**ABSTRACT**

People have a variety of reasons to use heart rate monitors. For example, patients in a hospital might have stationary, bedside equipment monitor their heart rate and alert medical staff in case of an emergency. Somebody going for a run might wear a portable heart rate monitor to keep track of their workout intensity. Heart rate monitors are not all the same—their appearance and function will vary depending on the intended use.

In order 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
* other way is to use a Heartbeat/Pulse Sensor.

In this project we have designed, build, and program our own Heart Rate Monitor that fits the needs using Arduino and Pulse Sensor.

A **pulse sensor** is a hardware device that can be used to measure **heart rate** in real-time. When paired with an **Arduino microcontroller**, we can create a simple yet effective heart rate monitor. 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.

In this project, we will interface **Pulse Sensor** with **Arduino** to Measure **Pulse Rate (BPM)** or **Heart Beat** value. The Pulse rate will be displayed on **16×2 LCD** Display.

|  |  |  |
| --- | --- | --- |
|  | **Content** |  |
| **Chapter No** | **Tittle** | **Page No.** |
|  | **Introduction** | **8** |
| **1** | **Pulse Sensor** | **9** |
| **2** | **Arduino Uno** | **12** |
| **3** | **16 x 2 LCD Display** | **14** |
| **4** | **I2C Module** | **15** |
| **5** | **Interfacing Pulse Sensor with Arduino** | **15** |
| **6** | **Displaying Pulse Rate Value on LCD** | **16** |
| **7** | **Problem Statement** | **18** |
| **8** | **Project Objectives** | **18** |
| **9** | **Architecture of Project** | **18** |
| **10** | **Codes Used** | **22** |
| **11** | **Testing Results** | **25** |
| **12** | **Literature survey** | **26** |
| **13** | **Methodology** | **26** |
| **14** | **Observation done** | **26** |
| **15** | **Photographs taken in Company** | **26** |
| **16** | **Complete analysis of Project done** | **26** |
| **17** | **Technology Used** | **26** |
| **18** | **Project Photos** | **27** |
| **19** | **Conclusion** | **29** |
| **20** | **References** | **30** |

**Introduction**

People have a variety of reasons to use heart rate monitors. For example, patients in a hospital might have stationary, bedside equipment monitor their heart rate and alert medical staff in case of an emergency. Somebody going for a run might wear a portable heart rate monitor to keep track of their workout intensity. Heart rate monitors are not all the same—their appearance and function will vary depending on the intended use.

In order 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
* other way is to use a Heartbeat/Pulse Sensor.

In this project we have designed, build, and program our own Heart Rate Monitor that fits the needs using Arduino and Pulse Sensor.

A **pulse sensor** is a hardware device that can be used to measure **heart rate** in real-time. When paired with an **Arduino microcontroller**, we can create a simple yet effective heart rate monitor. 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.

In this project, we will interface **Pulse Sensor** with **Arduino** to Measure **Pulse Rate (BPM)** or **Heart Beat** value. The Pulse rate will be displayed on **16×2 LCD** Display.

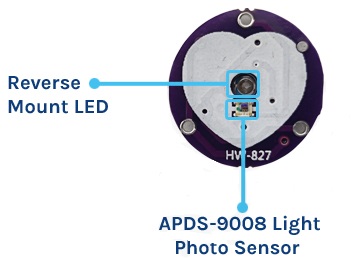
Details of components used here are given below.

1. **Pulse Sensor**

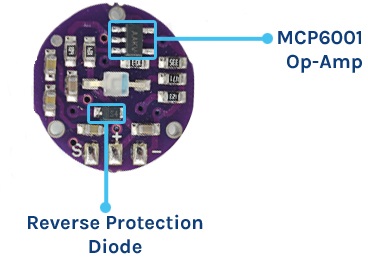
The Pulse Sensor is a plug-and-play heart-rate sensor for Arduino. It can be used by students, artists, athletes, makers, and game & mobile developers who want to easily incorporate live heart-rate data into their projects.

[](https://how2electronics.com/wp-content/uploads/2019/02/SEN-11574-heart-pulse-sensor.jpg)

The essence is an integrated **optical amplifying circuit** and **noise eliminating circuit** sensor. Clip the **Pulse Sensor** to your earlobe or fingertip. Then it into your **Arduino**, you are now ready to read **heart rate**.

