

**DEPARTMENT** **OF** **ELECTRONICS** **AND** **TELECOMMUNICATION**

**ENGINEERING, IIIT** **BHUBANESWAR**

**B.** **Tech** **(3rd** **YEAR,** **6TH** **SEMESTER)**

**PulseMAX: Building a ESP8266 NodeMCU based MAX30102 Pulse oximeter webserver**

**ACKNOWLEDGEMENT**

After having Interaction with our professor Mrs. S. K. Nayak Mam. We explored many types of IOT projects and many topics made us feel fascinated, one of the most interesting fields is Pulse Oximeter. It inspired us to use the knowledge we got in the respective subject and work on it.

We want to express our sincere gratitude to Mrs. S. K. Nayak Mam for providing us with proper guidance, resources, and motivation to complete our project successfully. Thank you so much for all the knowledge you have shared with us. We are grateful for the mentorship we have received.

Thank you!

*By:*

*Sudip Nayak (B220061)*

*Under* *the* *guidance* *of*

**Mrs.** **S.** **K.** **Nayak**

**ABSTRACT**

A pulse oximeter is a medical device that is used to measure the oxygen saturation level in a person's blood. It is a non-invasive device that uses a sensor to measure the absorption of light by the blood.

The pulse oximeter works by emitting two wavelengths of light, typically red and infrared, through a person's skin and into their blood vessels. The amount of light that is absorbed by the blood is proportional to the amount of oxygen in the blood. The pulse oximeter then calculates the oxygen saturation level by comparing the amount of absorbed light at the two wavelengths.

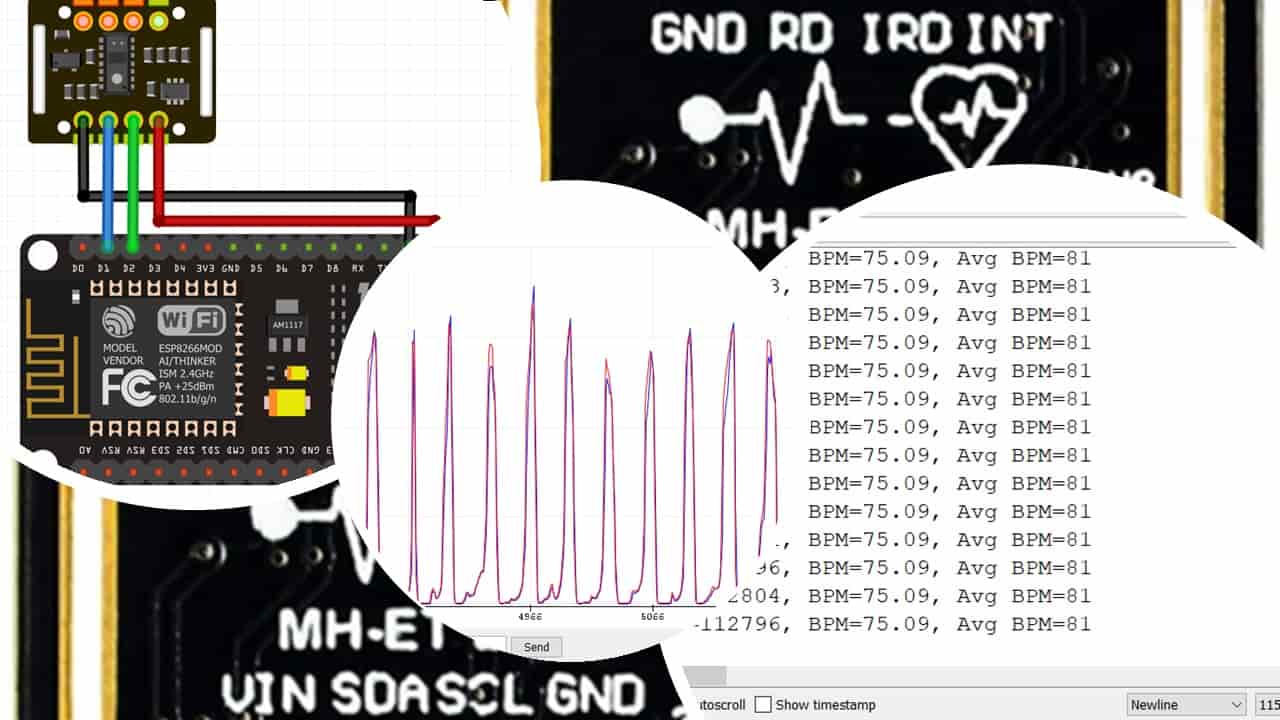
The oxygen saturation level is an important parameter to monitor, especially for individuals with respiratory problems or those undergoing certain medical procedures. It is a measure of the amount of oxygen that is being carried by the blood, which is important for the body's overall function and health.

