Microprocessor System – CSY2015
Electrical and Electronic Engineering

TCA1

Faisal Abdulwahed Alsaeed

UON ID: 23856001

GU ID: 220212100070

Instructor: DR. Osama Alrawi

Submission Date: 09/12/2024

Table of Contents

Part 1: Gas and Heat Index Monitoring System

YouTube Link:

https://youtu.be/9vmkeZRUH-0

Introduction:

This project is very useful to use in many places like and it will safe people life. I have been chosen this project because most of the houses they are keeping the LPG cylinder gas in the garage also some of the people they are keeping the car running in the house garage without opening the ventilation or shelter and there's some incident's happen in Bahrain because of co2 and LPG cylinder, so if every house have like this device in garage it will monitor them to be carful that the gas level are high to take immediate action like opening the shelter and inserting a fan circulate the air. I have been insert the HDT11 sensor to this project because I'm an employee in petroleum company and we faced a problems with gases and also with the heat index, if the heat index reached 54 Celsius we have to stop the hot work immediately which is any work related to the construction or inspections, then we postponed the hot work to 4:00pm up to 04:00am, this is what happened in the industries environment but what about the private shops and the car scraps they don't have any idea about this they are working continuously so then will have an incident because of the heat index but if they have device to monitor them that the heat index reach up to 54 Celsius " Stop The Work Immediately" it will reduce the incident's cause if heat index.

How do we can develop this project?

By replacing Arduino UNO to ESP32 to receive an update continuously about the condition by Gmail.

Project Objectives:

- Gas leak detection.
- Heat index monitoring.
- Workplace safety.

Components Of The Project:

1. Arduino UNO
2. Bread Board
3. Wires
4. 330 Ohm resistors
5. LCD I2C
6. HDT11
7. MQ2
8. Red LED
9. Green LED
10. Buzzer

Problems faced:

I faced a problem with Tinker cad and Wokwie to create the circuit to be clear but I didn't find 2 sensors there which is MQ2 and HDT11.

