Project.no: 02	DESIGN AND DEVOLOP THE MEDICAL BASED APPLICATION
Date:	Heart rate sensing

OBJECTIVE

- To detect abnormalities such as arrhythmias and provide diagnostic readings to address underlying heart issues.
- To develop a product that helps in development of 3rd SDG (Good health and Well being).s

AIM

To find your target heart rate to maximize your cardiovascular exercise, the first step is determining your maximum heart rate. Your maximum heart rate is 220 minus your age. Your target heart rate for moderate to vigorous exercise is about 50%–85% of your maximum heart rate.

INTRODUCTION

The heart is one of the most important organs in the human body. It acts as a pump for circulating oxygen and blood throughout the body, thus keeping the functionality of the body intact. A heartbeat can be defined as a two-part pumping action of the heart which occursfor almost a second. It is produced due to the contraction of the heart. When blood collects in upper chambers, the SA(Sino Atrial) node sends out an electrical signal which in turn causes the atria to contract. This contraction then pushes the blood through tricuspid and the mitral valves; this phase of the pumping system is called diastole. The next phase begins when the ventricles are completely filled with blood. The electrical signals generating fromSA node reach the ventricle and cause them to contract. In today's scenario, health problems related to heart are very common. Heart diseases are one of the most important causes of death among men and women. It claims approximately 1 million deaths every year. Heart rate is a critical parameter in the functioning of the heart. Therefore heart rate monitoringis crucial in the study of heart performance and thereby maintaining heart health

CIRCUIT DIAGRAM

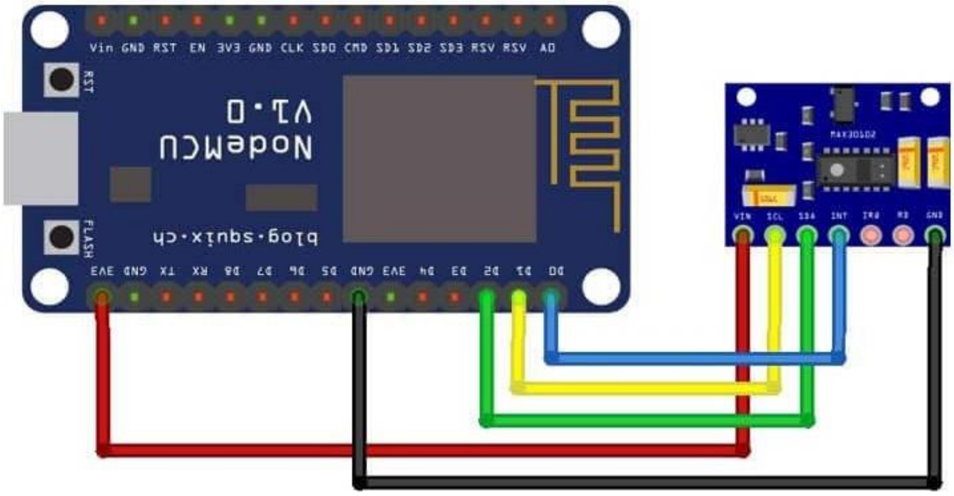

Figure 2.3 Circuit Diagram for Heart Rate Sensor

WORKING

This health monitoring system project includes various components such as a microcontroller, 5V regulated power supply, heartbeat sensor, WiFi module, receiver module and LCD display. The microcontroller is used as the brain of the entire project to monitor the patient's heart rate and pulse rate. How this monitoring system project works is demonstrated using a block diagram containing various blocks such as: B. A power block that powers the entire circuit, a temperature sensor that calculates the patient, and a heart rate sensor that monitors the patient's heart rate. The heart rate sensor module has lights to help you measure your heart rate. When you place your finger on the pulse sensor, the reflected light changes according to the amount of blood in the capillaries. This variation in light transmission and datareceived by the reflections of the Blynk app can be obtained as a pulse from the pulse sensor output. You can then adjust this pulse to measure your heart rate and program it to read as yourheart rate using the Node MCU accordingly. In the transmission part, there is a heartbeat sensorthat monitors the heartbeat, and the data is transmitted to the microcontroller. The data is sent first and then wirelessly encoded into serial data by the radio frequency module. Your heart rate per minute is shown on the LCD display. The data is sent to the receiver using IOT. In thereceiver section, the receiver is placed on the other end to receive the data, the received data isdecoded using the decoder, and the transmitted data (heartbeat pulse) is stored in the microcontroller. Compare with and get the resulting data on the displayed LCD screen. The doctor receives the data, continuously reads the patient's health status such as heart rate and pulse rate, and displays the result on the doctor's mobile phone.

