IE2070
Embedded Systems
2nd Year, 2nd Semester

Final Report

Assignment (Individual)

Submitted to

Sri Lanka Institute of Information Technology

In partial fulfilment of the requirements for the

Bachelor of Science Special Honors Degree in Information Technology

05/03/2024

Declaration

I certify that this report does not incorporate without acknowledgement, any material previously submitted for a degree or diploma in any university, and to the best of my knowledge and belief it does not contain any material previously published or written by another person, except where due reference is made in text.

Registration Number: IT22571298

Name: P.K.A.Silva

Table of Contents

Figure Table

1. Introduction

In the project, we design a portable illumination device powered by batteries. The operation of the device is facilitated by the infrared remote control. The fundamental will include an Arduino UNO microcontroller board that governs a light-emitting diodes (LEDs) array. First of all, let's examine the basic requirements of the setup which include four white LEDs and one RGB (red, green, blue) LED with different colors.

One can turn on the LEDs in an orderly fashion by pressing ‘volume up’ or ‘channel up’ button on the Infrared remote control device. Such a design enables various lighting alternatives including the cyclic reproduction of a visible spectrum of the white color that is provided by RGB LED. The report has several parts and you can explore the methodology of design, programming as well as testing .

The design phase included selection of the components, a circuit layout planning and finding compatibility between the elements of a circuit. The programming part contained project tasks like writing code for Arduino to detect IR signals, control LED behavior, and implement lighting patterns. Tough testing was conducted to confirm initial capabilities, improve performance characteristics, and define the potential for fine-tuning.

2. Design Methodology

2.2 Components Used

- Arduino Uno Board
- 4 white LEDs
- 1 Multicolor LED
- Breadboard
- Dartboard
- HX-1838 IR receiver with Remote
- Jump wires.
- Resistors(270Ω)
- 9v battery

2.3 Methodology

In the Project requirements, they were expected us to,

- Design an LED light that can be remotely operated using an IR remote controller used at home. The Power for the LED light should be provided using batteries. The Controller for the lights should be an Arduino UNO.

- The light consists of 4 white-coloured LEDs and an RGB LED.

- Use the volume up/down button button to operate the light
- Using the UP button, we should get the following results.

  Off ->Up->1 LED->Up->2 LEDs->Up->3LED s->Up->4 LED s ->Up->Multi Color LED Color 1 -> Up->Multi Color LED Color 2 -> Up->Multi Color LED Color 3-> Up->Multi Color LED Color 1-> Up->Multi Color LED Color 2 -> Up->Multi Color LED Color 3

- And the Down button to reduce brightness and to switch off the light.

So, after considering project requirements and real-world issues in the designing part of the project,

- First, I decided to use PORTB and PORTD registers because a single port cannot be set to hold 7 LEDs. Therefore, I decided to use the 4 ports of PORTD for white LEDs and 3 ports of PORTB for RGB LEDs.
- And as LEDs I used 3.0v white LEDs with 330Ω and 3.0v RGB LED with 100Ω resistors for each.
- Then I used an IR Receiver to take input readings from an IR remote and connected the receiver pin for the PIND3. because external interrupts work in only INT0 and INT1 pins for PORTD.
- I used 5v voltage regulator to keep the voltage from power supply within a range that is compatible with the other electric components.

3. Code Implementation

In the code implementation part, I have come to the following decisions to write the program according to the given instructions,

- I have used Atmel Studio 7 with C programming language in order to implement code of the project.

- First of all, I have used a library file named “IR_Reciever.h” that can be used to decode the commands given from remote control using NEC protocol for Atmega AVR microcontrollers.

- Then in the code I have used Timer0 in CTC mode with no-prescaler to control brightness of the LED using pulse-width-modulation. It’s better to use Timer1 (which provide larger time resolution than other timers) in the program but, it has already been used by the IR library file that I have included.

- Then I have set our target frequency value for OCR0A and a variable for OCR0B to control brightness of LED using PWM.

