HEALTH MONITORING SYSTEM

J Component Project Report for the course

CSE3066 INTERNET OF THINGS

by

NOEL BENNY 20MIA1020
ANTONY GEORGE MATHEW K 20MIA1022
HIDESH MATHEW SEBI 20MIA1064

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

NOVEMBER 2022

Table of Contents

ABSTRACT

This project will discuss health services in the field of diagnostic tools and life support systems in the form of photoplethysmography. We designed a system capable of providing heart pumping activity information through a phenomenon known as photoelectric so the user's health condition. We collect the data pulse using a heart rate sensor (fingertip sensor). This system works to retrieve data from the bloodstream on the index finger and for the simulation the data is retrieved from the sensor through a test pin giving arbitrary values.

ACKNOWLEDGEMENT

We wish to express our sincere thanks and deep sense of gratitude to our project guide, Dr. Pradeep kumar and Dr. Arun Kumar for his consistent encouragement and valuable guidance offered to us in a pleasant manner throughout the course of the project work.

We also take this opportunity to thank all the faculty of the school for their support and their wisdom imparted to us throughout the course.

We thank our parents, family, and friends for bearing with us throughout the course of our project and for the opportunity they provided us in undergoing this course in such a prestigious institution.

INTRODUCTION

The diagnostic tool measuring heart rate is a medical device that will be used to help nurses and very useful to know the state of the patient's condition. The tool further helps in remote monitoring of a patient through a Telegram bot. The principle of this diagnostic tool is to count the number of heartbeats in minutes and also provide the oxygen level. The Telegram bot also updates the nurses regarding the state of the patients when not in vicinity. Then from the heart beat count will be determined whether the patient is in normal condition or not. Usually, people who have arrhythmia abnormalities of heart disease, specific values will deviate from the range of 60-100 BPM. Monitoring heart rate is very important for athletes, patients as it determines the condition of the heart (just heart rate). There are many ways to measure heart rate and the most precise one is using an Electrocardiography but the easier way to monitor the heart rate is to use a Heartbeat Sensor. It comes in different shapes and sizes and allows an instant way to measure the heartbeat.

Heartbeat Sensors are available in Wrist Watches (Smart Watches), Smart Phones, chest straps, etc. The heartbeat is measured in beats per minute or bpm, which indicates the number of times the heart is contracting or expanding in a minute.

1.1 OBJECTIVE

Ever since humans have existed, ‘health’ has undoubtedly been the single most important factor in one’s life. Knowing how our body parts or internal organs are functioning is vital to keep track of ourselves and remain healthy.

It is much easier and more convenient to check ourselves. Parameters like heart rate, pulse rate, temperature, BMI, etc. can be conveniently calculated. This becomes more relevant in this current scenario amidst the pandemic which we are facing now. Although getting tested is the best way to know if one is infected or not, testing our pulse and heart rate can give us hints of possible symptoms of the virus.

1.2 PRINCIPLE

The principle of pulse oximetry is based on the differential absorption characteristics of oxygenated and the deoxygenated hemoglobin. Oxygenated hemoglobin absorbs more infrared light and allows more red light to pass through. Whereas Deoxygenated hemoglobin absorbs more red light and allowing more infrared light to pass through.

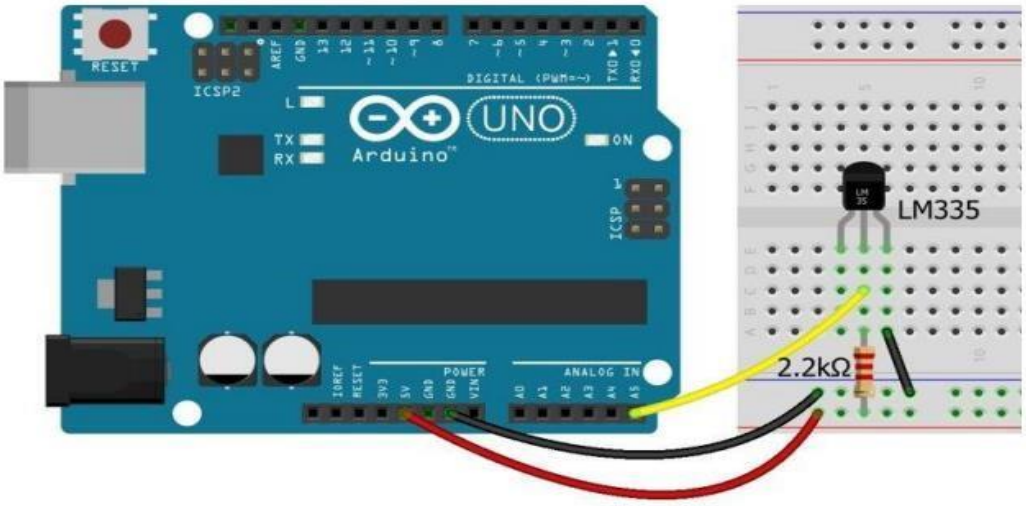
Each pulse oximeter sensor probe contains two light emitting diode one emitting red light and the other emitting near infrared light, it also has a photo-detector. The photo-detector measures the intensity of transmitted light at each wavelength. And using the differences in the reading the blood oxygen content is calculated. The probe is placed on a suitable part of the body, usually a fingertip or ear lobe

The basic principle of working of the temperature sensors is the voltage across the diode terminals. The change in temperature is sensed by the specially built Encardio-rite vibrating wire sensor and is converted to an electrical signal which is transmitted as a frequency to the read-out unit.

