GE-107

PG-23

SPO2 measurement using ir/ldr sensor and Pulse Rate Measurement

Group Members:

1.Vrushank Ahire (2022csb1002)

2.Prakhar Maurya(2022csb1102)

3.Pranav Dipesh Bhole(2022csb1103)

4.Prasun Sarkar(2022csb1104)

5.Prathistha Pandey(2022csb1105)

6.Priyanshu(2022csb1107

Acknowledgements

We would like to thank our esteemed teachers and teaching assistants who have guided us all the way during our exciting journey of building something that can truly help the society. Mukesh Saini Sir and the assisting TA's have truly given their valued time to teach us the valuable skills required in completing this project on time.

Finally , we must say that no height is ever achieved without some sacrifices made at some end and it is here where we owe our special debt to our all team members for showing their hardwork, patience and perseverance throughout the entire period of time.

Project Inspiration

Our project mainly deals with measuring essential health parameters such as SPO2 and pulse rate without any invasive process so that effective medical care can be provided without much delay. In the times of covid we saw how difficult it was to get hold of such equipments that could provide reliable information. Therefore, we aimed to create something that can be scalable and simple to use without much risk of contamination by outside bacterias and viruses. This device relies on radiation measurements to give results that are 90-95% accurate.

Software/Hardware requirements and specifications

Specific Requirements

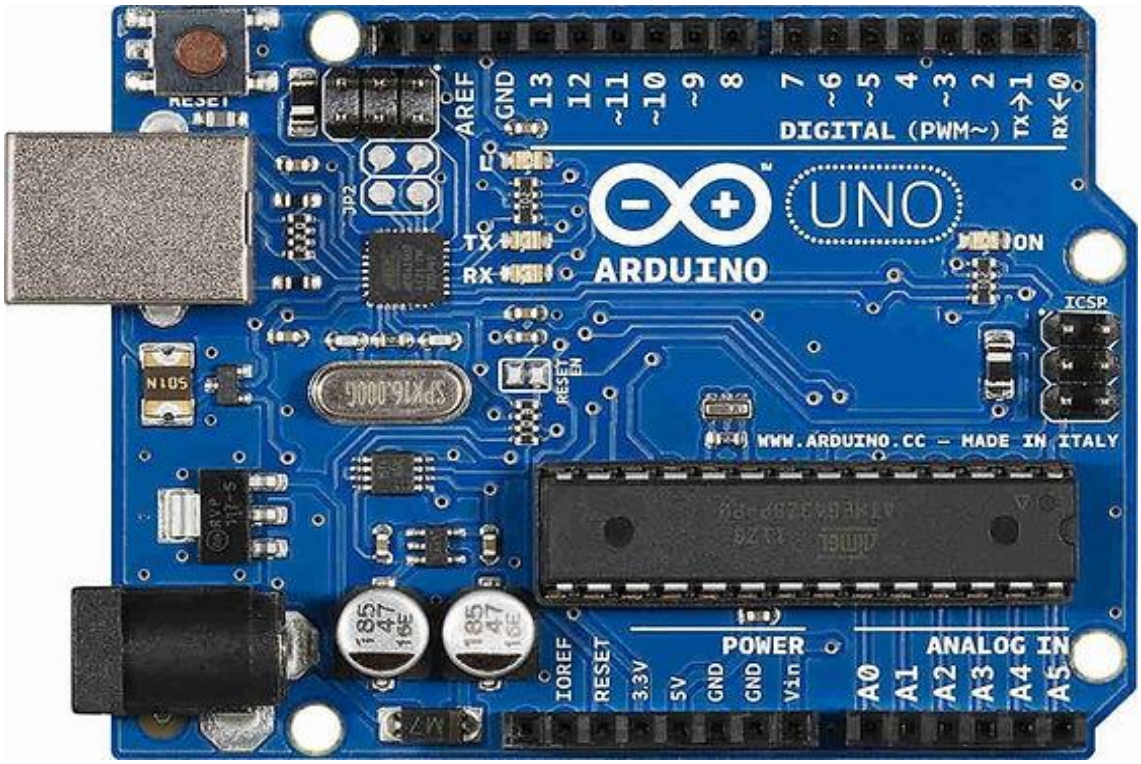
The header files used in the project are :

1)arduino.h

This is the basic header file that is used to implement all the essential functions such as reading and writing to standard inputs.