SUPPORTING SYSTEM – SOFTWARE DETAILS/HARDWARE DETAILS

SOFTWARE DETAILS

Arduino IDE

Arduino Integrated Development Environment (IDE) is an open source IDE that allows users to write code and upload it to any Arduino board. Arduino IDE is written in Java and is compatible with Windows, macOS and Linux operating systems.

Wokwi

The wokwi is an online Electronics simulator. You can use it to simulate Arduino, ESP32, STM32, and many other popular boards, parts and sensors. Here are some quick examples of things you can make with Wokwi: Arduino Uno "Hello World"

Proteus (for circuit diagram and simulation)

Proteus incorporates many functions derived from several other languages: C, BASIC, Assembly, Clipper/dBase; it is especially versatile in dealing with strings, having hundreds of dedicated functions; this makes it one of the richest languages for text manipulation.

HARDWARE DETAILS

ESP32

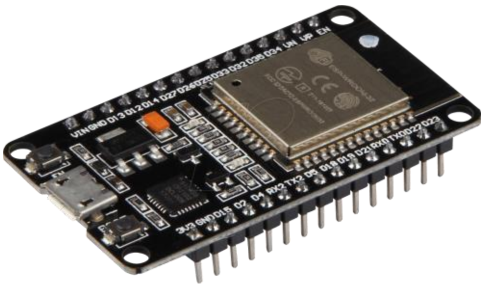

Figure 2.1: ESP32 Development Board

The ESP32 is a highly versatile microcontroller renowned for its dual-core processor and built-in Wi-Fi and Bluetooth connectivity. It is widely used in IoT projects due to its low power consumption and rich set of peripherals. With its robust performance and compatibility with multiple programming languages, the ESP32 is suitable for a wide range of applications, from home automation to industrial monitor.

PIN CONFIGURATION

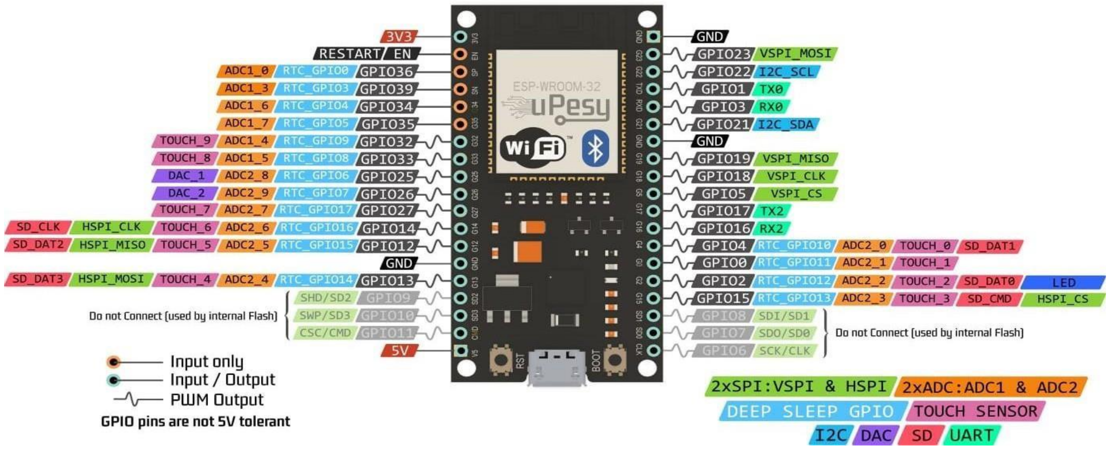

Figure 2.2 ESP32 Development Board

- GPIO Pins: Typically labeled GPIO0 to GPIO39, these pins can be configured as digital inputs or outputs.
- Analog Pins: Usually labeled A0 to A# (depending on the board), these pins support analog input.
- Power Pins: Includes VCC (3.3V) and GND pins for power supply connections.
- Communication Interfaces: Pins dedicated to SPI, I2C, UART, and other communication protocols.
- Special Function Pins: Such as EN (enable), BOOT (boot mode selection), and RST (reset).