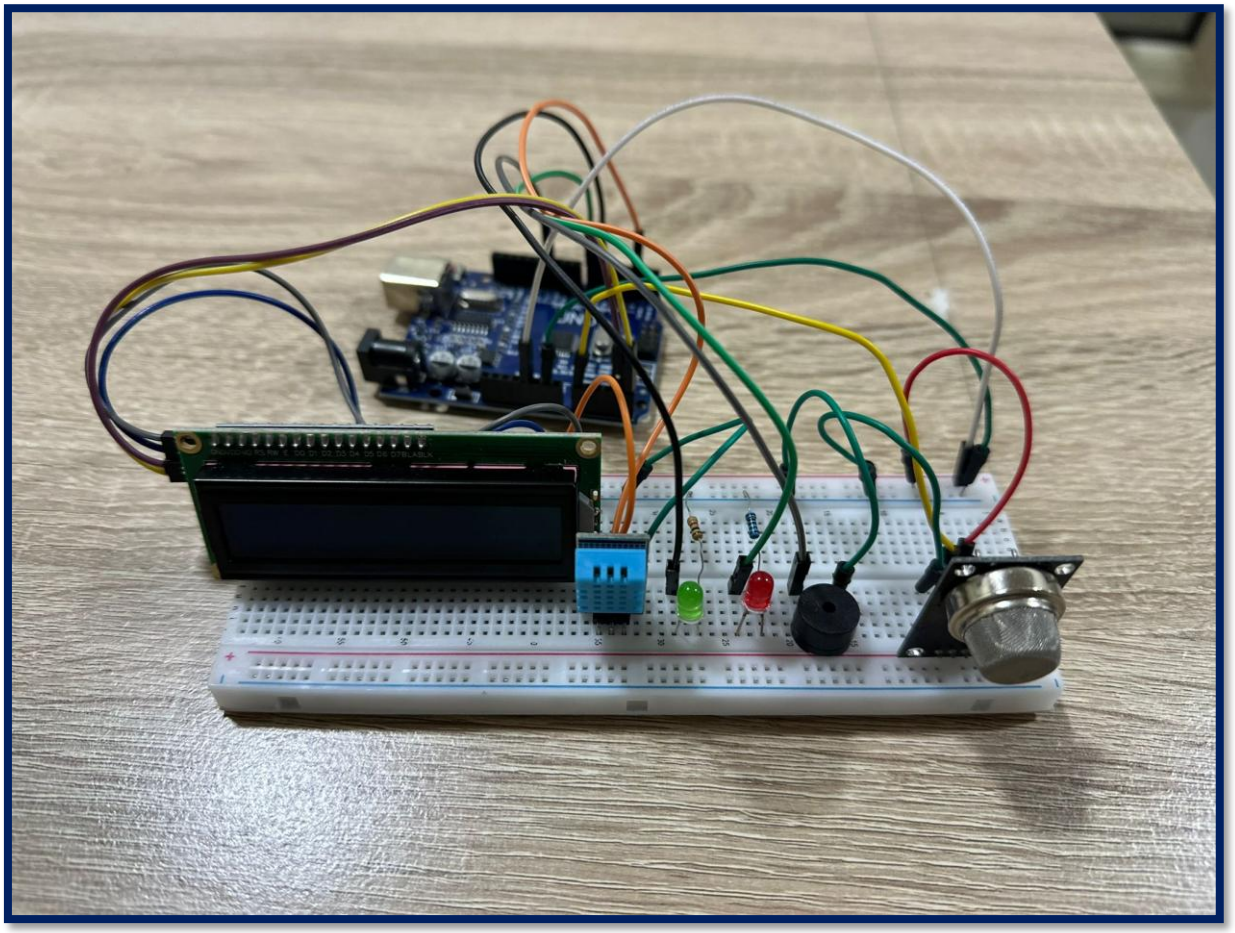

Figure1: Circuit of The Gas and Heat Index Monitoring System

Flow Chart of Gas and Heat Index Monitoring System

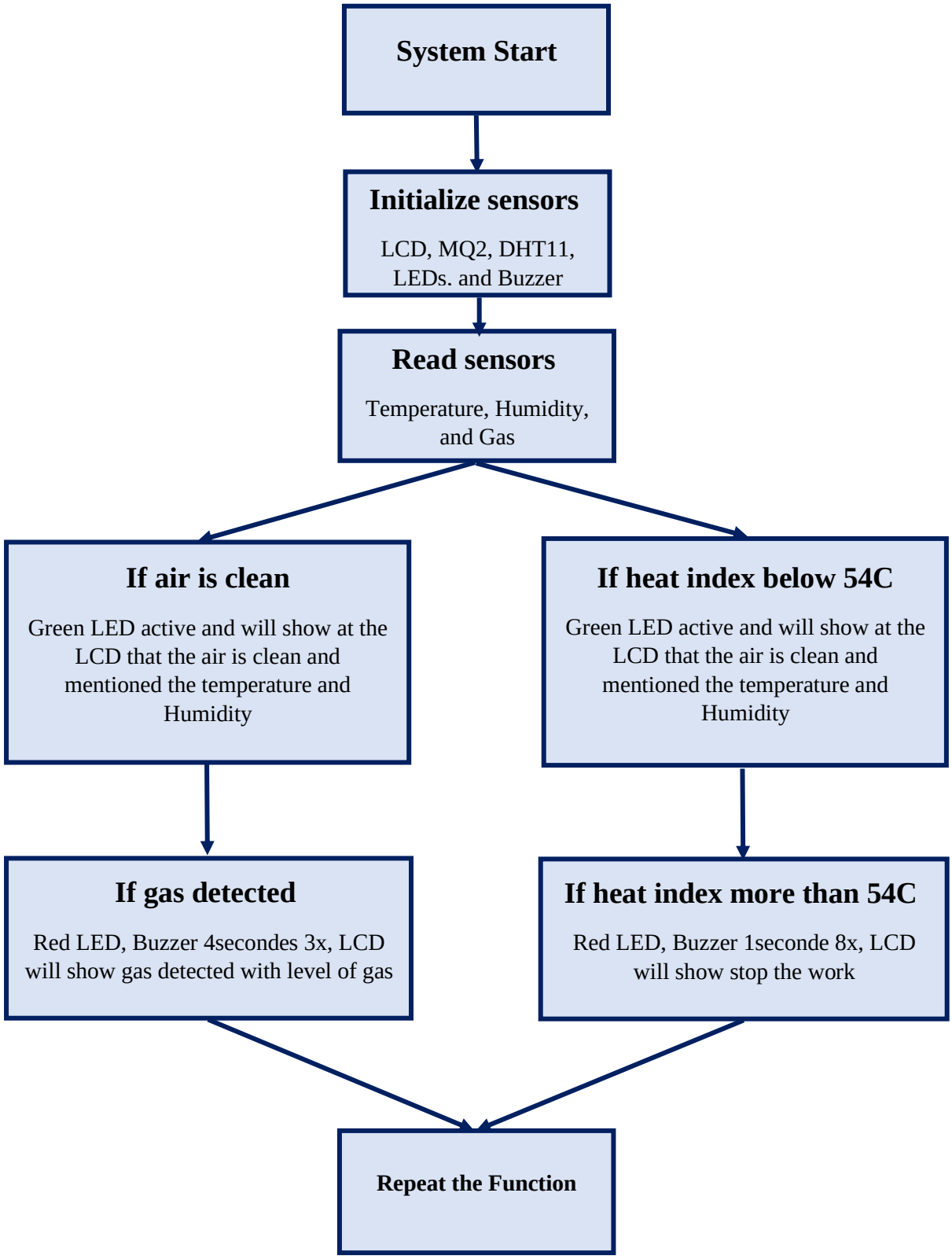

Figure2: Gas and Heat Index Monitoring System Flow Chart

Project Documentation

1- What does the project do?