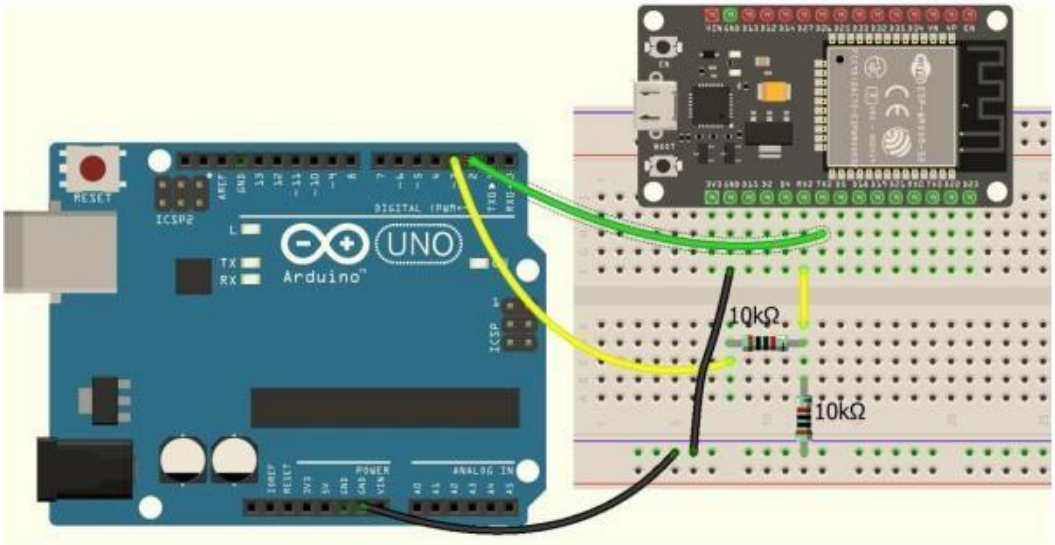
CHAPTER 2

CIRCUIT DESIGN

21. BLOCK DIAGRAM



(Fig 1 Interfacing with LM335)



(Fig 2. Interfacing with ESP32)

2.2 SOFTWARE SPECIFICATIONS

COMPONENTS:

- Pulse oximeter sensor (MAX30100)
- Temperature sensor (LM335)
- ESP32
- Software used: Proteus, Arduino IDE
- Telegram Bot

2.3 COMPONENTS SPECIFICATIONS:

Pulse oximeter sensor(MAX30100):

The MAX30100 is an integrated pulse oximetry and heartrate monitor sensor solution. It combines two LEDs, a photodetector, optimized optics, and low-noise analog signal processing to detect pulse oximetry and heart-rate signals.

The MAX30100 operates from 1.8V and 3.3V power supplies and can be powered down through software with negligible standby current, permitting the power supply to remain connected at all times.

Arduino UNO:

The Arduino Uno is a microcontroller board based on a removable, dual-inline-package (DIP) ATmega328 AVR microcontroller. It has 20 digital input/output pins (of which 6 can be used as PWM outputs and 6 can be used as analog inputs).

The Arduino Uno is programmed using the Arduino Software (IDE), our Integrated Development Environment common to all our boards and running both online and offline.

Temperature sensor (LM335):

The LM335 is a precision temperature sensor which can be easily calibrated. They operate as a 2terminal Zener and the breakdown voltage is directly proportional to the absolute temperature at 10mV/°K. At the temperature ranging from -40 degree Celsius to 100 degree

Celsius, LM335 can be used for each type of temperature sensing purposes. This device is available in different dimensions and in different sizes.

LM-335 can be used at several different places. Its applications include Heat Ventilation and Air Conditioning (HVAC), power supplies, battery management systems, home appliances, embedded systems etc.

ESP32:

ESP32 is a series of low-cost, low-power system on a chip microcontrollers with integrated Wi-Fi and dual-mode Bluetooth. The ESP32 series employs either a Tensilica Xtensa LX6 microprocessor in both dual-core and single-core variations, Xtensa LX7 dual-core microprocessor or a single-core RISC-V microprocessor and includes built-in antenna switches, RF balun, power amplifier, low-noise receive amplifier, filters, and power-management modules.

ESP32 can perform as a complete standalone system or as a slave device to a host MCU, reducing communication stack overhead on the main application processor. ESP32 can interface with other systems to provide Wi-Fi and Bluetooth functionality through its SPI / SDIO or I2C / UART interfaces.

Proteus:

The Proteus Design Suite is a proprietary software tool suite used primarily for electronic design automation. The software is used mainly by electronic design engineers and technicians to create schematics and electronic prints for manufacturing printed circuit boards.

Arduino IDE:

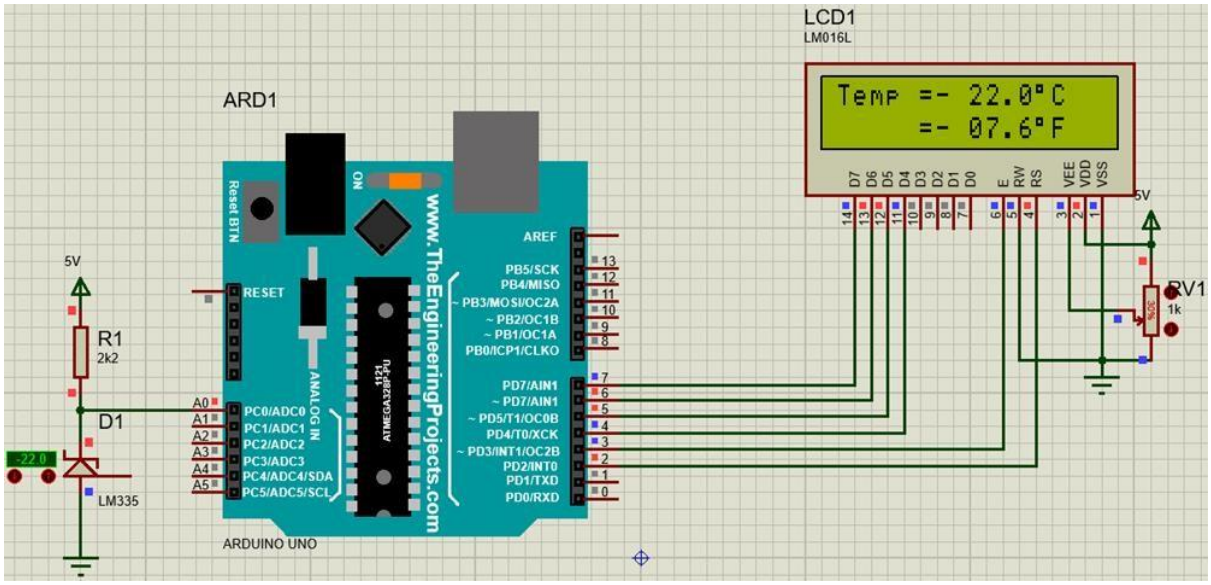
The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino hardware to upload programs and communicate with them

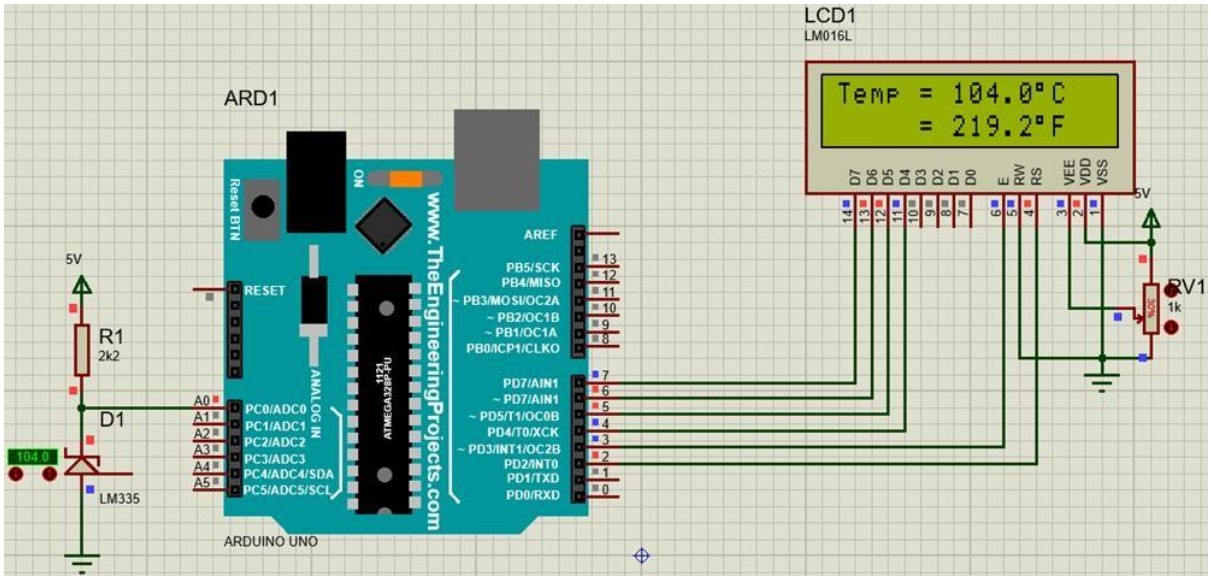
Telegram Bot:

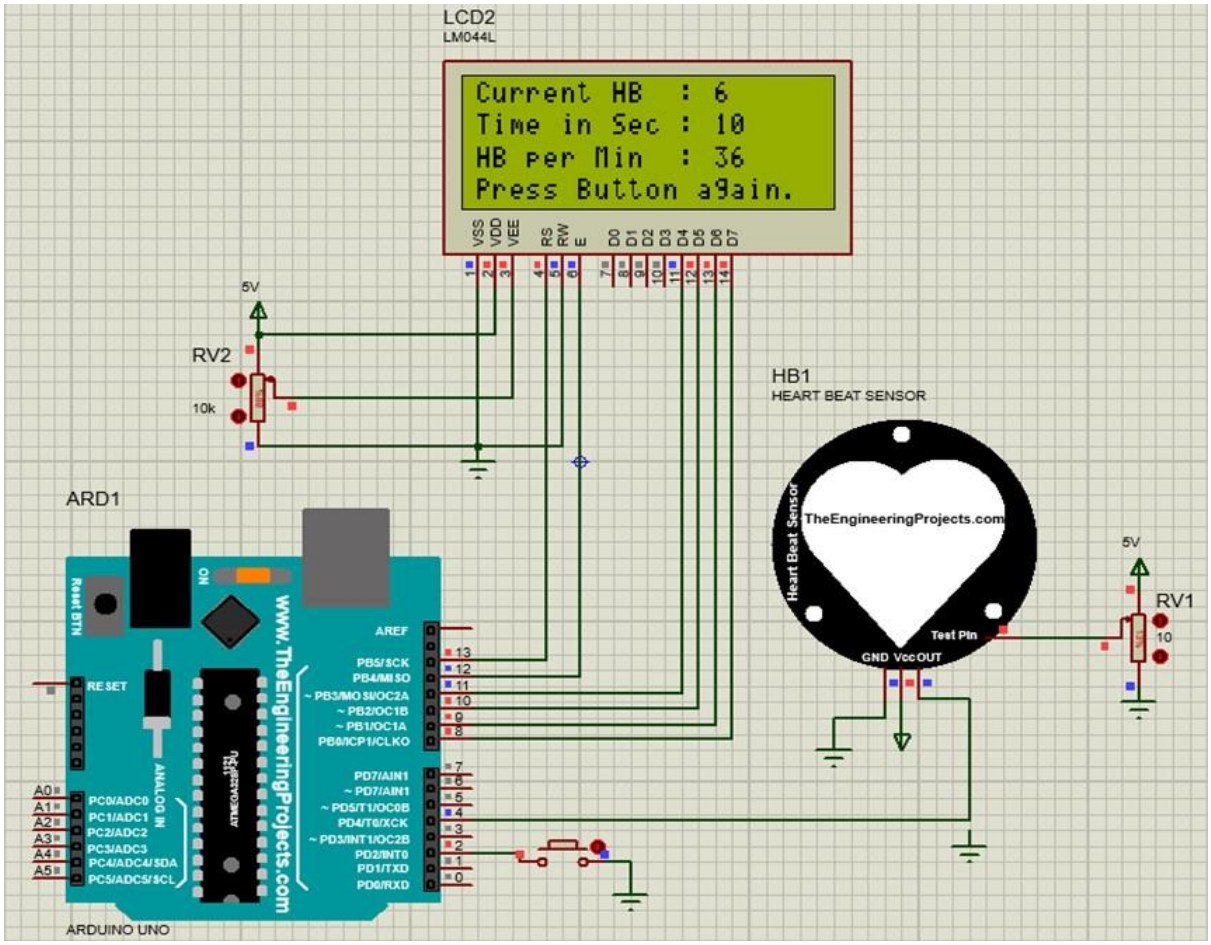
Telegram is a freeware, cross-platform, cloud-based instant messaging (IM) service. The service also provides end-to-end encrypted video calling,[11] VoIP, file sharing and several other features.

In our project, the ESP32 is used to send data to a telegram bot. Commands can be given to telegram bot which is received and handled by the ESP32. ESP32 sends the requested data to telegram bot API.(Code attached in 4.1 APPENDIX)

2.4 CIRCUIT DIAGRAM (PROTEUS)







Fig(a): Interfacing between Arduino and pulse oximeter sensor(MAX30100)

2.5 CIRCUIT DESIGN EXPLANATION:

Consider the circuit shown in Fig(a), let us divide it in 2 segments.

First consists of the input or the right side of the screen having a heartbeat and pulse sensor, potentiometer and the Vcc power terminal to operate the sensor.

Second segment consists of the arduino uno and LCD display.

By default, the VCC of Arduino is internally connected to +5V in this protues model.

There are 3 legs/pins of the heartbeat/pulse sensor .

Leftmost pin is a Ground

Middle pin is a Vcc

Rightmost pin à Signal.

Vcc is connected to a power terminal to operate the sensor. Giving Vcc power supply to the sensor so that it operates.

Next, we have in connection to the test pin the potentiometer.

So, if this project were on hardware, the input pulses or the input signal in real time can be varied by the heart beat so similarly, in this simulation we vary the voltage level with the help of this POT and observe the changes in the output of the heart rate sensor.

The voltmeter shows potential difference between Output pin and GND

If we even remove it, the circuit would work because the input to our microcontroller depends upon the POT values and the signal terminal of the heartbeat sensor.

Coming to the second segment.

As the power terminals provide the VCC for the whole circuit, the circuit is running and the analog signal is given to the A0 pin of the Arduino microcontroller and the signal is being plotted by waveform analyzer. The oscilloscope gives us the information about what type of analog signal we are getting as the wave of the heart which is sensed by the heart beat sensor.

The A0 to A5 all are analog ics , so we can give the input to any of the pin and can desired changes in the code.

The virtual terminal is basically simulating the RX TX pin of the Arduino.

RX= it receives data from outside the environment. (receiver)

TX= it sends data. (transmitter)

So, the virtual terminal is connected to TX which also is pin 1 of Arduino and the desired output is to be shown on the virtual screen.

This is our whole circuit providing the heartbeat and pulse readings as our output.

CHAPTER 3 SYSTEM IMPLEMENTATION AND ANALYSIS

(This section describes system implementation and results with inferences.)

3.1 CODE:

```
#include <Wire.h>
#include "MAX30100_PulseOximeter.h"

#define UPDATE_TIME   1000

// variables and pin defination
byte heart [8] = {0b00000, 0b01010, 0b11111, 0b11111, 0b11111,
0b01110,
0b00100, 0b00000};
uint32_t previous_update_time = 0;

PulseOximeter pulse;

void on_pulse_detected()
{
  Serial.println("Pulse Detected!");
}

void setup() {
Serial.begin(115200);
  Serial.print("Initializing Pulse Oximeter..");

  delay(3000);

  if (!pulse.begin()) {
    Serial.println("Sensor begin Failed");
for (;;);
  } else {
    Serial.println("Sensor begin Success");
  }
  //set current
  pulse.setIRLedCurrent(MAX30100_LED_CURR_7_6MA);

  // for the pulse detection
  pulse.setOnBeatDetectedCallback(on_pulse_detected);
}  void
loop() {
pulse.update(
);

  if (millis() - previous_update_time > UPDATE_TIME) {
previous_update_time = millis();

    // Display Result on Serial Monitor
    Serial.print("Heart ♥ Rate:");
Serial.print(pulse.getHeartRate());
    Serial.println("bpm");
```

```
    Serial.print(" SpO2 Level  :");
    Serial.print(pulse.getSpO2());
Serial.println("%");
    previous_update_time = millis();
  }
}
```

3.2 WORKING:

Heartbeat Sensor is an electronic device that is used to measure the heart rate. Monitoring body temperature, heart rate and blood pressure are the basic things that we do in order to keep us healthy. To measure the body temperature, we use thermometers and a sphygmomanometer to monitor the Arterial Pressure or Blood Pressure.