MAX30102 SP02 Heart Rate Sensor

The MAX30102 is a highly integrated sensor module designed for measuring blood oxygen saturation (SpO2) and heart rate (HR), commonly used in wearable technology and fitness applications. It employs an optical sensing method known as photoplethysmography (PPG), using red and infrared LEDs to detect changes in light absorption due to blood flow. This technology allows for non-invasive monitoring of cardiovascular health, making it suitable for fitness trackers, smartwatches, and medical devices. The sensor is compact and designed for low power consumption, ideal for battery-powered devices. Additionally, it supports I2C communication, facilitating integration with microcontrollers like Arduino and Raspberry Pi. Its versatility and ease of have made it a popular choice for developers creating health-related applications and wearable technology.

Figure 2.2 : Heartrate sensor

PROGRAM

```
#include <OneWire.h>

//MAX30100 ESP8266 WebServer
#include <ESP8266WebServer.h>
#include <Wire.h>
#include "MAX30100_PulseOximeter.h"


#define REPORTING_PERIOD_MS     1000

float BPM, SpO2;

/*Put your SSID & Password*/
const char* ssid = "xxxxxxx";  // Enter SSID here
const char* password = "xxxxxxx";  //Enter Password here

PulseOximeter pox;
uint32_t tsLastReport = 0;

ESP8266WebServer server(80);

void setup() {
  Serial.begin(115200);
  pinMode(16, OUTPUT);
  delay(100);

  Serial.println("Connecting to ");
  Serial.println(ssid);

  //connect to your local wi-fi network
  WiFi.begin(ssid, password);
```

```
  //check wi-fi is connected to wi-fi network
  while (WiFi.status() != WL_CONNECTED) {
  delay(1000);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected..!");
  Serial.print("Got IP: ");  Serial.println(WiFi.localIP());

  server.on("/", handle_OnConnect);
  server.onNotFound(handle_NotFound);

  server.begin();
  Serial.println("HTTP server started");
  .print("Initializing pulse oximeter..");

  if (!pox.begin()) {
    Serial.println("FAILED");
    for (;;);
  } else {
    Serial.println("SUCCESS");

  }
}
void loop() {
  server.handleClient();
  pox.update();
  if (millis() - tsLastReport > REPORTING_PERIOD_MS) {

    BPM = pox.getHeartRate();
    SpO2 = pox.getSpO2();

    Serial.print("BPM: ");
    Serial.println(BPM);

    Serial.print("SpO2: ");
    Serial.print(SpO2);
    Serial.println("%");

    Serial.println("*********************************");
    Serial.println();
    tsLastReport = millis();
  }


}

void handle_OnConnect() {
```

```
  server.send(200, "text/html", SendHTML(BPM, SpO2));
}

void handle_NotFound() {
  server.send(404, "text/plain", "Not found");
}

String SendHTML(float BPM, float SpO2) {
  String ptr = "<!DOCTYPE html>";
  ptr += "<html>";
  ptr += "<head>";
  ptr += "<title>ESP8266 WebServer</title>";
  ptr += "<meta name='viewport' content='width=device-width, initial-scale=1.0'>";
  ptr += "<link rel='stylesheet' href='https://cdnjs.cloudflare.com/ajax/libs/font
/5.7.2/css/all.min.css'>";
  ptr += "<link rel='stylesheet' type='text/css' href='styles.css'>";
  ptr += "<style>";
  ptr += "body { background-color: #fff; font-family: sans-serif; color: #333333; font:
14px Helvetica, sans-serif box-sizing: border-box;}";
  ptr += "#page { margin: 20px; background-color: #fff;}";
  ptr += ".container { height: inherit; padding-bottom: 20px;}";
  ptr += ".header { padding: 20px;}";
  ptr += ".header h1 { padding-bottom: 0.3em; color: #008080; font-size: 45px; font-
weight: bold; font-family: Garmond, 'sans-serif'; text-align: center;}";
  ptr += "h2 { padding-bottom: 0.2em; border-bottom: 1px solid #eee; margin: 2px; text-
align: left;}";
  ptr += ".header h3 { font-weight: bold; font-family: Arial, 'sans-serif'; font-size: 17px;
color: #b6b6b6; text-align: center;}";
  ptr += ".box-full { padding: 20px; border 1px solid #ddd; border-radius: 1em 1em 1em
1em; box-shadow: 1px 7px 7px 1px rgba(0,0,0,0.4); background: #fff; margin: 20px;
width: 300px;}";
  ptr += "@media (max-width: 494px) { #page { width: inherit; margin: 5px auto; }
#content { padding: 1px;} .box-full { margin: 8px 8px 12px 8px; padding: 10px; width:
inherit;; float: none; } }";
  ptr += "@media (min-width: 494px) and (max-width: 980px) { #page { width: 465px;
margin 0 auto; } .box-full { width: 380px; } }";
  ptr += "@media (min-width: 980px) { #page { width: 930px; margin: auto; } }";
  ptr += ".sensor { margin: 12px 0px; font-size: 2.5rem;}";
  ptr += ".sensor-labels { font-size: 1rem; vertical-align: middle; padding-bottom:
15px;}";
  ptr += ".units { font-size: 1.2rem;}";
  ptr += "hr { height: 1px; color: #eee; background-color: #eee; border: none;}";
  ptr += "</style>";

  //Ajax Code Start
  ptr += "<script>\n";
  ptr += "setInterval(loadDoc,1000);\n";
  ptr += "function loadDoc() {\n";
  ptr += "var xhttp = new XMLHttpRequest();\n";
  ptr += "xhttp.onreadystatechange = function() {\n";
```

```
  ptr += "if (this.readyState == 4 && this.status == 200) {\n";
  ptr += "document.body.innerHTML =this.responseText}\n";
  ptr += "};\n";
  ptr += "xhttp.open(\"GET\", \"/\", true);\n";
  ptr += "xhttp.send();\n";
  ptr += "}\n";
  ptr += "</script>\n";
  //Ajax Code END

  ptr += "</head>";
  ptr += "<body>";
  ptr += "<div id='page'>";
  ptr += "<div class='header'>";
  ptr += "<h1>MAX30100 ESP8266 WebServer</h1>";
  href='https://iotprojectsideas.com'>https://theiotprojects.com</a></h3>
"; ptr += "</div>";
  ptr += "<div id='content' align='center'>";
  ptr += "<div class='box-full' align='left'>";
  ptr += "<h2>Sensor Readings</h2>";
  ptr += "<div class='sensors-container'>";

  //For Heart Rate
  ptr += "<p class='sensor'>";
  ptr += "<i class='fas fa-heartbeat' style='color:#cc3300'></i>";
  ptr += "<span class='sensor-labels'> Heart Rate </span>";
  ptr += (int)BPM;
  ptr += "<sup class='units'>BPM</sup>";
  ptr += "</p>";
  ptr += "<hr>";

  //For Sp02
  ptr += "<p class='sensor'>";
  ptr += "<i class='fas fa-burn' style='color:#f7347a'></i>";
  ptr += "<span class='sensor-labels'> Sp02 </span>";
  ptr += (int)SpO2;
  ptr += "<sup class='units'>%</sup>";
  ptr += "</p>";

  ptr += "</div>";
  ptr += "</div>";
  ptr += "</div>";
  ptr += "</div>";
  ptr += "</div>";
  ptr += "</body>";
  ptr += "</html>";
  return ptr;
}
```

