Heart Beat Monitoring Using Arduino

A Project Report

Submitted as part of the Course

Distributed Real-Time Systems

(CSE2021)

Winter Semester: 2020-2021

Course Faculty: Dr. Venkatraman.S

Submitted By

PRIYADARSHINI.S 19BPS1054

AKSHARA MAGESH 19BPS1061

ABSTRACT:

Heart is the most wanted part of human being to live in a world, at the same time the heart rate analysis is increased in medical field and the heart analysis is important parameter of human health.

The various heart rate analysis method is available in medical field like ECG and pulse sensing system this pulse analysis is depends on the blood force of heart artery. This artery is closed to the skin in that reason the pulse is identified easily.

Heartbeat is measured in beats per minute or bpm, which indicates the number of times the heart is contracting or expanding in a minute.

The proposed system analysis the pulse rate in the way of fingertip using Arduino controller, and it’s based on photo phelthysmography principle.

According to this principle, the changes in the volume of blood in an organ is measured by the changes in the intensity of the light passing through that organ.

Usually, the source of light in a heartbeat sensor would be an IR LED and the detector would be any Photo Detector like a Photo Diode, an LDR (Light Dependent Resistor) or a Photo Transistor.

INTRODUCTION:

In recent year the mortality rate is increased through the heart attack is occurred in the human being, so the heart rate analysis is very important to reduce the mortality rate in the world.

The heart rate is monitoring with help of the real time sensors like heart beat sensor which is used to monitor human heart every few seconds, and the sensed data is send to the controller if any variation is occur in the data the alert signal is send to the medical person.

The heart rate is varied with respect to the human age, like the normal person having 72 bpm (beats per minute), the aged person having 90 bpm and the child having 120 bpm.

In that heart rate is increasing when the human doing an exercise and the rest of the time it is going to a normal condition, but the heart rate is lower when compared to the normal range is called has a bradycardia and higher range is called as tachycardia.

The heart rate analyser is fixed to the human fingertip to counting the pulse over every 30 seconds and the signal given to the controller.

In existing methods the analysed data is multiple by 2 because it have more error, many type of electronic device is measure the heart rat like ECG but the cost of this device is high.

The low cost device is available in the wristwatch type or pulse oximeter is to measure the temperature also, so this type of device is efficient and cost effective.

In this project, we will design Heartbeat/Pulse/BPM Rate Monitor using Arduino & Pulse Sensor. You can interface the pulse sensor with Arduino for monitoring Heartbeat/Pulse/BPM Rate. We used a *LCD panel to display the pulse rate in BPM.*

This sensor is quite easy to use and operate. Place your finger on top of the sensor and it will sense the heartbeat by measuring the change in light from the expansion of capillary blood vessels.

OBJECTIVES:

The objective of this experiment is to build a system or device that will measure the rate of heart beat of human body & detect heart attack. The device must be able to monitor all the heart rate in a continuous interval length of time.

For the device to monitor the heart rate in a continuous interval length of time, it is important for the device to be able to display the information regarding the heart rate to the patient on the liquid-crystal display (LCD) screen as well.

MOTIVATION:

In a clinical environment, heart rate is measured under controlled conditions like blood measurement, heart beat measurement, listening to heartbeats using Stethoscope and Electrocardiogram (ECG), but these methods are expensive and need to be carryout by an experience medical personnel.

Drawbacks with ECG method are:

too many sensors and cables connections, fluctuations in the ECG signal baseline, power line noise, and interference due to muscular activities and high cost of procurement. More so, ECG is not suitable for continuous monitoring on burnt victims and the conduction gel used may cause discomfort and inflammation on the skin.

So to overcome this we thought of a SMART HEART RATE MONITORING SYSTEM which predicts the heart rate using infrared transmitter and receiver circuits where PhotoPlethysmography (PPG) is implemented.

CHALLENGES :

Several challenges were faced throughout the entire life cycle of the project.

The first one was with the pulse sensor. It didn't detect accurate readings if it was placed with excessive or loose pressure on the body

The next challenge was the fact that some of the purchased components were provided without datasheet, specifically the pulse sensor used for this project which made it difficult to fully understand the sensor specifications.

Hence, we depended on the basic information supplied by the vendors on their website.

Another challenge was with the Android development environment, which took considerable time for installation and setting up. In addition to the time spent fixing inconvenient Gradle errors that seemed never ending keeping in mind it was working correctly after it has been installed.

However working with AS was much better when we tried reinstalling and troubleshooting

LITERATURE SURVEY:

Different researchers used different methods and technologies to carry out the process of heart rate monitoring.