By using MQ-2 sensor it's well detects 5 types of gases which is: Methane, Propane, Butane, Natural Gas, and LPG. Also, it can detect the smoke. Then the HDT11 sensor well measure the temperature and humidity. So, by merging these 2 sensors well protect people from heat index and gases hazards.

2- How to run a program?

By installing these 3 libraries which is HDT.h, LiquidCrystal_I2C, and Wire.h . then setting a maximum gas value and maximum heat index in Celsius. Then the program is ready to uploaded but firs we have to select the board and the port.

3- What are the inputs required of this project?

- Temperature and Humidity which is configured by HDT11 sensor and connected to pin 2 in the Arduino UNO.
- Gas level which is configured by MQ-2 sensor and connected to pin A0.

4- What is the output of the program?

- Red LED active when gas detected and heat index exceeds the limits.
- Buzzer activate when gas detected or heat index reached up.
- Green LED active when air is clean and heat index below 54C.
- LCD I2C to shown if gas detected, heat index reached up, if air is clean and temperature below 54C.
- Serial monitor printing gas and heat index values with messages to the serial monitor.

Part1 code:

```
//Produced by: Faisal Alsaeed/ ID:23856001
//Purpose: To identify if the gas or heat index reach up
//How to run: Install Wire.h, LiquidCrystal_I2C.h, DHT.h then
select "Tools" and go to "Boards" and choose "Arduino Uno" and
go to "ports" and choose "COM3" then click the arrow on the
top left "Upload"
//Active Program Requirments:  Adjust the "gasThreshold" and
"heatIndexThreshold" based on your need
//Program Output: Red LED and buzzer if gas is detected or the
heat index reached the specified "heatIndexThreshold"

#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <DHT.h>

// Define pins and sensor types
#define DHTPIN 2          // DHT11 data pin
#define DHTTYPE DHT11     // DHT11 sensor type
DHT dht(DHTPIN, DHTTYPE);

LiquidCrystal_I2C lcd(0x27, 16, 2); // Initialize LCD

// Pin definitions
```

```
const int gasAnalogPin = A0; // MQ2 AOUT pin
const int redLedPin = 7;     // Red LED
const int greenLedPin = 6;   // Green LED
const int buzzerPin = 11;    // Buzzer

// Gas and heat index thresholds
const int gasThreshold = 10;  // Adjusted gas threshold
const float heatIndexThreshold = 54; // Heat index threshold
in °C

//This function are prepares the Arduino for the main program
loop, and it will run once when the program starts
void setup() {
  lcd.begin(16, 2);
  lcd.backlight();
  pinMode(redLedPin, OUTPUT);
  pinMode(greenLedPin, OUTPUT);
  pinMode(buzzerPin, OUTPUT);
  dht.begin();
  Serial.begin(9600);
  lcd.setCursor(0, 0);
  lcd.print("System Starting...");
  delay(2000);
  lcd.clear();
}

//This function is continuously monitors gas levels,
temperature, and humidity, displaying values and triggering
alerts if gas levels exceed the threshold or the heat index
becomes too high, otherwise showing a "safe" status.
void loop() {
  int gasValue = analogRead(gasAnalogPin); // Read MQ2 analog
value
  float temperature = dht.readTemperature();
  float humidity = dht.readHumidity();
  float heatIndex = dht.computeHeatIndex(temperature,
humidity, false);

  Serial.print("Gas Analog: ");
  Serial.println(gasValue);
  Serial.print("Heat Index: ");
  Serial.println(heatIndex);

  if (isnan(temperature) || isnan(humidity)) {
    lcd.setCursor(0, 0);
    lcd.print("Sensor Error");
    delay(2000);
    return;
```

```
  }

  // Check heat index condition
  if (heatIndex > heatIndexThreshold) {
    digitalWrite(redLedPin, HIGH);//ON
    digitalWrite(greenLedPin, LOW);//OFF
    lcd.setCursor(0, 0);
    lcd.print("Attention: Stop ");
    lcd.setCursor(0, 1);
    lcd.print("the Work       ");
    for (int i = 0; i < 8; i++) {
      tone(buzzerPin, 1000);
      delay(1000);
      noTone(buzzerPin);
      delay(1000);
    }
    delay(5000);
    return;
  }

  // Check gas analog value
  if (gasValue > gasThreshold + 30) { // Add a buffer to the
threshold
    digitalWrite(redLedPin, HIGH);///
    digitalWrite(greenLedPin, LOW);
    lcd.setCursor(0, 0);
    lcd.print("Gas Detected!  ");
    lcd.setCursor(0, 1);
    lcd.print("Gas: ");
    lcd.print(gasValue);
    for (int i = 0; i < 3; i++) {
      tone(buzzerPin, 1000);
      delay(4000);
      noTone(buzzerPin);
      delay(1000);
    }
    delay(5000);
    return;
  }

  // Safe condition
  digitalWrite(redLedPin, LOW);
  digitalWrite(greenLedPin, HIGH);
  lcd.setCursor(0, 0);
  lcd.print("Air is Clean   ");
  lcd.setCursor(0, 1);
  lcd.print("Temp: ");
  lcd.print(temperature);
```

```
        lcd.print("C Hum: ");
        lcd.print(humidity);
        lcd.print("%");
        delay(5000);
```

