IR Remote Controlled LED Lamp

***Abstract* — This project report features the development of a remote-controlled LED lamp, which comprises 4 White LEDs and one multi-color LED, using an Arduino UNO R3 and a simple IR remote.**

I. Introduction

Light Emitting Diodes, commonly known as LEDs have become a prevailing lighting solution in the present world due to their energy efficiency, long lifespan, and versatility. This project proposes designing, developing, and implementing a remote-controlled LED lamp, utilizing an Arduino UNO R3 microcontroller, and an easily available Infrared (IR) Remote control. The lamp comprises five LEDs; four White LEDs and a single multi-color (RGB) LED. The state of the LED lamp is controlled by the user via the IR remote, using the appropriate channel or volume up/down buttons.

This project highlights feasible functionalities by combining the related components that produce an embedded system. The designated design offers a user-friendly acquaintance with inherent button presses and demonstrates the control of the LED lamp. The subsequent section will discuss the project's components, design methodology, code implementation, and testing procedures.

II. Design Methodology

The design methodology discusses the proceeding with the implementation of the circuitry which indentures a combination of hardware selection, initial circuit design, and software selection for code implementation.

A. Hardware Selection

Aspects that were taken into consideration for hardware selection are user-friendliness, functionality, and cost-efficiency. The Arduino UNO R3 microcontroller serves as the computational center due to its beginner-friendly platform, extensive libraries that support the designated procedures, and ample computing power that controls the states of the LEDs. An appropriate IR receiver module, unanimous with a standard IR remote control, is chosen to decode the user input IR signal from the remote. Four white LEDs were chosen to provide basic illumination, and one multi-color (RGB) LED was chosen to provide a range of colorful lighting effects. To prevent the LEDs from being damaged due to excess current, 220-ohm resistors are embodied into each LED to ensure the safe and proper operation of the LED lamp. This hardware selection established a balanced functionality, convenient use, and cost-efficiency, resulting in a robust foundation for the remote-controlled LED Lamp.

B. Initial Circuit Design

The initial circuit design highlights the establishment of a connection between the Arduino UNO board and the appropriate components. All the LEDs were planned for connection in a parallel configuration, enabling all the LEDs to connect and use a common ground on the Arduino Board. PWM pins 5, 6, 9, and 10 on the Arduino board are particularly allotted to the 4 White LEDs sequentially. This specific allotment was designated to allow the user to control and adjust the brightness of the four White LEDs using the IR remote. Pins 11, 12, and 13 are allotted to the multi-color (RGB) LED correspondingly connecting to the RED pin, GREEN pin, and BLUE pin of the RGB LED. To prevent potential reimbursement for the LEDs due to excess current, 220-ohm resistors were encompassed in the initial circuit design, planned for connection with the positive leg of each LED, in a serial configuration. The IR Receiver module is equipped with three designated pin configurations, (i) VCC pin connected to the 5V pin on Arduino, (ii) GND pin connected to the GND pin on Arduino, and (iii) SO (Signal Out) pin connected to Analog Input A0 pin on Arduino.

C. Software Selection

To implement the code for the proper operation of the LED Lamp, Microchip Studio, a user-friendly Integrated Development Environment (IDE) is chosen. This software immaculately integrates with Arduino boards and provides a solidly built platform for developing the project's functionalities. Microchip Studio provides the facilities to execute Arduino C++ programs, eliminating the necessity for supplementary language setup. In addition, Microchip Studio supports built-in libraries for interrelation with hardware components. This adequate environment enables efficient code development, testing, debugging, and project management feasible across the development phase.