Heart Rate can be monitored in two ways: one way is to manually check the pulse either at wrists or neck and the other way is to use a Heartbeat Sensor.

In this project, we have designed a Heart Rate Monitor System using Arduino and Heartbeat Sensor. You can find the Principle of Heartbeat Sensor, working of the Heartbeat Sensor and Arduino based Heart Rate Monitoring System using a practical heartbeat Sensor.

For ease we have displayed the signals of the heat bit sensed on the oscilloscope. Waveforms generated are plotted as analog signals, giving us a brief about the health of that person. Arbitrary digital values are shown in the minitor to track the health of the patient.

This is a brief of what we have plotted on the virtual terminal in proteus simulationà

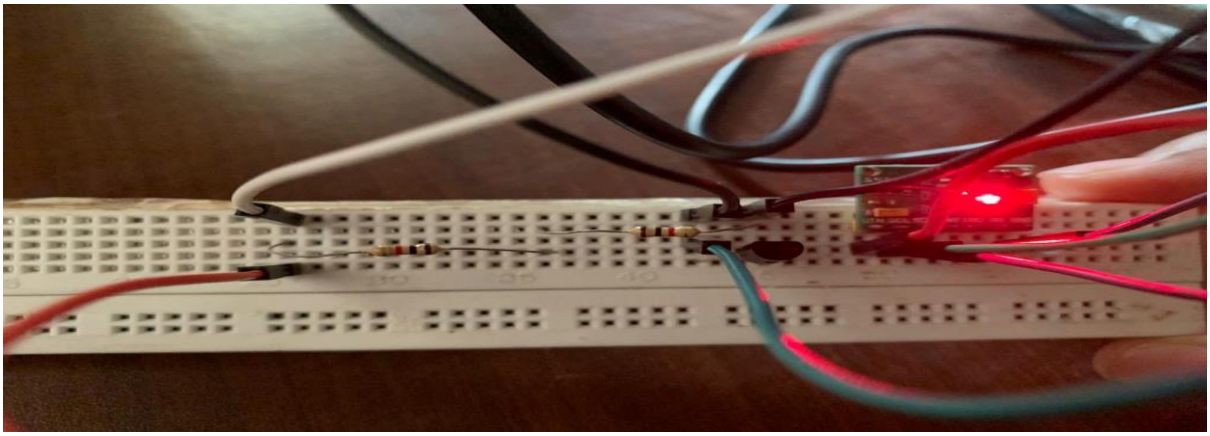
1) Displays user's live and changing BPM, Beats Per Minute, in Arduino's native Serial Monitor.

2) Print: " A Heartbeat Happened!" when a beat is detected, live.

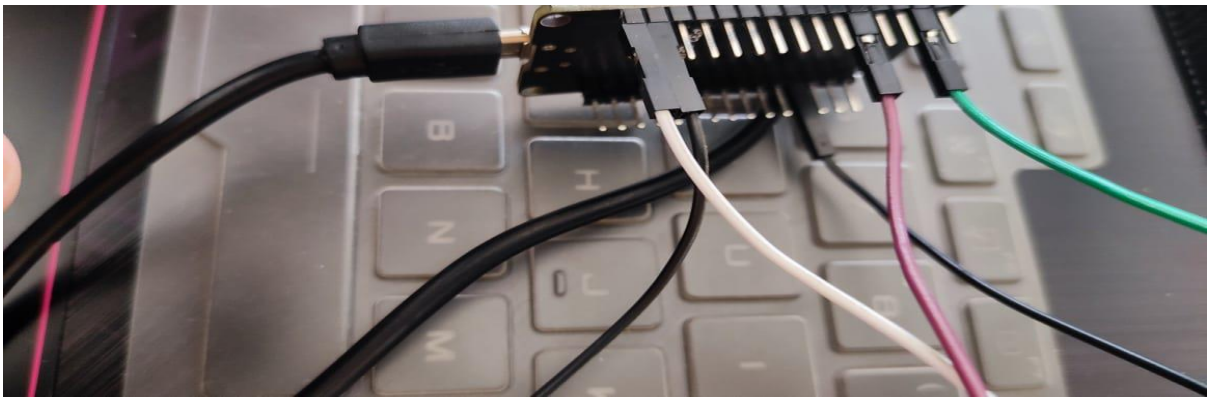
3) Learn about using a Pulse Sensor Library "Object".

4) Blinks LED on PIN 13 with user's Heartbeat.