Part 2: Sustainable Flood Light Systems

YouTube Link:

https://youtu.be/WQQttpPnYsE?si=b6E3UfoC3i6yF3mL

Introduction:

This project is supporting the saving energy goal. And it's very sustainable we can use it in many places in example like a car parks, gardens, hallway and other places. This projects have many features which we control it manually by the IR remote and that well work with the IR receiver, Second feature that the led will activate when there's a motion at minimum distance 20 cm and that configured by the ultrasonic, thirdly it will activate automatically when the LDR detect a darkness and I set it as an 90 in the program, and finally we can control the brightness by the potentiometer and while adjusting the brightness white led will activate to see what is the brightness that needs.

How do we can develop this project:

By adding a small solar panel with a battery to be active all the time without external source.

Project Objectives:

- Energy efficiency.
- Motion detection.
- Automation and smart control.

Components of The Project:

1- Arduino UNO
2- Bread Board
3- Wires
4- RGB
5- Ultrasonic
6- LDR
7- IR receiver
8- IR remote
9- Potentiometer
10- 4pcs of resisters 440Ohm

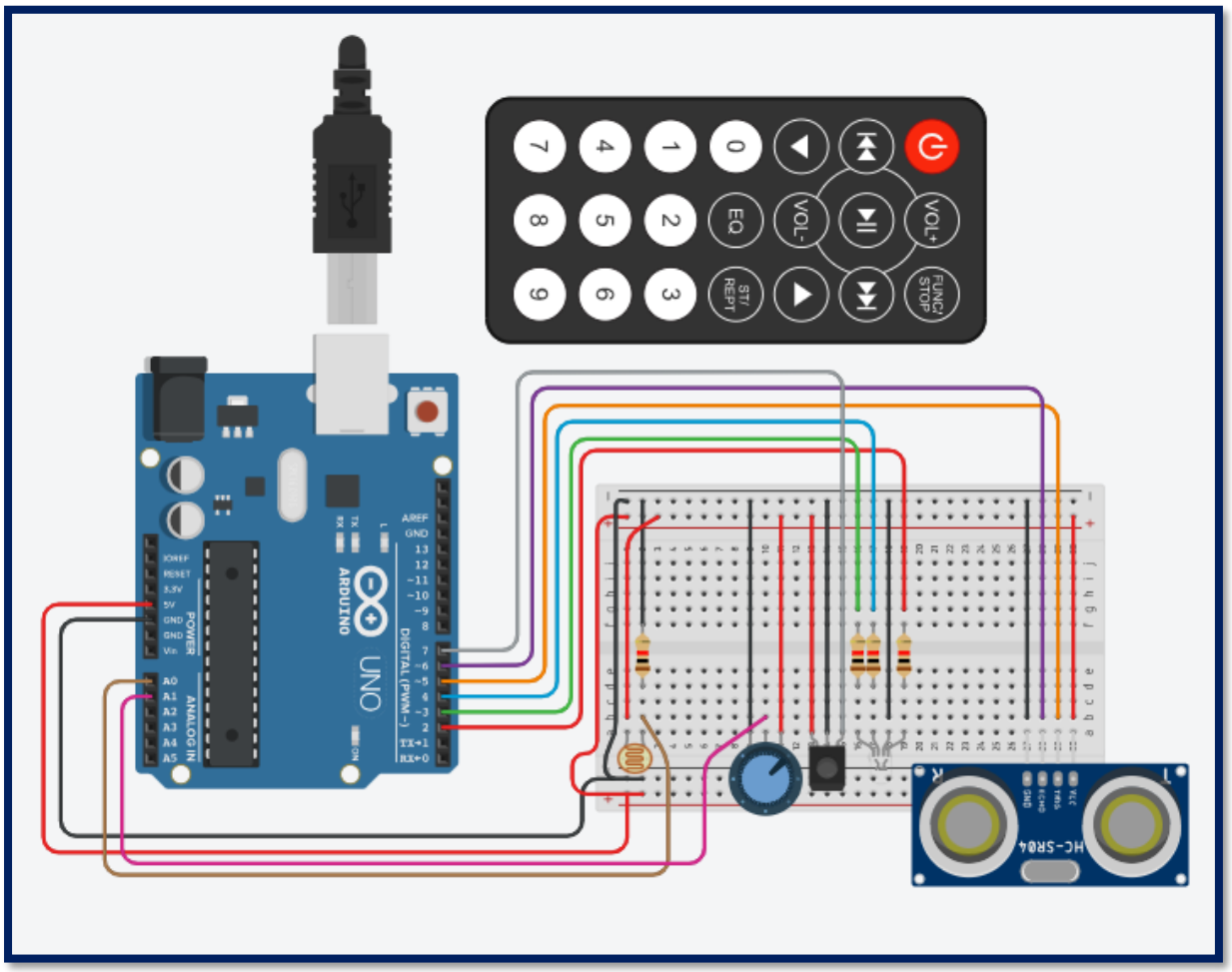

Figure 3: The Circuit of The Flood Light Project by TinkerCad

Flow Chart of The Flood Light Project:

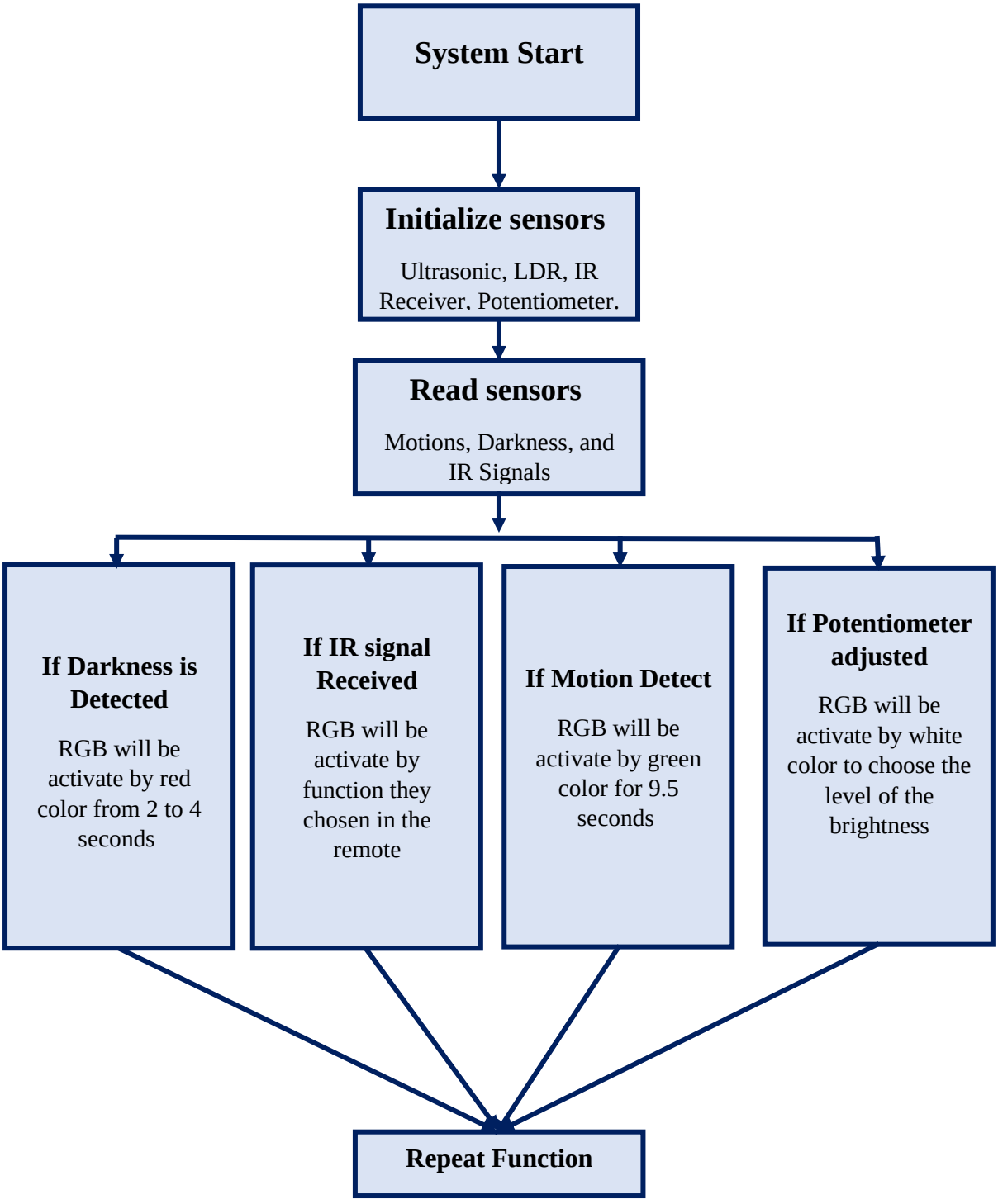

Figure 4: Sustainable Flood Light System Flow Chart

Project Documentation

1- What does the project do?