In [1] , Ultra-wideband (UWB) radio detection and ranging is used in almost all the major sensing signals works. It is widely used in wireless networking process which consumes only low power with more number of bandwidths can be formed in it. UWB remote sensing can also helps in detecting the problems arising in respiratory functions as well as the cardiac counts occurs.

In [2] , This paper relates about the monitoring of signals by using the aortic blood vessesls to take out organic signals from arterial waves. The sensor is fixed near to the heart valve that detects the blocks or any other problem arises in the human body. the other method is known as catheter which is inserted into our human body to find any sort of health disorders in sequential time difference.

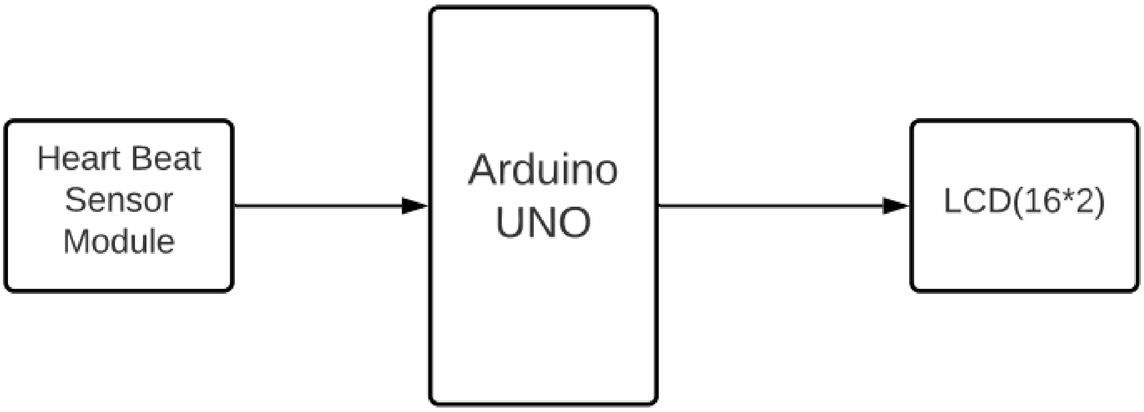
In [3] , Sleep observance is used to observe the sleep timings in a whole day of human’s life span. In this paper, respiratory and cardiac problems can be noted. Especially, during the night time there is a emergency purpose, the watch monitor is found to measure the patients health. This is also known as sleep monitoring using the wearable watch type.

In [4], In this paper, they have used a pressure mask with ECG device for the better application. The signal can be measured using the oximeter which is used to find the amount of oxygen flow in the blood. The viscous observation is the fluid resistance flow in the blood. The viscous observation is the fluid resistance flow in the blood. The pressure therapy helps to cures any type of disorder in the human body. There are two types of pressure of airways which is negative and positive.

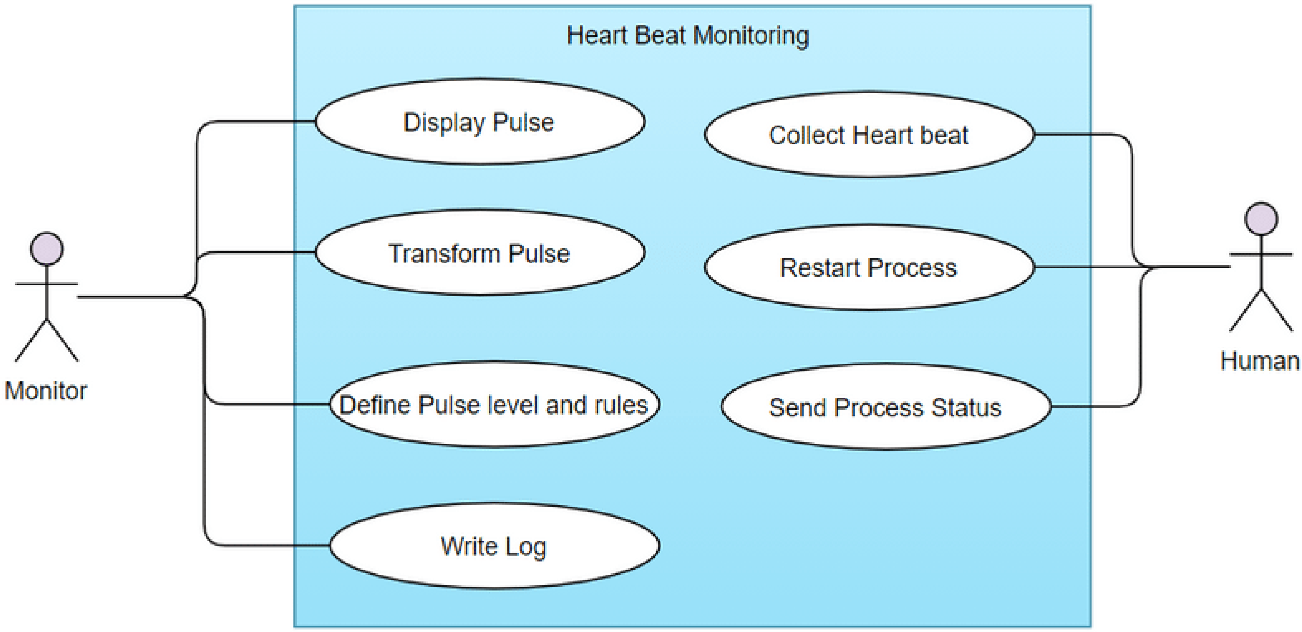
In [5] , In this project, smart chair monitoring is used with the advanced features used here is telemonitoring with additional sytem is implemented. This can be urged to use because of the emergency health problems. The patients can be monitored by keeping them in smart chair very comfortably for the aged people. This can also be connected by using GSM, blutooth,wi-fi and other network connections.

