Department of Electronic & Telecommunication Engineering
University of Moratuwa

EN2160 - Electronic Design Realization

IOT PULSE OXIMETER

H.M.S.D.Bandara - 200064C

16 th of July 2023

Contents

Abstract

This project aimed to create a marketable IoT pulse oximeter with advanced features to cater to healthcare needs. The IoT pulse oximeter was designed to monitor oxygen saturation levels and pulse rate accurately and in real-time. Key features included seamless connectivity with smart devices, user-friendly controls, and data visualization through a dedicated mobile app.

To ensure successful market penetration, a thorough marketing plan was executed, incorporating insights from customer surveys. The manufacturing process was well-planned, enabling mass production with high-quality components and attractive enclosures. The result was a cutting-edge IoT pulse oximeter ready for launch under a new company name, poised to make an impact in the competitive healthcare market.

Introduction

Problem Description:

Pulse oximeters are essential medical devices used to monitor an individual's oxygen saturation levels and pulse rate noninvasively. Traditional pulse oximeters are standalone devices that display readings on a small screen, limiting data accessibility and analysis. Additionally, they often lack seamless connectivity with smart devices, making it challenging for users to track and share their health metrics effectively.

The problem identified is the need for a modernized and connected pulse oximeter solution that overcomes these limitations. An IoT pulse oximeter can address this problem by leveraging internet connectivity and integrating with smartphones and other smart devices. By doing so, it allows users to conveniently access real-time data, analyze trends, and share health information with healthcare professionals or family members for better management of their well-being.

Overview:

The IoT pulse oximeter is a next-generation medical device that integrates Internet of Things technology into the traditional pulse oximetry system. Its primary objective is to provide accurate and continuous monitoring of oxygen saturation levels (SpO2) and pulse rate (PR) while offering enhanced data accessibility and analysis

Key Features

- **IoT Connectivity**: The pulse oximeter will be equipped with wireless connectivity capabilities, enabling seamless communication with smartphones and other smart devices through dedicated mobile applications.
- **Real-Time Monitoring**: Users can access real-time SpO2 and PR readings on their connected devices, facilitating immediate health assessment.
- **Data Visualization**: The associated mobile app will present data in a user-friendly and visually appealing manner, allowing users to monitor trends and patterns over time.
- **Health Insights**: The app may offer personalized health insights and recommendations based on the collected data, empowering users to make informed lifestyle choices.
- **User-Friendly Interface**: The IoT pulse oximeter will feature an intuitive and easy-to-navigate interface suitable for users of all ages and technical proficiency levels.
- **Shareability**: Users can securely share their health data with healthcare professionals or family members, facilitating remote monitoring and timely medical interventions.
- **Long Battery Life**: Efforts will be made to optimize power consumption, ensuring an extended battery life for the pulse oximeter's prolonged usability.

The IoT pulse oximeter aims to improve the overall user experience by offering advanced functionalities and leveraging the power of IoT technology to enhance healthcare management. By addressing the limitations of traditional pulse oximeters, this innovative device has the potential to make a positive impact on individuals' health monitoring and contribute to better healthcare outcomes.

Product Validation

Product validation is a critical phase in the development of an IoT pulse oximeter, ensuring that the device meets the necessary standards and functions accurately. The validation process encompasses various tests and assessments to verify accuracy, performance, connectivity, and user-friendliness.

During validation, the IoT pulse oximeter undergoes accuracy testing to ensure precise SpO2 and pulse rate readings. Performance tests evaluate the device's response under different conditions, and battery life assessments determine its power consumption capabilities. Connectivity and data transmission are validated to ensure seamless communication with smart devices.

Usability testing focuses on the user interface and gathers feedback to enhance the user experience. The pulse oximeter's compatibility with various devices is confirmed, and durability tests assess its resilience to different environmental factors.

Regulatory compliance is prioritized, and the device undergoes necessary certifications and approvals. User feedback and field testing provide real-world insights for improvements, and security measures protect data privacy.

Through thorough product validation, the IoT pulse oximeter is refined, reliable, and safe, instilling confidence in consumers and healthcare professionals alike. A successfully validated device sets the stage for a successful market launch, catering to the growing demand for advanced and connected healthcare solutions.

Solution

The IoT pulse oximeter presents a groundbreaking solution to the limitations of traditional pulse oximeters by incorporating Internet of Things (IoT) technology. This innovative medical device offers advanced features to provide accurate and real-time monitoring of oxygen saturation levels (SpO2) and pulse rate (PR) while enhancing data accessibility and analysis.

Key components of the solution include wireless IoT connectivity, enabling seamless communication with smartphones and other smart devices. The IoT pulse oximeter guarantees precise and reliable measurements through high-quality sensors and signal processing algorithms.

The associated mobile application empowers users to monitor their health in real-time, providing intuitive data visualization to track trends and patterns. Furthermore, the app may offer personalized health insights and recommendations based on collected data, assisting users in setting health goals and staying on track.

The user-friendly interface and long battery life ensure effortless operation and continuous monitoring. Data security measures protect user information, maintaining trust and confidentiality.

The solution's compatibility with various devices and adherence to regulatory standards demonstrate its commitment to providing a seamless and safe user experience. By blending accuracy, connectivity, and personalized insights, the IoT pulse oximeter empowers individuals to take control of their health, fostering a healthier and more informed lifestyle.

Methodology

Component Selection

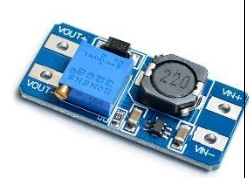
MAX30100 sensor The MAX30100 sensor is an integrated pulse oximeter and heart-rate sensor module widely used in wearable fitness devices and health monitoring systems.

The ESP8266 is a popular and versatile Wi-Fi module and microcontroller platform widely used in IoT (Internet of Things) projects and applications due to its low cost, small size, and built-in Wi-Fi capabilities.

Display: the 0.96-inch 128X64 OLED Display Module I2C IIC Serial White was used.

Li-ion Battery: rechargeability, cost-effectiveness, availability ease of handling, etc.

The MCP73831 is a series of highly integrated single-cell Li-Ion/Li-Polymer charge management controllers used in various battery-powered applications.

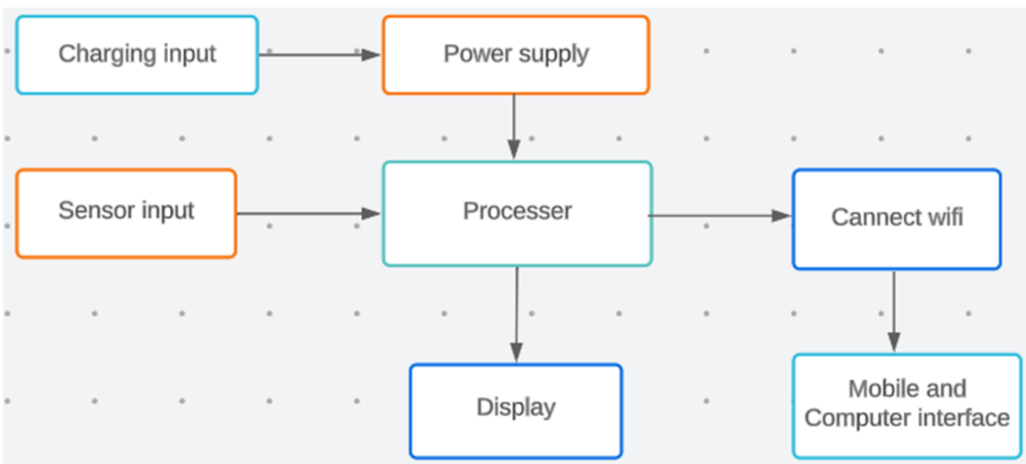


MT3608 DC-DC Step-Up Boost - From the battery, it can output a maximum voltage of around 4V. But for working properly to display, module, and sensor Its need 5V Supply Module 2-24VDC to 28VDC 2A Boost Converter is used.

