AMERICAN INTERNATIONAL UNIVERSITY-BANGLADESH

408/1, Kuratoli, Khilkhet, Dhaka 1229, Bangladesh

Assignment Title: Familiarization with microcontroller, study of blink test using and implementation of a traffic control system using microcontrollers and Tinkercad.

Assignment No: 02		Date of Submission: 28-02-2021
Course Title: Microprocessor and Embedded Systems.		
Course Code:		Section: C
Semester: Spring	2020-2021	Course Teacher: NADIA NOWSHIN

Declaration and Statement of Authorship:

1. I/we hold a copy of this Assignment/Case-Study, which can be produced if the original is lost/damaged.
2. This Assignment/Case-Study is my/our original work and no part of it has been copied from any other student's work or from any other source except where due acknowledgement is made.
3. No part of this Assignment/Case-Study has been written for me/us by any other person except where such collaborationhas been authorized by the concerned teacher and is clearly acknowledged in the assignment.
4. I/we have not previously submitted or currently submitting this work for any other course/unit.
5. This work may be reproduced, communicated, compared and archived for the purpose of detecting plagiarism.
6. I/we give permission for a copy of my/our marked work to be retained by the Faculty for review and comparison, including review by external examiners.
7. I/we understand thatPlagiarism is the presentation of the work, idea or creation of another person as though it is your own. It is a formofcheatingandisaveryseriousacademicoffencethatmayleadtoexpulsionfromtheUniversity. Plagiarized material can be drawn from, and presented in, written, graphic and visual form, including electronic data, and oral presentations. Plagiarism occurs when the origin of them arterial used is not appropriately cited.
8. I/we also understand that enabling plagiarism is the act of assisting or allowing another person to plagiarize or to copy my/our work.

* *Student(s) must complete all details except the faculty use part.*
** Please submit all assignments to your course teacher or the office of the concerned teacher.

Group Name/No.: 02

No	Name	ID	Program	Signature
1	Mohammed Afridul Haque	18-38794-3		
2	Tania Rahman	17-33841-1		
3	Nowrin Muhaimin Shailee	18-37259-1		
4	Md Refatul Islam	18-37818-2		
5	Rahman, Md. Ashiqur	17-35477-3		
6	Ragib,a Azmaill Damil	17-35503-3		
7	Anwar Hossain	17-35487-3		
8	Raihan MD Hemel	17-33607-1		
9	Chowdhury Ullas	17-35528-3		

Faculty use only		
FACULTYCOMMENTS	**Marks Obtained**	
	Total Marks	

Title:

Familiarization with microcontroller, study of blink test using and implementation of a traffic control system using microcontrollers and Tinkercad.

Introduction:

The objective of this experiment is to get familiarized with Microcontroller using Proteus 8.9.

- Learning to make the LED blink using Arduino and the delay functions.
- Implementation of a traffic control system using Arduino.

Equipment:

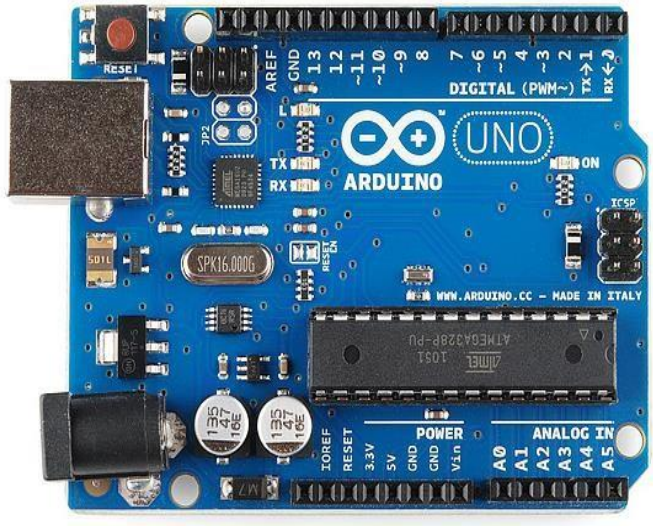
- PC
- Tinkercad

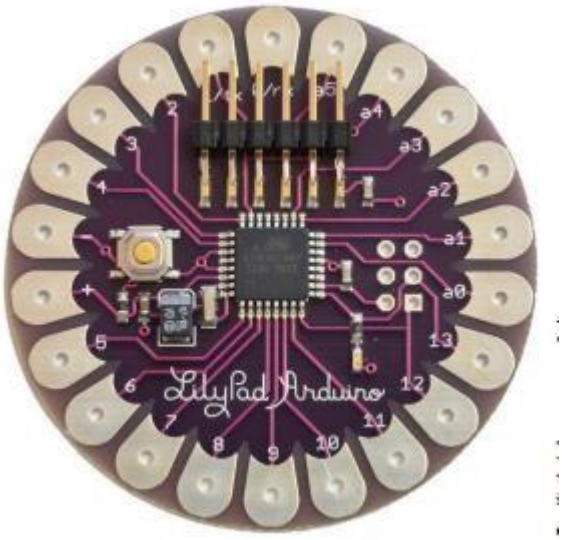
Theory and Methodology:

Arduino is an open-source platform used for creating interactive electronics projects. Arduino consists of both a programmable microcontroller and a piece of software, or IDE (Integrated Development Environment) that runs on our computer, used to write and upload computer code to the microcontroller board. Arduino Uno also doesn‟t need a hardware circuit (programmer/ burner) to load a new code into the board. We can easily load a code into the board just using a USB cable and the Arduino IDE (that uses an easier version of C++ to write a code).

Arduino Family:

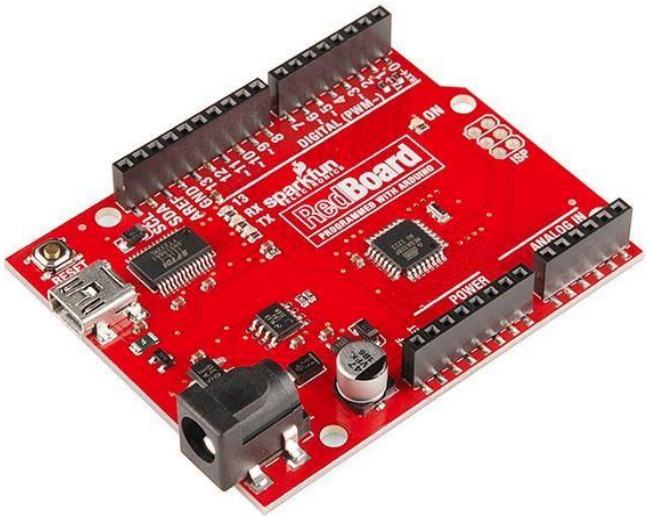
Arduino makes several different boards, each with different capabilities. In addition, part of being open source hardware means that others can modify and produce derivatives of Arduino boards that provide even more form factors and functionality. Here are a few options that are well-suited to someone new to the world of Arduino:

Figures of different types of Arduino boards	Configuration & Features
	**Arduino Uno (R3)** The Uno is a great choice for our first Arduino. It's got everything we need to get started, and nothing us don‟t. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a USB connection, a power jack, a reset button and more. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.



Lilypad Arduino

LilyPad is a wearable e-textile technology developed by Leah Buechley and cooperatively designed by Leah and SparkFun. Each LilyPad was creatively designed with large connecting pads and a flat back to allow them to be sewn into clothing with conductive thread. The LilyPad also has its own family of input, output, power, and sensor boards that are also built specifically for e-textiles. They're even washable. The LilyPad is unique because it is designed to be sewn into clothing. Using conductive thread, we can wire it up to sewable sensors, LEDs,and more. To keep size down, it can be programmed by using an FTDI cable.