ENGINEERING DESIGN:

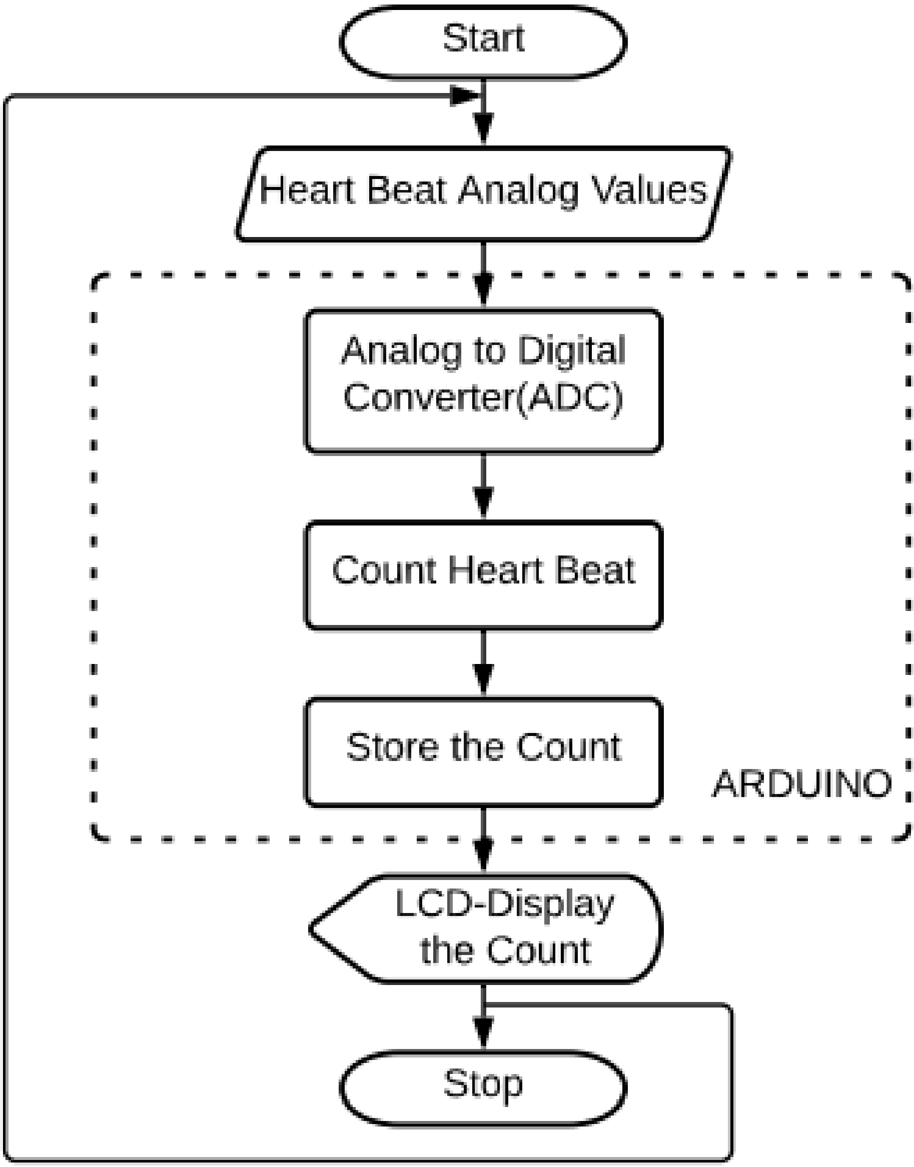
ER DIAGRAM:



USE CASE DIAGRAM:



DATA FLOW DIAGRAM:



TECHNICAL IMPLEMENTATION

In this project we are developing a Heart rate monitor using Arduino and a pulse sensor which follows the principle of Photoplethysmograph.