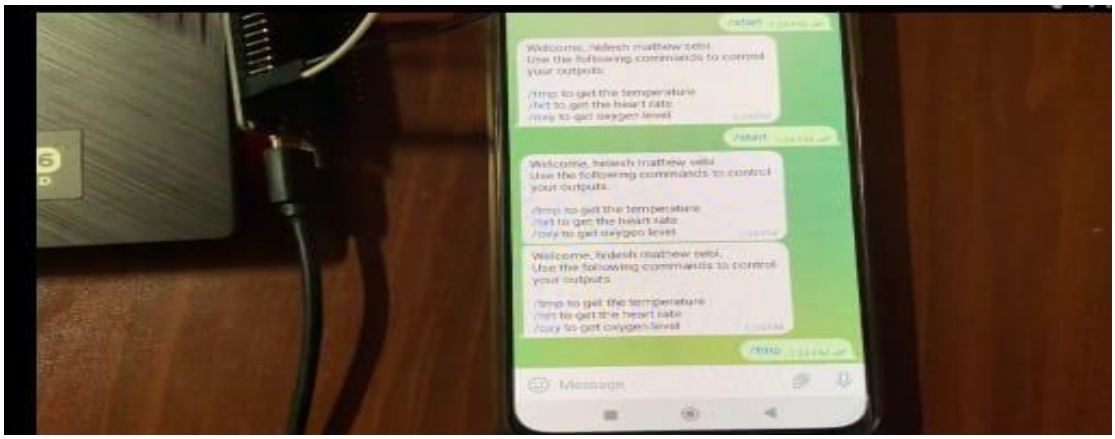
3.3 HARDWARE IMPLEMENTATION:



(Pulse oximeter sensor (MAX30100 in breadboard)



(EXP32)



(Telegram Bot)

3.4 LIMITATIONS:

The LM335 sensor and the MAX30100 couldn't be used at the same time due to the limitations of L2C interfacing.

4 . CONCLUSION:

Hence, we have designed a simple health monitoring system which calculates and displays the temperatur, heart rate and pulse of a person. Thus, since our Proteus simulation is running and the hardware simulation as demonstrated in the video is working, it proves the fact that our circuit is thus valid and can give an alternate device as a solution to monitoring their health on a daily basis.

Also, we have managed to send the temperatur, heart rate and pulse of a person through telegram bot but only one at a single time, which can be accessed worldwide provided there is a wifi connection.

5.APPLICATIONS:

- A simple project involving Arduino UNO and Heartbeat Sensor Module is designed here which can calculate the heart rate of a person.
- This project can be used as an inexpensive alternative to Smart Watches and other expensive Heart Rate Monitors.
- In this covid scenario outside, the physiological parameters related to heart are important to ponder upon and they need to be checked on an everyday basis. So, our project is easy to operate by anyone and can give the readings of BPM and pulse of a person.

5 REFERENCES:

- https://www.electronicshub.org/heartbeat-sensor-using-arduino-heart-ratemonitor/#:~:text=First%2C%20in%20order%20to%20display,LCD%20(contrast%20 adjust%20pin).
- https://www.engineersgarage.com/electronic-projects/arduino-based-heartbeat-andbody-temperature-monitoring-iot-device/

APPENDIX:

Arduino code -

```
#include <LiquidCrystal.h>//For LCD display
#include <Wire.h>
#include "MAX30100_PulseOximeter.h"
#include <Arduino.h>

#include "Talkie.h"

Talkie voice;

#define UPDATE_TIME   10000

// variables and pin definition
byte heart [8] = {0b00000, 0b01010, 0b11111, 0b11111, 0b11111,
0b01110,
0b00100, 0b00000};
uint32_t previous_update_time = 0;
const int rs = 12, en = 10, d4 = 4, d5 = 5, d6 = 6, d7 = 7;   /* pins
used for LCD display      */

LiquidCrystal lcd(rs, en, d4, d5, d6, d7); //Declare a LiquidCrystal
object


//LPC codes
const uint8_t spZERO[]  PROGMEM =
{0x69,0xFB,0x59,0xDD,0x51,0xD5,0xD7,0xB5,0x6F,0x0A,0x78,0xC0,0x52,0x01
,0x0
F,0x50,0xAC,0xF6,0xA8,0x16,0x15,0xF2,0x7B,0xEA,0x19,0x47,0xD0,0x64,0xE
B,0x
AD,0x76,0xB5,0xEB,0xD1,0x96,0x24,0x6E,0x62,0x6D,0x5B,0x1F,0x0A,0xA7,0x
B9,0
xC5,0xAB,0xFD,0x1A,0x62,0xF0,0xF0,0xE2,0x6C,0x73,0x1C,0x73,0x52,0x1D,0
x19,
0x94,0x6F,0xCE,0x7D,0xED,0x6B,0xD9,0x82,0xDC,0x48,0xC7,0x2E,0x71,0x8B,
0xBB
,0xDF,0xFF,0x1F};
const uint8_t spONE[]   PROGMEM =
{0x66,0x4E,0xA8,0x7A,0x8D,0xED,0xC4,0xB5,0xCD,0x89,0xD4,0xBC,0xA2,0xDB
,0xD
1,0x27,0xBE,0x33,0x4C,0xD9,0x4F,0x9B,0x4D,0x57,0x8A,0x76,0xBE,0xF5,0xA
9,0x
AA,0x2E,0x4F,0xD5,0xCD,0xB7,0xD9,0x43,0x5B,0x87,0x13,0x4C,0x0D,0xA7,0x
75,0
xAB,0x7B,0x3E,0xE3,0x19,0x6F,0x7F,0xA7,0xA7,0xF9,0xD0,0x30,0x5B,0x1D,0
x9E,
0x9A,0x34,0x44,0xBC,0xB6,0x7D,0xFE,0x1F};
const uint8_t spTWO[]   PROGMEM =
{0x06,0xB8,0x59,0x34,0x00,0x27,0xD6,0x38,0x60,0x58,0xD3,0x91,0x55,0x2D
,0xA
A,0x65,0x9D,0x4F,0xD1,0xB8,0x39,0x17,0x67,0xBF,0xC5,0xAE,0x5A,0x1D,0xB
```

```
5,0x
7A,0x06,0xF6,0xA9,0x7D,0x9D,0xD2,0x6C,0x55,0xA5,0x26,0x75,0xC9,0x9B,0x
DF,0 xFC,0x6E,0x0E,0x63,0x3A,0x34,0x70,0xAF,0x3E,0xFF,0x1F}; const
uint8_t spTHREE[] PROGMEM =
{0x0C,0xE8,0x2E,0x94,0x01,0x4D,0xBA,0x4A,0x40,0x03,0x16,0x68,0x69,0x36
,0x1
C,0xE9,0xBA,0xB8,0xE5,0x39,0x70,0x72,0x84,0xDB,0x51,0xA4,0xA8,0x4E,0xA
3,0x
C9,0x77,0xB1,0xCA,0xD6,0x52,0xA8,0x71,0xED,0x2A,0x7B,0x4B,0xA6,0xE0,0x
37,0
xB7,0x5A,0xDD,0x48,0x8E,0x94,0xF1,0x64,0xCE,0x6D,0x19,0x55,0x91,0xBC,0
x6E,
0xD7,0xAD,0x1E,0xF5,0xAA,0x77,0x7A,0xC6,0x70,0x22,0xCD,0xC7,0xF9,0x89,
0xCF ,0xFF,0x03};
const uint8_t spFOUR[]  PROGMEM =
{0x08,0x68,0x21,0x0D,0x03,0x04,0x28,0xCE,0x92,0x03,0x23,0x4A,0xCA,0xA6
,0x1
C,0xDA,0xAD,0xB4,0x70,0xED,0x19,0x64,0xB7,0xD3,0x91,0x45,0x51,0x35,0x8
9,0x
EA,0x66,0xDE,0xEA,0xE0,0xAB,0xD3,0x29,0x4F,0x1F,0xFA,0x52,0xF6,0x90,0x
52,0
x3B,0x25,0x7F,0xDD,0xCB,0x9D,0x72,0x72,0x8C,0x79,0xCB,0x6F,0xFA,0xD2,0
x10,
0x9E,0xB4,0x2C,0xE1,0x4F,0x25,0x70,0x3A,0xDC,0xBA,0x2F,0x6F,0xC1,0x75,
0xCB
,0xF2,0xFF};
const uint8_t spFIVE[]  PROGMEM =
{0x08,0x68,0x4E,0x9D,0x02,0x1C,0x60,0xC0,0x8C,0x69,0x12,0xB0,0xC0,0x28
,0xA
B,0x8C,0x9C,0xC0,0x2D,0xBB,0x38,0x79,0x31,0x15,0xA3,0xB6,0xE4,0x16,0xB
7,0x
DC,0xF5,0x6E,0x57,0xDF,0x54,0x5B,0x85,0xBE,0xD9,0xE3,0x5C,0xC6,0xD6,0x
6D,0
xB1,0xA5,0xBF,0x99,0x5B,0x3B,0x5A,0x30,0x09,0xAF,0x2F,0xED,0xEC,0x31,0
xC4,
0x5C,0xBE,0xD6,0x33,0xDD,0xAD,0x88,0x87,0xE2,0xD2,0xF2,0xF4,0xE0,0x16,
0x2A
,0xB2,0xE3,0x63,0x1F,0xF9,0xF0,0xE7,0xFF,0x01};
const uint8_t spSIX[]   PROGMEM =
{0x04,0xF8,0xAD,0x4C,0x02,0x16,0xB0,0x80,0x06,0x56,0x35,0x5D,0xA8,0x2A
,0x6
D,0xB9,0xCD,0x69,0xBB,0x2B,0x55,0xB5,0x2D,0xB7,0xDB,0xFD,0x9C,0x0D,0xD
8,0x
32,0x8A,0x7B,0xBC,0x02,0x00,0x03,0x0C,0xB1,0x2E,0x80,0xDF,0xD2,0x35,0x
20,0 x01,0x0E,0x60,0xE0,0xFF,0x01}; const uint8_t spSEVEN[] PROGMEM =
{0x0C,0xF8,0x5E,0x4C,0x01,0xBF,0x95,0x7B,0xC0,0x02,0x16,0xB0,0xC0,0xC8
,0xB
A,0x36,0x4D,0xB7,0x27,0x37,0xBB,0xC5,0x29,0xBA,0x71,0x6D,0xB7,0xB5,0xA
B,0x
A8,0xCE,0xBD,0xD4,0xDE,0xA6,0xB2,0x5A,0xB1,0x34,0x6A,0x1D,0xA7,0x35,0x
37,0
xE5,0x5A,0xAE,0x6B,0xEE,0xD2,0xB6,0x26,0x4C,0x37,0xF5,0x4D,0xB9,0x9A,0
x34,
0x39,0xB7,0xC6,0xE1,0x1E,0x81,0xD8,0xA2,0xEC,0xE6,0xC7,0x7F,0xFE,0xFB,
0x7F
};
```

```
PulseOximeter pulse;
void on_pulse_detected(){Serial.println("Pulse Detected!");}
 void audio(int
val){   int temp1
= val%10;   int
temp2 = val/10;
   if (temp2 == 0){voice.say(spZERO);}
else if (temp2 == 1){voice.say(spONE);}
else if (temp2 ==
2){voice.say(spTWENTY);}   else if (temp2
== 3){voice.say(spTHIRTY);}   else if
(temp2 == 4){voice.say(spFOURTY);}   else
if (temp2 == 5){voice.say(spFIFTY);}
else if (temp2 == 6){voice.say(spSIXTY);}
else if (temp2 ==
7){voice.say(spSEVENTY);}   else if
(temp2 == 8){voice.say(spEIGHTY);}   else
if (temp2 == 9){voice.say(spNINETY);}
      if (temp1 ==
1){voice.say(spONE);}   else if (temp1
== 2){voice.say(spTWO);}    else if
(temp1 == 3){voice.say(spTHREE);}
else if (temp1 ==
4){voice.say(spFOUR);}   else if (temp1
== 5){voice.say(spFIVE);}   else if
(temp1 == 6){voice.say(spSIX);}   else
if (temp1 == 7){voice.say(spSEVEN);}
else if (temp1 ==
8){voice.say(spEIGHT);}   else if
(temp1 == 9){voice.say(spNINE);}
 }  void

setup() {
  lcd.begin(16, 2);//Set up the LCD's number of columns and rows

  //Serial baud rates
  Serial.begin(115200);

  //set current in MAX30100
  pulse.setIRLedCurrent(MAX30100_LED_CURR_7_6MA);

  // for the pulse detection
  pulse.setOnBeatDetectedCallback(on_pulse_detected);
}
void loop(){
    //set current in MAX30100
  pulse.setIRLedCurrent(MAX30100_LED_CURR_7_6MA);
    //Temperature
    int rawvoltage= analogRead(A0);     float
millivolts= (rawvoltage/1024.0) * 5000;
float celsius= (millivolts/10)-273.15;
```

```
    String tmp="Temperature:";
tmp+=String(celsius);
tmp+="C";
    Serial.println(tmp);
lcd.clear();
lcd.print(tmp);
audio(int(celsius));
delay(2000);

    //pulse
    //float heart=pulse.getHeartRate();
float heart=73.8;
String hrt="Heart-Rate:";
hrt+=String(heart);
hrt+="bpm";
Serial.println(hrt);
lcd.clear();
lcd.print(hrt);
audio(int(heart));
delay(2000);

    //oxygen-level
    //float
oxygen=pulse.getSpO2();
float oxygen=96.5;     String
sp="SpO2-Level:";
sp+=String(oxygen);
sp+="%";
    Serial.println(sp);
lcd.clear();
lcd.print(sp);
audio(int(oxygen));
delay(2000);

    pulse.update();
 }
```