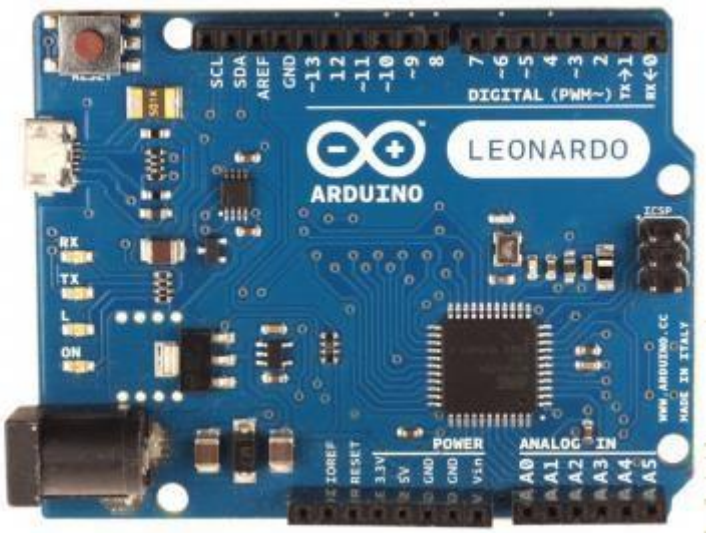


RedBoard

RedBoard can be programmed over a USB Mini-B cable using the Arduino IDE. It'll work on Windows 8 without having to change our security settings (we used signed drivers, unlike the UNO). It's more stable due to the USB/FTDI chip used in it, plus it's completely flat on the back, making it easier to embed in our projects. Just plug in the board, select "Arduino UNO" from the board menu and we are ready to upload code. We can power the RedBoard over USB or through the barrel jack. The on-board power regulator can handle anything from 7 to 15VDC

Arduino Mega (R3)

The Arduino Mega is like the UNO‟s big brother, it is mostly used in projects require digital i/o pins. It has lots (*54*) of digital input/output pins (14 can be used as PWM outputs), 16 analog inputs, a USB connection, a power jack, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Mega 2560 employs an ATMega 2560 as the main MCU, which has 54 general I/Os to enable us to interface with many more devices. The Mega also has more ADC channels, and has four hardware serial interfaces (unlike the one serial interface found on the Uno).