Pulse oximeters are commonly used in hospitals, clinics, and homes. They are small, portable, and easy to use, making them a convenient device for monitoring a person's oxygen saturation level. Some pulse oximeters also measure the heart rate, which can provide additional information about a person's health.

Overall, pulse oximeters are an important medical device that can provide valuable information about a person's oxygen saturation level and heart rate, which can aid in the diagnosis and management of various medical conditions.

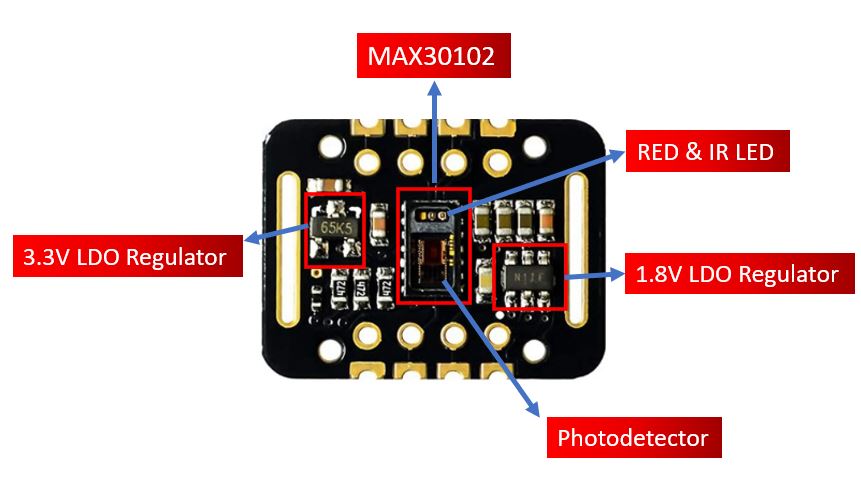
**OBJECTIVE**

In this project, we will learn about MAX30102 high sensitivity pulse oximeter and heart rate sensor and how to interface it with ESP8266 NodeMCU board to measure BPM and blood oxygen concentration (SpO2). This sensor is used to measure heartbeat/pulse rate in BPM. Firstly, we will discuss the introduction, pinout, working, and connection diagram of the sensor with ESP8266. After that, we will see different examples sketches from the MAX3010x Pulse and Proximity Sensor Library. These will include finding BPM reading, plotting raw data on serial plotter, presence sensing, and temperature measurement. So let us begin!



## **MAX30102 SENSOR**

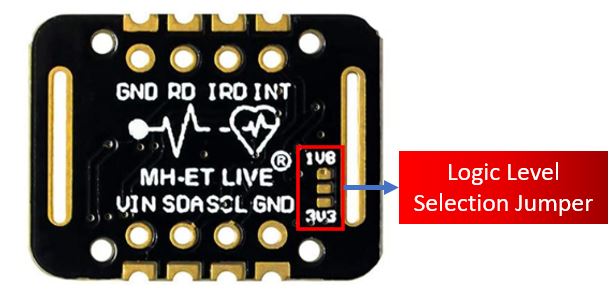
The MAX30102 sensor is the further optimized version of MAX30100 sensor; used as both a heart rate monitor and a pulse oximeter. These features are enabled by constructing this sensor which consists of two LEDs, a photodetector, optimized optics, and low noise signal processing components. It is easily used with microcontrollers such as Arduino, ESP32, NodeMCU, etc. to build an efficient heartbeat and oxygen saturation device.