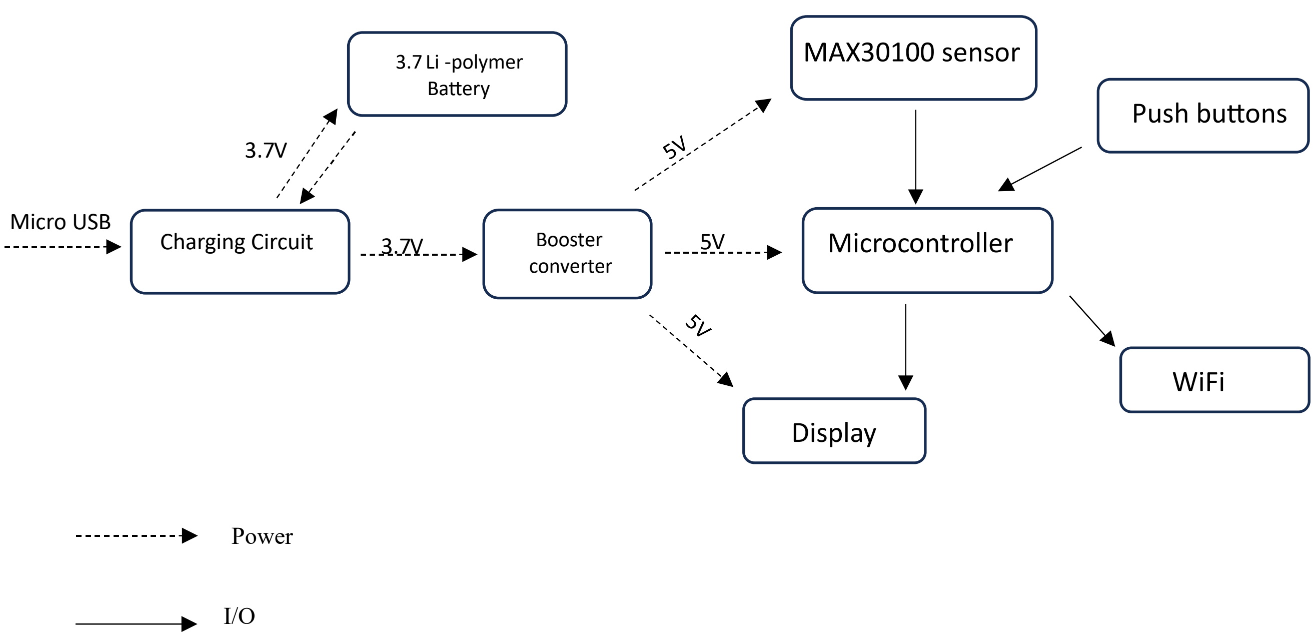
Functionality Block Diagram

The Functionality Block Diagram of the IoT pulse oximeter illustrates the integration of the MAX30100 sensor module, microcontroller (ESP8266), wireless connectivity (if applicable), display (optional), and battery management system to provide accurate real-time monitoring of SpO2 and heart rate with wireless data transmission capabilities.

- **The block diagram of design is as follows.**



- **Circuit Block diagram**

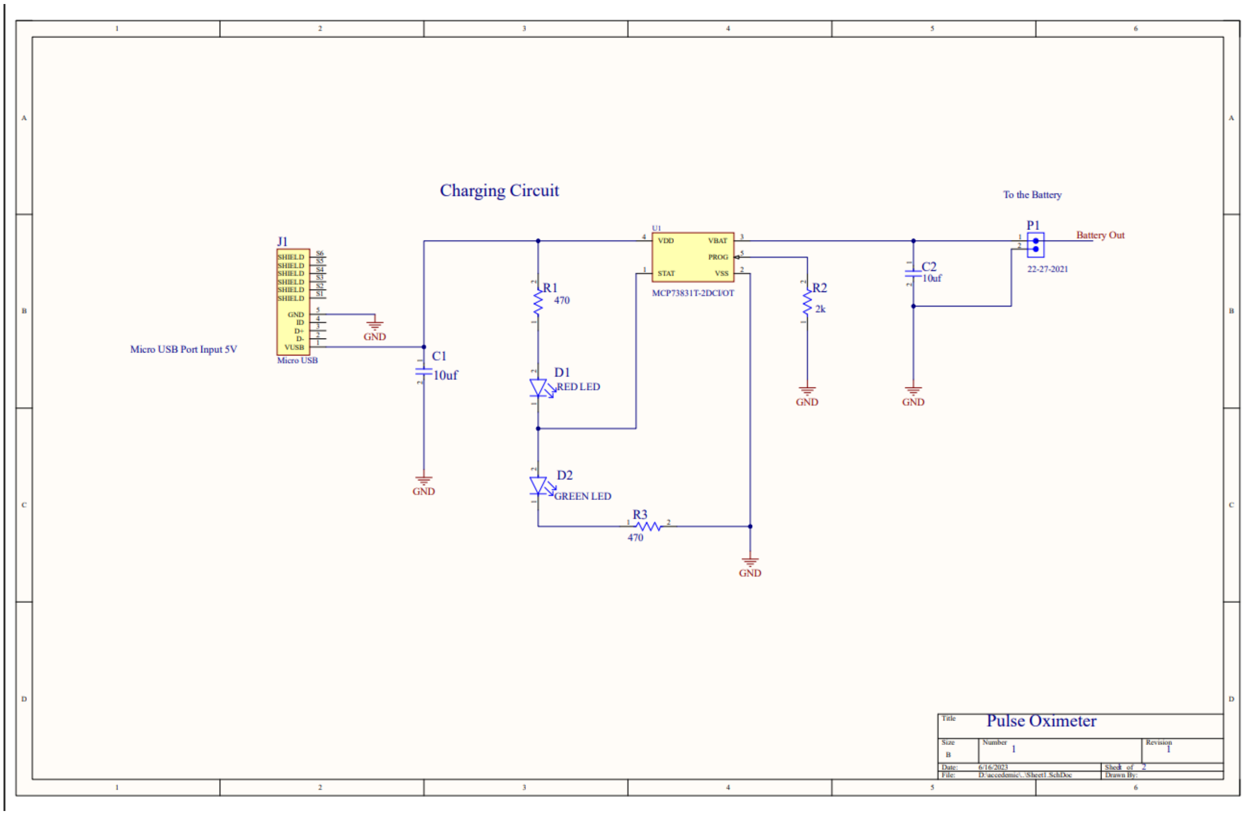


Schematics, PCB, and Encloser

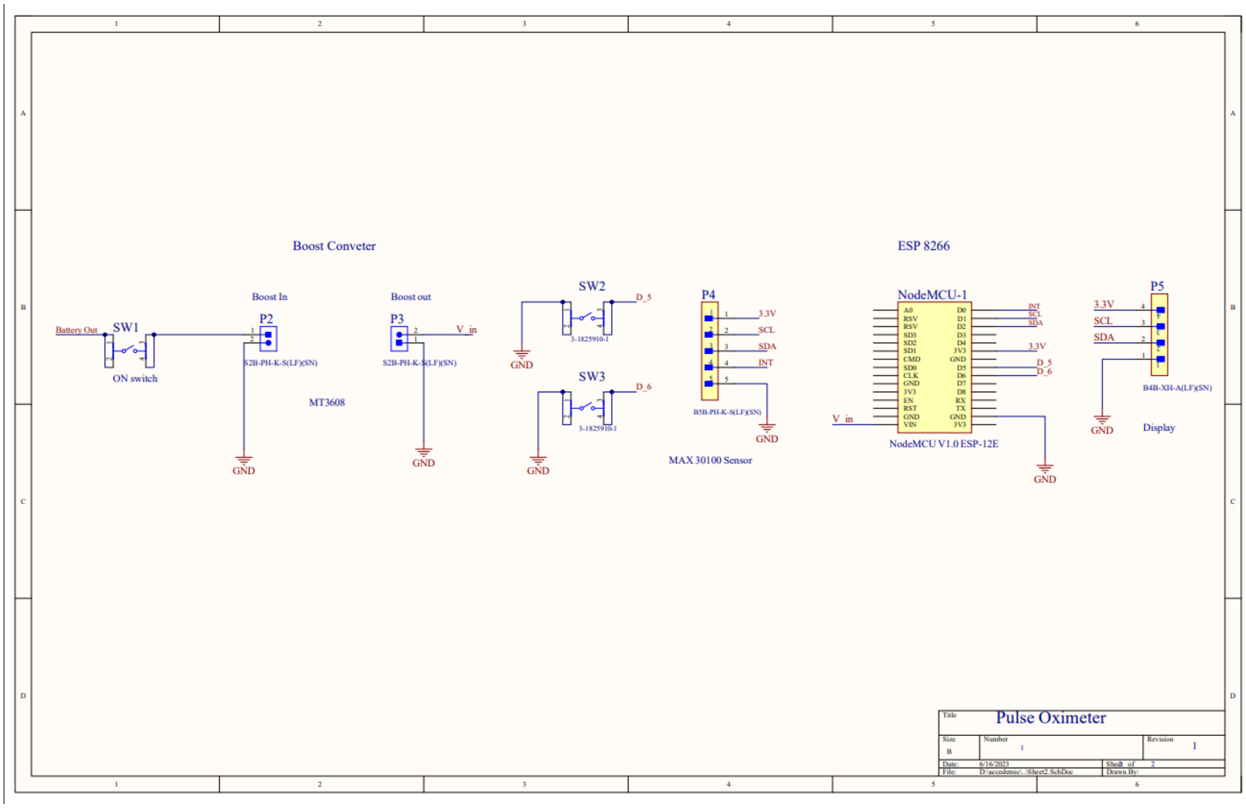
Schematic

For the PCB design of the IoT pulse oximeter, Altium software was utilized to create a well-structured and optimized layout, ensuring proper integration and precise connections of components for efficient functioning of the device.

- Charging Circuit

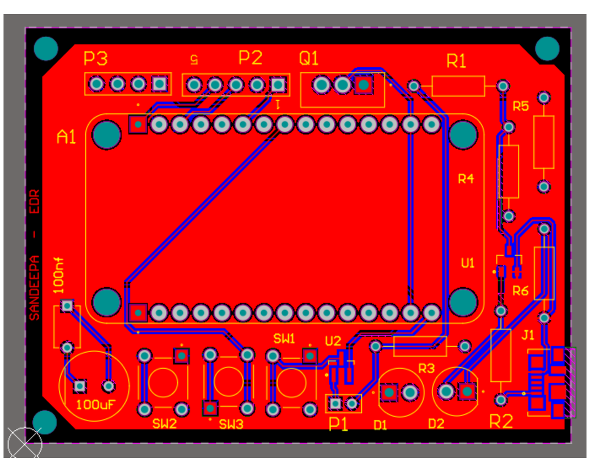


- Main circuit

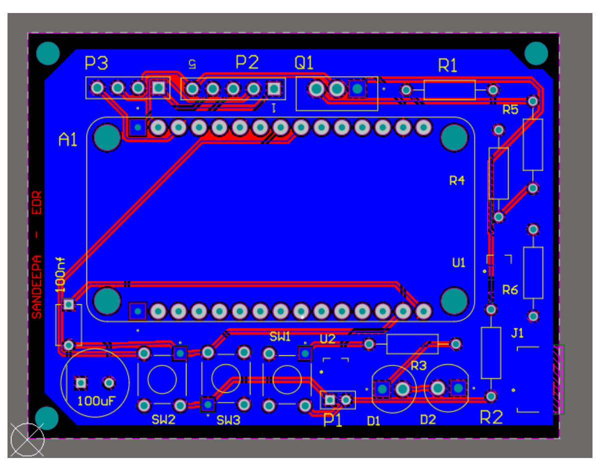


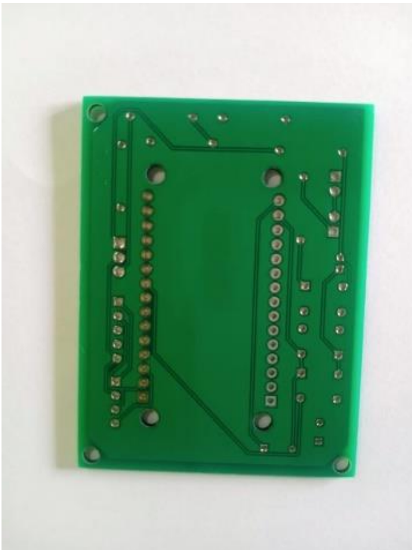
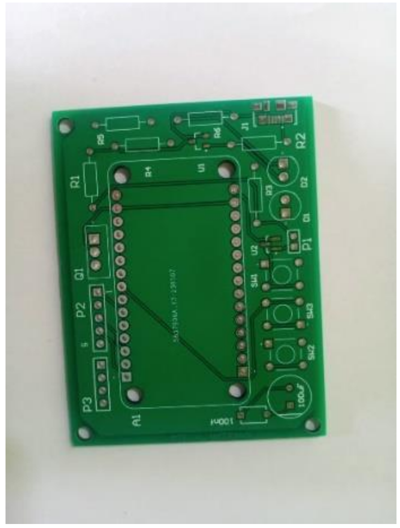
PCB layout

Top layer

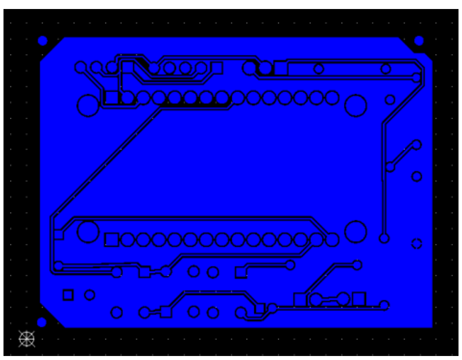
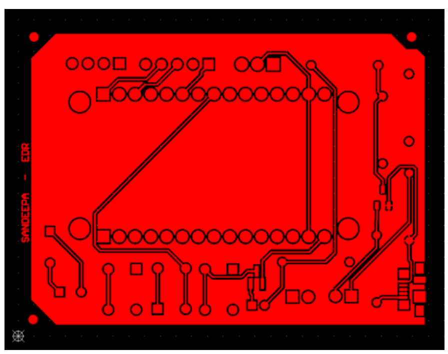
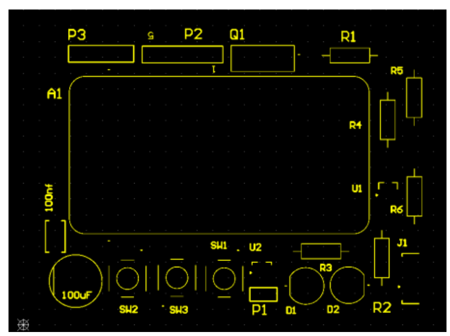


Bottom layer



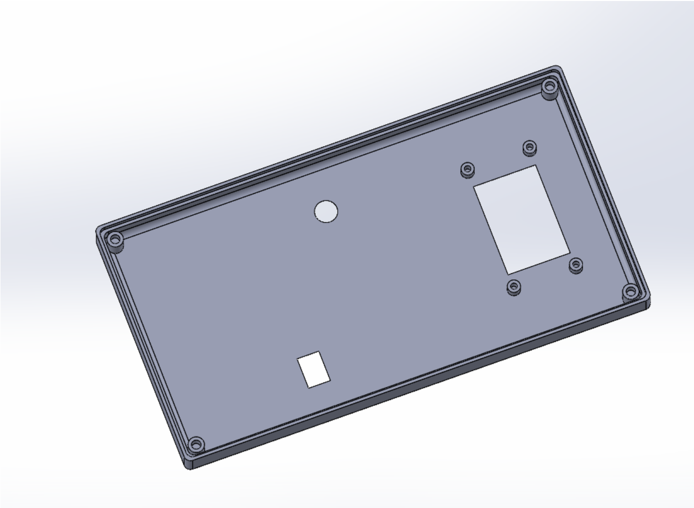
PCB Gerber File

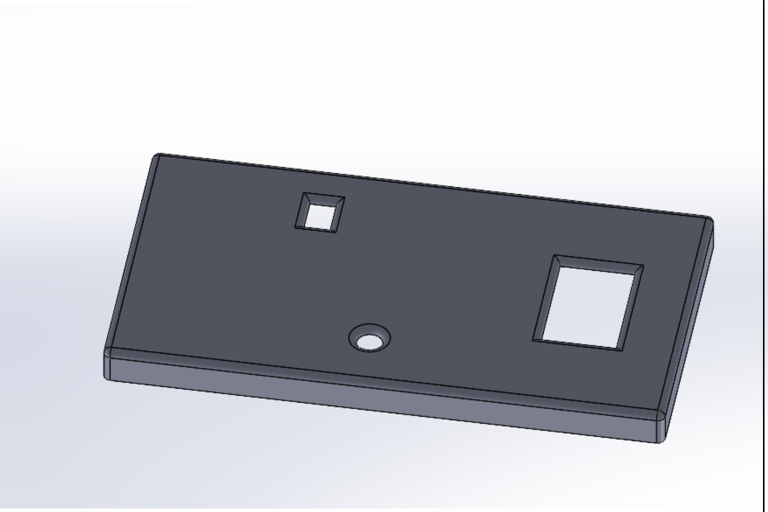
  

Encloser designs

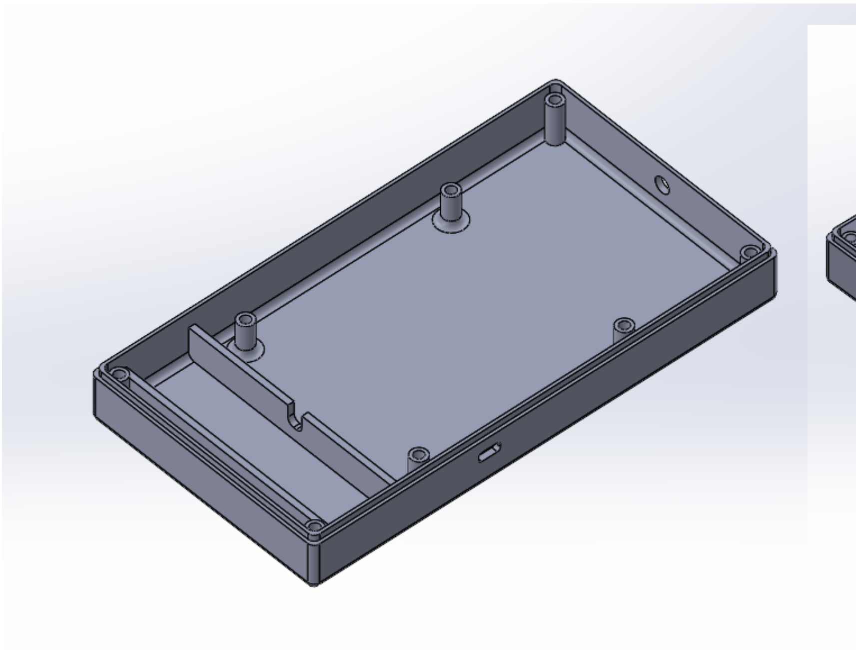
Enclosure design was designed with SOLIDWORKS 2020.enclosure consist of Lid and the Body. Plastic is used in the enclosure. This was designed to contain the main PCB and battery pack as well as the boost converters and buck converters. All of them can be mounted on the body by means of 3mmx 5mm screws.

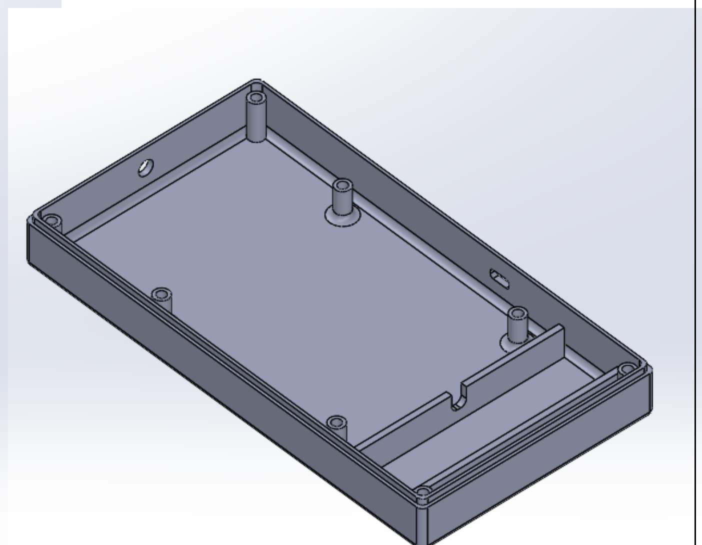
- **Lid Part**

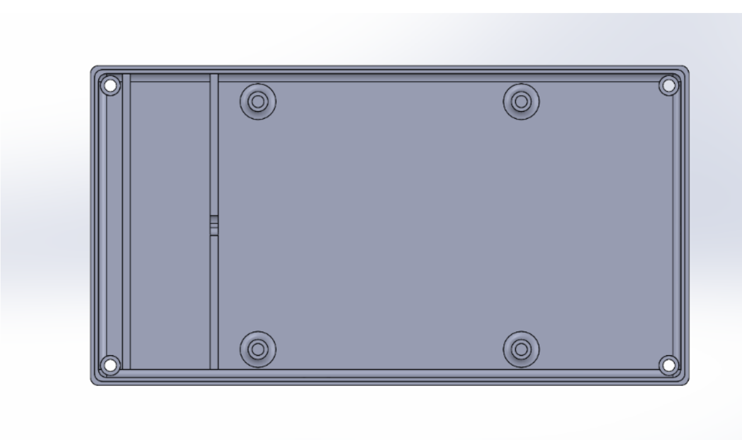




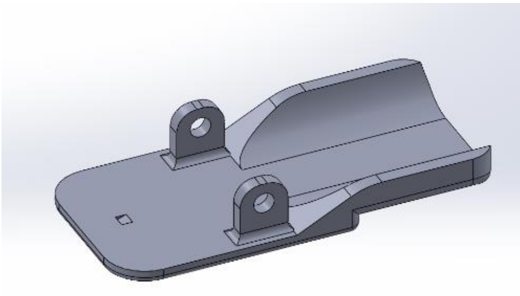
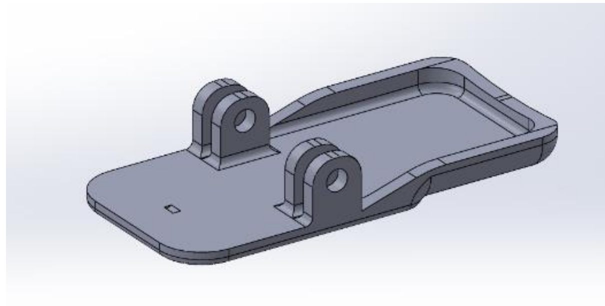
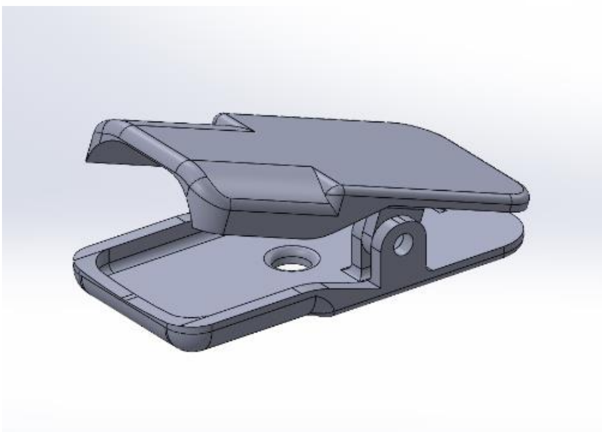
- **Body part**







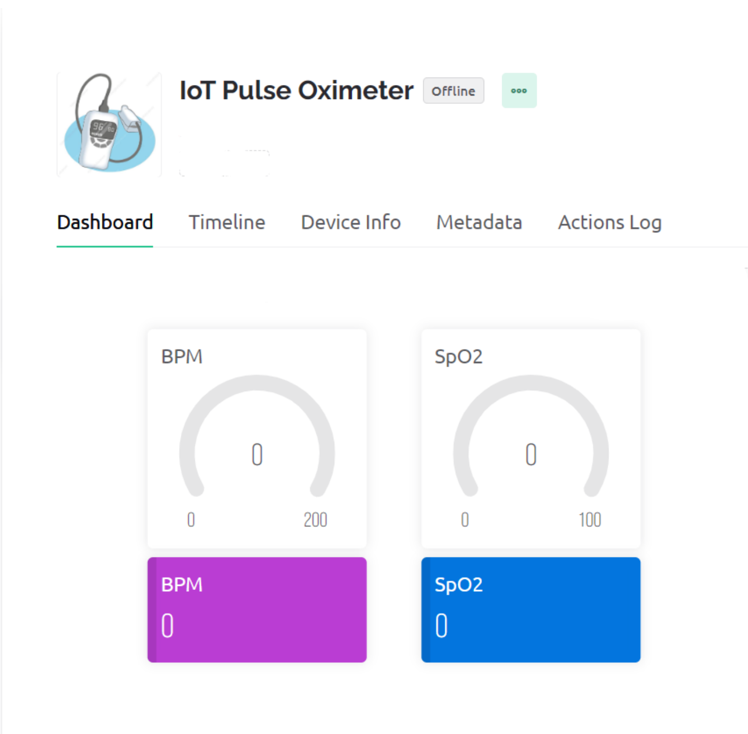
- **Finger clip**

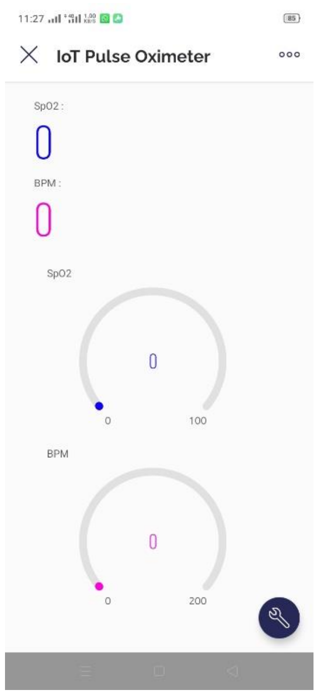

- **Final enclosure**

User Interface

The IoT pulse oximeter is a state-of-the-art medical device that integrates traditional pulse oximetry with Internet of Things (IoT) technology. The user interface of the pulse oximeter is designed for simplicity, providing real-time and accurate readings of oxygen saturation levels (SpO2) and heart rate (HR) in a clear and intuitive format. The OLED display, touch-responsive controls, and customizable settings enhance user-friendliness, allowing users to navigate easily through various features, set alarms, and access historical data for comprehensive health monitoring. The mobile application associated with the pulse oximeter further expands the user experience by offering detailed data visualization, trends, and personalized health insights. With its focus on clarity, accessibility, and continuous monitoring capabilities, the IoT pulse oximeter represents a promising and impactful addition to modern healthcare management and personal well-being.