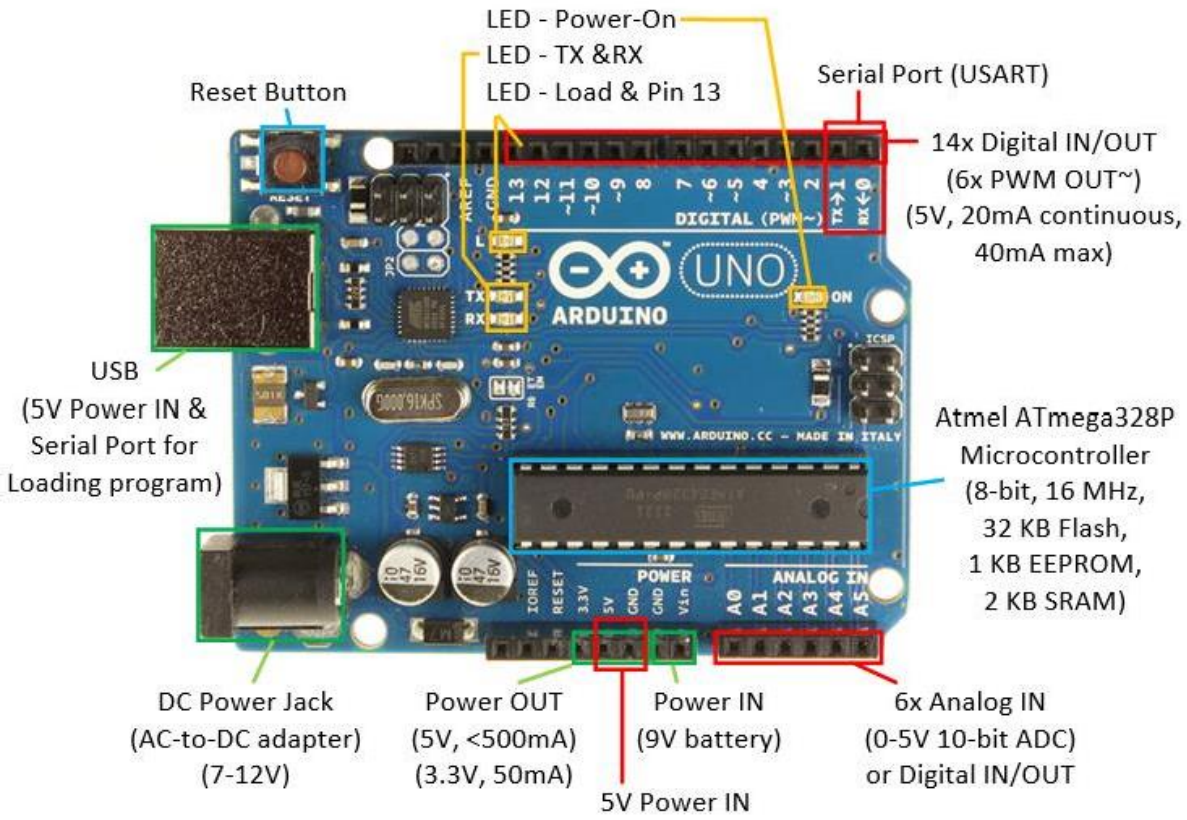


Arduino Leonardo

The Leonardo is Arduino‟s first development board to use one microcontroller with built-in USB. This means that it can be cheaper and simpler. Also, because the board is handling USB directly, code libraries are available which allow the board to emulate a computer keyboard, mouse, and more. The Leonardo uses the 32U4 as the main microcontroller, which has a USB interface built in. Therefore, it doesn‟t need a secondary MCU to perform the serial-to-USB conversion. This cuts down on the cost and enables us to do unique things like

	emulate a joystick or a keyboard instead of a simple serial device. We will learn how to use these features in Chapter 6, "USB and Serial Communication."
	Unlike all the other Arduino variants, which use 8-bit AVR MCUs, the Due uses a 32-bit ARM Cortex M3 SAM3X MCU. The Due offers higher-precision ADCs, selectable resolution pulse-width modulation (PWM), Digital-to-Analog Converters (DACs), a USB host connector, and an 84 MHz clock speed.
	The Nano is designed to be mounted right into a breadboard socket. Its small form factor makes it perfect for use in more finished projects.

Overview of the board (Arduino Uno R3):