[](https://how2electronics.com/wp-content/uploads/2019/02/Front-Side.jpg)

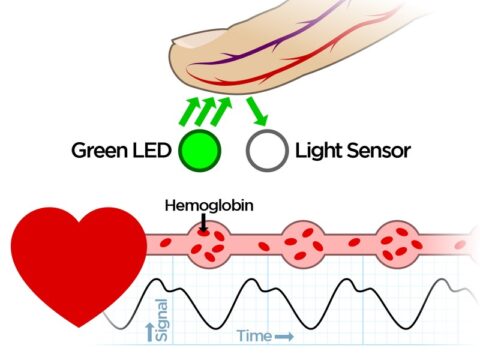
The front of the sensor comes with the heart logo. This is where you place your finger. On the front side, you will see a small round hole, from where the **green LED** shines. Just below the LED is a small **ambient light photosensor** [**APDS9008**](https://www.mouser.com/datasheet/2/678/av02-1169en-1828127.pdf) which adjust the brightness in different light conditions.

[](https://how2electronics.com/wp-content/uploads/2019/02/BackSide.jpg)

On the back of the module you will find **MCP6001 Op-Amp IC**, a few resistors, and capacitors. This makes up the **R/C filter** network. There is also a **reverse protection diode** to prevent damage if you connect the power leads reverse.

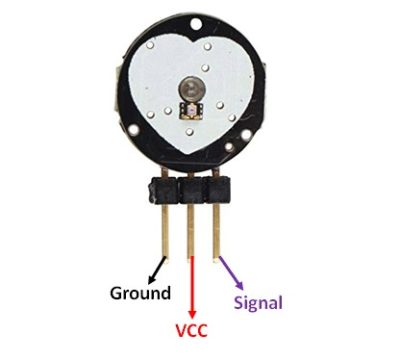
* 1. **Pulse Sensor Technical Specifications**
  2. **Physical Characteristics**
* **Dimensions**: Approximately 0.625″ (15.875mm) in diameter
* **Weight**: Lightweight, usually around a few grams
* **Material**: Biocompatible materials for safe skin contact
  1. **Electrical Characteristics**
* **Operating Voltage**: 3V – 5.5V
* **Current Consumption**: Typically around 4mA
* **Output Signal**: Analog (0.3V to VCC)
* **Signal Range**: 0-1023 (10-bit ADC output of Arduino)
  1. **Sensing Technology**
* **Sensor Type**: Photoplethysmogram (PPG)
* **Wavelength**: Typically around 565nm (Green LED)
  1. **Working of the Pulse Sensor**

The Pulse Sensor works on the principle of Photoplethysmography (PPG), which is a non-invasive method for measuring changes in blood volume under the skin. The sensor essentially consists of two main components: a light-emitting diode (LED) that shines light into the skin and a photodetector that measures the amount of light that is reflected back. Here’s a detailed explanation of its working:

[](https://how2electronics.com/wp-content/uploads/2019/02/Pulse-Sensor-Working.jpg)

1. **Light Emission**: A green LED emits light into the skin.
2. **Reflection & Detection**: The light interacts with blood and is partially reflected back, captured by a photodetector.
3. **Heart Rate**: Changes in reflected light create a waveform that correlates with heartbeats.
4. **Oxygen Level**: The amount of reflected light also indicates blood oxygen levels, as oxygenated blood absorbs more green light.
5. **Signal Filtering**: A Low Pass Filter cleans up the noisy, raw signal from the photodetector.
6. **Amplification**: An operational amplifier boosts the filtered signal for better accuracy.
7. **Data Reading**: Finally, an Arduino reads the amplified signal and software algorithms translate it into heart rate and potentially blood oxygen levels.
   1. **Pulse Sensor PinOut**

The pulse sensor has three pins: VCC, GND & Analog Pin.

[](https://how2electronics.com/wp-content/uploads/2019/02/Pulse-Sensor-Pinout.jpg)

1. **Arduino Uno:**

An Arduino is actually a microcontroller based kit which can be either used directly by purchasing from the vendor or can be made at home using the components, owing to its open source hardware feature. It is basically used in communications and in controlling or operating many devices.

