**TRAFFIC MANAGEMENT USING IOT**

**PHASE 3**

**SENSOR DESIGN SIMULATION USING WOKWI**

**INTRODUCTION:**

In phase 3, we have used the Arduino Uno in designing a circuit to manage traffic using IOT. The simulation of the circuit was done using WOKWI simulator. Simulations are also an essential tool for testing and refining Intelligent Transportation Systems (ITS) and Internet of Things (IoT) solutions, such as smart traffic lights, dynamic traffic signal control, and adaptive traffic management systems.

**REQUIREMENTS:**

Creating a traffic management system in an online simulation platform like Wokwi requires multiple components, including a microcontroller (such as an Arduino) and a way to visualize and control the system like LEDS and a pedestrian pushbutton.

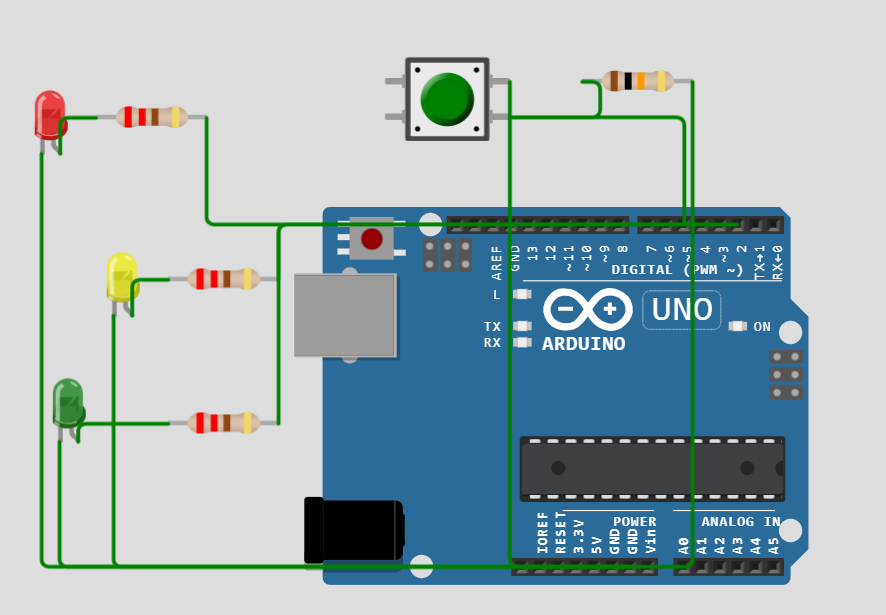
* Arduino Uno:

The Arduino Uno is an [open-source](https://en.wikipedia.org/wiki/Open-source) [microcontroller board](https://en.wikipedia.org/wiki/Single-board_microcontroller) based on the [Microchip](https://en.wikipedia.org/wiki/Microchip_Technology) [ATmega328P](https://en.wikipedia.org/wiki/ATmega328P) [microcontroller](https://en.wikipedia.org/wiki/Microcontroller) (MCU) and developed by [Arduino.cc](https://en.wikipedia.org/wiki/Arduino) and initially released in 2010.[[2]](https://en.wikipedia.org/wiki/Arduino_Uno#cite_note-2)[[3]](https://en.wikipedia.org/wiki/Arduino_Uno#cite_note-What_is_Arduino?-3) The [microcontroller board](https://en.wikipedia.org/wiki/Single-board_microcontroller) is equipped with sets of digital and analog [input/output](https://en.wikipedia.org/wiki/Input/output) (I/O) pins that may be interfaced to various [expansion boards](https://en.wikipedia.org/wiki/Expansion_board) (shields) and other circuits



* LEDS: to represent the traffic lights
* Breadboard and Wires
* Resistors: 220Ω(3N) , 10KΩ(1N)

**CIRCUIT DESIGN:**



**PROGRAM:**

const int redPin = 2;

const int yellowPin = 3;

const int greenPin = 4;

const int pedestrianButtonPin = 5;

void setup() {

  pinMode(redPin, OUTPUT);

  pinMode(yellowPin, OUTPUT);

  pinMode(greenPin, OUTPUT);

  pinMode(pedestrianButtonPin, INPUT\_PULLUP);

}

void loop() {

  // Normal traffic light sequence

  digitalWrite(redPin, HIGH);

  delay(5000);

  digitalWrite(redPin, LOW);

  digitalWrite(yellowPin, HIGH);

  delay(2000);

  digitalWrite(yellowPin, LOW);

  digitalWrite(greenPin, HIGH);

  delay(5000);

  digitalWrite(greenPin, LOW);

  // Check if the pedestrian button is pressed

  if (digitalRead(pedestrianButtonPin) == LOW) {

    // Pedestrian crossing sequence

    digitalWrite(redPin, HIGH);

    delay(2000);

    digitalWrite(redPin, LOW);

    delay(500);

    digitalWrite(yellowPin, HIGH);

    delay(1000);

    digitalWrite(yellowPin, LOW);

    delay(500);

    digitalWrite(redPin, HIGH);

    delay(2000);

    digitalWrite(redPin, LOW);

  }

}

**EXPLANATION:**

The provided Arduino code simulates a basic traffic management system with traffic lights and a pedestrian crossing button.

1. The code begins by defining the pins for the three traffic light LEDs (red, yellow, and green) and the pedestrian button. These pins are connected to the Arduino.

2. In the `setup` function, the pins are configured:

- `redPin`, `yellowPin`, and `greenPin` are set as OUTPUT pins, indicating that they control the traffic light LEDs.

- `pedestrianButtonPin` is set as an INPUT\_PULLUP pin, enabling the internal pull-up resistor. This configuration ensures that when the pedestrian button is not pressed, the pin reads HIGH (due to the pull-up resistor).