As we may notice the MAX30102 IC lies at the centre of the module. The module consists of two different types of LEDs (Red and IR) and a photodetector. Blood oxygen saturation and heart rate are found using these two key features. We will later learn how the sensor works to obtain the BPM and SpO2 readings.

Another important feature you may notice is that the MAX30102 sensor module consists of two LDO regulators. This is because the MAX30100 IC requires 1.8V and the LEDs require 3.3V to function properly. With the addition of the voltage regulators, we can safely use microcontrollers that use 5/3.3/1.8V level input/outputs.

Moreover, if you view the module from the back, you can view a solder jumper to select the voltage logic level. By default, it is set to 3.3V but you can also change it to 1.8V according to your microcontroller’s logic requirements.



### **KEY FEATURES**

* The MAX30102 sensor module has an ultra-low power operation, uses 600μA (measurement mode and 0.7μA(standby mode). Therefore, a great choice to use in wearable devices such as smartwatches etc.
* It has a high sample rate capability along with fast data output capability.
* Additionally, the sensor features integrated ambient light cancellation as well.
* One additional feature that the MAX30102 sensor module possess is the inclusion of an on-chip temperature sensor. This gives us the die temperature (-40˚C to +85˚C) which is ± 1˚C accurate.
* For communicating with microcontrollers, the sensor uses the I2C pins SCL and SDA.
* Another feature of this sensor is that it uses a **32 sample FIFO buffer** to store data as compared to MAX30100 which has only 16 sample FIFO buffer. In other words, it further reduces power consumption as it already holds a maximum of thirty-two heart rate and SPO2 values.
* The MAX30102 can also be used with interrupts which can be enabled for several sources such as power ready, new data ready, ambient light cancellation, temperature ready and FIFO almost full. With the generation of interrupts, the microcontroller can perform other events which do not happen during the sequential execution of a program while the sensor keeps obtaining new data samples.

The table below shows the specifications of this sensor:

|  |  |
| --- | --- |
| Maximum Current Consumption | 6mA |
| Voltage | 3.3-5V |
| Sample Rate | 50Hz – 3200Hz |
| Temperature Range | -40°C to +85°C |
| Temperature Accuracy | ±1˚C |
| ADC Resolution | 18 bits |
| IR LED peak wavelength | 880nm |
| Red LED peak wavelength | 660nm |

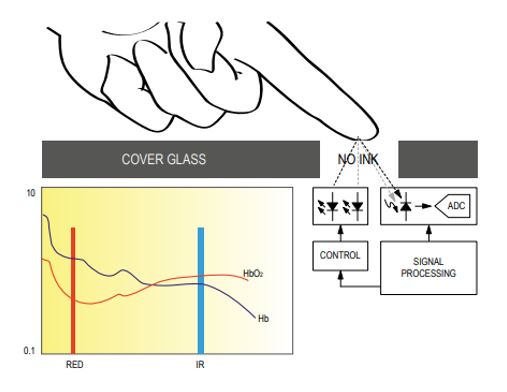
## **MAX30102 Sensor Working**

In this section, let us discuss how the MAX30102 heartbeat monitor and pulse oximeter works.

### **Heart Rate Measurement**

To measure the heart rate, we do not require the Red LED, only the IR LED is needed. This is because oxygenated hemoglobin absorbs more infrared light.

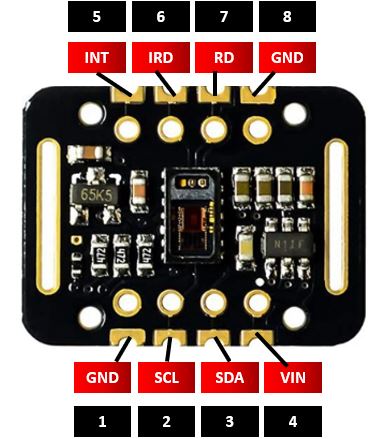
The heartbeat rate is the ratio of time between two consecutive heartbeats. Similarly, when the human blood is circulated in the human body then this blood is squeezed in capillary tissues. As a result, the volume of capillary tissues is increased but this volume is decreased after each heartbeat. This change in volume of capillary tissues affects the infrared light of the sensor, which transmits light after each heartbeat.



The working of this sensor could be checked by placing a human finger in front of this sensor. When a finger is placed in front of this pulse sensor then the reflection of infrared light is changed based on the volume of blood change inside capillary vessels. This means during the heartbeat, the volume of blood in capillary vessels will be high and then will be low after each heartbeat. So, by changing this volume, the LED light is changed. This change of the LED light measures the heartbeat rate of a finger. This phenomenon is known as “Photoplethysmogram.”

## **MAX30102 SENSOR PINOUT**

The MAX30102 module consists of eight pins.



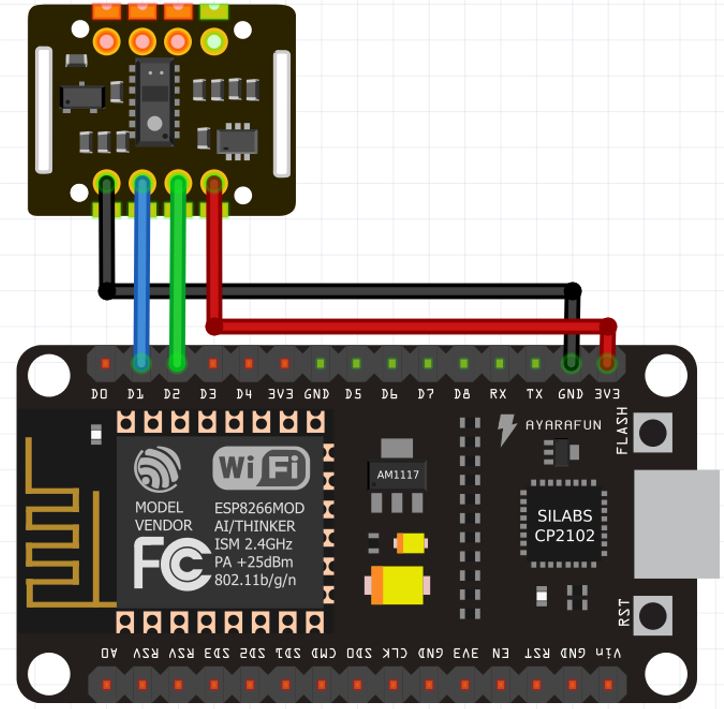
| **Pin** | **Explanation** |
| --- | --- |
| VIN | This pin is used to supply power to the sensor. This sensor is powered on at 3.3-5V. |
| SCL | This is the I2C serial clock pin. |
| SDA | This is the I2C serial data pin. |
| INT | This is the active low interrupt pin. It is pulled HIGH by the onboard resistor but when an interrupt occurs it goes LOW until the interrupt clears. |
| IRD | IR LED Cathode and LED Driver Connection Point |
| RD | Red LED Cathode and LED Driver Connection Point |
| GND | This is used for supplying ground to this sensor and it is connected to the source ground pin. |

## **INTERFACING MAX30102 BPM SENSOR WITH ESP8266 NodeMCU**

In this section, we will learn how to interface MAX30102 sensor module with ESP8266. We will use only four pins of the sensor module to connect with our microcontroller. The connections between the sensor module and ESP8266 are as follows:

| **MAX30102 Module** | **ESP8266** |
| --- | --- |
| VCC | 3.3V |
| SCL | D1 |
| SDA | D2 |
| GND | GND |

**CIRCUIT DESIGN/DIAGRAM**



## **MEASURE HEART RATE(BPM) WITH MAX30102 AND ESP8266**

The following program code will open. This example sketch will display the BPM values in the serial monitor when you hold the sensor in between your fingers.