* Digital pins: 14 (These pins have only 2 states i.e. high or low or in simple words either 5 V or 0 V no in between values. These pins are mostly used to sense the voltage presence when switch is open or close)
* Analog pins: 6 (A0 to A5 and they come up with a resolution of 10 bits and they provide flexibility of connecting any external device via these pins. These pins are configured from 0 V to 5 V but they can be configured to high range by using AREF pin or analogReference () function. ADC (analog to digital convertor) is used to sample these pins. These pins take analog signal and by using ADC convertor they convert this analog signal to number between 0 – 1023)
* 16 MHz crystal oscillator
* Out of 14 digital pins, 6 can be used for PWM (pulse width modulation)
* USB port
* TX and RX pins (for serial communication)
* Arduino is an open-source hardware and software company, project and user community that designs and manufactures single-board microcontrollers and microcontroller kits for building digital devices. Its products are licensed under the GNU Lesser General Public License (LGPL) or the GNU General Public License (GPL), permitting the manufacture of Arduino boards and software distribution by anyone. Arduino boards are available commercially in preassembled form or as doityourself (DIY) kits.
* Arduino board designs use a variety of microprocessors and controllers. The boards are equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards ('shields') or breadboards (For prototyping) and other circuits. The boards feature serial communications interfaces, including Universal Serial Bus (USB) on some models, which are also used for loading programs from personal computers. The microcontrollers can be programmed using C and C++ programming languages. In addition to using traditional compiler toolchains, the Arduino project provides an integrated development environment (IDE) based on the Processing language project.
* The Arduino project started in 2005 as a program for students at the Interaction Design Institute Ivrea in Ivrea, Italy, aiming to provide a low-cost and easy way for novices and professionals to create devices that interact with their environment using sensors and actuators. Common examples of such devices intended for beginner hobbyists include simple robots, thermostats and motion detectors.
* The name Arduino comes from a bar in Ivrea, Italy, where some of the founders of the project used to meet. The bar was named after Arduin of Ivrea, who was the margrave of the March of Ivrea and King of Italy from 1002 to 1014.

**Different Types Of Arduino Boards**

* + Arduino Uno
  + Arduino due
  + Arduino Mega (R3)
  + Arduino Leonardo

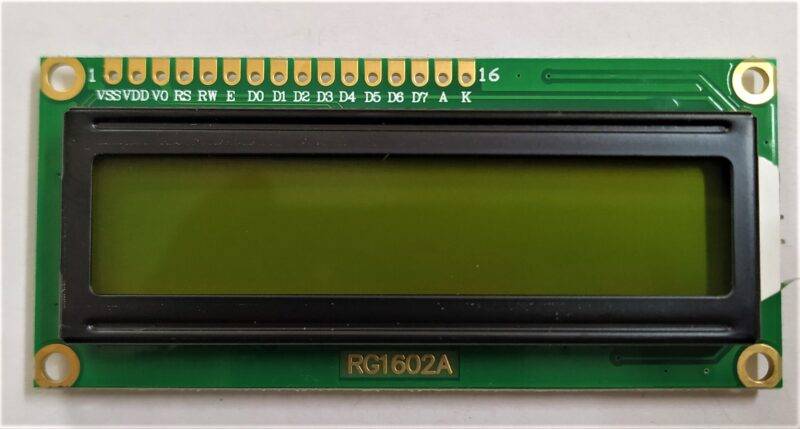
**Arduino Uno**

****

The Uno is a huge option for your initial Arduino. It consists of 14-digital I/O pins, where 6- pins can be used as PWM(pulse width modulation outputs), 6-analog inputs, a reset button, a power jack, a USB connection and more. It includes everything required to hold up the microcontroller; simply attach it to a PC with the help of a USB cable and give the supply to get started with a AC-to-DC adapter or battery.

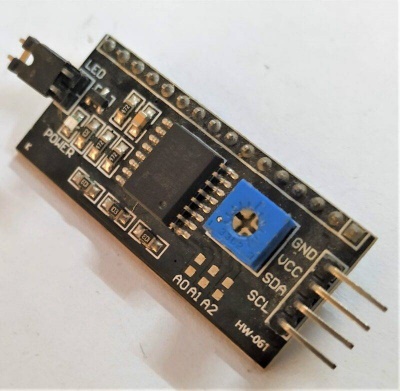
1. **16×2 LCD display**

LCD stands for liquid crystal display and there are crystals inside the display which illuminates the full display and the character as those which crystals are not illuminated.



* An LCD (Liquid Crystal Display) screen is an electronic display module and has a wide range of applications. It’s having a Hitachi driver.
* 16×2 LCD display content 2 rows and 16 columns. where you can print 16 characters into one row. There are RW and RS pin. so, to **interface LCD with Arduino** you need to know some pin on the LCD.
* RS ( Resistor selected, enables a user to select the instruction mode or the character mode)
* R/w ( Read/write, enables a user to select the Read or Write mode)
* E( Enable, Enable driver to on the LCD)
* From D0 to D7 all pins are used for data transfer.