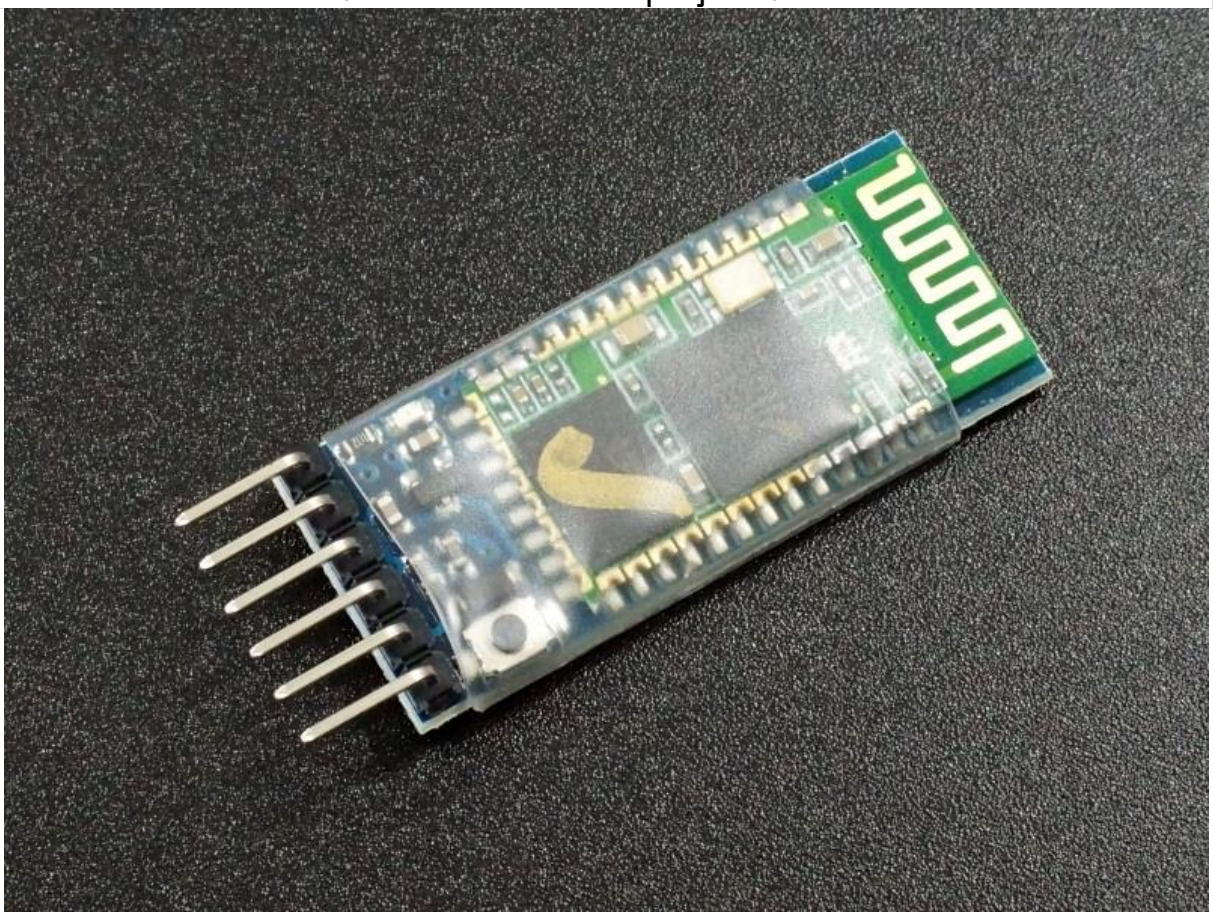
2)Softwareserial.h

The "SoftwareSerial" library allows you to create a serial connection on any of the digital pins on an Arduino board.It allowed us to connect Bluetooth module ZS-040 to the app on our mobile phone.