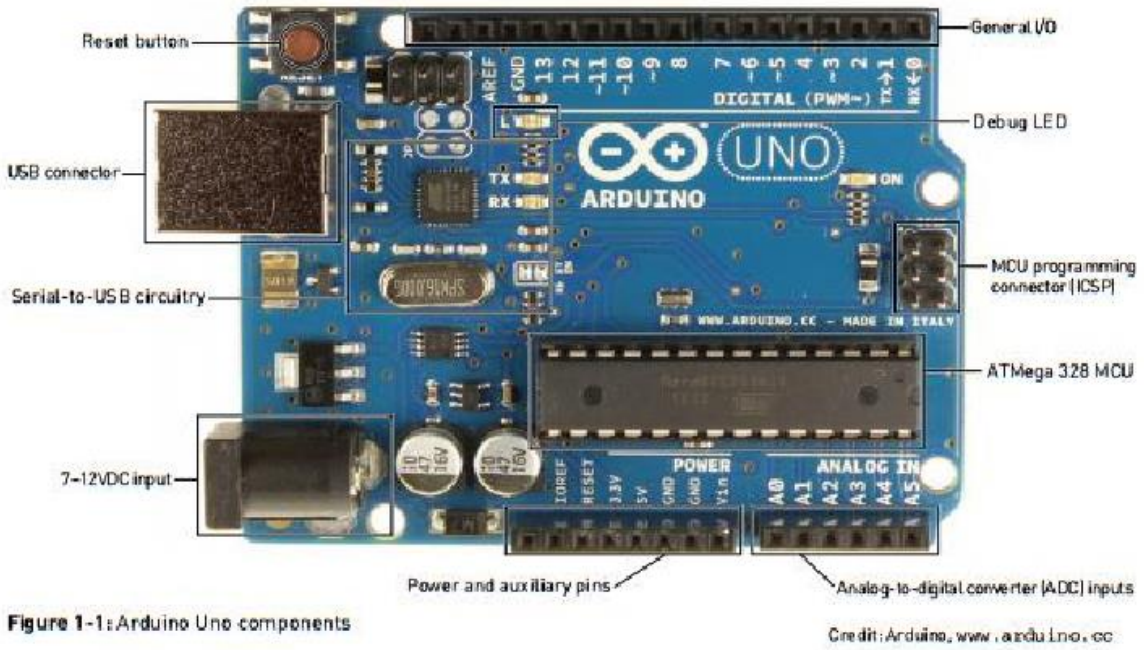


Figure 1-1: Arduino Uno components

Credit: Arduino, www.arduino.cc

1. DC Power Jack and USB connector

Every Arduino board needs a way to be connected to a power source. The Arduino UNO can be powered from a USB cable coming from our computer or a wall power supply that is terminated in a barrel jack.

2. Pins (5V, 3.3V, GND, Analog, Digital, PWM, AREF):

The pins on our Arduino are the places where we connect wires to construct a circuit (probably in conjuction with a breadboard and some wire. They usually have black plastic „headers" that allow us to just plug a wire right into the board. The Arduino has several different kinds of pins, each of which is labeled on the board and used for different functions.

a. **GND** : Short for „Ground". There are several GND pins on the Arduino, any of which can be used to ground our circuit.
b. **5V & 3.3V** : As we might guess, the 5V pin supplies 5 volts of power, and the 3.3V pin supplies 3.3 volts of power. Most of the simple components used with the Arduino run happily off of 5 or 3.3 volts.
c. **Analog**: The area of pins under the „Analog In" label (A0 through A5 on the UNO) are Analog Input pins. These pins can read the signal from an analog sensor (like a temperature sensor) and convert it into a digital value that we can read.
d. **Digital** : Across from the analog pins are the digital pins (0 through 13 on the UNO). These pins can be used for both digital input (like telling if a button is pushed) and digital output (like powering an LED).
e. **PWM** : We may have noticed the tilde (~) next to some of the digital pins (3, 5, 6, 9, 10, and 11 on the UNO). These pins act as normal digital pins, but can also be used for something called Pulse-Width Modulation (PWM, think of these pins as being able to simulate analog output (like fading an LED in and out).
f. **AREF** : Stands for Analog Reference. Most of the time we can leave this pin alone. It is sometimes used to set an external reference voltage (between 0 and 5 Volts) as the upper limit for the analog input pins.

3. Reset Button:

Just like the original Nintendo, the Arduino has a reset button . Pushing it will temporarily connect the reset pin to ground and restart any code that is loaded on the Arduino. This can be very useful if our code doesn't repeat, but we want to test it multiple times. Unlike the original Nintendo however, blowing on the Arduino doesn't usually fix any problems.

4. Power LED indicator:

Just beneath and to the right of the word "UNO" on our circuit board, there's a tiny LED next to the word „ON". This LED should light up whenever we plug our Arduino into a power source. If this light doesn't turn on, there's a good chance something is wrong. Time to re-check your circuit.