By using Ultrasonic sensor it's well detects a motion with minimum distance 20cm if it detect a motion the green Led well activate for 9.5 seconds, Then the LDR sensor well measure the darkness level if it’s less than 90 as per I set it in my program the red LED will activate from 2 to 4 seconds. Also, the IR receiver if received a signal from the remote the RGB will activate depend on the function have been chosen in the remote. And the potentiometer will activate the RGB by white colour while adjusting the brightness. So, by merging these sensors we are have amazing project to save energy and to reduce the pollution." The engineers always created s sustainable projects which will help the environment".

2- How to run a program?

By installing this library which is IR. remote version (4.4.1) for the IR receiver. then choosing the function from the remote the RGB will activate on the function have been chosen. By setting a maximum darkness value the RGB will activate by red LED from 2 to 4 seconds. Then by setting a minimum distance for the ultrasonic I set it in my program 20 cm, if it detects a motion after 20 cm the RGB will activate by green LED for 9.5 seconds. And last one is the potentiometer we can choose the brightness and also will activate white LED to see the brightness if is it good. Then the program is ready to be uploaded but first we have to select the board and the port type.

3- What are the inputs required of this project?

- Motion which is configured by ultrasonic sensor and it's connected to the Arduino: trig to pin 5, and echo to pin 6.
- Darkness level which is configured by LDR sensor and connected to pin A0.
- IR signal which is configured by IR receiver and it's connected to pin 7.
- Brightness adjusted which is configured by potentiometer and it's connected to pin A1.

4- What is the output of the program?

- Red LED active when darkness detected from 2 to 4 seconds.
- Green Led activate when motion detected within 20 cm for 9.5 seconds.
- White LED active when there is adjusting of the brightness.
- About the IR receiver I will set a table of every function with what it does and the time delay.

IR remote:

First have been recorded all of the decimal values buttons from my remote, and then I set every function with a decimal value I token from the remote and I will attach the of how did I get the decimals values.

IR Remote Code:

```
#include <IRremote.hpp>
const int IR_RECEIVE_PIN = 7;
int redPin = 9;

decode_results results;
void setup() {
  // run once:
  Serial.begin(9600);
  IrReceiver.enableIRIn();
  Serial.println("IR RECEIVE TEST");
 IrReceiver.begin(IR_RECEIVE_PIN); // start IrReceiver

 pinMode(redPin, OUTPUT);
 digitalWrite(redPin, LOW);
}

void loop() {
// run repetadley:
   if(IrReceiver.decode()){
    Serial.println(IrReceiver.decodedIRData.command);

  int IRD = IrReceiver.decodedIRData.command;
 switch(IRD){
  case 12:
     digitalWrite(redPin, HIGH);
  break;
  case 82:
    digitalWrite(redPin, LOW);
  break;
  }

  }
 IrReceiver.resume();
  delay(2000);
}
```

Decimal Values	Action	Delay
70	Will pause and play the function	----------
28	Red LED	3.5 Seconds
24	Green LED	3.5 Seconds
12	Blue LED	3.5 Seconds
82	White LED	3.5 Seconds
68	Previous	3.5 Seconds
64	Next	3.5 Seconds
7	Jumping	1.1 Seconds
21	Stroboscopic	3 flashes each colour and 200 milli seconds between flashes