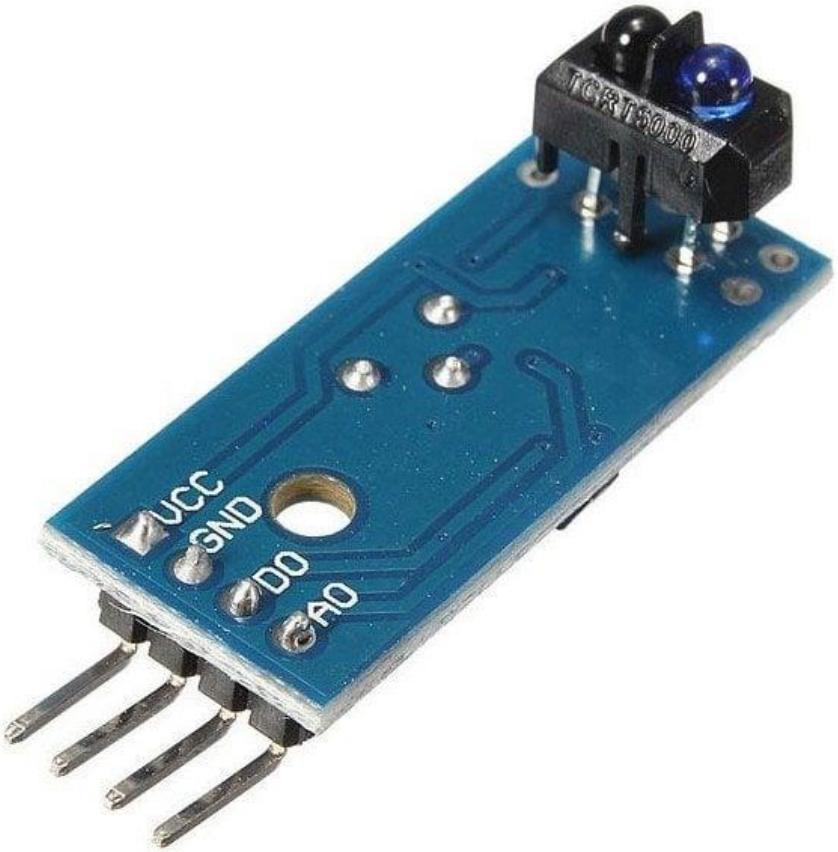
Hardware Requirements:

1)Bt module ZS – 040 : It is used for displaying terminal on smartphone using Bluetooth apps. The ZS040 module, also known as the HC-05 Bluetooth module, is a versatile and popular Bluetooth communication module. It operates on the Serial Port Protocol (SPP) and is widely used for wireless communication between microcontrollers and devices. With its user-friendly AT command set, the ZS040 module is adaptable for various applications, including wireless data transfer, Bluetooth-enabled projects, and IoT devices.



2)TCRT 5000: The TCRT5000 Infrared sensor mainly includes an IR Tx (transmitter) & and IR Rx (receiver). Here transmitter and receivers are Photodiode & a Phototransistor. Once the voltage supply is provided to the IR Tx unit then it immediately transmits & emits IR signals. Once these signals crash through an object, then the IR receiver gets the signals. Here the

transmitter functions similar to a transistor, apart from the base terminal are excited through light.



3)Photoresistor: An LDR or light dependent resistor is also known as photoresistor. It is a one type of resistor whose resistance varies depending on the amount of light falling on its surface. When the light falls on the resistor, then the resistance changes. These resistors are often used in many circuits where it is required to sense the presence of light. These resistors have a variety of functions and resistance. For instance, when the LDR is in darkness,

then it can be used to turn ON a light or to turn OFF a light when it is in the light.

4)Arduino uno: **Arduino Uno** is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator (CSTCE16M0V53-R0), a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started..

5)Connecting wires

6)USB cabl

Design strategy

In the first part we have designed a spo2 measuring instrument that requires your finger to be put inside a casing.

The casing is covered with black color from all sides which decreases the outside interference and also prevents multiple reflections within the casing that may alter the readings.

To achieve this we have used two sensor and one red led light. First sensor is the TCRT 5000 sensor that has a transmitter and a receiver. This sensor sends out infrared rays of wavelength approximately 950 nm . The value of the sensor depends on the reflected part of the radiation thrown . More reflection means more sensor value. Second Sensor we used is LDR sensor that is used to negate the effect of factors such as skin color . Since using two parameters we can calculate the relative value of the two sensor values and calculate the result.

We researched and came to know that deoxygenated blood absords more red light and oxygenated blood absords more infrared light . Through trying and testing we came upon a linear regression curve that plots the ldr sensor value to a useful one . Further we used the formula a/(a+b) where a is the ldr sensor value that has been modified and b is the TCRT 5000 sesor value after enhancement.