- Next, I have set 2 external interrupt routines compare match A (TIMER0_COMPA_vect) to on LEDs and compare match B (TIMER0_COMPA_vect) to set all ports to 0 in order to off the LED.

```c
#define F_CPU 16000000UL

#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#include <string.h>
#include <stdlib.h>

#include "IR_Receiver.h"



struct IR_Packet received_packet;
uint8_t led_count = 0;

uint8_t pwm_count = 200;

void LED_ON()
{

	if (led_count == 1){
		PORTB |= 0b00000001;
	}
	else if (led_count == 2){
		PORTB |= 0b00000011;
	}
	else if (led_count == 3){
		PORTB |= 0b00000111;
	}
	else if (led_count == 4){
		PORTB |= 0b00001111;
	}
	else if (led_count == 5){
		PORTB |= 0b00001111;
		PORTD |= 0b10000000;
	}
	else if (led_count == 6){
		PORTB |= 0b00001111;
		PORTD |= 0b01000000;
else if (led_count == 7){
		PORTB |= 0b00001111;
		PORTD |= 0b00100000;
	}
	else if (led_count == 8){
		PORTB |= 0b00001111;
		PORTD |= 0b10000000;
	}
	else if (led_count == 9){
		PORTB |= 0b00001111;
		PORTD |= 0b01000000;
	}
	else if (led_count == 10){
		PORTB |= 0b00001111;
		PORTD |= 0b00100000;
	}


}

ISR (TIMER0_COMPA_vect) {//ISR for Timer0 compare match A
	LED_ON();   // function controls the LEDs based on the led_count variable.
}
```

```
ISR (TIMER0_COMPB_vect) {   //ISR for Timer0 compare match B
	PORTB = 0x00;
	PORTD = 0x00;

}



int main(void)
{
	DDRB |= 0xFF;
	DDRD |= 0xF0;

	OCR0A = 254; //set frequency
	OCR0B = pwm_count; //duty cycle     register controls the PWM duty cycle (brightness) of the LEDs.

	init_receiver();     // function initializes the IR receiver.
	led_count = 0;  // different LEDs are turned on by setting specific bits in the PORTB and PORTD registers.

	TCCR0A = 0b00000010; //CTC mode, internal clk, no prescaler
	TCCR0B = 0b00000001;
	TIMSK0 = 0b00000110; //enable Timer0 compare match A int and compare match B
	                        register enables Timer0 interrupts.
	sei();
	while (1)
	{
		cli();
		uint8_t check_result = check_new_packet(&received_packet);// function checks if a new IR packet
		sei();                                                        has been received.



if (check_result)
		{
			// up key packet received
			if (received_packet.command == 0x18)
			{

				if ( led_count == 10)
				{
					led_count = 10;
				}
				else
				{
					led_count++;
				}
				_delay_ms(300);   // function introduces a delay after changing the LED count

			}

			// down key packet received
			else if(received_packet.command == 0x52)
			{
				if ( led_count == 0)
				{
					led_count = 0;
				}
				else
				{
					led_count--;
				}
```

```
                }

                // back key packet received
                else if (received_packet.command == 0x08)
                {

                        if (pwm_count == 0)
                        {
                                TIMSK0 = 0x00;

                        }
                        else{
                                pwm_count = pwm_count - 20;

                        }
                        OCR0B = pwm_count;

                }


              // forward key packet received
              else if (received_packet.command == 0x5a)
              {

                      if (pwm_count == 200)
                      {
                              TIMSK0 = 0x06;

                      }
                      else{
                              pwm_count = pwm_count + 20;

                      }
                      OCR0B = pwm_count;

              }

            }

        }

            //_delay_ms(50);
}
```

4. Circuit Diagram

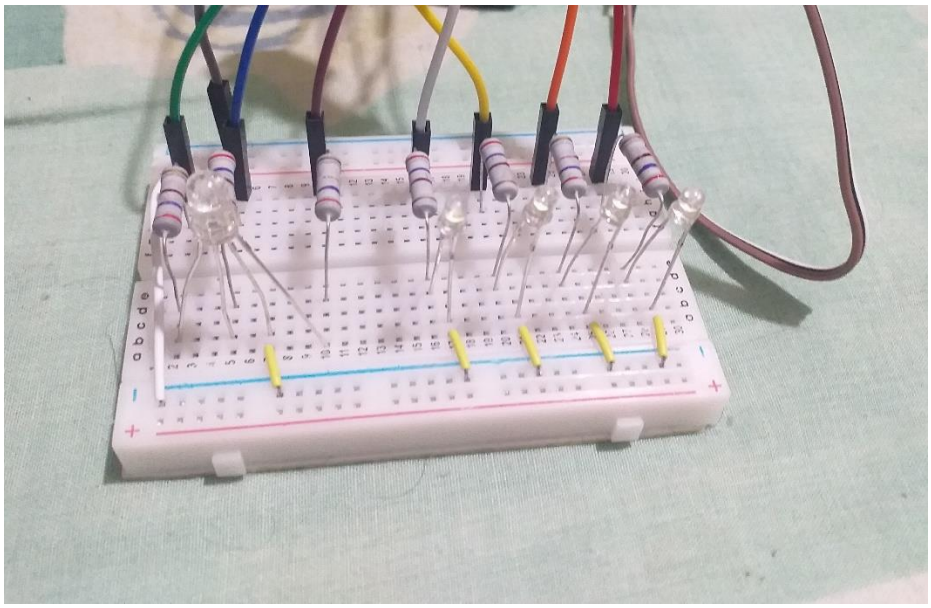

Figure 1 In the hardware implementation phase of the project, I have connected all LEDs to the bread board with resistors for each of them