1. **I2C Module (LCD Interfacing with Arduino)**



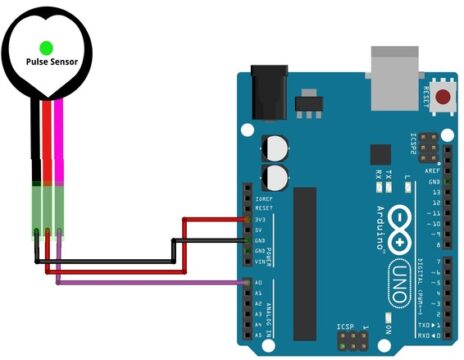
* The display originally uses multiple wires communication with the Arduino but we can use an I2C module to make it simple and easier to program.
* The brightness of the display can be controlled if using multiple wires connection or it can be programmed in code if using a **liquidcrystal\_i2c** module.
* LCD stands for liquid crystal display and there are crystals inside the display which illuminates the full display and the character as those which crystals are not illuminated.
* if you see the display carefully it works as reverse as other displays. here the character is not illuminating they are inactive when we apply the signal to them.
* for example, if we are sending character A then the A-shaped crystal remains inactive, and the remaining crystal activates.

1. **Interfacing Pulse Sensor with Arduino**

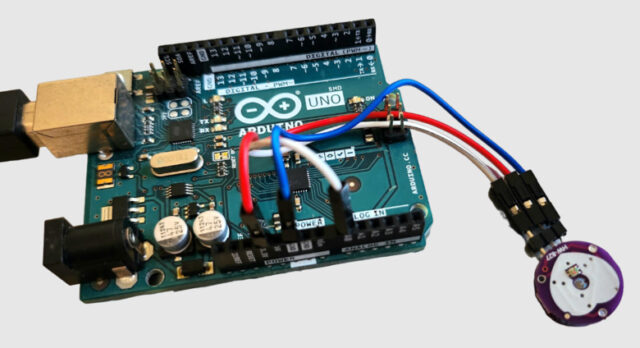
Let us interface the Pulse Sensor with Arduino and start measuring the Pulse Rate/Heart Rate/BPM Value.

**Hardware Wiring Diagram**

The connection diagram between Pulse Sensor and Arduino is so easy.

[](https://how2electronics.com/wp-content/uploads/2019/02/Pulse-Sensor-Arduino-Connection.jpg)

* Connect the RED wire (Power) of the Pulse Sensor to the 5V pin on the Arduino.
* Connect the BLACK wire (Ground) to the GND pin on the Arduino.
* Connect the PURPLE wire (Signal) to Analog Pin 0 (A0) on the Arduino.

[](https://how2electronics.com/wp-content/uploads/2019/02/Pulse-Sensor-Arduino-Interfacing.jpg)

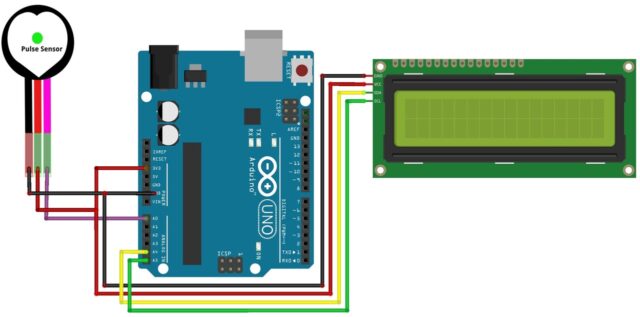
Using the Jumper Wires you can directly connect the Pulse Sensor with Arduino.

1. **Displaying Pulse Rate (BPM) Value on LCD Display**

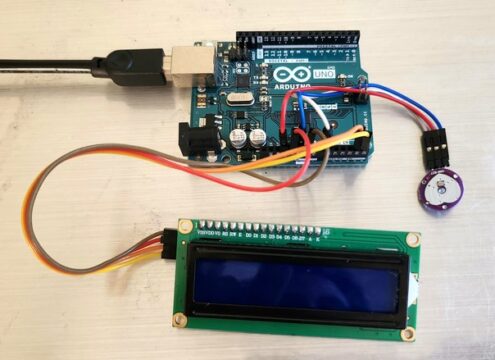
Instead of displaying the BPM value on Serial Monitor, we can display the value on LCD Display. We can use a 16×2 I2C LCD Display Code and interface with Arduino Board to display Pulse Sensor BPM Value.

**Hardware Wiring Diagram**

Since we are using an I2C LCD Display, connect it to the Arduino I2C Pins.

[](https://how2electronics.com/wp-content/uploads/2019/02/Arduino-Pulse-Sensor-LCD.jpg)

* **VCC (Power)**: Connect the VCC pin on the I2C LCD to the 5V pin on the Arduino.
* **GND (Ground)**: Connect the GND pin on the I2C LCD to the GND pin on the Arduino.
* **SCL (Clock)**: Connect the SCL pin on the I2C LCD to the SCL pin (A5) on the Arduino.
* **SDA (Data)**: Connect the SDA pin on the I2C LCD to the SDA (A4) pin on the Arduino.