III. Code Implementation

```
#include <Arduino.h>
#include <IRremote.h>

//============= External Functions pre-definitions =============
void RGB_OFF();
void RED();
void GREEN();
void BLUE();
void RED_GREEN();
void RED_BLUE();
void GREEN_BLUE();
void RGB_WHITE();
//End

#define CH_UP 16769565     // Decimal code of Channel UP
#define CH_DOWN 16753245   // Decimal code of Channel DOWN
#define VOL_UP 16754775    // Decimal code of Volume UP
#define VOL_DOWN 16769055  // Decimal code of Volume UP

#define R 11
#define G 12
#define B 13
int whiteLED[4] = { 5, 6, 9, 10 };

const int RECEIVER_PIN = A0;
int w_val = 0;
int rgb_val = 0;
int rgb_state = 0;
int rgb_cycle = -1;
int brightness = 255;

IRrecv irrecv(RECEIVER_PIN);
decode_results results;
```

```
void setup() {
  //========== Defining all LED pins as OUTPUT pins ==========
  for (int i = 0; i < 4; i++) {
    pinMode(whiteLED[i], OUTPUT);
  }

  for (int i = 11; i <= 13; i++) {
    pinMode(i, OUTPUT);
  }

//========== Upon start-up turn all LEDs OFF ==========
  for (int i = 0; i < 4; i++) {
              digitalWrite(whiteLED[i], LOW);
  }

  for (int i = 11; i <= 13; i++) {
              digitalWrite(i, LOW);
  }

  irrecv.enableIRIn();  // Start the IR receiver
}

void loop() {
  if (irrecv.decode(&results)) {

  //========== Turn ON the LEDs ==========

    if (results.value == CH_UP) {
      if (w_val >= 0 && w_val < 4) {
        analogWrite(whiteLED[w_val], brightness);
        w_val++;
      } else if (w_val == 4 && rgb_val >= 0 && rgb_val < 7) {
        rgb_state = 1;
        switch (rgb_val) {
          case 0:
            RED();
            rgb_val++;
            rgb_cycle++;
            break;

          case 1:
            GREEN();
            rgb_val++;
            break;

          case 2:
            BLUE();
            rgb_val++;
            break;

          case 3:
            RED_GREEN();
            rgb_val++;
            break;

          case 4:
            RED_BLUE();
            rgb_val++;
            break;

          case 5:
            GREEN_BLUE();
            rgb_val++;
            break;

          case 6:
            RGB_WHITE();
            rgb_val = 0;
            break;
        }
      }
    }

//==========Turn OFF the LEDs ==========

if (results.value == CH_DOWN) {
      if (w_val == 4 && rgb_state == 1) {
        if(rgb_cycle > 0){
          switch(rgb_val){
            case 0:
              GREEN_BLUE();
              rgb_val=6;
              break;

            case 1:
              RGB_WHITE();
              rgb_val--;
              rgb_cycle--;
              break;

            case 2:
              RED();
              rgb_val--;
              break;

            case 3:
              GREEN();
              rgb_val--;
              break;

            case 4:
              BLUE();
              rgb_val--;
              break;

            case 5:
              RED_GREEN();
              rgb_val--;
              break;

            case 6:
              RED_BLUE();
              rgb_val--;
              break;
          }
        }
        if (rgb_cycle == 0) {
          switch (rgb_val) {
            case 0:
              GREEN_BLUE();
              rgb_val = 6;
              break;

            case 6:
              RED_BLUE();
              rgb_val--;
              break;

            case 5:
              RED_GREEN();
              rgb_val--;
              break;

            case 4:
              BLUE();
              rgb_val--;
              break;

            case 3:
              GREEN();
              rgb_val--;
              break;

            case 2:
              RED();
              rgb_val--;
              break;

            case 1:
              RED();
              rgb_val--;
              rgb_cycle--;
              break;
          }
        }
        if (rgb_cycle == -1){
          RGB_OFF();
          rgb_state = 0;
        }
      } else if (w_val <= 4 && w_val > 0) {
        analogWrite(whiteLED[w_val - 1], 0);
        w_val--;
      }
    }

//========== Increase the Brightness of White LEDs ==========

if (results.value == VOL_UP) {
      if (brightness > 0 && brightness <= 225) {
        brightness += 25;
        for (int i = 0; i < w_val; i++) {
          analogWrite(whiteLED[i], brightness);
        }
      }
    }

//========== Reduce the Brightness of White LEDs ==========

if (results.value == VOL_DOWN) {
      if (brightness <= 255 && brightness >= 25) {
        brightness -= 25;
        for (int i = 0; i < w_val; i++) {
          analogWrite(whiteLED[i], brightness);
        }
      }
    }

    irrecv.resume();  // Prepare to receive the next value
  }
}  // End of Main (Loop) Function


//=================== RGB OFF ===================
void RGB_OFF() {
  digitalWrite(R, LOW);
  digitalWrite(G, LOW);
  digitalWrite(B, LOW);
}
```

```
//==================== RGB RED ====================
void RED() {
  digitalWrite(R, HIGH);
  digitalWrite(G, LOW);
  digitalWrite(B, LOW);
}

//==================== RGB GREEN ====================
void GREEN() {
  digitalWrite(R, LOW);
  digitalWrite(G, HIGH);
  digitalWrite(B, LOW);
}

//==================== RGB BLUE ====================
void BLUE() {
  digitalWrite(R, LOW);
  digitalWrite(G, LOW);
  digitalWrite(B, HIGH);
}

//==================== RGB YELLOW ====================
void RED_GREEN() {
  digitalWrite(R, HIGH);
  digitalWrite(G, HIGH);
  digitalWrite(B, LOW);
}

//==================== RGB MAGENTA ====================
void RED_BLUE() {
  digitalWrite(R, HIGH);
  digitalWrite(G, LOW);
  digitalWrite(B, HIGH);
}

//==================== RGB CYAN ====================
void GREEN_BLUE() {
  digitalWrite(R, LOW);
  digitalWrite(G, HIGH);
  digitalWrite(B, HIGH);
}

//==================== RGB WHITE ====================
void RGB_WHITE() {
  digitalWrite(R, HIGH);
  digitalWrite(G, HIGH);
  digitalWrite(B, HIGH);
}
```