ESP 32 Code -

```
#include <WiFi.h>
#include <WiFiClientSecure.h>
#include <UniversalTelegramBot.h>   // Universal Telegram Bot Library
written by Brian Lough: https://github.com/witnessmenow/Universal-
Arduino-
Telegram-Bot
#include <ArduinoJson.h>

// Replace with your network
credentials char* ssid =
"YOUR_SSID"; char* password =
"WiFi_Password";

// Initialize Telegram BOT
```

```
#define BOTtoken "1234xyz"  // your Bot Token (Get from
Botfather)
// Use @myidbot to find out the chat ID of an individual or a group
// Also note that you need to click "start" on a bot before it can
// message you
#define CHAT_ID "YOUR_Telegram_Chat-ID"

WiFiClientSecure client;
UniversalTelegramBot bot(BOTtoken, client);

// Checks for new messages every 1 second.
 int botRequestDelay = 1000; unsigned long lastTimeBotRan

//Variable to store temperature,heart rate and oxygen level
String tmp;
String hrt;
String oxy;

// Handle what happens when you receive new messages
void handleNewMessages(int numNewMessages) {
Serial.println("handleNewMessages");
  Serial.println(String(numNewMessages));
   for (int i=0; i<numNewMessages;
i++) {
    // Chat id of the requester
    String chat_id =
String(bot.messages[i].chat_id);     if (chat_id !=
CHAT_ID){
      bot.sendMessage(chat_id, "Unauthorized user", "");
continue;
    }

    // Print the received message
    String text = bot.messages[i].text;
    Serial.println(text);

String from_name = bot.messages[i].from_name;

    if (text == "/start") {
      String welcome = "Welcome, " + from_name + ".\n";
welcome += "Use the following commands to control your
outputs.\n\n";
      welcome += "/tmp to get the temperature \n";
welcome += "/hrt to get the heart rate \n";
welcome += "/oxy to get oxygen level \n";
bot.sendMessage(chat_id, welcome, "");

}
    if (text == "/tmp"){
String tmp="Temperature:";
      float temp = ((analogRead(A0) * resolution) * 100)-
283.15;       tmp+=String(temp);       tmp+="C";
      bot.sendMessage(chat_id,tmp, "");
```

```
    }      if (text == "/hrt") {
hrt="Sensor not connected";
bot.sendMessage(chat_id,hrt, "");
    }      if (text == "/oxy") {
oxy="Sensor not connected";
bot.sendMessage(chat_id,oxy, "");
}
  }
}
void setup() {
Serial.begin(9600);

  // Connect to Wi-Fi
  WiFi.begin(ssid, password);


// Add root certificate for api.telegram.org
client.setCACert(TELEGRAM_CERTIFICATE_ROOT);

  //Checking if WiFi connected
while (WiFi.status() != WL_CONNECTED)
{     delay(1000);
    Serial.println("Connecting to WiFi..");
 }

// Print ESP32 Local IP Address
  Serial.println(WiFi.localIP());
}  void
loop() {
  if (millis() > lastTimeBotRan + botRequestDelay){
    int numNewMessages = bot.getUpdates(bot.last_message_received +
1);

    while(numNewMessages) {
      Serial.println("got response");
handleNewMessages(numNewMessages);
      numNewMessages = bot.getUpdates(bot.last_message_received + 1);
    }
    lastTimeBotRan = millis();
  }
 }
```