[](https://how2electronics.com/wp-content/uploads/2019/02/Arduino-Pulse-Sensor-BPM-LCD-Connection.jpg)

You may use breadboard or jumper wire for connection.

1. **Problem statement**

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 more easy 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.

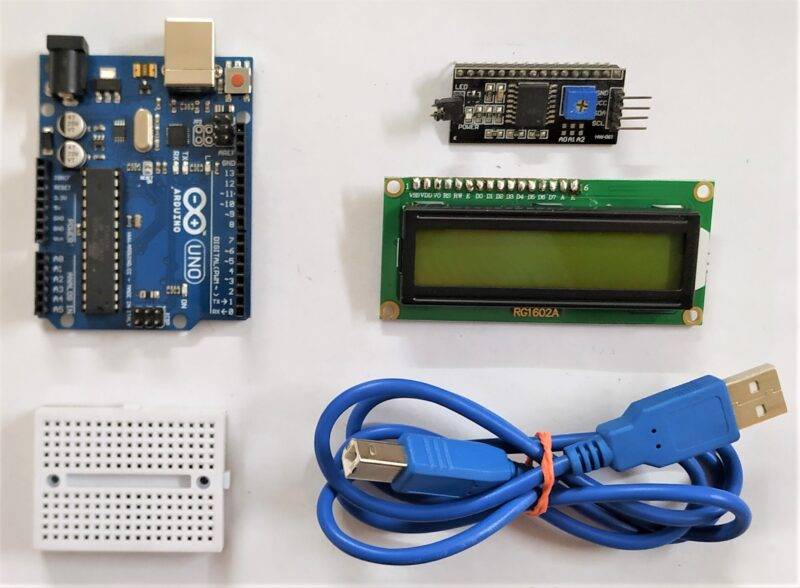
Heartbeat Sensors are available in Wrist Watches (Smart Watches), Smart Phones, cheststraps, etc. The heartbeat is measured in beats per minute or bpm, which indicates the number of times the heart is contracting or expanding in a minute.

1. **Project Objectives**

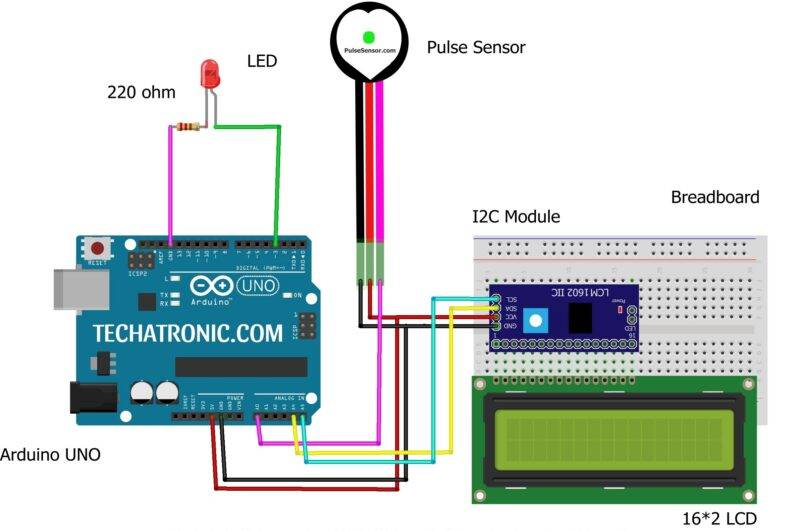
To design a Heart Rate Monitor.

1. **Architecture of Project**
   * **Components Required**

* [Arduino UNO](https://techatronic.com/what-is-arduino-brief-description/)
* [Pulse sensor](https://techatronic.com/?s=sensor)
* [LED](https://techatronic.com/diode/)
* [220 ohm resistor](https://techatronic.com/what-is-a-resistor-and-its-basic-type/)
* [I2C module](https://techatronic.com/lcd-interfacing-with-arduino-using-i2c/)
* [16×2 LCD](https://techatronic.com/interface-lcd-with-arduino-16x2/)
* Jumper wires and a breadboard
* USB cable for uploading the code