The code snippet demonstrates the implementation logic for the anticipated IR remote-controlled LED Lamp project. The following figures illustrate the implementation of the provided code snippet in Microchip Studio, the specified Development Environment for writing, compiling, and uploading for the Arduino UNO.

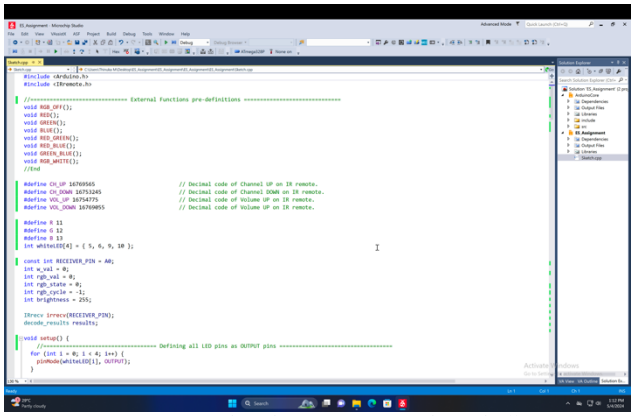

Figure 3. 1 - Code implementation on Microchip Studio (i).

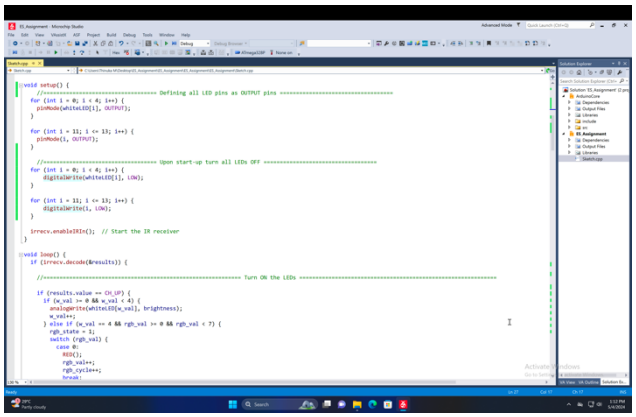

Figure 3. 2 - Code implementation on Microchip Studio (ii).

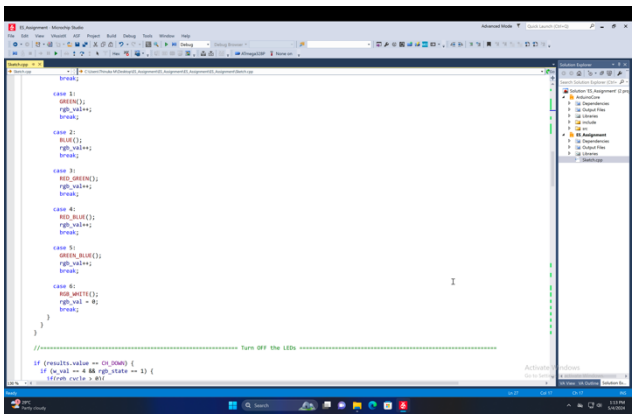

Figure 3. 3 - Code implementation on Microchip Studio (iii).