HOW IT WORKS?:

The proposed system is based on the working of infrared light is passing to the blood value and analysis the heart rate.

In this device is placed on the human fingertip and measure the heart rate through heart beat sensor and send the signal to the controller. First the sensor is fixed into the human fingertip the blood is circulated to the fingertip at the time sensor infrared light is passing to the photo diode via blood value to measure the pressure of blood and this measured value is given to the arduino controller. The controller analysis the sensed value and threshold value if any difference occur in the output the arduino controller display the value of sensor output in the LCD display.

The heart beat sensor having the photo diode and IR sensor, the working of this sensor is IR passed to the finger one side and the photo diode is receiving the signal and measure the pulse, blood count for 30 seconds. The intensity of the blood is decrease and increase is respect to the heart rate, so easily found the heart is normal or abnormal. The sensor measure value is converted into the voltage variation respect to the op-amp function and the output value is given to the controller in DC voltage from.

In this proposed system the output value is displayed in the LCD display to the consumer. The LCD interfacing to the arduino is very simple and easy steps, the display coding is return in the controller to show the output value of the sensors. The 16*2 LCD display is mostly used in the proposed system, the ‘16’ is denote the how many characters in the line and ‘2’ is denote to the how many rows in the display, 20*4 display is also available in the market ‘20’ is denote 20 characters of the line is available and ‘4’ is the how many rows are present in the display is shown in LCD. The 14 pins are available in the LCD display ‘8’ pins are data, ‘2’ pins are power and ‘3’ pins are control pins contrast adjustment is having one pin this is the pin details of the LCD display.

COMPONENTS:

- Arduino Uno Board and USB Cable
- Pulse Sensor
- Jumper Wires
- LCD
- Potentiometer 10K
- Breadboard
- Pushbutton

Circuit Design of Interfacing Heartbeat Sensor with Arduino :

The circuit design of Arduino based Heart rate monitor system using Heart beat Sensor is very simple. First, in order to display the heartbeat readings in bpm, we have to connect a 16×2 LCD Display to the Arduino UNO.

The 4 data pins of the LCD Module (D4, D5, D6 and D7) are connected to Pins 1, 1, 1 and 1 of the Arduino UNO. Also, a 10KΩ Potentiometer is connected to Pin 3 of LCD (contrast adjust pin). The RS and E (Pins 3 and 5) of the LCD are connected to Pins 1 and 1 of the Arduino UNO.

Next, connect the output of the Heartbeat Sensor Module to the Analog Input Pin (Pin 1) of Arduino.

Working of the Circuit

Upload the code to Arduino UNO and Power on the system. The Arduino asks us to place our finger in the sensor and press the switch.

Place any finger (except the Thumb) in the sensor clip and push the switch (button). Based on the data from the sensor, Arduino calculates the heart rate and displays the heartbeat in bpm.

While the sensor is collecting the data, sit down and relax and do not shake the wire as it might result in a faulty values.

After the result is displayed on the LCD, if you want to perform another test, just push the rest button on the Arduino and start the procedure once again.

CODE:

pulse_sensor

```
#include <LiquidCrystal.h>

//  Variables
int pulsePin = 0;                 // Pulse Sensor purple wire connected to analog pin 0
int blinkPin = 13;                // pin to blink led at each beat
int fadePin = 8;                  // pin to do fancy classy fading blink at each beat
int fadeRate = 0;                 // used to fade LED on with PWM on fadePin

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

// Volatile Variables, used in the interrupt service routine!
volatile int BPM;                   // int that holds raw Analog in 0. updated every 2mS
volatile int Signal;                // holds the incoming raw data
volatile int IBI = 600;             // int that holds the time interval between beats! Must be seeded!
volatile boolean Pulse = false;     // "True" when User's live heartbeat is detected. "False" when not a "live beat".
volatile boolean QS = false;        // becomes true when Arduoino finds a beat.

// Regards Serial OutPut  -- Set This Up to your needs
static boolean serialVisual = true;   // Set to 'false' by Default.  Re-set to 'true' to see Arduino Serial Monitor ASCII Visual Pulse

volatile int rate[10];                      // array to hold last ten IBI values
volatile unsigned long sampleCounter = 0;          // used to determine pulse timing
volatile unsigned long lastBeatTime = 0;           // used to find IBI
volatile int P = 512;                      // used to find peak in pulse wave, seeded
volatile int T = 512;                     // used to find trough in pulse wave, seeded
volatile int thresh = 525;                // used to find instant moment of heart beat, seeded
volatile int amp = 100;                   // used to hold amplitude of pulse waveform, seeded
volatile boolean firstBeat = true;        // used to seed rate array so we startup with reasonable BPM
volatile boolean secondBeat = false;      // used to seed rate array so we startup with reasonable BPM
```