* + **Circuit Diagram:**



* Take a **pulse sensor** and connect its VCC pin with the 5 volt pin of the [Arduino](https://techatronic.com/types-of-arduino-boards-arduino-uno-mega-mini-specification/).
* Join the GND pin of the pulse sensor with the GND pin of the Arduino.
* Attach the OUT/signal pin of the heart beat sensor to the Analog-0 pin of the Arduino.
* Now take an LED and connect its positive leg with the digital-3 pin of the Arduino.
* Join the negative leg of the LED with the GND pin of the Arduino via a [220 ohm resisto](https://techatronic.com/what-is-a-resistor-and-its-basic-type/)r.
* Then Connect the I2C module with the 16×2 LCD module.
* You can also check the interfacing of the [I2C module with Arduino](https://techatronic.com/lcd-interfacing-with-arduino-using-i2c/).
* Join the VCC pin of the I2C module with the 5 volt pin of the Arduino and the GND pin of the Arduino with the GND pin of the I2C module.
* Connect the SDA and SCK pins of the [I2C module](https://techatronic.com/i2c-scanner/) with the analog-4 and analog-5 pins of the Arduino as shown in the diagram.
* Make sure that the connections are correct and tight.

|  |  |
| --- | --- |
| Arduino UNO | Pulse Sensor / heart beat sensor |
| A0 Pin | OUT Pin |
| VCC | VCC |
| GND | GND |
| Arduino UNO | I2C Module |
| A4 Pin  ( SDA ) | SDA Pin |
| A5 Pin  ( SCL ) | SCL Pin |
| VCC | VCC |
| GND | GND |
| 16\*2 LCD Display | I2C Module |
| 16 Pin connect | 16 Pin connect |
| Arduino UNO | LED | 220-ohm Resistor |
| 3 Pin | Anode Terminal |  |
| GND |  | Terminal 1 |
|  | Cathode Terminal | Terminal 2 |

|  |  |
| --- | --- |
| Arduino UNO | Pulse Sensor / heart beat sensor |
| A0 Pin | OUT Pin |
| VCC | VCC |
| GND | GND |
| Arduino UNO | I2C Module |
| A4 Pin  ( SDA ) | SDA Pin |
| A5 Pin  ( SCL ) | SCL Pin |
| VCC | VCC |
| GND | GND |
| 16\*2 LCD Display | I2C Module |
| 16 Pin connect | 16 Pin connect |
| Arduino UNO | LED | 220-ohm Resistor |
| 3 Pin | Anode Terminal |  |
| GND |  | Terminal 1 |
|  | Cathode Terminal | Terminal 2 |

* + **Heart Rate Monitor - How does it work?**

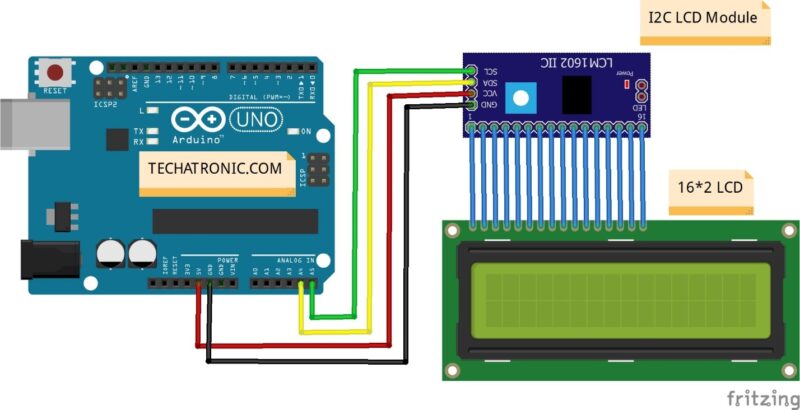
The working of the heart rate monitor and sensor. the pulse rate sensor having a green color light on which we have to put our finger so that the light can incident on our skin. after that, the sensor read the value and sens this value to the Arduino. Arduino process on this value and after all the calculation it send the data to 16X2 LCD display

Heart rate sensor. Heart rate sensor work on the optical principle. there are one light and one photodiode that sense the light intensity. the sensitivity is very high of this photodiode so that can easily sense the low-intensity light also. now, when we place our finger at the top of the sensor the light incident from the light source on the finger and back to the photodiode. when the blood suddenly increases the light sense by the photodiode also be change. so that the sensor can send this data to the process. this is how the heart rate monitor systems work.

* + **Interface LCD with Arduino (16×2)  :**

Two methods to interface LCD one is directly another is using [LCD connect with I2](https://techatronic.com/interface-lcd-with-arduino-16x2/)C.