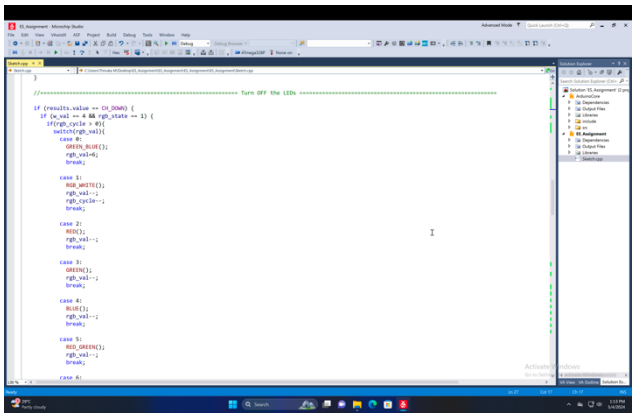

Figure 3. 4 - Code implementation on Microchip Studio (iv).

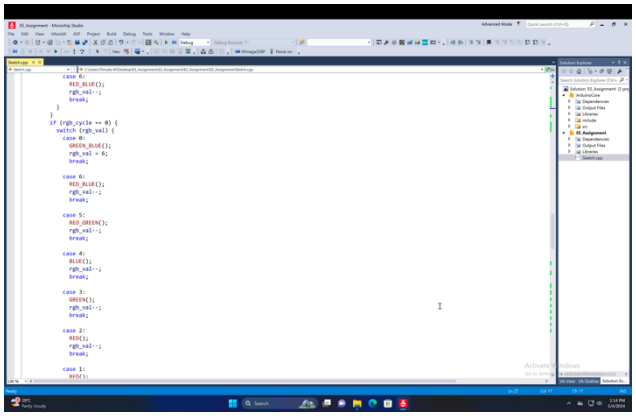

Figure 3. 5 - Code implementation on Microchip Studio (v).

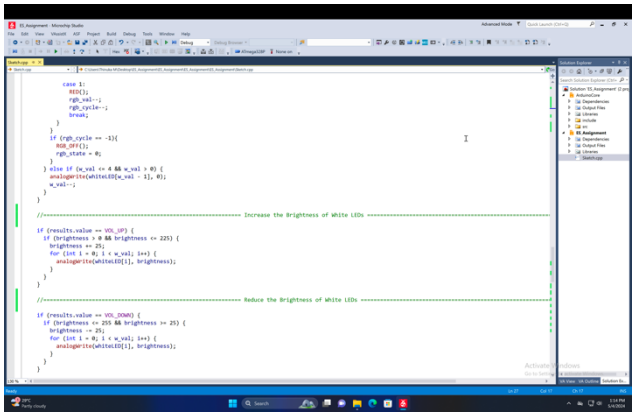

Figure 3. 6 - Code implementation on Microchip Studio (vi).

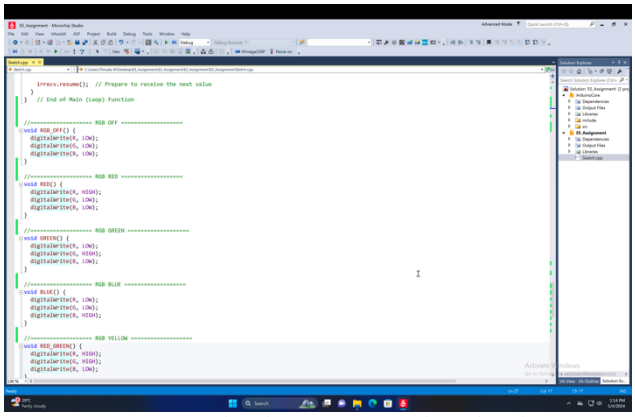

Figure 3. 7 - Code implementation on Microchip Studio (vii).