Figure 5: Decimal Values of the IR Remote

Part 2 Code:

```
//Produced by: Faisal Alsaeed/ ID:23856001
//Purpose: To activate the RGB if there's a motion, darkness or signal from the IR remote
//How to run: Install from the library "IRremote.hpp" then select "Tools" and go to "Boards" and choose "Arduino Uno" and go to "ports" and choose "COM5" then click the arrow on the top left "Upload"
//Active Program Requirements:  Adjust the darkness value and minimum distance based on your need.
//Program Output: If detect a motion green led will active for 9.5 seconds, Red LED will active if detect darkness, White Led will active while adjusting brightness level. RGB will activat if get a signal from the remote.

#include <IRremote.hpp>

// Function Prototypes
void cycleColors(unsigned long delayTime);
void setRGBColor(int RED, int GREEN, int BLUE);
float getDistance();
void Potentiometer_Control();
void Ultrasonic_Control();
void LDR_Control();
void processIRSignal(uint32_t command);
void stroboscopicEffect();

// Pin Assignments
const int IR_Receiver_Module = 7; // Ir pin
const int EC_Pin = 6;             // Ultrasonic echo pin
const int TR_Pin = 5;             // Ultrasonic trig pin
const int PO_Pin = A1;            // Potentiometer pin
const int RE_Pin = 2;             // Red pin from RGB
const int GR_Pin = 3;             // Green pin from RGB
const int BL_Pin = 4;             // Blue pin from RGB
const int LDR_Pin = A0;           // LDR pin

// Variables
const int darknessLevel = 90;  // Level of darkness
const int minimumRange = 20;   // Minimum range for ultrasonic
const unsigned long ultrasonicRGBTime = 9500; // RGB activation time for ultrasonic
const unsigned long ldrMinDuration = 2000;   // Minimum activation time for LDR RGB in milliseconds
const unsigned long ldrMaxDuration = 4000;   // Maximum activation time for LDR RGB in milliseconds

unsigned long rgbOffTime = 0; // RGB deactivation timer
bool rgbActive = false;       // RGB dectivation state
bool paused = false;          // RGB pausing state
```

```
int brightness = 255;         // Standard brightness

// IR Instructions (decimal codes)
const uint32_t IR_INS_PA = 70; // Pause
const uint32_t IR_INS_RE = 28; // Red
const uint32_t IR_INS_NE = 64; // Next
const uint32_t IR_INS_GR = 24; // Green
const uint32_t IR_INS_PR = 68; // Previous
const uint32_t IR_INS_BL = 12;  // Blue
const uint32_t IR_INS_JUMPING = 7;
const uint32_t IR_INS_SCROBOSCOBIC = 21;
const uint32_t IR_INS_WH = 82; // White

// RGB Colors
int colors[][3] = {
  {255, 255, 255}, // White color
  {255, 0, 0},     // Red color
  {0, 0, 255},     // Blue color
  {0, 255, 0}      // Green color
};

int activeColorIndex = 0;

IRrecv signalReceiver(IR_Receiver_Module); // Setting up the IR receiver
decode_results results;

unsigned long irOperationDuration = 0; // IR triggered RGB activation timer
bool irActive = false;

void setup() {
  Serial.begin(9600);
  signalReceiver.begin(IR_Receiver_Module);

  pinMode(TR_Pin, OUTPUT);
  pinMode(EC_Pin, INPUT);
  pinMode(LDR_Pin, INPUT);
  pinMode(PO_Pin, INPUT);
  pinMode(RE_Pin, OUTPUT);
  pinMode(GR_Pin, OUTPUT);
  pinMode(BL_Pin, OUTPUT);

  // Initialize RGB off
  setRGBColor(0, 0, 0);

  Serial.println("System Startup Complete");
}

void loop() {
```

```
  if (signalReceiver.decode()) { // this is to process the ir signal and
prints the details and passes the signal for the action
    Serial.print("Protocol: ");
    Serial.println(signalReceiver.decodedIRData.protocol);// thus will show me
the protocol type in the serial monitor
    Serial.print("Raw Data: ");
    Serial.println(signalReceiver.decodedIRData.decodedRawData, DEC);// this
will print the decimals values in the serial moniter
    Serial.print("Command: ");
    Serial.println(signalReceiver.decodedIRData.command); // this will print
the value of the recived signal in the serial monitor
    processIRSignal(signalReceiver.decodedIRData.command);
    signalReceiver.resume();
  }

  if (rgbActive && millis() > rgbOffTime && !paused) {
    rgbActive = false;
    setRGBColor(0, 0, 0);
  }// If RGB is active, timer expired, and not paused, turn off RGB and set
LEDs to off.
  if (irActive && millis() > irOperationDuration && !paused) {
    irActive = false;
    Serial.println("Leaving IR mode. Resuming sensor operations");
    setRGBColor(0, 0, 0); // Turning off the RGB
  }

//If not paused, run Potentiometer, Ultrasonic, and LDR controls, then delay
122micro seconds.
  if (!paused) {
    Potentiometer_Control();
    Ultrasonic_Control();
    LDR_Control();
  }

  delay(122);
}

//Read the the value and map the potentiometer from 0 to 255
void Potentiometer_Control() {
  int potValue = analogRead(PO_Pin);
  int newBrightness = map(potValue, 0, 1023, 0, 255);

  // testing and updating the brightness value if it changed and activate
white LED while adjusting
  if (newBrightness != brightness) {