/\*

Optical Heart Rate Detection (PBA Algorithm) using the MAX30105 Breakout

\*/

#include <Wire.h>

#include "MAX30105.h"

#include "heartRate.h"

MAX30105 particleSensor;

const byte RATE\_SIZE = 4; //Increase this for more averaging. 4 is good.

byte rates[RATE\_SIZE]; //Array of heart rates

byte rateSpot = 0;

long lastBeat = 0; //Time at which the last beat occurred

float beatsPerMinute;

int beatAvg;

void setup()

{

Serial.begin(115200);

Serial.println("Initializing...");

// Initialize sensor

if (!particleSensor.begin(Wire, I2C\_SPEED\_FAST)) //Use default I2C port, 400kHz speed

{

Serial.println("MAX30102 was not found. Please check wiring/power. ");

while (1);

}

Serial.println("Place your index finger on the sensor with steady pressure.");

particleSensor.setup(); //Configure sensor with default settings

particleSensor.setPulseAmplitudeRed(0x0A); //Turn Red LED to low to indicate sensor is running

particleSensor.setPulseAmplitudeGreen(0); //Turn off Green LED

}

void loop()

{

long irValue = particleSensor.getIR();

if (checkForBeat(irValue) == true)

{

//We sensed a beat!

long delta = millis() - lastBeat;

lastBeat = millis();

beatsPerMinute = 60 / (delta / 1000.0);

if (beatsPerMinute < 255 && beatsPerMinute > 20)

{

rates[rateSpot++] = (byte)beatsPerMinute; //Store this reading in the array

rateSpot %= RATE\_SIZE; //Wrap variable

//Take average of readings

beatAvg = 0;

for (byte x = 0 ; x < RATE\_SIZE ; x++)

beatAvg += rates[x];

beatAvg /= RATE\_SIZE;

}

}

Serial.print("IR=");

Serial.print(irValue);

Serial.print(", BPM=");

Serial.print(beatsPerMinute);

Serial.print(", Avg BPM=");

Serial.print(beatAvg);

if (irValue < 50000)

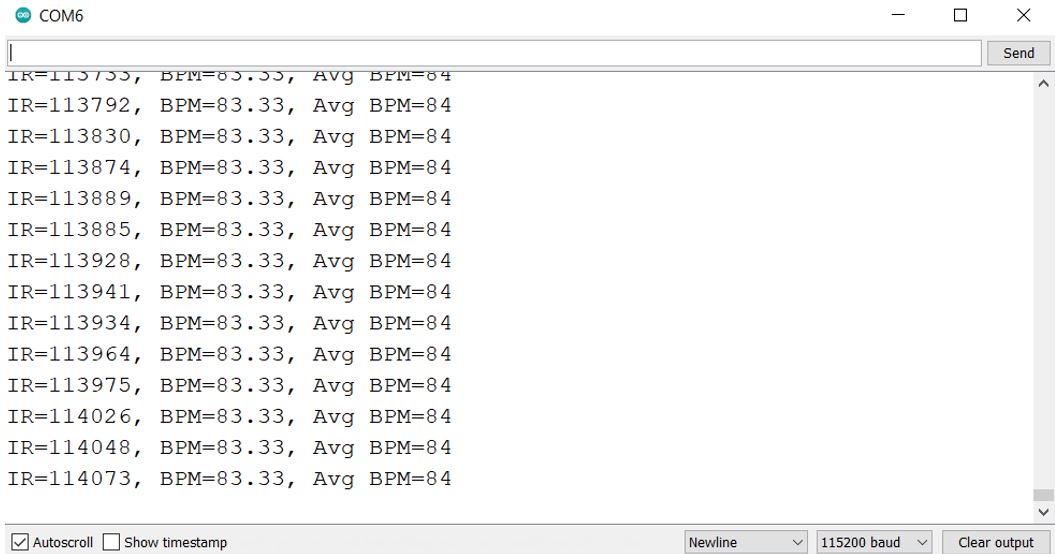
Serial.print(" No finger?");

Serial.println();

}

**OUTPUT**

The BPM readings will continuously update to new ones in the Arduino IDE compiler.



We also have used RemoteXY to show the results of BPM ( Beats per minute) whose readings taken within a second and then multiplied with 60 to make it per minute and then the average BPM is found out after few BPM readings.



**THE FINAL CODE INTEGRATING WITH REMOTEXYAPP**

#include <Wire.h>

#include "MAX30105.h"

#include "heartRate.h"

#define REMOTEXY\_MODE\_\_ESP8266WIFI\_LIB\_POINT

#include <ESP8266WiFi.h>

#include <RemoteXY.h>

// RemoteXY connection settings

#define REMOTEXY\_WIFI\_SSID "RemoteXY"

#define REMOTEXY\_WIFI\_PASSWORD "12345678"

#define REMOTEXY\_SERVER\_PORT 6377

// RemoteXY configurate

#pragma pack(push, 1)

uint8\_t RemoteXY\_CONF[] =   // 30 bytes

  { 255,0,0,8,0,23,0,16,31,1,68,50,0,0,63,98,8,36,135,66,

  80,77,0,66,80,77,97,118,103,0 };

// this structure defines all the variables and events of your control interface

struct {

    // output variables

  float onlineGraph\_1\_var1;

  float onlineGraph\_1\_var2;

    // other variable

  uint8\_t connect\_flag;  // =1 if wire connected, else =0

} RemoteXY;

#pragma pack(pop)

MAX30105 particleSensor;

const byte RATE\_SIZE = 4; //Increase this for more averaging. 4 is good.

byte rates[RATE\_SIZE]; //Array of heart rates

byte rateSpot = 0;

long lastBeat = 0; //Time at which the last beat occurred

float beatsPerMinute;

float beatAvg;

void setup()

{

  RemoteXY\_Init ();

  Serial.begin(115200);

  Serial.println("Initializing...");

  // Initialize sensor

  if (!particleSensor.begin(Wire, I2C\_SPEED\_FAST)) //Use default I2C port, 400kHz speed

  {

    Serial.println("MAX30102 was not found. Please check wiring/power. ");

    while (1);

  }

  Serial.println("Place your index finger on the sensor with steady pressure.");

  particleSensor.setup(); //Configure sensor with default settings

  particleSensor.setPulseAmplitudeRed(0x0A); //Turn Red LED to low to indicate sensor is running

  particleSensor.setPulseAmplitudeGreen(0); //Turn off Green LED

}

void loop()

{

  RemoteXY\_Handler();

  long irValue = particleSensor.getIR();

  if (checkForBeat(irValue) == true)

  {

    //We sensed a beat!

    long delta = millis() - lastBeat;

    lastBeat = millis();

    beatsPerMinute = 60 / (delta / 1000.0);

    if (beatsPerMinute < 255 && beatsPerMinute > 20)

    {

      rates[rateSpot++] = (byte)beatsPerMinute; //Store this reading in the array

      rateSpot %= RATE\_SIZE; //Wrap variable

      //Take average of readings

      beatAvg = 0;

      for (byte x = 0 ; x < RATE\_SIZE ; x++)

        beatAvg += rates[x];

      beatAvg /= RATE\_SIZE;

    }

  }

  Serial.print("IR=");

  Serial.print(irValue);

  Serial.print(", BPM=");

  Serial.print(beatsPerMinute);

  Serial.print(", Avg BPM=");

  Serial.print(beatAvg);

RemoteXY.onlineGraph\_1\_var1=beatsPerMinute;

RemoteXY.onlineGraph\_1\_var2=beatAvg;

  if (irValue < 50000)

    Serial.print(" No finger?");

  Serial.println();

}

**ADVANTAGES**

* Cost-effective: The ESP8266 NodeMCU and MAX30102 sensor are relatively inexpensive compared to other medical-grade pulse oximeters, making them a cost-effective solution for healthcare providers, patients, and researchers.
* Easy to use: The NodeMCU and MAX30102 are easy to use and require minimal setup. They can be programmed using the Arduino IDE, which is a popular and user-friendly programming environment.
* Wireless connectivity: The NodeMCU has built-in Wi-Fi connectivity, which enables the pulse oximeter to transmit data wirelessly to a computer or smartphone, allowing for remote monitoring and analysis of data.
* Real-time monitoring: The MAX30102 sensor provides real-time data on heart rate and blood oxygen levels, enabling quick and accurate assessment of a patient's condition.
* Data logging: The NodeMCU can be programmed to log data, allowing for long-term monitoring of a patient's condition and analysis of trends over time.
* Customizable: The ESP8266 NodeMCU and MAX30102 sensor can be customized and programmed to meet specific needs, making them a versatile solution for various applications in healthcare, fitness, and research.

Overall, the use of ESP8266 NodeMCU and MAX30102 in pulse oximetry offers a cost-effective, easy-to-use, and customizable solution for healthcare providers, researchers, and patients alike.

**DISADVANTAGES**

* Accuracy: The accuracy of the readings obtained from the pulse oximeter using these components may not be as reliable as medical-grade devices. The accuracy of the readings can be affected by factors such as movement, ambient light, and skin pigmentation.
* Regulatory compliance: The ESP8266 NodeMCU and MAX30102 sensor are not certified as medical devices by regulatory agencies such as the FDA, which could limit their use in clinical settings or for diagnostic purposes.
* Technical expertise required: Building and programming a pulse oximeter using these components requires technical expertise, which may not be readily available to all users.
* Power consumption: The ESP8266 NodeMCU and MAX30102 sensor require a power source to operate, and their power consumption may be higher than that of other pulse oximeters, which could limit their use in remote or resource-limited settings.
* Data security: Transmitting health data wirelessly using the NodeMCU could pose potential risks to data security and patient privacy.

Overall, while the use of ESP8266 NodeMCU and MAX30102 sensor for pulse oximetry offers several advantages, it is essential to consider the potential disadvantages and limitations before using them for clinical or diagnostic purposes.

**REFERENCES**

* [**https://microcontrollerslab.com/esp8266-heart-rate-pulse-oximeter-max30102/**](https://microcontrollerslab.com/esp8266-heart-rate-pulse-oximeter-max30102/)
* [**https://github.com/har-in-air/ESP8266\_MAX30102\_SPO2\_PULSE\_METER**](https://github.com/har-in-air/ESP8266_MAX30102_SPO2_PULSE_METER)
* [**https://www.analog.com/media/en/technical-documentation/data-sheets/MAX30102.pdf**](https://www.analog.com/media/en/technical-documentation/data-sheets/MAX30102.pdf)
* [**https://lastminuteengineers.com/max30102-pulse-oximeter-heart-rate-sensor-arduino-tutorial/**](https://lastminuteengineers.com/max30102-pulse-oximeter-heart-rate-sensor-arduino-tutorial/)
* [**http://www.esp8266learning.com/max30102-pulse-and-heart-rate-monitor-sensor-and-esp8266.php**](http://www.esp8266learning.com/max30102-pulse-and-heart-rate-monitor-sensor-and-esp8266.php)