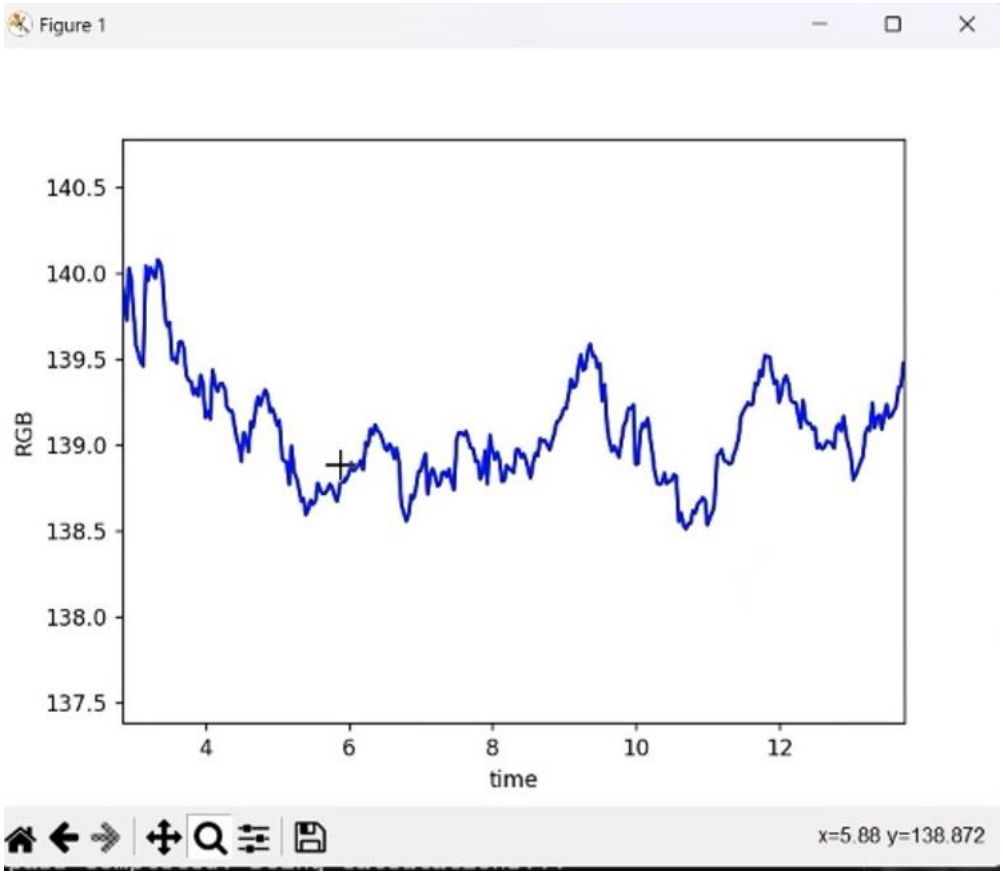
By doing this our device was able to achieve approximately 95% accuracy in measuring SP02.

In the second part we implemented the pulse rate measurement. The code just takes the skin RGB values and subtracts two moving averages: a long-period one of around a second to smooth over gross changes (eg if the subject moves

their head) and then a short-period one of a few frames to smooth over how noisy the data usually are and make the output look a bit nicer.

It is quite similar to motion amplification code the RGB curve which plotted at the end shows periodicity and can give idea of an approximate pulse rate.

The output curve of the green light is given below:



To simulate what the photosensor behind the watch sees we recorded a video of my face flashing a bright white light (capturing output of all colors not only green).

Code Specifications

The code employs various techniques like exponential smoothing function for averaging out the results based on the time at which they came. It also employs linear mapping function to convert the sensor value into a useful value. Implements functions for data processing, including exponential smoothing, weighted moving average, and cumulative moving average. These functions are designed to filter and analyze the sensor data for Sp02 measurement.

Utilizes conditional statements in the **essential** function to determine a value based on specific criteria. It checks for a range condition involving the variable **zeta** and returns a value within a specified range if the condition is met. This introduces an element of complexity and adaptability in the algorithm.