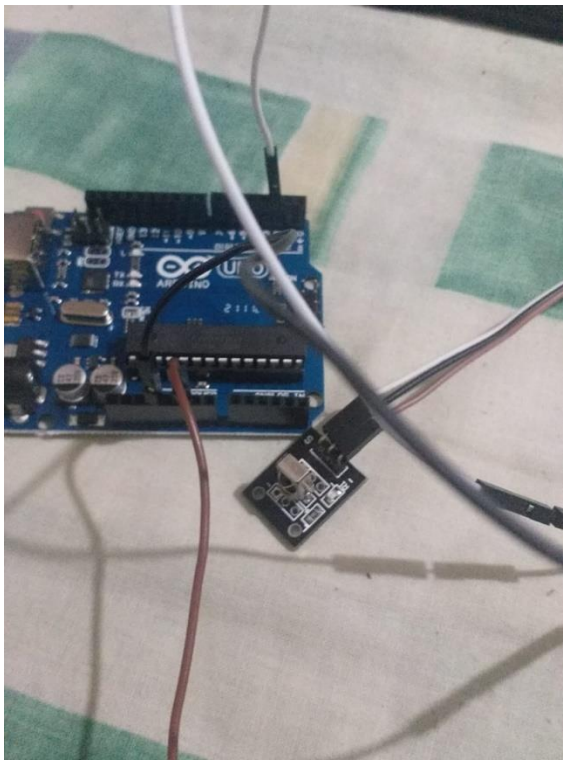

Figure 2 Then I have connected IR receiver signal pin to the PIND2 of Arduino board, Vcc pin to the 3.3v pin and GND pin to one of ground pin in the Arduino.

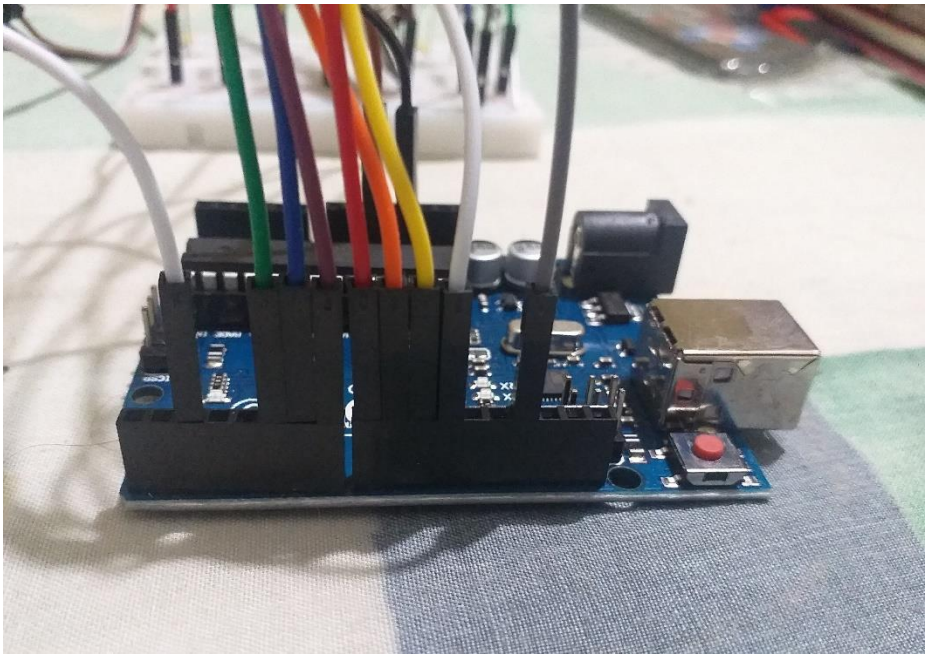

Figure 3 Next, I have connected the jumper wires of the LEDs to the relevant digital port of Arduino board as I mention in the design decision part.

- Then I have used USB serial port as the power source by connecting it to a COM port of the computer.