* + **LCD Connection with Arduino through I2C Module:-**



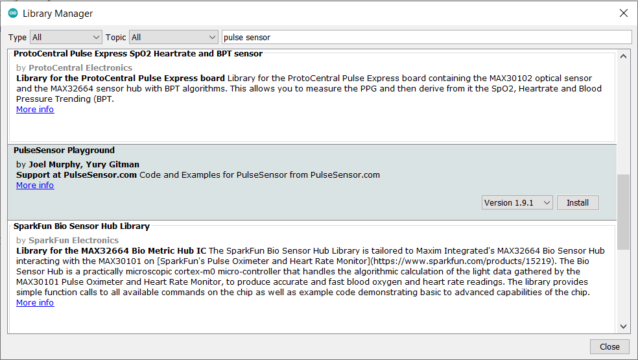
* + **Connection table**

|  |  |
| --- | --- |
| Arduino UNO | I2C Module |
| ( +5V ) | VCC |
| GND | GND |
| A4 Pin  ( SDA ) | SDA Pin |
| A5 Pin  ( SCL ) | SCL Pin |
| 16\*2 LCD Display | I2C Module |
| 16 Pin connect | 16 Pin connect |

I2C connection of the display to the I2C module as pin count is the same and we have to just connect it to parallelly with I2C module and from I2C module connect 4 Wires VCC, GND to VCC and GND of Arduino and SCL, SDA pins of I2c module to A5and A4 analog pins or I25C pins of Arduino Uno or any other microcontroller similar to Arduino.

* + **Pulse Sensor Library Installation**

Before moving to the coding part, you need to add the Pulse Sensor Library on your Arduino Library Folder.

[](https://how2electronics.com/wp-content/uploads/2019/02/Library.png)

Download the PulseSensor Playground Library from the Arduino IDE (Go to Sketch -> Include Library -> Manage Libraries, then search for “PulseSensor Playground” and install it).

1. **Codes Used**
   1. [PulseSensorPlayground.h](https://github.com/WorldFamousElectronics/PulseSensorPlayground)
   2. [LiquidCrystal\_I2C.h](https://github.com/fdebrabander/Arduino-LiquidCrystal-I2C-library)

We have to install <[PulseSensorPlayground.h](https://github.com/WorldFamousElectronics/PulseSensorPlayground)> and <[LiquidCrystal\_I2C.h](https://github.com/fdebrabander/Arduino-LiquidCrystal-I2C-library)> libraries first.

**#define USE\_ARDUINO\_INTERRUPTS true //--> Set-up low-level interrupts for most acurate BPM math.**

**#include <PulseSensorPlayground.h> //--> Includes the PulseSensorPlayground Library.**

**#include <LiquidCrystal\_I2C.h> //--> Includes the LiquidCrystal Library.**

**LiquidCrystal\_I2C lcd(0x27,16,2);**

**const int PulseWire = 0; //--> PulseSensor PURPLE WIRE connected to ANALOG PIN 0**

**int LED\_3 = 3; //--> LED to detect when the heart is beating. The LED is connected to PIN 3 on the Arduino UNO.**

**int Threshold = 550; //--> Determine which Signal to "count as a beat" and which to ignore.**

**//--> Use the "Gettting Started Project" to fine-tune Threshold Value beyond default setting.**

**//--> Otherwise leave the default "550" value.**

**byte heart1[8] = {B11111, B11111, B11111, B11111, B01111, B00111, B00011, B00001};**

**byte heart2[8] = {B00011, B00001, B00000, B00000, B00000, B00000, B00000, B00000};**

**byte heart3[8] = {B00011, B00111, B01111, B11111, B11111, B11111, B11111, B01111};**

**byte heart4[8] = {B11000, B11100, B11110, B11111, B11111, B11111, B11111, B11111};**

**byte heart5[8] = {B00011, B00111, B01111, B11111, B11111, B11111, B11111, B11111};**

**byte heart6[8] = {B11000, B11100, B11110, B11111, B11111, B11111, B11111, B11110};**

**byte heart7[8] = {B11000, B10000, B00000, B00000, B00000, B00000, B00000, B00000};**

**byte heart8[8] = {B11111, B11111, B11111, B11111, B11110, B11100, B11000, B10000};**

**//----------------------------------------**

**int Instructions\_view = 500; //--> Variable for waiting time to display instructions on LCD.**

**PulseSensorPlayground pulseSensor; //--> Creates an instance of the PulseSensorPlayground object called "pulseSensor"**

**//--------------------------------------------------------------------------------void setup**

**void setup() {**

**Serial.begin(9600);//--> Set's up Serial Communication at certain speed.**

**lcd.begin(); //--> Initializes the interface to the LCD screen, and specifies the dimensions (width and height) of the display**