pulse_sensor

```
void setup()
{
  pinMode(blinkPin,OUTPUT);         // pin that will blink to your heartbeat!
  pinMode(fadePin,OUTPUT);          // pin that will fade to your heartbeat!
  Serial.begin(115200);             // we agree to talk fast!
  interruptSetup();                 // sets up to read Pulse Sensor signal every 2mS
                                    // IF YOU ARE POWERING The Pulse Sensor AT VOLTAGE LESS THAN THE BOARD VOLTAGE,
                                    // UN-COMMENT THE NEXT LINE AND APPLY THAT VOLTAGE TO THE A-REF PIN
                                    //   analogReference(EXTERNAL);
}


//  Where the Magic Happens
void loop()
{
   serialOutput();

  if (QS == true) // A Heartbeat Was Found
    {
     // BPM and IBI have been Determined
     // Quantified Self "QS" true when arduino finds a heartbeat
     fadeRate = 255; // Makes the LED Fade Effect Happen, Set 'fadeRate' Variable to 255 to fade LED with pulse
     serialOutputWhenBeatHappens(); // A Beat Happened, Output that to serial.
     QS = false; // reset the Quantified Self flag for next time
    }

  ledFadeToBeat(); // Makes the LED Fade Effect Happen
  delay(20); //  take a break
}

void ledFadeToBeat()
```

```
void ledFadeToBeat()
{
   fadeRate -= 15;                         //  set LED fade value
   fadeRate = constrain(fadeRate,0,255);   //  keep LED fade value from going into negative numbers!
   analogWrite(fadePin,fadeRate);          //  fade LED
}

void interruptSetup()
{
  // Initializes Timer2 to throw an interrupt every 2mS.
  TCCR2A = 0x02;     // DISABLE PWM ON DIGITAL PINS 3 AND 11, AND GO INTO CTC MODE
  TCCR2B = 0x06;     // DON'T FORCE COMPARE, 256 PRESCALER
  OCR2A = 0X7C;      // SET THE TOP OF THE COUNT TO 124 FOR 500Hz SAMPLE RATE
  TIMSK2 = 0x02;     // ENABLE INTERRUPT ON MATCH BETWEEN TIMER2 AND OCR2A
  sei();             // MAKE SURE GLOBAL INTERRUPTS ARE ENABLED
}

void serialOutput()
{   // Decide How To Output Serial.
 if (serialVisual == true)
  {
     arduinoSerialMonitorVisual('-', Signal);   // goes to function that makes Serial Monitor Visualizer
  }
 else
  {
      sendDataToSerial('S', Signal);     // goes to sendDataToSerial function
   }
}
```

pulse_sensor

```
      sendDataToSerial('S', Signal);     // goes to sendDataToSerial function
   }
}

void serialOutputWhenBeatHappens()
{
 if (serialVisual == true) //  Code to Make the Serial Monitor Visualizer Work
   {
     Serial.print("*** Heart-Beat Happened *** ");  //ASCII Art Madness
     Serial.print("BPM: ");
     Serial.println(BPM);
     lcd.clear();
     lcd.print("BPM: ");
     lcd.print(BPM);
   }
 else
   {
     sendDataToSerial('B',BPM);   // send heart rate with a 'B' prefix
     sendDataToSerial('Q',IBI);   // send time between beats with a 'Q' prefix
   }
}

void arduinoSerialMonitorVisual(char symbol, int data )
{
  const int sensorMin = 0;      // sensor minimum, discovered through experiment
  const int sensorMax = 1024;    // sensor maximum, discovered through experiment
  int sensorReading = data; // map the sensor range to a range of 12 options:
  int range = map(sensorReading, sensorMin, sensorMax, 0, 11);
  // do something different depending on the
  // range value:
  switch (range)
```

```
void sendDataToSerial(char symbol, int data )
{
   Serial.print(symbol);
   Serial.println(data);
}

ISR(TIMER2_COMPA_vect) //triggered when Timer2 counts to 124
{
  cli();                                      // disable interrupts while we do this
  Signal = analogRead(pulsePin);              // read the Pulse Sensor
  sampleCounter += 2;                         // keep track of the time in mS with this variable
  int N = sampleCounter - lastBeatTime;       // monitor the time since the last beat to avoid noise
                                              //  find the peak and trough of the pulse wave
  if(Signal < thresh && N > (IBI/5)*3) // avoid dichrotic noise by waiting 3/5 of last IBI
    {
      if (Signal < T) // T is the trough
      {
        T = Signal; // keep track of lowest point in pulse wave
      }
    }

  if(Signal > thresh && Signal > P)
    {          // thresh condition helps avoid noise
      P = Signal;                             // P is the peak
    }                                        // keep track of highest point in pulse wave

  //  NOW IT'S TIME TO LOOK FOR THE HEART BEAT
  // signal surges up in value every time there is a pulse
  if (N > 250)
  {                                   // avoid high frequency noise
```

