

```
#include "BluetoothSerial.h"
```

```
#include <SPI.h>
```

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

```
#include <Adafruit_GFX.h>
```

```
#include <Adafruit_SSD1306.h>
```

```
//-----Configure OLED screen size in pixels
```

```
#define SCREEN_WIDTH 128 //--> OLED display width, in pixels
```

```
#define SCREEN_HEIGHT 64 //--> OLED display height, in pixels
```

```
//-----
```

```
#define OLED_RESET -1 // Reset pin # (or -1 if sharing Arduino reset pin)
```

```
#define LED_BUILTIN 2 //pin with LED to turn on when BT connected
```

```
#define audio 18
```

```
#define buttonPin 15
```

```
#define SENSOR A0 // Set the A0 as SENSOR
```

```
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire,-1);
```

BluetoothSerial ESP\_BT; // Object for Bluetooth

// global vars

boolean BT\_cnx = false;

//-----

unsigned long previousMillisGetHR = 0; //--> will store the last time Millis (to get Heartbeat) was updated.

unsigned long previousMillisResultHR = 0; //--> will store the last time Millis (to get BPM) was updated.

const long intervalGetHR = 20; //--> Interval for reading heart rate (Heartbeat) = 10ms.

const long intervalResultHR = 10000; //--> The reading interval for the result of the Heart Rate calculation is in 10 seconds.

int PulseSensorSignal; //--> Variable to accommodate the signal value from the sensor

const int PulseSensorHRWire = A0; //--> PulseSensor connected to ANALOG PIN 0 (A0 / ADC 0).

const int

LED\_A1 = 19; //--> LED to detect when the heart is beating. The LED is connected to PIN A1 on the Arduino UNO.

//const int audio = 18;

//const int buttonPin = 17;

int buttonState;

int audio\_state = LOW;

int UpperThreshold = 2250; //--> Determine which Signal to "count as a beat", and which to ignore.

int LowerThreshold = 1778;

int cntHB = 0; //--> Variable for counting the number of heartbeats.

boolean ThresholdStat = true; //--> Variable for triggers in calculating heartbeats.

int BPMval = 0; //--> Variable to hold the result of heartbeats calculation.

int x=0; //--> Variable axis x graph values to display on OLED

int y=0; //--> Variable axis y graph values to display on OLED

int lastx=0; //--> The graph's last x axis variable value to display on the OLED

int lasty=0; //--> The graph's last y axis variable value to display on the OLED

//-----'LogoChannel', 128x64px

// A complete demo of 128x64 OLED can be seen in this video:

<https://www.youtube.com/watch?v=CuusKoCBoUE>

const unsigned char wifisymbol [] PROGMEM = {

0x00, 0x1e, 0x80, 0x01, 0x80, 0x00, 0xc0, 0x38, 0x40, 0x0c, 0x40, 0x62, 0x40, 0x32, 0x40, 0xd2, 0x40,  
0xd2, 0x40, 0x32, 0xc0, 0x64, 0x80, 0x0c, 0x80, 0x38, 0x80, 0x01, 0x00, 0x01, 0x00, 0x1e

};

const unsigned char LogoChannel [] PROGMEM = {

0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0xe0, 0x00, 0x00, 0x00, 0x00, 0x07, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0xc0, 0x00, 0x00, 0x00, 0x00, 0x03, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0x87, 0xff, 0xff, 0xff, 0xe1, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0x8f, 0xff, 0xff, 0xff, 0xf1, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0x8f, 0xff, 0xf8, 0x7f, 0xff, 0xf1, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0x89, 0x00, 0x13, 0x00, 0xff, 0xf1, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0x89, 0x00, 0x17, 0x00, 0x7f, 0xf1, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0x89, 0x00, 0x13, 0x00, 0x3f, 0xf1, 0xff, 0xff, 0xff, 0xff,

[illegible]

0xff, 0xff, 0xff, 0xff, 0xfc, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x1f, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xfc, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x3f, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xfe, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x3f, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x7f, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0x80, 0x00, 0x00, 0x00, 0x00, 0x01, 0xff, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0xe0, 0x00, 0x00, 0x00, 0x00, 0x07, 0xff, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0xb2, 0x08, 0x13, 0x7e, 0x10, 0x41, 0xff, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0x92, 0x08, 0x12, 0x7c, 0x10, 0x40, 0xff, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0x93, 0x99, 0xf2, 0x7c, 0x9c, 0xcc, 0xff, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0x93, 0x99, 0xf2, 0x7c, 0xfc, 0xcc, 0xff, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0x93, 0x98, 0x30, 0x7c, 0x1c, 0xcc, 0xff, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0x93, 0x98, 0x30, 0x7f, 0x9c, 0xc0, 0xff, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0x93, 0x99, 0xf2, 0x7c, 0x9c, 0xc9, 0xff, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0x83, 0x98, 0x12, 0x7c, 0x1c, 0xc9, 0xff, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0x87, 0x98, 0x12, 0x7c, 0x1c, 0xcc, 0xff, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,  
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff

};

//-----

//-----'Heart\_Icon', 16x16px

// I drew this heart icon at : <http://dotmatrixtool.com/>

```
const unsigned char Heart_Icon [] PROGMEM = {  
    0x00, 0x00, 0x18, 0x30, 0x3c, 0x78, 0x7e, 0xfc, 0xff, 0xfe, 0xff, 0xfe, 0xee, 0xee, 0xd5, 0x56,  
    0x7b, 0xbc, 0x3f, 0xf8, 0x1f, 0xf0, 0x0f, 0xe0, 0x07, 0xc0, 0x03, 0x80, 0x01, 0x00, 0x00, 0x00  
};  
//-----
```

```
void callback(esp_spp_cb_event_t event, esp_spp_cb_param_t *param){  
    if(event == ESP_SPP_SRV_OPEN_EVT){  
        Serial.println("Client Connected");  
        digitalWrite(LED_BUILTIN, HIGH);  
        BT_cnx = true;  
    }  
}
```

```
if(event == ESP_SPP_CLOSE_EVT ){  
    Serial.println("Client disconnected");  
    digitalWrite(LED_BUILTIN, LOW);  
    BT_cnx = false;  
    // ESP.restart();  
}  
}
```

```
void setup() {  
  
    digitalWrite(audio,LOW);  
    // initialize digital pin 2 as an output.  
    pinMode(LED_BUILTIN, OUTPUT);  
}
```

```

pinMode(LED_A1,OUTPUT); //--> Set LED_3 PIN as Output.

pinMode(audio,OUTPUT);

pinMode(buttonPin,INPUT_PULLUP);

// initialize the serial communication:
Serial.begin(9600);

//-----SSD1306_SWITCHCAPVCC = generate display voltage from 3.3V
internally

// Address 0x3C for 128x32 and Address 0x3D for 128x64.

// But on my 128x64 module the 0x3D address doesn't work. What works is the 0x3C address.

// So please try which address works on your module.

if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
    Serial.println(F("SSD1306 allocation failed"));
    for(;;); //--> Don't proceed, loop forever
}

//-----

//-----Show initial display buffer contents on the screen

// the library initializes this with an Adafruit splash screen.

display.display();

delay(1000);

//-----

//-----Display Bitmap Images

display.clearDisplay(); //--> for Clearing the display

display.drawBitmap(0, 0, LogoChannel, 128, 64, WHITE); //--> display.drawBitmap(x position, y
position, bitmap data, bitmap width, bitmap height, color)

display.display();

delay(2000); // Pause for 3 seconds

//-----

```

```
//-----Displays BPM value reading information
```

```
display.clearDisplay();
```

```
display.setTextSize(1.9);
```

```
display.setTextColor(WHITE);
```

```
display.setCursor(45, 5); //--> (x position, y position)
```

```
display.print("WELCOME");
```

```
display.setCursor(30, 25); //--> (x position, y position)
```

```
display.print("THIS IS OUR");
```

```
display.setCursor(5, 45); //--> (x position, y position)
```

```
display.print("PROJECT 2B -- ECG");
```

```
display.display();
```

```
delay(3000);
```

```
display.clearDisplay();
```

```
display.setCursor(16, 5); //--> (x position, y position)
```

```
display.print("PLEASE WAIT 10S");
```

```
display.setCursor(23, 25); //--> (x position, y position)
```

```
display.print("FOR THE HEART");
```



```
display.setCursor(26, 45); //--> (x position, y position)
display.print("RATE VALUE ^^");
```

```
// display.setCursor(0, 42); //--> (x position, y position)
// display.print(" the Heart Rate value");
```

```
display.display();
delay(3000);
```

```
display.clearDisplay();
```

```
display.setTextSize(1);
display.setTextColor(WHITE);
display.setCursor(0, 25); //--> (x position, y position)
display.print(" BLUETOOTH CONNECTING");
display.display();
delay(3000);
```

```
//-----
```

```
//-----Displays the initial display of BPM value
display.clearDisplay(); //--> for Clearing the display
display.setTextSize(2);
display.drawBitmap(0, 50, Heart_Icon, 16, 16, WHITE); //--> display.drawBitmap(x position, y position,
bitmap data, bitmap width, bitmap height, color)

display.drawLine(0, 46, 127, 46, WHITE); //--> drawLine(x1, y1, x2, y2, color)
```

```
display.setTextSize(2);  
display.setTextColor(WHITE);  
display.setCursor(20, 51); //--> (x position, y position)  
display.print(": 0 BPM");  
display.display();  
//-----
```

```
Serial.println();  
Serial.println("Please wait 10 seconds to get the BPM Value");
```

```
//FOR BLUETOOTH-----
```

```
Serial.println(); // blank line in serial ...  
pinMode(41, INPUT); // Setup for leads off detection LO +  
pinMode(40, INPUT); // Setup for leads off detection LO -  
// initialize the serial BT communication:  
ESP_BT.register_callback(callback);  
if(!ESP_BT.begin("ESP32_ECG")){  
    Serial.println("An error occurred initializing Bluetooth");  
}else{  
    Serial.println("Bluetooth initialized... Bluetooth Device is Ready to Pair...");  
}  
}
```

```
void loop() {
```

```
    GetHeartRate(); //--> Calling the GetHeartRate() subroutine
```

```
if((digitalRead(40) == 1) || (digitalRead(41) == 1)){  
    Serial.println('!');  
    ESP_BT.println('!');  
}
```

```
else{  
    // send the value of analog input 0 to serial:  
    Serial.println(analogRead(A0));  
    //Do the same for bluetooth
```

```
if(BT_cnx){  
    display.setTextSize(1);  
    display.setTextColor(WHITE);  
    display.fillRect(0, 0, 128, 8, BLACK);  
    display.setCursor(0, 0);  
    display.print("Bluetooth connected");  
    display.display();
```