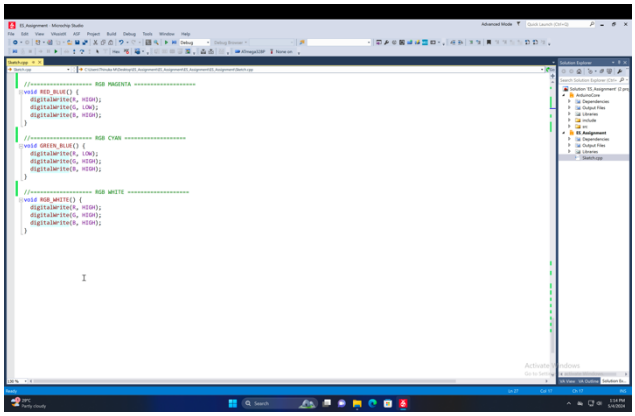

Figure 3. 8 - Code implementation on Microchip Studio (viii).

IV. CIRCUIT DIAGRAM

Upon ascertaining the relevant components to build the project, a lucid circuit diagram is essential for establishing a proper comprehension of the connection between components. To achieve the intended objective, a schematic diagram derived with the aid of the Tinkercad Online platform is presented herewith.

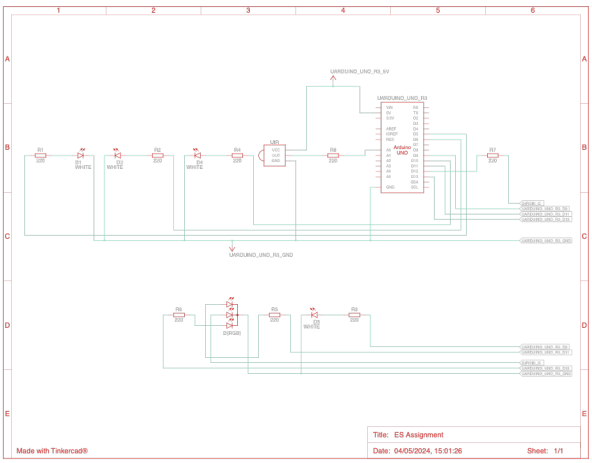

Figure 4. 1 - Schematic Diagram of the Circuit.

Following a profound analysis of the schematic diagram, a prototype of the circuit was developed to examine the proper functionalities of the peripherals before concluding a finalized product integrated onto a dot board. The circuit diagram of the prototype, along with the physical implementation of the prototype, are as follows.

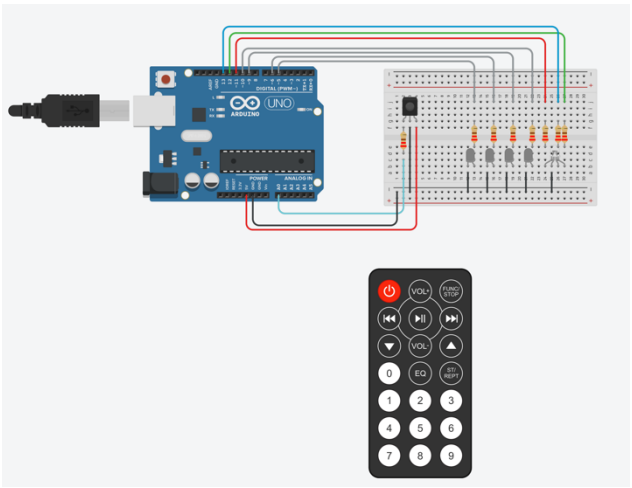

Figure 4. 2 - Circuit Diagram of the prototype.

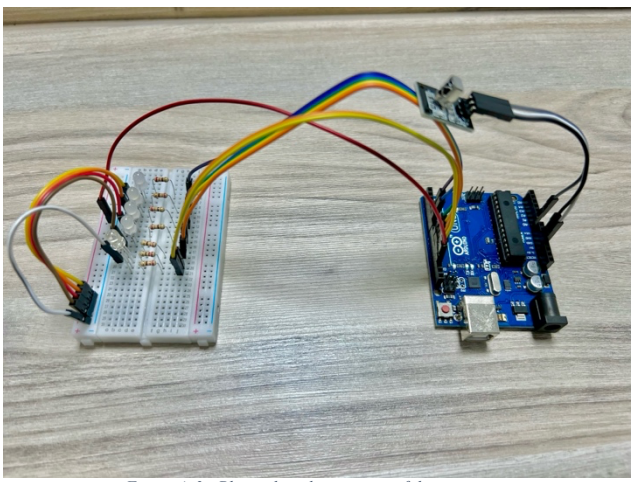

Figure 4. 3 - Physical implementation of the prototype.