5. TX RX LEDs:

TX is short for transmit, RX is short for receive. These markings appear quite a bit in electronics to indicate the pins responsible for serial communication. In our case, there are two places on the Arduino UNO where TX and RX appear – once by digital pins 0 and 1, and a second time next to the TX and RX indicator LEDs. These LEDs will give us some nice visual indications whenever our Arduino is receiving or transmitting data (like when we're loading a new program onto the board).

6. Main IC:

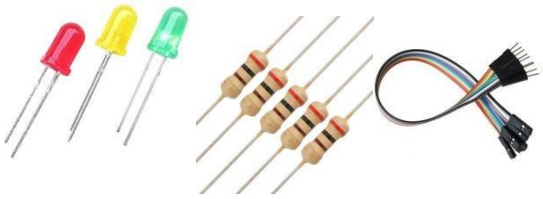
The black thing with all the metal legs is an IC, or Integrated Circuit . Think of it as the brains of our Arduino. The main IC on the Arduino is slightly different from board type to board type but is usually from the ATmega line of IC's from the ATMEL company. This can be important, as we may need to know the IC type (along with your board type) before loading up a new program from the Arduino software.

7. Voltage Regulator:

The voltage regulator is not actually something we can (or should) interact with on the Arduino. But it is potentially useful to know that it is there and what it's for. The voltage regulator does exactly what it says – it controls the amount of voltage that is let into the Arduino board. Think of it as a kind of gatekeeper; it will turn away an extra voltage that might harm the circuit. Of course, it has its limits, so don't hook up our Arduino to anything greater than 20 volts.

Apparatus:

- LED lights (RED, GREEN and YELLOW)
- Three 220 ohms resisters and
- Jumper wires



CODE for Traffic Control system :

```
void setup()
{
  pinMode(13, OUTPUT);
  pinMode(8, OUTPUT);
  pinMode(7, OUTPUT);
}

void loop()
{
  digitalWrite(13, HIGH);
  delay(4000); // Wait for 4000 millisecond(s)
  digitalWrite(13, LOW);
  delay(2000); // Wait for 2000 millisecond(s)

  digitalWrite(8, HIGH);
  delay(3000); // Wait for 3000 millisecond(s)
  digitalWrite(8, LOW);
  delay(2000); // Wait for 2000 millisecond(s)

  digitalWrite(7, HIGH);
  delay(4000); // Wait for 4000 millisecond(s)
  digitalWrite(7, LOW);
  delay(2000); // Wait for 2000 millisecond(s)
}
```

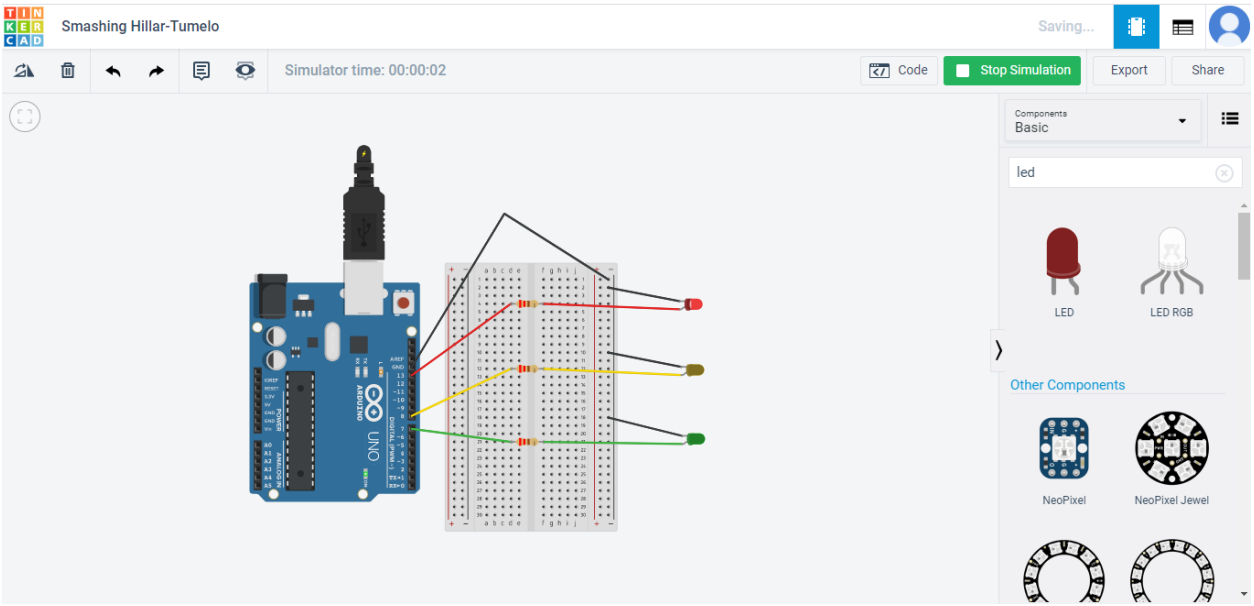
Experimental Procedure :

1. First, we open the online Tinkercad and then we go to the Circuits-> Create new Circuits -> then took the arduinounor3, small size breadboard, three LED (Red, Yellow, Green), three resistors (220 ohm) and rotate them to design comfortably. Then we connected the three resistors with arduinounor3 board digital pin 7,8,13 by using small breadboard. After that we shorted the arduinounor3 ground pin with the three LED and finally we have done our circuit connection.

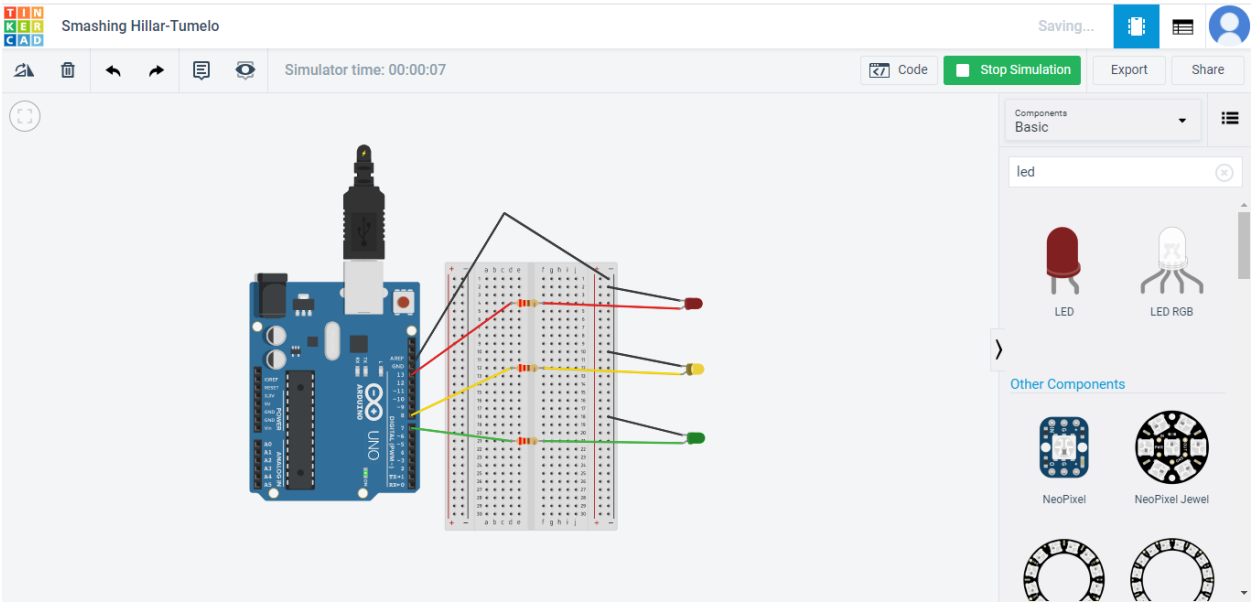
2. Second, we have done the code in Tinkercad code section. At this section, we have done the setup and loop function. In the setup function we took three outputs as pinMode (13, 8, 7) and then in the loop function, at first, we digitaWrite for pin 13 as Red LED and kept it high and waiting for 4000 millisecond(s) and then digitaWrite for pin 13 as Red LED and kept it low and waiting for 2000 millisecond(s) then we digitaWrite for pin 8 as Yellow LED and kept it high and waiting for 4000 millisecond(s) and then digitaWrite for pin 8 as Yellow LED and kept it low and waiting for 2000 millisecond(s) and then we digitaWrite for pin 7 as Green LED and kept it high and waiting for 4000 millisecond(s) and then digitaWrite for pin 7 as Green LED and kept it low and waiting for 2000 millisecond(s).