ESP\_BT.print('E'); //make the app Bluetooth Graphics  
([https://play.google.com/store/apps/details?id=com.emrctn.BluetoothGraphics&hl=en\\_US](https://play.google.com/store/apps/details?id=com.emrctn.BluetoothGraphics&hl=en_US)) work (as  
specified by the app)

```
ESP_BT.println(analogRead(A0));
```

```
}
```

```
else {  
    display.setTextSize(1);  
    display.setTextColor(WHITE);
```

```

display.fillRect(0, 0, 128, 8, BLACK);

display.setCursor(0, 0);

display.print("Bluetooth connecting");


display.display();
}

}

//Wait a little to keep serial data from saturating
delay(1);
}

void GetHeartRate() {
    //-----Process of reading heart rate.
    unsigned long currentMillisGetHR = millis();

    if (currentMillisGetHR - previousMillisGetHR >= intervalGetHR) {
        previousMillisGetHR = currentMillisGetHR;

        PulseSensorSignal = analogRead(PulseSensorHRWire); //--> holds the incoming raw data. Signal value
        can range from 0-1024

        if (PulseSensorSignal > UpperThreshold && ThresholdStat == true) {
            cntHB++;

            ThresholdStat = false;

            digitalWrite(LED_A1,HIGH);

            digitalWrite(audio,HIGH);
        }
    }
}

```

```
if (PulseSensorSignal < UpperThreshold) {  
    ThresholdStat = true;  
    digitalWrite(LED_A1,LOW);  
    digitalWrite(audio,LOW);  
}
```

```
DrawGraph(); //--> Calling the DrawGraph() subroutine  
}
```

```
//-----
```

```
//-----The process for getting the BPM value.
```

```
unsigned long currentMillisResultHR = millis();
```

```
if (currentMillisResultHR - previousMillisResultHR >= intervalResultHR) {  
    previousMillisResultHR = currentMillisResultHR;
```

```
    BPMval = cntHB * 6; //--> The taken heart rate is for 10 seconds. So to get the BPM value, the total  
    heart rate in 10 seconds x 6.
```

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

```
    Serial.println(BPMval);
```

```
display.fillRect(20, 48, 108, 18, BLACK);
```

```
display.drawBitmap(0, 50, Heart_Icon, 16, 16, WHITE); //--> display.drawBitmap(x position, y position,  
bitmap data, bitmap width, bitmap height, color)
```

```
display.drawLine(0, 46, 127, 46, WHITE); //--> drawLine(x1, y1, x2, y2, color)
```

```

display.setTextSize(2);
display.setTextColor(WHITE);
display.setCursor(20, 51); //--> (x position, y position)
display.print(": ");
display.print(BPMval);
display.print(" BPM");
display.display();

cntHB = 0;
}
//-----
}
//-----

//-----Subroutines for drawing or displaying
heart rate graphic signals
void DrawGraph() {
    //-----Condition to reset the graphic display if it fills the width of the OLED
    screen
    if (x > 127) {
        display.fillRect(0, 0, 128, 44, BLACK);
        x = 0;
        lastx = 0;
    }
    //-----

    //-----Process signal data to be displayed on OLED in graphic form
    int ySignal = PulseSensorSignal;

```

```
if (ySignal > 4095) ySignal = 4095;
```

```
if (ySignal < 1485) ySignal = 1485;
```

```
int ySignalMap = map(ySignal, 1485, 4095, 3, 46); //--> The y-axis used on OLEDs is from 0 to 40
```

```
y = 44 - ySignalMap;
```

```
//-----
```

```
//-----Displays the heart rate graph
```

```
display.drawLine(lastx,lasty,x,y,WHITE);
```

```
display.display();
```

```
//-----
```

```
lastx = x;
```

```
lasty = y;
```

```
x++;
```

```
}
```