3. The `loop` function contains the main logic for the traffic light and pedestrian crossing sequences:

- The traffic light sequence follows a standard pattern: red light for 5 seconds, yellow light for 2 seconds, and green light for 5 seconds. This pattern repeats continuously.

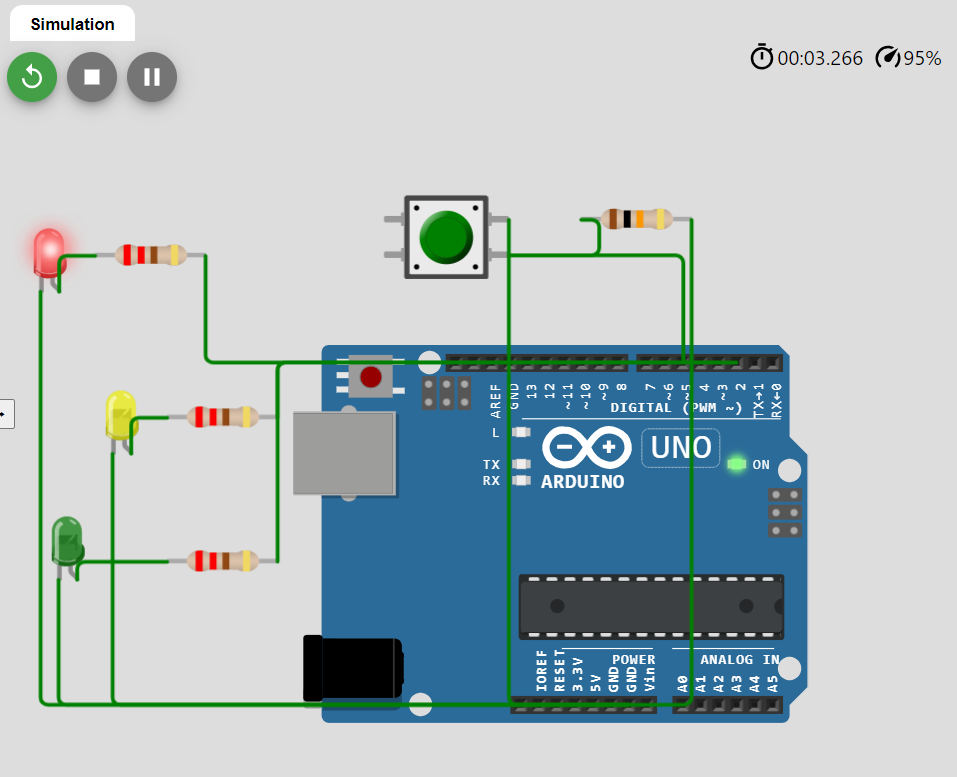
- It then checks if the pedestrian button is pressed by monitoring the state of `pedestrianButtonPin`. If the button is pressed (reads LOW), it initiates the pedestrian crossing sequence.

- During the pedestrian crossing sequence, the red light turns on for 2 seconds, followed by a 0.5-second delay, then the yellow light for 1 second, another 0.5-second delay, and finally the red light for 2 seconds. This sequence allows pedestrians to safely cross the road.

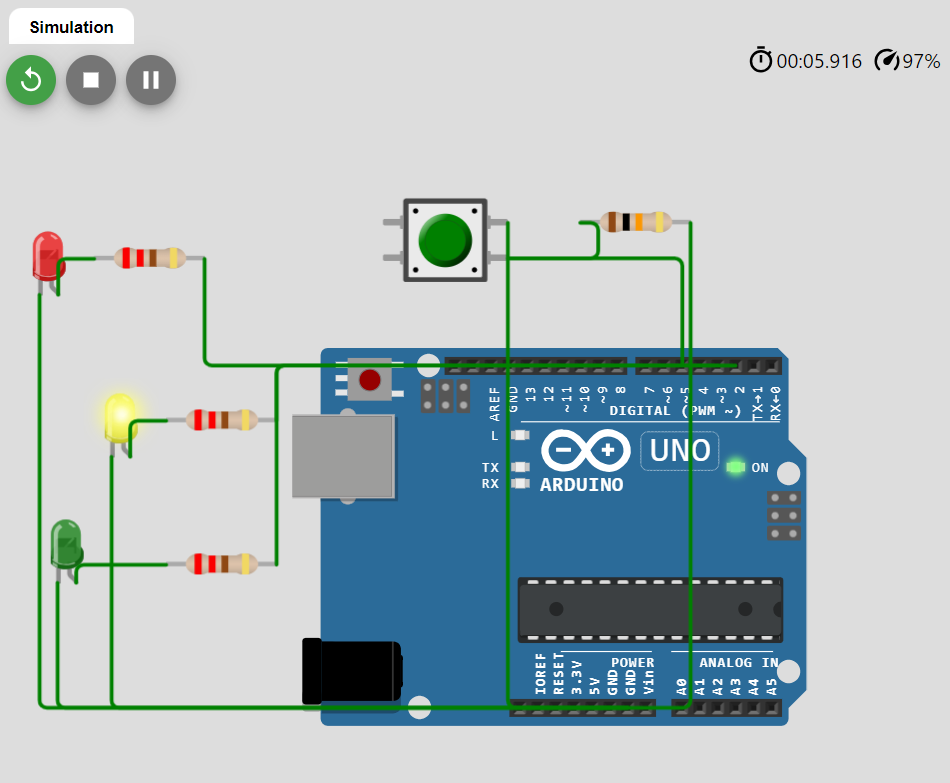
4. The `delay` function is used to control the timing of the different phases in the traffic light and pedestrian crossing sequences.

**OUTPUT:**

Red Light:



Yellow Light:



Green Light:

