Experiment No. 01:

Traffic Light Controller Implementation on Arduino UNO

By: Abhilash Kashyap Balasubramanyam

Lab Project Partner: Adnan Patel

Instructor: Dr. Jafar Saniie

ECE 510

Lab Date: 05-16-2024

Due Date: 05-24-2024

Acknowledgment: I acknowledge all of the work (including figures and codes) belongs to me and/or persons who are referenced.

Signature: 

I. Introduction

A. Purpose

The purpose of this experiment is to acquaint oneself better with the basic operations of the Arduino Board UNO and the Arduino IDE. This experiment also aids an introductory understanding towards the Tinkercad online tool by Autodesk which helps in simulating circuits and testing code's functionality before its actual application. The main project deals with the designing and building of a simple and advanced Traffic Controller Circuit(s) that operate on timing specified in the code and the switch provided.

B. Background

Arduino is an open-source company that designs and manufactures microcontrollers and kits for creating digital devices and interactive objects. These boards can be connected to expansion boards and other circuits using their digital and analog input/output pins. Additionally, they provide serial communication interfaces for program loading, like USB. The most common programming languages for microcontrollers are C and C++. Furthermore, an integrated development environment based on the Processing Language Project is made available through the Arduino project.

Microcontrollers are essential for managing the timing of the lights in complicated traffic signals. It is difficult to coordinate the various signals for turns, pedestrian crossings, and each direction. Timing can be easily adjusted based on traffic sensors and the time of day thanks to microcontrollers. Not all these tasks are appropriate for simple logic circuits. In smaller applications, however, where the expense of a microcontroller system is not warranted, the same functionality can be achieved with a sequential logic circuit.

A seven-segment display effectively depicts numbers ranging from 0 to 9. Ten inputs total seven segments for the digits, a decimal point segment, and power pins. LEDs are positioned in a suitable

way on the display to show numbers. To protect the device, a resistor (minimum 220 Ω) must be used for each input, excluding the power pins since it does not include resistors at input points.

Simple Two-Way Intersection Traffic Light

In this laboratory experiment, a traffic signal at a basic intersection is controlled by a logic. There are two roads at the intersection: an East/West Road and a North/South Road. Three lights are present on each traffic signal: green for forward motion, yellow for changing, and red for stop. A timer operates the lights without the need for any extra sensor data. Compared to the East/West Road (five time-units), the North/South Road receives longer green lights (ten time-units) due to its higher traffic volume. A green light in one direction indicates a red light in the other. The change from green to red is signaled by a yellow light, and for one moment only, all directions have a red light for safety.

Two-Way Intersection with Traffic Sensors

The prior traffic light system is insufficient to alter traffic patterns in the second half of this experiment. The prior scheduling may have resulted in needless delays if there was more traffic on the North/South street but less on the East/West street. The North/South signal shouldn't become red if no cars are stopped at the East/West street's red light. At the intersection, automobile sensors are installed to adjust to changes in traffic and guarantee seamless operation when there are no waiting cars.

II. Lab Procedure and Equipment List

A. Equipment

Equipment

- 1 x Arduino UNO board and USB Cable
- 2 x Red LED
- 2 x Yellow LED
- 2 x Green LED
- 2 x Pushbutton
- 6 x 220 ohm & 2 x 10k ohm Resistors
- 1 x Breadboard

B. Procedure

Simple Two-Way Intersection Traffic Light

1. Using the list of components, the circuit is built including 6 LED bulbs for each intersection of colors red, yellow, green (two of each).
2. The code to run the program as mentioned below is loaded on to the Arduino IDE which is then uploaded onto the Arduino UNO board.
3. The results are then recorded and presented.

Two-Way Intersection with Traffic Sensors

1. Using the list of components, the circuit is built including 6LED bulbs for each intersection of colors is red, yellow, green (two of each), pushbutton switches that doubles down as automobile sensors are also included.
2. The court to run the program as mentioned below is loaded on the Arduino IDE which is then uploaded onto the Arduino UNO board.
3. The results are checked by clicking on the pushbuttons which doubles down as automobile sensors. Based on the addition of input of these pushbuttons the appropriate results are then recorded.

III. Code for main experiments

Simple Two-Way Intersection Traffic Light

```
//North-South LEDs
#define r_ns 2
#define y_ns 3
#define g_ns 4

//East-West LEDs
#define r_ew 5
#define y_ew 6
#define g_ew 7

//Definition of running speed
#define time_base 500

void setup()
{
  //Initializing all LEDs to be OUTPUT
  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);
  pinMode(6, OUTPUT);
  pinMode(7, OUTPUT);
}

void ChangeLedValue(byte number)
{
  digitalWrite(r_ns, bitRead(number, 5));
  digitalWrite(y_ns, bitRead(number, 4));
  digitalWrite(g_ns, bitRead(number, 3));
  digitalWrite(r_ew, bitRead(number, 2));
  digitalWrite(y_ew, bitRead(number, 1));
  digitalWrite(g_ew, bitRead(number, 0));
}

void LightSequence()
{
beginning:
  ChangeLedValue(B001100);
  delay(time_base * 10);

  ChangeLedValue(B010100);
  delay(time_base * 1);

  ChangeLedValue(B100100);
  delay(time_base * 1);

  ChangeLedValue(B100001);
  delay(time_base * 5);

  ChangeLedValue(B100010);
  delay(time_base * 1);

  ChangeLedValue(B100100);
  delay(time_base * 1);
}

void loop()
{
  LightSequence()
}
```

Figure 1: Code for Simple Traffic Light demonstration.

Two-Way Intersection with Traffic Sensors

```
//North-South LEDs
#define r_ns 2
#define y_ns 3
#define g_ns 4

//East-West LEDs
#define r_ew 5
#define y_ew 6
#define g_ew 7

//Sensors
#define as_w 12
#define as_e 13

//Definition of running speed
#define time_base 500

void setup()
{
  //Initializing all LEDs to be OUTPUT
  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);
  pinMode(6, OUTPUT);
  pinMode(7, OUTPUT);
  pinMode(12, INPUT);
  pinMode(13, INPUT);
}

void ChangeLedValue(byte number)
{
  digitalWrite(r_ns, bitRead(number, 5));
  digitalWrite(y_ns, bitRead(number, 4));
  digitalWrite(g_ns, bitRead(number, 3));
  digitalWrite(r_ew, bitRead(number, 2));
  digitalWrite(y_ew, bitRead(number, 1));
  digitalWrite(g_ew, bitRead(number, 0));
}

void LightSequence()
{
beginning:
  ChangeLedValue(B001100);
  long start = millis();
  long timeelapsed = 0;
  long waittime = time_base * 10;
  while(timeelapsed < waittime)
  {
    if (not(digitalRead(as_e) or digitalRead(as_w)))
    {
      goto beginning;
    }
  timeelapsed = millis() - start;
  }

  ChangeLedValue(B010100);
  delay(time_base * 1);

  ChangeLedValue(B100100);
  delay(time_base * 1);

  ChangeLedValue(B100001);
  delay(time_base * 5);

  ChangeLedValue(B100010);
  delay(time_base * 1);

  ChangeLedValue(B100100);
  delay(time_base * 1);
}

void loop()
{
  LightSequence();
}
```

Figure 2: Code for Traffic Intersection with Automobile Sensors.

III. Results and Demonstration

The results and the demo video is available at this Link.

A. Discussion

1. The following reasons state the need for and against the use of Arduino for the experiment:

With Arduino	Without Arduino
1. **Simple programming**: The Arduino IDE, which is user-friendly for beginners and supports the C and C++ programming languages, makes it simple to program Arduino boards.	1. Limited capabilities: The memory, computing power, and I/O pins of Arduino boards are specifically limited. For complex traffic management systems, specialized hardware may occasionally provide higher performance and more sophisticated functionality.
2. **Flexibility and expandability**: The traffic light controller system may be customized and expanded thanks to the digital and analog input/output pins on Arduino boards, which can be interfaced with a variety of expansion boards and circuits.	2. **Cost considerations**: Although Arduino boards are reasonably priced, there can be some situations in which the expense of specialized hardware makes sense. Dedicated hardware may offer more customized capabilities and greater value for money in large-scale projects or specialist applications.
3. **Integration with sensors**: In order to adjust to shifting traffic patterns, Arduino boards may readily interface with sensors, including automotive sensors. This makes it possible for the traffic light controller to optimize traffic flow and react dynamically.	3. **Specific requirements**: Hardware that is not easily accessible inside the Arduino ecosystem may be required, depending on the specifications of the project. In order to ensure optimal performance and compatibility with other components, dedicated hardware might be specifically built and configured to fit specific needs.

2. To inculcate a free left turn signal for all the four intersections, the following steps can be taken to achieve the same:
    a. There needs to be an addition of two more sets of lamps (2 x red, 2 x yellow, 2 x green). This is added with one more green lamp (dedicated for left turn) at each intersection (4 x green - additional).
    b. The code is to be modified such as when the north and south intersection is supposed to be green (10 units in time), the north intersection is first set to green for half the time (5 units in time) and then north is set to red while the south is made to turn green for the remaining half the time (5 units in time).
    c. Subsequently, the east and west intersection repeats the same logic that is like the north and south intersection.

3. The following is the implementation of an accelerating LED performed on the Tinkercad platform:

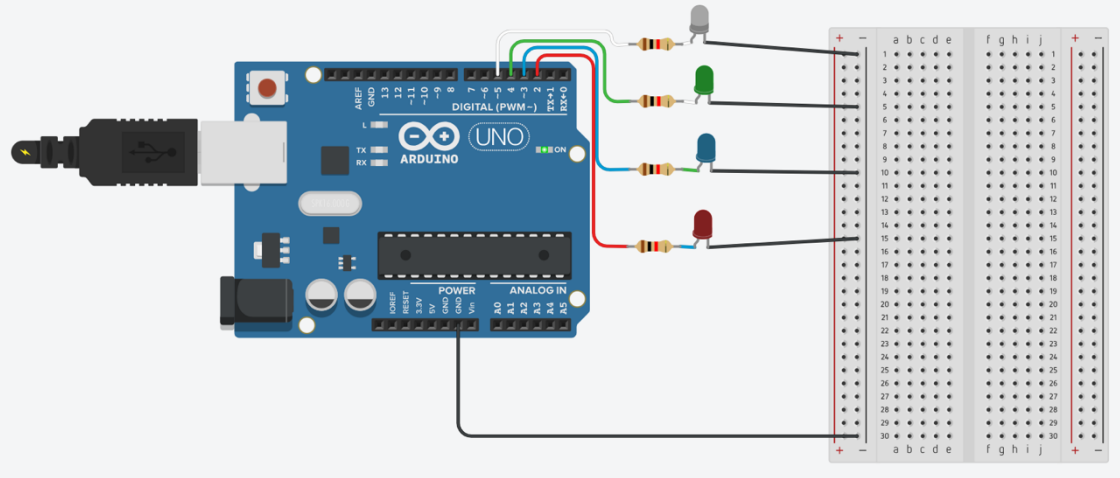

Figure 3: Screenshot of Tinkercad circuit for accelerating LEDs.

```
// Define the LED pins
const int ledPins[] = {2, 3, 4, 5};

// Define the delay times in milliseconds
const int delays[] = {500, 400, 300, 200, 100};

void setup() {
  // Set the LED pins as OUTPUT
  for (int i = 0; i < 4; i++) {
    pinMode(ledPins[i], OUTPUT);
  }
}

void loop() {
  // Blink each LED in order
  for (int i = 0; i < 4; i++) {
    digitalWrite(ledPins[i], HIGH);
    delay(delays[0]);
    digitalWrite(ledPins[i], LOW);
    delay(delays[0]);
  }

  // Accelerate the blink rate
  for (int i = 1; i < 5; i++) {
    for (int j = 0; j < 4; j++) {
      digitalWrite(ledPins[j], HIGH);
      delay(delays[i]);
      digitalWrite(ledPins[j], LOW);
      delay(delays[i]);
    }
  }
}
```

Figure 4: Code for accelerating LEDs.

IV. Additional Experiments

The additional experiments are also found in the Link.

V. Conclusions

Therefore, the purpose of the experiment is accomplished. Through the implementation, analysis and modification of the simple and automobile sensor-based traffic control system, the fundamentals of familiarizing with the Arduino board and the Arduino IDE has been accomplished. Additionally, the use of Tinkercad is also understood with successful implementations of the same. This will aid towards the overall understanding of IoT and its devices for the project and future laboratory experiments as well.

References

[1] Experiment 01 Lab Manual