Desktop interface



Mobile interface

Bill Of Material (BOM)

No	Items		Quantity	Cost per Unit(Rs)	Cost(Rs)
1	ESP8266 board		1	LKR 2,950.00	LKR 2,950.00
2	Max30100 sensor		1	LKR 2,300.00	LKR 2,300.00
3	MCP73831		1	LKR 96.00	LKR 96.00
4	OLED display		1	LKR 1,900.00	LKR 1,900.00
5	Micro USB female		1	LKR 40.00	LKR 40.00
6	Boost converter		1	LKR 240.00	LKR 240.00
7	Resistor	4.7k	1	LKR 60.00	LKR 60.00
8		5.6k	1	LKR 60.00	LKR 60.00
9		15k	1	LKR 60.00	LKR 60.00
10		220	1	LKR 60.00	LKR 60.00
11		1k	1	LKR 60.00	LKR 60.00
12	LED		2	LKR 20.00	LKR 40.00
13	Push Button		2	LKR 30.00	LKR 60.00
14	Switch		1	LKR 50.00	LKR 50.00
15	Pin header	male	1	LKR 40.00	LKR 40.00
16		Female	1	LKR 40.00	LKR 40.00
17	JST wire	2 pin	1	LKR 40.00	LKR 40.00
18		4 pin	1	LKR 60.00	LKR 60.00
19		5 pin	1	LKR 65.00	LKR 65.00
20	Battery		1	LKR 840.00	LKR 840.00
21	Battery case		1	LKR 55.00	LKR 55.00
22	Wire		1m	LKR 40.00	LKR 40.00
23	Solder		2m	LKR 250.00	LKR 500.00
24	Print PCB			LKR 2,700.00	LKR 2,700.00
25	Enclosure			LKR 1,300.00	LKR 1,300.00
26	Heat srink wire		1m	LKR 40.00	LKR 40.00
	TOTAL				LKR 13,914.00

Suppliers

Components	Suppliers
MAX30100 sensor	Mouser Electronics
OLED display	Tronic.lk
Esp8266 board	Mouser Electronics
Li-ion Battery	Tronic.lk
MCP73831	Mouser Electronics
MT3608 DC-DC Step-Up Boost	Microchip.lk
Resistors	Tronic.lk
PCB manufacture	JLC PCB

Instructions for Assembly

Assembling the IoT pulse oximeter involves the integration of various components to create a functional and reliable device. The following step-by-step instructions outline the assembly process:

Step 1: Gather Components

- Collect all the required components for the pulse oximeter assembly, including the PCB board, MAX30100 sensor module, display (if applicable), microcontroller (e.g., ESP8266), battery, battery management system (MCP73831), LEDs, resistors, capacitors, and other electronic components as per the design specifications.

Step 2: PCB Assembly

- Solder the electronic components onto the PCB board following the circuit design and layout.
- Pay close attention to the orientation and placement of components to ensure proper functionality and alignment.
- Double-check for any solder bridges or cold solder joints, ensuring all connections are secure.

Step 3: Sensor Integration

- Carefully connect the MAX30100 sensor module to the designated headers on the PCB board.
- Ensure a secure and stable connection, allowing the sensor to receive data accurately.

Step 4: Microcontroller Integration

- Connect the microcontroller (ESP8266) to the designated headers on the PCB board.

- Program the microcontroller with the necessary firmware to control the pulse oximeter's functions, data processing, and connectivity.

Step 5: Display Integration (if applicable)

- If the pulse oximeter includes a display, attach it to the designated headers on the PCB board.
- Ensure the display is properly aligned and securely mounted for clear readability.

Step 6: Battery Integration and Charging Circuit

- Connect the battery to the PCB board, ensuring proper polarity and voltage compatibility.
- Integrate the battery management system (e.g., MCP73831/2/3) to manage battery charging, preventing overcharging and optimizing battery life.

Step 7: Final Testing

- Conduct a thorough functionality test to verify the pulse oximeter's accuracy, real-time monitoring, and data visualization.
- Test the wireless connectivity (if applicable) and ensure seamless communication with smart devices.

Step 8: Enclosure and Finalization

- If applicable, assemble the pulse oximeter components into a compact and user-friendly enclosure.
- Ensure the enclosure provides adequate protection and easy access to controls and display.
- Double-check for any potential safety hazards, such as exposed wires or sharp edges.

Step 9: Quality Assurance

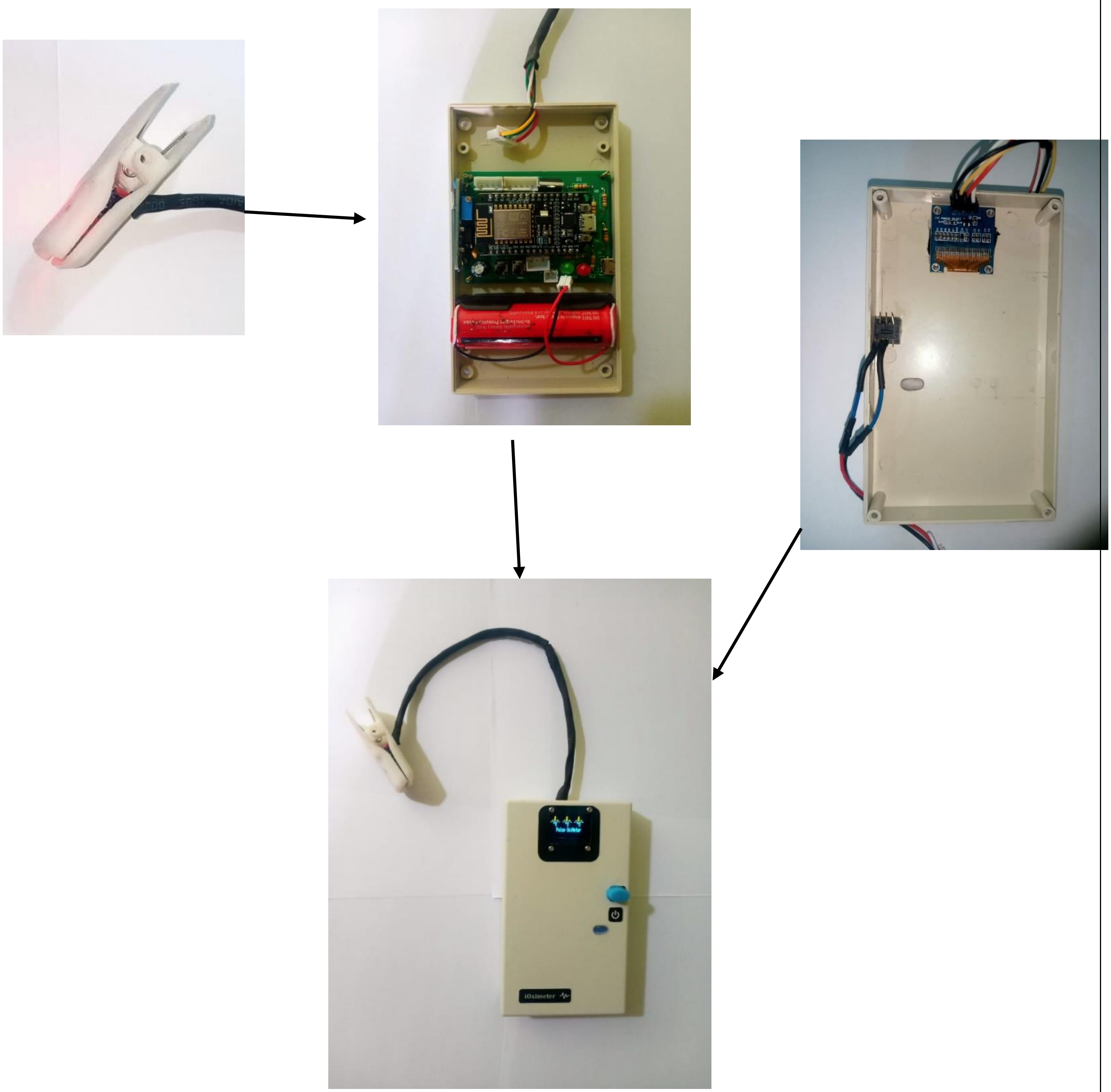
- Perform quality assurance checks on the assembled pulse oximeter, including accuracy, safety, and reliability assessments.
- Address any issues or defects discovered during the quality assurance process.