Upon careful analysis of the operations and functionalities of the prototype, a finalized user-friendly product was integrated into the dot-board. The physical layout of the finalized product is as follows.

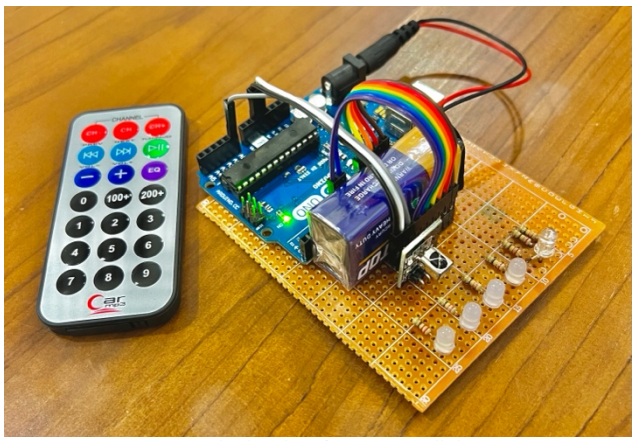

Figure 4. 4 - Finalized product of Remote-controlled LED Lamp.

V. Testing and Validation

Before Integrating the prototype into the breadboard, meticulous testing was conducted on the prototype to ensure the LED Lamp functioned as intended. The following key testing areas were observed during this phase.

A. IR Signal Reception

Testing verified the proper connection of the IR Receiver module with the Arduino UNO and the successful reception of IR signals from the standard IR remote. Furthermore, signal reception of the IR signal from the remote was tested by placing the remote at various distances to ensure the reliability of the user interaction with the lamp despite the remote position.

B. LED Sequence

Button presses on the remote (channel UP and DOWN) were extensively tested to ensure the programmed LED sequence. This included verification of LEDs turning ON upon pressing the channel UP button, multi-color LED toggling available colors upon repeatedly pressing the channel UP button, reversing the toggled sequence of the multi-color LED accordingly upon pressing the channel DOWN button, and finally turning OFF the LEDs sequentially upon pressing the channel DOWN button on the IR remote.

C. Brightness Control

Testing was conducted for the validation of the brightness-controlling functionality for the White LEDs subsequent to the volume UP/DOWN button presses. This includes the verification of the brightness reduction of the four White LEDs upon pressing the volume DOWN button and the brightness increase of the four White LEDs upon pressing the volume UP Button on the IR remote.

Upon conducting these comprehensive tests at each stage of development, potential issues were identified and addressed accordingly to ensure the smooth operation of the LED lamp before transitioning to a finalized product.

VI. Discussion

The project successfully deployed a user-friendly remote-control LED lamp utilizing an Arduino UNO and an IR remote. Under the discussion, potential challenges that were faced during the development of the project, strengths, and limitations of the product are highlighted.

As mentioned, transitioning from the breadboard prototype to a permanent solution required the circuit to be integrated into a dot board. This requires soldering the required components such as the LEDs, resistors, IR receiver module, and connection pins onto the dot board. One of the main challenges faced was to execute precise soldering in connecting the IR received module and the LEDs along with the resistors to the connection pins as depicted in Figure 4.4. Furthermore, in the first attempt after successfully soldering the required components, during the testing phase, it was observed that the RGB LED was damaged, resulting in an obvious reduction of the brightness in the red color. This arose a significant challenge in replacing the RGB LED, by conscientiously removing the damaged RGB LED and precisely soldering the new RGB LED.

From the perspective of the project's outcome, user-friendliness, customizable lighting effects, versatility, cost-efficiency, and mobilization of the lamp can be considered strengths.

It can be acknowledged that the LED Lamp will operate smoothly under the condition that the IR remote must be placed within a distance of one meter from the IR receiver to precisely receive the IR remote signal. Furthermore, to power up the LED lamp, users must carry a power supply such as a 9v battery. These are some of the limitations of the project.

References

1) How to setup an IR Remote and Receiver on an Arduino - https://www.circuitbasics.com/arduino-ir-remote-receiver-tutorial/