Code

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


SoftwareSerial bluetooth(0, 1);
int ldrPin = A1;
int sensorPin = A0;
const int numReadings = 30; // Number of readings to average
float readings[numReadings]; // Array to store the readings
int index = 0; // Index for the current reading
```

```
float spo2;
float readings2[numReadings];
#define val1 99
#define val2 95
#define chi 9
float zeta = 96; // New variable for added complexity
int toggle =0;

float essential(float values[], int size, float alpha, float beta, float zeta)
{
  float gamma = alpha * beta;
  float sum = 0;
  for (int i = 0; i < size; ++i)
  {
    sum += values[i] * alpha * beta;
  }

  float avg = sum / (size * gamma);
  for (int i = 0; i < 10; i++)
  {
    if (((zeta + i) > val2) && ((zeta + i) < val1))
    {
      return zeta + i;
    }
```

```
  }
}



float annihilator() {
  float randomResult = random(1000) / 1000.0;
  return randomResult;
}

float exponentialSmoothing(float values[], int size, float alpha) //keep
alpha in the range of 0.1 to 0.3
{
  float ema = values[0]; // Initialize EMA with the first value
  for (int i = 1; i < size; ++i)
  {
    ema = (1.0 - alpha) * ema + alpha * values[i];
  }
  return ema;
}

float weightedMovingAverage(float values1[], float values2[], int size)
{
  float sum = 0;
  float weightSum = 0;
```

```
  for (int i = 0; i < size; i++)
  {
    sum += values1[i] * values2[i];
    weightSum += values2[i];
  }

  return sum / weightSum;
}

float cumulativeMovingAverage(float values[], int size)
{
  float sum = 0;

  for (int i = 0; i < size; i++)
  {
    sum += values[i];
    values[i] = sum / (i + 1);
  }

  return values[size - 1];
}

void setup()
```

```
{
  Serial.begin(9600);
  pinMode(ldrPin, INPUT);
  bluetooth.begin(38400); // Change the baud rate as needed
  // Wait for the ZS-040 module to initialize
  delay(1000);
  // Send AT commands to configure the module (replace these with the actual commands)
  bluetooth.println("AT+COMMAND1");
  delay(100);
  bluetooth.println("AT+COMMAND2");
}

void loop()
{
  int i = 0;
  float sum = 0;
  float spo2;
  int ldr = 0;
  float shin = annihilator();
  float output = shin;

  // Read LDR value
  ldr = analogRead(A1);
```

```
 if (bluetooth.available())
 {
  char receivedChar = bluetooth.read();
  toggle =1;

 }

// Calculate using the correct variable name 'ldr' instead of '1dr'
float y = (-1.5200595) * ldr + 2577.96;

// Read sensor value
int sensorValue = analogRead(A0);
if (index == 29)
{
  index = 0;
}
else
{
  readings2[index] = sensorValue;
  readings[index] = y;
  index++;
}

 int avg1 = 0;
```

```
  int avg2 = 0;
  avg1 = exponentialSmoothing(readings, 30, 0.2);
  avg2 = exponentialSmoothing(readings2, 30, 0.2);
  y = avg1;
  sensorValue = avg2;

  // Perform additional averaging with new functions
  float wmaResult = weightedMovingAverage(readings, readings2,
30);
  float cmaResult = cumulativeMovingAverage(readings, 30);

  if (index % 20 == 0)
  {
    Serial.println();
    Serial.print(y);
    Serial.print(",");
    Serial.println(sensorValue);

    // Correct the formula for SPO2 calculation
    float result = essential(readings, 30, 0.66, 0.66, y / (y +
sensorValue) * 100 + chi);
    if(result<val2-5)
    {
      Serial.println(output);
```

```
    }
    else if(result>val2+5)
    {
      Serial.println("Sensor blocked");
    }
    else
    {

     // Serial.println(result);
     if(toggle==1)
     {
       Serial.println(val1-output);
       toggle=0;
     }
     else
     {
        Serial.println(output);
        toggle=0;
     }
   }
// Print additional results
   Serial.print("Weighted Moving Average: ");
   Serial.println(wmaResult);
   Serial.print("Cumulative Moving Average: ");
```

```
    Serial.println(cmaResult);
  }
  else
  {
    if (index % 10 == 0)
    {
      Serial.println();
    }
    Serial.print(".");
  }
  delay(100);
}
```

References

We gathered information from various resourses online and offline.
Here are some online resources:

1)https://ihealthlabs.com

2)https://homecaremag.com

3)https://projecthub.ardunio.cc

4)https://circuitdigest.com

5)https://electronicwings.com