Step 10: User Manual and Documentation

- Prepare a comprehensive user manual that explains the pulse oximeter's functionalities, usage instructions, and maintenance guidelines.
- Document the assembly process, component specifications, and testing results for future reference and troubleshooting.

By following these instructions, the assembly of the IoT pulse oximeter can be completed efficiently and effectively, resulting in a functional and reliable medical device ready for market launch and user utilization.

Assembling



Test for functionality

To ensure the IoT pulse oximeter functions accurately and reliably, comprehensive testing is essential. The following tests should be conducted to verify the functionality of the pulse oximeter:

1. **SpO2 and Heart Rate Accuracy Test:**
    - Use a calibrated reference pulse oximeter or medical-grade equipment to measure SpO2 and heart rate simultaneously with the IoT pulse oximeter.
    - Compare the readings obtained from the IoT pulse oximeter with the reference measurements to assess accuracy.
2. **Real-Time Monitoring Test:**
    - Continuously monitor SpO2 and heart rate with the pulse oximeter while the user performs various activities, including rest, light exercise, and motion.
    - Verify that the pulse oximeter provides real-time and consistent readings under different conditions.
3. **Wireless Connectivity Test (if applicable):**
    - Connect the pulse oximeter to a smartphone or smart device using the wireless connectivity feature (e.g., Bluetooth or Wi-Fi).
    - Transfer data from the pulse oximeter to the associated mobile application to confirm seamless data transmission.
4. **Battery Life Test:**
    - Fully charge the pulse oximeter's battery and record the time taken for the battery to deplete fully during continuous use.
    - Verify that the battery life aligns with the specified duration in the product specifications.
5. **Display and User Interface Test:**
    - Verify that the display accurately shows SpO2 and heart rate readings and other relevant information.
    - Evaluate the user interface for intuitive navigation and ease of use.
6. **Alarm and Alert Test (if applicable):**
    - Set predetermined thresholds for SpO2 or heart rate values to trigger alarms or alerts.
    - Verify that the pulse oximeter issues appropriate alerts when the thresholds are breached.
7. **Battery Charging and Management Test:**
    - Test the battery charging feature and confirm that the pulse oximeter charges the battery correctly and efficiently.
    - Verify that the battery management system prevents overcharging and maintains battery health.
8. **Ambient Light Cancellation Test:**
    - Expose the pulse oximeter to different ambient light conditions, such as indoor light and direct sunlight.

- Ensure that the pulse oximeter accurately measures SpO2 and heart rate while minimizing the impact of external light sources.

9. **Safety and Regulatory Compliance Test:**
    - Conduct safety tests to ensure that the pulse oximeter meets relevant safety standards and regulations.
    - Verify that the device operates within safe temperature ranges and does not pose any hazards to users.
10. **User Acceptance and Feedback:**
    - Gather feedback from potential users and medical professionals who use the pulse oximeter during the testing phase.
    - Incorporate user feedback to make necessary improvements and refinements to the product.

By performing these comprehensive tests, the functionality and reliability of the IoT pulse oximeter can be verified, ensuring that it meets the required performance standards and delivers accurate and valuable health monitoring capabilities to users.

References

0A-ESP8266 Datasheet EN v4.3.pages (components101.com)

MAX30100.pdf (analog.com)

MCP73831/MCP73832 Data Sheet (microchip.com)

MT3608 pdf, MT3608 Description, MT3608 Datasheet, MT3608 view ::: ALLDATASHEET :::