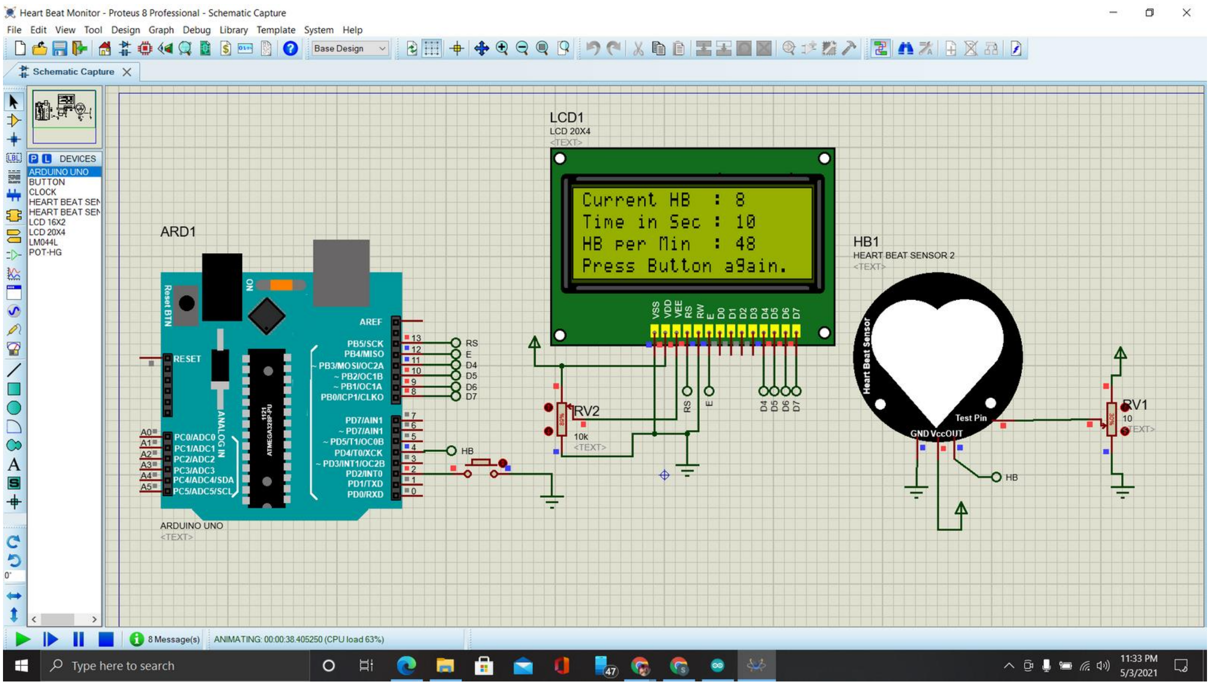
pulse_sensor

```
  {                                   // avoid high frequency noise
    if ( (Signal > thresh) && (Pulse == false) && (N > (IBI/5)*3) )
      {
        Pulse = true;                               // set the Pulse flag when we think there is a pulse
        digitalWrite(blinkPin,HIGH);                // turn on pin 13 LED
        IBI = sampleCounter - lastBeatTime;         // measure time between beats in mS
        lastBeatTime = sampleCounter;               // keep track of time for next pulse

        if(secondBeat)
        {                        // if this is the second beat, if secondBeat == TRUE
          secondBeat = false;                  // clear secondBeat flag
          for(int i=0; i<=9; i++) // seed the running total to get a realisitic BPM at startup
          {
            rate[i] = IBI;
          }
        }

        if(firstBeat) // if it's the first time we found a beat, if firstBeat == TRUE
        {
          firstBeat = false;                   // clear firstBeat flag
          secondBeat = true;                   // set the second beat flag
          sei();                               // enable interrupts again
          return;                              // IBI value is unreliable so discard it
        }
      // keep a running total of the last 10 IBI values
      word runningTotal = 0;                  // clear the runningTotal variable

      for(int i=0; i<=8; i++)
        {                // shift data in the rate array
          rate[i] = rate[i+1];                  // and drop the oldest IBI value
          runningTotal += rate[i];              // add up the 9 oldest IBI values
```