   Finally, after building the project we pressed Start Simulation and seen the results.

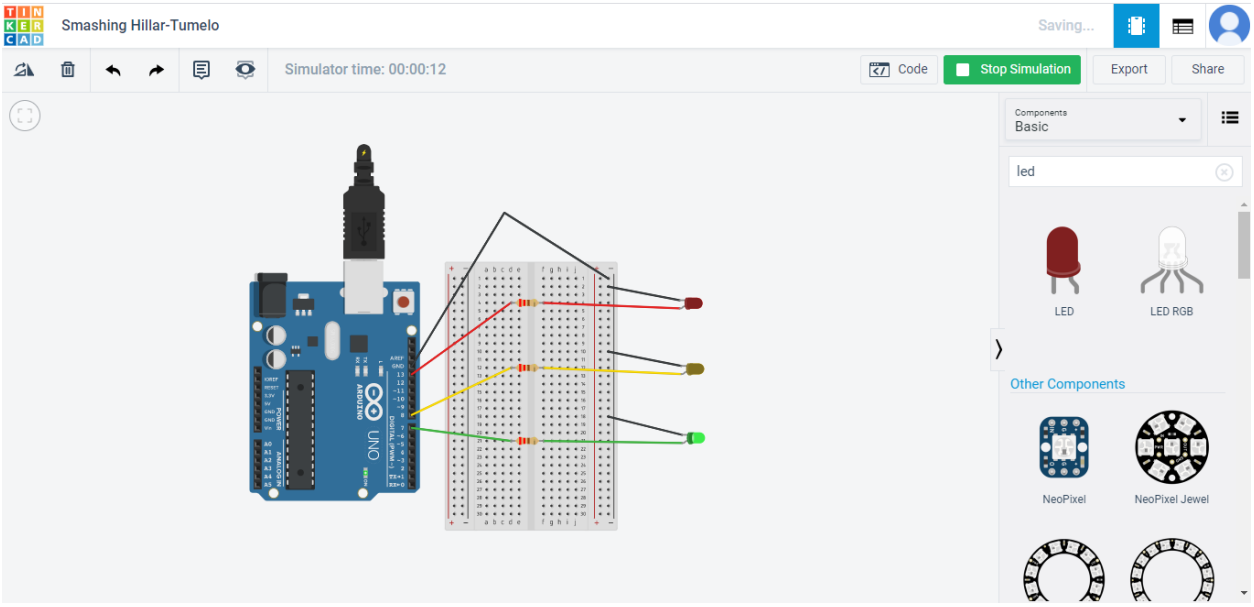
Results :



Red LED on for 4000ms.



Yellow LED on for 3000ms.



Green LED on for 4000ms.

Discussion :

In the result section we found that the Red LED on for 4000ms, Yellow LED on for 3000ms and Green LED on for 4000ms.

Referance(s) :

1) https://www.arduino.cc/.
2) https://www.coursera.org/learn/arduino/lecture/ei4ni/1-10-first-glance-at-a-program
3) Jeremy Blue; Exploring Arduino: Tools and Techniques for Engineering Wizardry