    brightness = newBrightness;
    setRGBColor(brightness, brightness, brightness);
    Serial.print("Potentiometer adjusted, brightness: ");
```

```
    Serial.println(brightness);
  }
}
// Testing the distance If it's within 20cm and RGB green color will activate
for 9.5 secondes.
void Ultrasonic_Control() {
  float distance = getDistance();
  if (distance > 0 && distance <= minimumRange && !rgbActive) {
    Serial.println("Ultrasonic sensor detected movement. Enabling RGB");
    rgbActive = true;
    rgbOffTime = millis() + ultrasonicRGBTime;
    setRGBColor(0, 255, 0);
  }
}

// if detecting a darkness level below 90 the red LED will activate
void LDR_Control() {
  int lightLevel = analogRead(LDR_Pin);
  if (lightLevel < darknessLevel && !rgbActive) {
    Serial.println("Low light detected. Enabling RGB");
    rgbActive = true;
    rgbOffTime = millis() + random(ldrMinDuration, ldrMaxDuration);
    setRGBColor(255, 0, 0);
  }
}

void processIRSignal(uint32_t Signal) {
  const uint32_t validSignals[] = {
    IR_INS_RE, IR_INS_GR, IR_INS_BL, IR_INS_WH,
    IR_INS_SCROBOSCOBIC, IR_INS_JUMPING, IR_INS_PR, IR_INS_NE, IR_INS_PA
  };

//here to set isvalid false and loop of ValidSignals if it simillar to the
mentioned set is valid true.
  bool isValid = false;
  for (int i = 0; i < sizeof(validSignals) / sizeof(validSignals[0]); i++) {
    if (Signal == validSignals[i]) {
      isValid = true;
      break;
    }
  }

//// if it's getting a value doesn't has been mentioned in decimals will print
Unrecognized signal
  if (!isValid) {
    Serial.println("Unrecognized Signal. Ignoring.");
    return;
  }
```

```
  if (Signal == IR_INS_PA) {
    paused = !paused;
    Serial.print("Pause state: ");
    Serial.println(paused ? "ON" : "OFF");
    return;
  }

  irActive = true;
  switch (Signal) {
    case IR_INS_JUMPING:
      cycleColors(1100);
      break;
    case IR_INS_SCROBOSCOBIC:
      stroboscopicEffect();
      break;
    case IR_INS_GR:
      Serial.println("Green for 3.5 seconds.");
      setRGBColor(colors[3][0], colors[3][1], colors[3][2]);
      irOperationDuration = millis() + 3500;
      break;
    case IR_INS_RE:
      Serial.println("Red for 3.5 seconds.");
      setRGBColor(colors[1][0], colors[1][1], colors[1][2]);
      irOperationDuration = millis() + 3500;
      break;
    case IR_INS_WH:
      Serial.println("White for 3.5 seconds.");
      setRGBColor(colors[0][0], colors[0][1], colors[0][2]);
      irOperationDuration = millis() + 3500;
      break;
    case IR_INS_BL:
      Serial.println("Blue for 3.5 seconds.");
      setRGBColor(colors[2][0], colors[2][1], colors[2][2]);
      irOperationDuration = millis() + 3500;
      break;
    case IR_INS_PR:
      activeColorIndex = (activeColorIndex - 1 + 4) % 4;
      setRGBColor(colors[activeColorIndex][0], colors[activeColorIndex][1],
colors[activeColorIndex][2]);
      irOperationDuration = millis() + 3500;
      break;
    case IR_INS_NE:
      activeColorIndex = (activeColorIndex + 1) % 4;
      setRGBColor(colors[activeColorIndex][0], colors[activeColorIndex][1],
colors[activeColorIndex][2]);
      irOperationDuration = millis() + 3500;
      break;
```

```
    default:
      Serial.println("Unexpected error in IR");
      break;
  }
}

void setRGBColor(int RED, int GREEN, int BLUE) {
  analogWrite(BL_Pin, BLUE);
  analogWrite(RE_Pin, RED);
  analogWrite(GR_Pin, GREEN);
}

//// trigger sensor, measure puls then return distance in cm.
float getDistance() {
  digitalWrite(TR_Pin, LOW);
  delayMicroseconds(4);
  digitalWrite(TR_Pin, HIGH);
  delayMicroseconds(10);
  digitalWrite(TR_Pin, LOW);
  long duration = pulseIn(EC_Pin, HIGH);
  return (duration * 0.034) / 2;
}

//Setting the color and waiting for the mentioned delay
void cycleColors(unsigned long delayTime) {
  for (int i = 0; i < 4; i++) {
    setRGBColor(colors[i][0], colors[i][1], colors[i][2]);
    delay(delayTime);
  }
}

void stroboscopicEffect() {
  int strobeDelay = 200; // Delay between flashes in milliseconds

  for (int i = 0; i < 3; i++) {
    setRGBColor(255, 255, 255); // White
    delay(strobeDelay);
    setRGBColor(0, 0, 0);
    delay(strobeDelay);
  }
  for (int i = 0; i < 3; i++) {
    setRGBColor(255, 0, 0); // Red
    delay(strobeDelay);
    setRGBColor(0, 0, 0);
    delay(strobeDelay);
  }
  for (int i = 0; i < 3; i++) {
    setRGBColor(0, 255, 0); // Green
```

```
    delay(strobeDelay);
    setRGBColor(0, 0, 0);
    delay(strobeDelay);
  }
  for (int i = 0; i < 3; i++) {
    setRGBColor(0, 0, 255); // Blue
    delay(strobeDelay);
    setRGBColor(0, 0, 0);
    delay(strobeDelay);
  }
}
```

References:

As a reference I has been used the Arduino UNO website to see the functions for part 1 and 2.
Using Functions in a Sketch | Arduino Documentation
Then I click to electronics sections to see the circuits and how I can create the programs.