OUTPUT

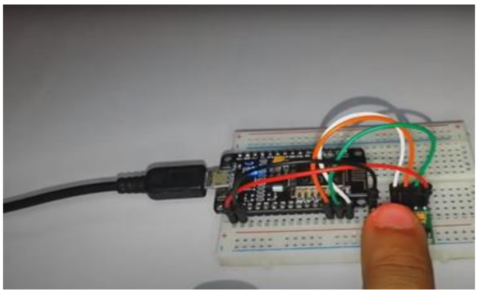

Figure 2.4: Output for Heart Rate Sensor

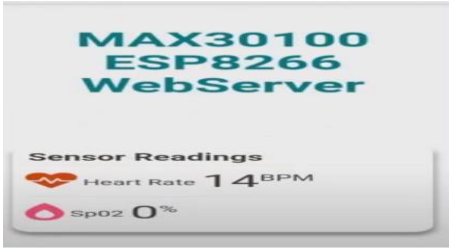

Figure 2.5: Output For Heart Rate Sensor in Cloud Data

RESULT

The DS18B20, when integrated with the ESP32, provides accurate heart rate readings with a simple one-wire connection. The result is a reliable and scalable solution for a range of applications, from IoT projects to industrial monitoring. This combination facilitates easy data communication, enabling real-time heart rate monitoring and remote connectivity

CONCLUSION AND FUTURE SCOPE

In conclusion, We analyzed a microcontroller-based heart rate monitoring system using IoT. Heart rate abnormalities are directly recognizable and communicatedto each patient advisor via the internet. Our system is simple, energy efficient and easy to understand. It helps to closely monitor patients 24 hours a day, 7 days a week throughour system. The project hardware has been installed and the initial results have been fully verified.