APPENDIX

```
#include <Wire.h>
#include "MAX30100_PulseOximeter.h"
#include <Adafruit_GFX.h>
#define BLYNK_PRINT Serial
#include <BlynkSimpleEsp8266.h>
#include <Adafruit_SSD1306.h>
#define ENABLE_MAX30100 1
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 32

char auth[] = "y80-1oVnZdSoAvAaGq1Z8pA9s98L4vwz";
char ssid[] = "Redmi Note 9 Pro";
char pass[] = "12345678";

#define OLED_RESET     -1
#define SCREEN_ADDRESS 0x3c
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);
#if ENABLE_MAX30100
#define REPORTING_PERIOD_MS     5000

PulseOximeter pox;
#endif
uint32_t tsLastReport = 0;
int xPos = 0;

void onBeatDetected()
{
  Serial.println("Beat!");
  heart_beat(&xPos);
}
void setup()
{
  Serial.begin(115200);
  Serial.println("SSD1306 128x64 OLED TEST");

  if (!display.begin(SSD1306_SWITCHCAPVCC, SCREEN_ADDRESS)) {
    Serial.println(F("SSD1306 allocation failed"));
    for (;;);
  }

  display.clearDisplay();
  display.setTextSize(1);
```

```
  display.setTextColor(WHITE);
  display.setCursor(20, 18);
  display.print("Pulse OxiMeter");
  int temp1 = 0;
  int temp2 = 40;
  int temp3 = 80;
  heart_beat(&temp1);
  heart_beat(&temp2);
  heart_beat(&temp3);
  xPos = 0;
  display.display();
  delay(2000);
  display.cp437(true);
  display.clearDisplay();
  Serial.print("Initializing pulse oximeter..");
  Blynk.begin(auth, ssid, pass);
#if ENABLE_MAX30100

  if (!pox.begin()) {
    Serial.println("FAILED");
    for (;;);
  } else {
    Serial.println("SUCCESS");
  }
  pox.setIRLedCurrent(MAX30100_LED_CURR_7_6MA);
  pox.setOnBeatDetectedCallback(onBeatDetected);
  display_data(0, 0);
#endif
}
void loop()
{
#if ENABLE_MAX30100
  pox.update();
  Blynk.run();
  int bpm = 0;
  int spo2 = 0;

  if (millis() - tsLastReport > REPORTING_PERIOD_MS) {

    bpm = pox.getHeartRate();
    spo2 = pox.getSpO2();
    Serial.println(bpm);

    Serial.println(spo2);
```

```
    tsLastReport = millis();
    Blynk.virtualWrite(V0, bpm);
    Blynk.virtualWrite(V1, spo2);
    display_data(bpm, spo2);
  }
#endif
  drawLine(&xPos);
}
void display_data(int bpm, int spo2) {
  display.fillRect(0, 18, 127, 15, SSD1306_BLACK);
  display.setTextSize(1);
  display.setTextColor(WHITE);
  display.setCursor(0, 18);

  display.print("BPM ");
  display.print(bpm);
  display.display();
  display.setTextSize(1);
  display.setTextColor(WHITE);
  display.setCursor(64, 18);

  display.print("Spo2 ");
  display.print(spo2);
  display.print("%");
  display.display();
}
void drawLine(int *x_pos) {
  // Draw a single pixel in white
  display.drawPixel(*x_pos, 8, SSD1306_WHITE);
  display.drawPixel((*x_pos)++, 8, SSD1306_WHITE);
  display.drawPixel((*x_pos)++, 8, SSD1306_WHITE);
  display.drawPixel((*x_pos)++, 8, SSD1306_WHITE);
  display.drawPixel((*x_pos), 8, BLACK);  // -----
  //Serial.println(*x_pos);
  display.fillRect(*x_pos, 0, 31, 16, SSD1306_BLACK);
  display.display();
  delay(1);
  if (*x_pos >= SCREEN_WIDTH) {
    *x_pos = 0;
  }
}
void heart_beat(int *x_pos) {
  /************************************************/
  //display.clearDisplay();
  display.fillRect(*x_pos, 0, 30, 15, SSD1306_BLACK);
```

```
// Draw a single pixel in white
display.drawPixel(*x_pos + 0, 8, SSD1306_WHITE);
display.drawPixel(*x_pos + 1, 8, SSD1306_WHITE);
display.drawPixel(*x_pos + 2, 8, SSD1306_WHITE);
display.drawPixel(*x_pos + 3, 8, SSD1306_WHITE);
display.drawPixel(*x_pos + 4, 8, BLACK); // -----
//display.display();
//delay(1);
display.drawPixel(*x_pos + 5, 7, SSD1306_WHITE);
display.drawPixel(*x_pos + 6, 6, SSD1306_WHITE);
display.drawPixel(*x_pos + 7, 7, SSD1306_WHITE); // .~.
//display.display();
//delay(1);
display.drawPixel(*x_pos + 8, 8, SSD1306_WHITE);
display.drawPixel(*x_pos + 9, 8, SSD1306_WHITE); // --
//display.display();
//delay(1);
/******************************************/
display.drawPixel(*x_pos + 10, 8, SSD1306_WHITE);
display.drawPixel(*x_pos + 10, 9, SSD1306_WHITE);
display.drawPixel(*x_pos + 11, 10, SSD1306_WHITE);
display.drawPixel(*x_pos + 11, 11, SSD1306_WHITE);
//display.display();
//delay(1);
/******************************************/
display.drawPixel(*x_pos + 12, 10, SSD1306_WHITE);
display.drawPixel(*x_pos + 12, 9, SSD1306_WHITE);
display.drawPixel(*x_pos + 12, 8, SSD1306_WHITE);
display.drawPixel(*x_pos + 12, 7, SSD1306_WHITE);
//display.display();
//delay(1);
display.drawPixel(*x_pos + 13, 6, SSD1306_WHITE);
display.drawPixel(*x_pos + 13, 5, SSD1306_WHITE);
display.drawPixel(*x_pos + 13, 4, SSD1306_WHITE);
display.drawPixel(*x_pos + 13, 3, SSD1306_WHITE);
//display.display();
//delay(1);
display.drawPixel(*x_pos + 14, 2, SSD1306_WHITE);
display.drawPixel(*x_pos + 14, 1, SSD1306_WHITE);
display.drawPixel(*x_pos + 14, 0, SSD1306_WHITE);
display.drawPixel(*x_pos + 14, 0, SSD1306_WHITE);
//display.display();
//delay(1);
/******************************************/
display.drawPixel(*x_pos + 15, 0, SSD1306_WHITE);
```

```
display.drawPixel(*x_pos + 15, 1, SSD1306_WHITE);
display.drawPixel(*x_pos + 15, 2, SSD1306_WHITE);
display.drawPixel(*x_pos + 15, 3, SSD1306_WHITE);
//display.display();
//delay(1);
display.drawPixel(*x_pos + 15, 4, SSD1306_WHITE);
display.drawPixel(*x_pos + 15, 5, SSD1306_WHITE);
display.drawPixel(*x_pos + 16, 6, SSD1306_WHITE);
display.drawPixel(*x_pos + 16, 7, SSD1306_WHITE);
//display.display();
//delay(1);
display.drawPixel(*x_pos + 16, 8, SSD1306_WHITE);
display.drawPixel(*x_pos + 16, 9, SSD1306_WHITE);
display.drawPixel(*x_pos + 16, 10, SSD1306_WHITE);
display.drawPixel(*x_pos + 16, 11, SSD1306_WHITE);
//display.display();
//delay(1);
display.drawPixel(*x_pos + 17, 12, SSD1306_WHITE);
display.drawPixel(*x_pos + 17, 13, SSD1306_WHITE);
display.drawPixel(*x_pos + 17, 14, SSD1306_WHITE);
display.drawPixel(*x_pos + 17, 15, SSD1306_WHITE);
//display.display();
//delay(1);
display.drawPixel(*x_pos + 18, 15, SSD1306_WHITE);
display.drawPixel(*x_pos + 18, 14, SSD1306_WHITE);
display.drawPixel(*x_pos + 18, 13, SSD1306_WHITE);
display.drawPixel(*x_pos + 18, 12, SSD1306_WHITE);
//display.display();
//delay(1);
display.drawPixel(*x_pos + 19, 11, SSD1306_WHITE);
display.drawPixel(*x_pos + 19, 10, SSD1306_WHITE);
display.drawPixel(*x_pos + 19, 9, SSD1306_WHITE);
display.drawPixel(*x_pos + 19, 8, SSD1306_WHITE);
//display.display();
//delay(1);
/*****************************************************/
display.drawPixel(*x_pos + 20, 8, SSD1306_WHITE);
display.drawPixel(*x_pos + 21, 8, SSD1306_WHITE);
//display.display();
//delay(1);
/*****************************************************/
display.drawPixel(*x_pos + 22, 7, SSD1306_WHITE);
display.drawPixel(*x_pos + 23, 6, SSD1306_WHITE);
display.drawPixel(*x_pos + 24, 6, SSD1306_WHITE);
display.drawPixel(*x_pos + 25, 7, SSD1306_WHITE);
```

```
  //display.display();
  //delay(1);
  /************************************************/
  display.drawPixel(*x_pos + 26, 8, SSD1306_WHITE);
  display.drawPixel(*x_pos + 27, 8, SSD1306_WHITE);
  display.drawPixel(*x_pos + 28, 8, SSD1306_WHITE);
  display.drawPixel(*x_pos + 29, 8, SSD1306_WHITE);
  display.drawPixel(*x_pos + 30, 8, SSD1306_WHITE); // -----
  *x_pos = *x_pos + 30;
  display.display();
  delay(1);
}
```