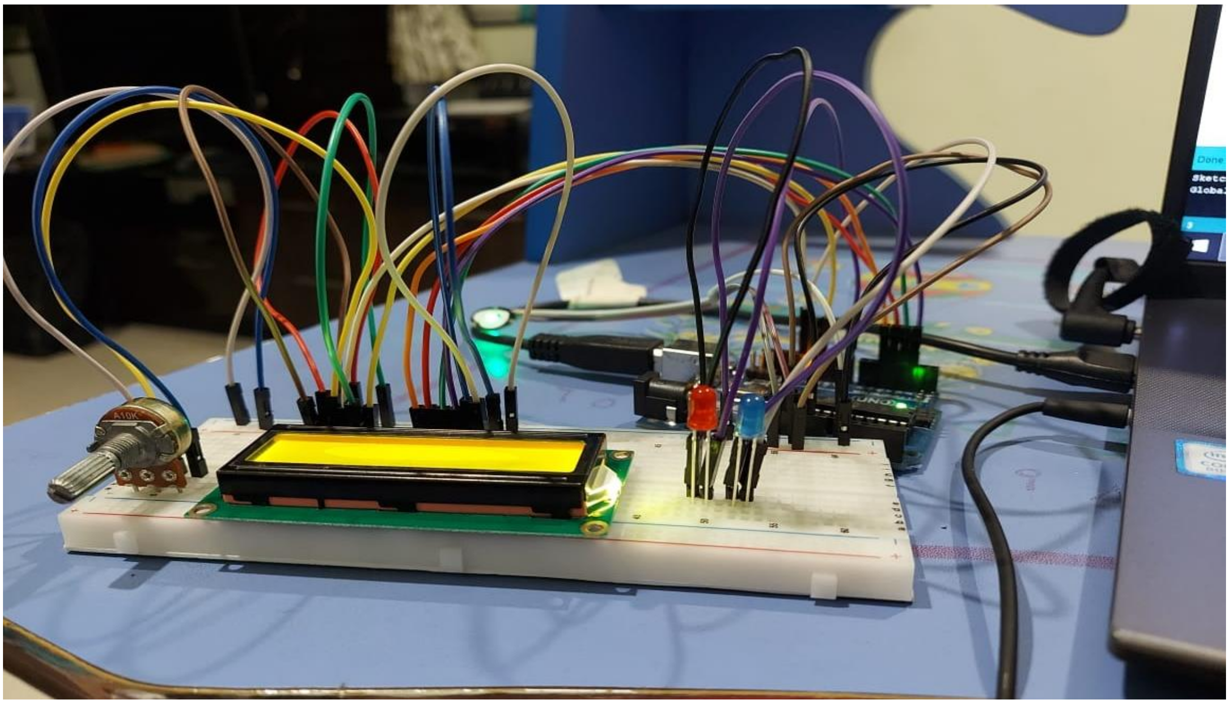
pulse_sensor

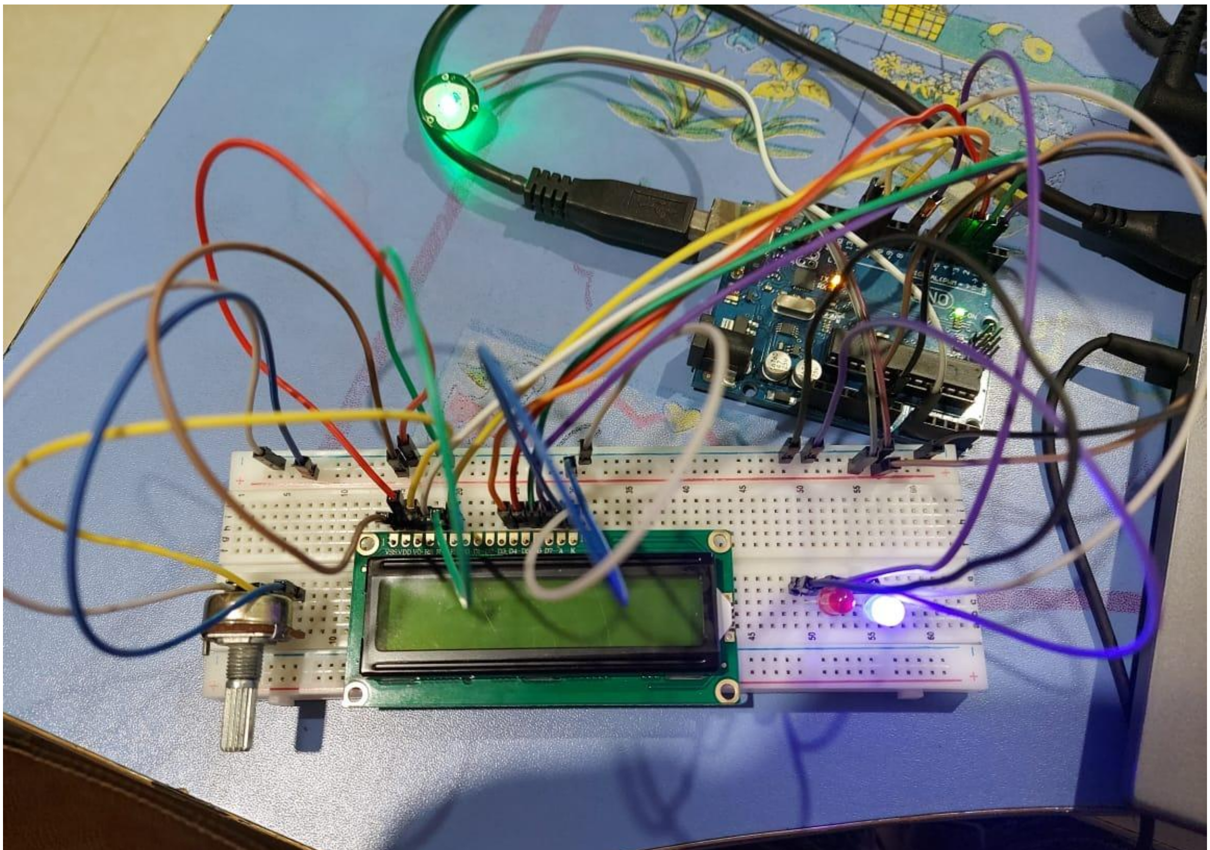
```
        rate[9] = IBI;                          // add the latest IBI to the rate array
        runningTotal += rate[9];                // add the latest IBI to runningTotal
        runningTotal /= 10;                     // average the last 10 IBI values
        BPM = 60000/runningTotal;               // how many beats can fit into a minute? that's BPM!
        QS = true;                              // set Quantified Self flag
        // QS FLAG IS NOT CLEARED INSIDE THIS ISR
      }
    }

    if (Signal < thresh && Pulse == true)
      {   // when the values are going down, the beat is over
        digitalWrite(blinkPin,LOW);            // turn off pin 13 LED
        Pulse = false;                         // reset the Pulse flag so we can do it again
        amp = P - T;                           // get amplitude of the pulse wave
        thresh = amp/2 + T;                    // set thresh at 50% of the amplitude
        P = thresh;                            // reset these for next time
        T = thresh;
      }

    if (N > 2500)
      {                             // if 2.5 seconds go by without a beat
        thresh = 512;                          // set thresh default
        P = 512;                               // set P default
        T = 512;                               // set T default
        lastBeatTime = sampleCounter;          // bring the lastBeatTime up to date
        firstBeat = true;                      // set these to avoid noise
        secondBeat = false;                    // when we get the heartbeat back
      }

    sei();                                  // enable interrupts when youre done!
  }// end isr
```

SOFTWARE SIMULATION:



HARDWARE IMPLEMENTATION:





CONCLUSION:

In this project, the design and development of a low cost HRM device has been presented.

The device is ergonomic, portable, durable, and cost effective. The HRM device is efficient and easy to use.