- In the code implementation I have did following things in the c program,
    - I have implemented a function to operate the state of LEDs according to a counter that indicate how many LEDs will be ON specifically.
    - Then I have set the data direction register value of PORTB as all output ports and data direction register value of PORTD as 4 output ports.
    - Then I have implemented 2 buttons to control LED state and 2 separate buttons to control brightness of LEDs as an added feature.
        - UP button of remote will switch ON the LEDs according to the led counter I have set.(pressing 1 times UP button will take as led counter incremented by 1).
        - DOWN button of remote will switch OFF the LEDs according to the led counter I have set.(pressing 1 times DOWN button will take as led counter decremented by 1).

- FORWARD and BACK buttons will control the brightness level of the LED according to the pwm_count variable I have given. It will increase and decrease the brightness of LEDs accordingly to 10 brightness levels.

5. Testing and Validation

- In the testing phase first, I find out the maximum current that an I/O pin of an Arduino UNO can produce is around 40mA, then I have calculated the power consumption of a 3v LED with 270ohm resistor is around 18mA and decided to use that because the required current for 3mm led is also same.(I/O pin output voltage is 5v)

  V = IR
  I= V/R
  I=5 / 270ohm
  I = 0.018A = 18mA

- Test results of the Arduino projects shown in below.

Figure 4 I have used this remote to generate signals in order to control LEDs of my project.

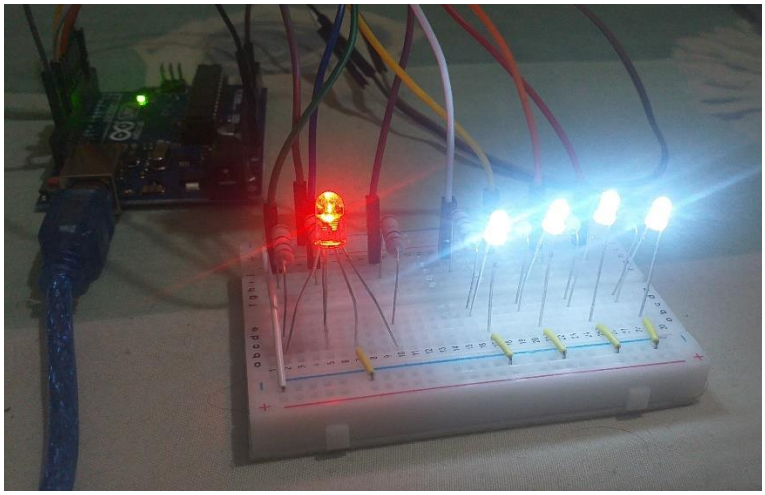

Figure 5 First using UP button of remote, LED will be ON sequentially to the given requirement.(colors of RGB LEDs in this order red - > green -> blue)

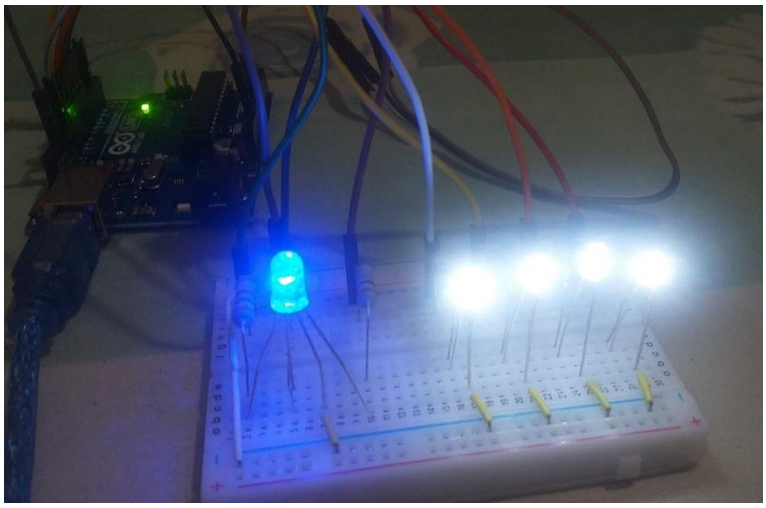

Figure 6 First using UP button of remote, LED will be ON sequentially to the given requirement.(colors of RGB LEDs in this order red - > green -> blue)

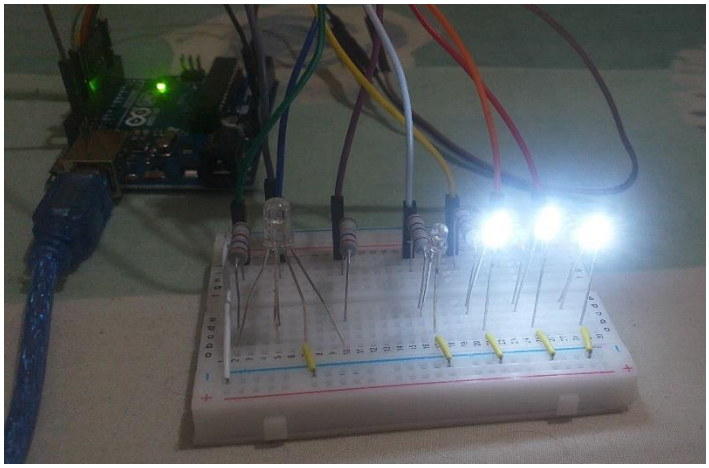

Figure 7 Then using DOWN button of remote, LED will be OFF from backward order of the LED light up.

