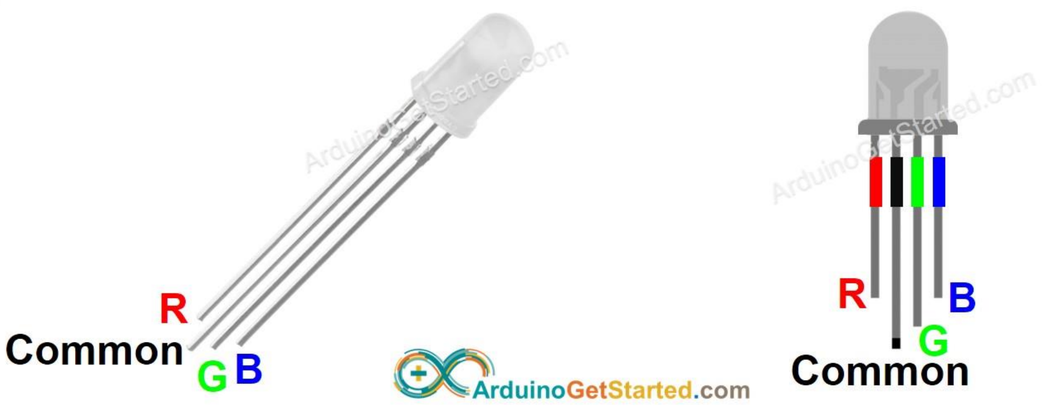
- The finished program should look like this:

```
        void setup() {

        pinMode(8, OUTPUT);

}

void loop() {

        digitalWrite(8, HIGH);

        delay(1000);

        digitalWrite(8, LOW);

        delay(1000);

}
```

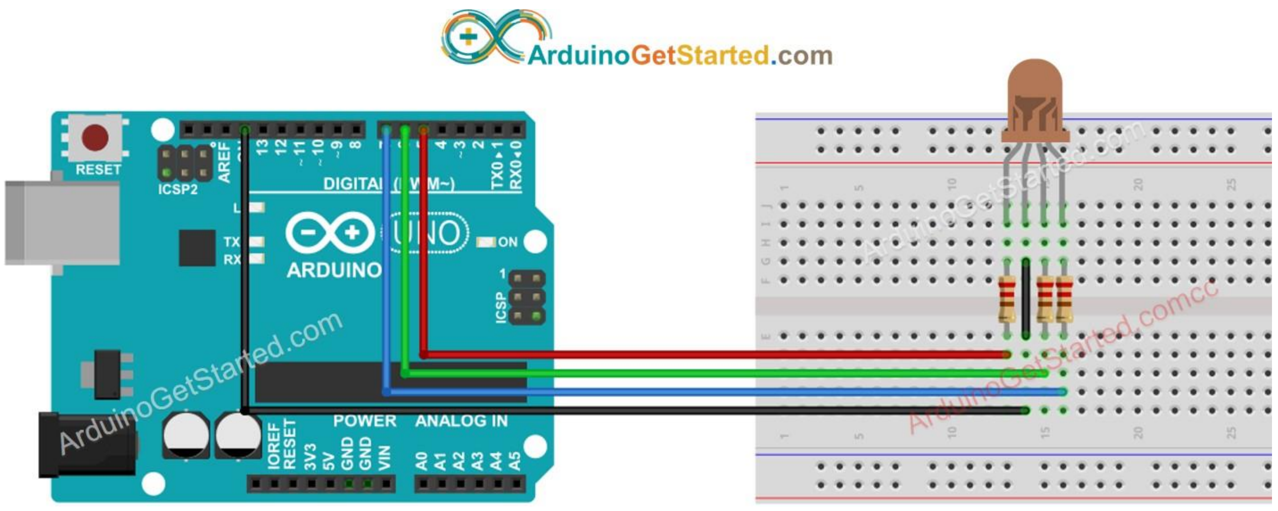
- After uploading, the LED should flash just like it did when it was connected to pin 13.

➢ RGB LED connections

Mainly three colours with RGB Pin



- In the nature of physics, a colour is composed of three colour values: Red (R), Green (G) and Blue (B). Each colour value ranges from 0 to 255.
- The mix of three values creates 256 x 256 x 256 colours in total.
- If we provide PWM signals (with duty cycle from 0 to 255) to R, G, B pins, we can make RGB LED displays any colour we want.
- The duty cycle of PWM signals to R, G and B pins correspond to colour values of Red (R), Green (G) and Blue (B)



CODE:

```
const int PIN_RED   = 5;

const int PIN_GREEN = 6;

const int PIN_BLUE  = 7;


void setup() {

  pinMode(PIN_RED,   OUTPUT);
  pinMode(PIN_GREEN, OUTPUT);
  pinMode(PIN_BLUE,  OUTPUT);
}

void loop() {
  // color code #00C9CC (R = 0,   G = 201, B = 204)
  analogWrite(PIN_RED,   0);
  analogWrite(PIN_GREEN, 201);
  analogWrite(PIN_BLUE,  204);

  delay(1000); // keep the color 1 second

  // color code #F7788A (R = 247, G = 120, B = 138)
  analogWrite(PIN_RED,   247);
  analogWrite(PIN_GREEN, 120);
  analogWrite(PIN_BLUE,  138);

  delay(1000); // keep the color 1 second

  // color code #34A853 (R = 52,  G = 168, B = 83)
  analogWrite(PIN_RED,   52);
  analogWrite(PIN_GREEN, 168);
  analogWrite(PIN_BLUE,  83);

  delay(1000); // keep the color 1 second
}
```