Tests have shown excellent agreement with actual heartbeat rates. This device could be used in clinical and nonclinical environments. It can also be easily used by individual users, e.g. athletes during sporting activities.

The device could also be used as a monitoring instrument exploiting the SMS capabilities provided by this system.

By using my heart beat rate monitor machine I get the heart beat counting in a minute.

In doing so variation of reading may be observed in a person's beat counted if I take the reading at different places of different fingers.

So for most appropriate reading the LED should be placed just beside the upper portion of the nail. T

he more the finger contacts with LED the more accurate the reading becomes.

The counted beat varies from person to person so no need to worry or think about the accuracy of the device.

FUTURE SCOPE

- Monitoring device that could be used to detect the heart beat anomalies of physically challenged individuals without hands.
- Also a graphical LCD can be used to display a graph of the change of heart rate over time.
- A serial output can be incorporated into the device so that the heart rates can be sent to a Personal Computer (PC) for further online or offline analysis.
- It could be integrated with mobile technology for e-health cloud transmission to health care providers.

REFERENCES:

- http://www.jcreview.com/fulltext/197-1590057169.pdf

- https://www.researchgate.net/publication/4083599_Personal_autonomic_computing_self-healing_tool

- https://www.generationrobots.com/media/DetecteurDePoulsAmplifie/PulseSensorAmpedGettingStartedGuide.pdf

- https://www.researchgate.net/publication/330831756_Design_and_Fabrication_of_Arduino_Based_Heart_Rate_Monitoring_System_Using_Reflectance_Photoplethysmography_Mubarak_Riaz

- https://pulsesensor.com/pages/open-hardware

VIDEO LINK:

https://drive.google.com/file/d/1AyjUUc-iULsIhMLtHSYxOWfEqthJF9R8/view?usp=sharing