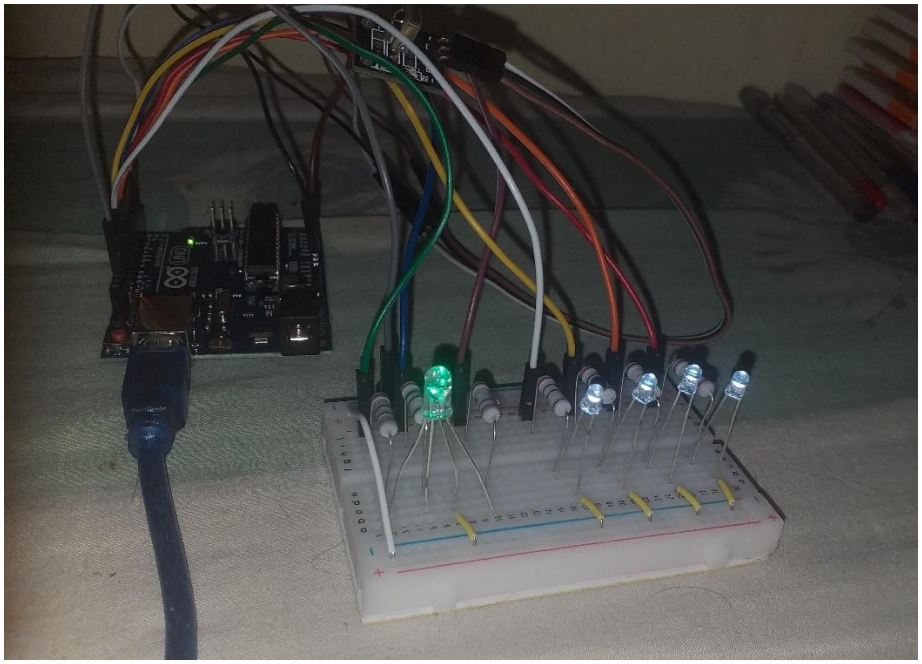

*Figure 8 The brightness of the LEDs is controlled by FORWARD and BACK button of the remote. Brightness increasing is carried by forward button and decreasing of brightness is carried by back button.(**decreased brightness levels cannot see clearly from the im*

6. Discussion

- Limitations of the design:

  - Maximum current output from Arduino pins

  - Limited number of LEDs due to pin availability

  - Reliance on IR remote for control

- Potential improvements:
  - Using a more powerful microcontroller for more LEDs/features
  - Implementing wireless control (e.g., Bluetooth, WiFi)
  - Adding additional functionalities (e.g., dimming, color mixing)

- Challenges faced:
  - Properly managing power consumption
  - Ensuring accurate IR signal detection and decoding
  - Optimizing code for real-time responsiveness

- Future enhancements:
  - Integration with smart home systems
  - Mobile app control
  - Scheduling and automation features

- Practical applications:
  - Decorative lighting
  - Emergency lighting
  - Portable lighting solutions

- Cost and power efficiency analysis

  - Comparison with commercial alternatives

  - Battery life and power consumption analysis

- Scalability and modularity

  - Potential for expanding the system

  - Modular design for easy maintenance and upgrades

7. References

 Atmel. (2009). ATmega328P Datasheet. Retrieved from https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

 Arduino. (n.d.). Arduino Uno Rev3. Retrieved from https://store.arduino.cc/products/arduino-uno-rev3

 Philips Semiconductors. (2001). IR receiver modules for remote control systems. Retrieved from https://www.vishay.com/docs/82659/tsop311.pdf

 Margolis, M. (2011). Arduino Cookbook. O'Reilly Media, Inc.

 Matic, N. (2017). Embedded Systems: Introduction to the Embedded World. Mango Publications.

 Kazemian, H. B. (2014). Introduction to embedded systems: A cyber-physical systems approach. MIT Press.

 Dhananjay, V. G. (2014). Embedded Systems: An Integrated Approach. Pearson Education India.

 Barr, M. (2001). Programming embedded systems in C and C++. O'Reilly Media, Inc.

Thank You.