**//----------------------------------------Create a custom character (glyph) for use on the LCD**

**lcd.createChar(1, heart1);**

**lcd.createChar(2, heart2);**

**lcd.createChar(3, heart3);**

**lcd.createChar(4, heart4);**

**lcd.createChar(5, heart5);**

**lcd.createChar(6, heart6);**

**lcd.createChar(7, heart7);**

**lcd.createChar(8, heart8);**

**//----------------------------------------**

**lcd.setCursor(0,0);**

**lcd.print(" HeartBeat Rate ");**

**lcd.setCursor(0,1);**

**lcd.print(" Monitoring ");**

**//----------------------------------------Configure the PulseSensor object, by assigning our variables to it.**

**pulseSensor.analogInput(PulseWire);**

**pulseSensor.blinkOnPulse(LED\_3); //--> auto-magically blink Arduino's LED with heartbeat.**

**pulseSensor.setThreshold(Threshold);**

**//----------------------------------------**

**//----------------------------------------Double-check the "pulseSensor" object was created and "began" seeing a signal.**

**if (pulseSensor.begin()) {**

**Serial.println("We created a pulseSensor Object !"); //--> This prints one time at Arduino power-up, or on Arduino reset.**

**}**

**//----------------------------------------**

**delay(2000);**

**lcd.clear();**

**}**

**//--------------------------------------------------------------------------------**

**//--------------------------------------------------------------------------------void loop**

**void loop() {**

**int myBPM = pulseSensor.getBeatsPerMinute(); //--> Calls function on our pulseSensor object that returns BPM as an "int". "myBPM" hold this BPM value now.**

**//----------------------------------------Condition if the Sensor does not detect the heart rate / the sensor is not touched.**

**if (Instructions\_view < 500) {**

**Instructions\_view++;**

**}**

**if (Instructions\_view > 499) {**

**lcd.setCursor(0,0);**

**lcd.print("Put your finger ");**

**lcd.setCursor(0,1);**

**lcd.print("on the sensor ");**

**delay(1000);**

**lcd.clear();**

**delay(500);**

**}**

**//----------------------------------------**

**//----------------------------------------Constantly test to see if "a beat happened".**

**if (pulseSensor.sawStartOfBeat()) { //--> If test is "true", then the following conditions will be executed.**

**Serial.println("♥ A HeartBeat Happened ! "); //--> Print a message "a heartbeat happened".**

**Serial.print("BPM: "); //--> Print phrase "BPM: "**

**Serial.println(myBPM); //--> Print the value inside of myBPM.**

**//----------------------------------------Displays a "Heart" shape on the LCD.**

**lcd.setCursor(1,1);**

**lcd.write(byte(1));**

**lcd.setCursor(0,1);**

**lcd.write(byte(2));**

**lcd.setCursor(0,0);**

**lcd.write(byte(3));**

**lcd.setCursor(1,0);**

**lcd.write(byte(4));**

**lcd.setCursor(2,0);**

**lcd.write(byte(5));**

**lcd.setCursor(3,0);**

**lcd.write(byte(6));**

**lcd.setCursor(3,1);**

**lcd.write(byte(7));**

**lcd.setCursor(2,1);**

**lcd.write(byte(8));**

**//----------------------------------------**

**//----------------------------------------Displays the BPM value on the LCD.**

**lcd.setCursor(5,0);**

**lcd.print("Heart Rate");**

**lcd.setCursor(5,1);**

**lcd.print(": ");**

**lcd.print(myBPM);**

**lcd.print(" ");**

**lcd.print("BPM ");**

**//----------------------------------------**

**Instructions\_view = 0;**

**}**

**//----------------------------------------**

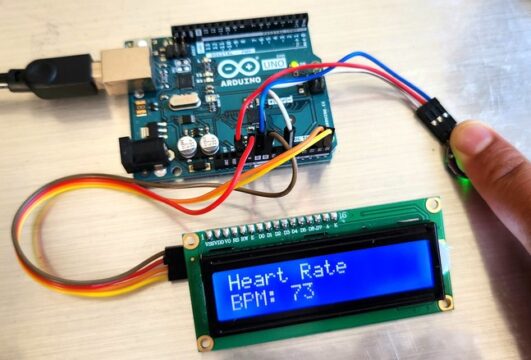
**delay(20); //--> considered best practice in a simple sketch.**

**}**

**//--------------------------------------------------------------------------------**

#### **Testing & Results**

Place your finger on Pulse Sensor and you may see the BPM Value displayed on LCD Screen.

[](https://how2electronics.com/wp-content/uploads/2023/09/Pulse-Sensor-Heart-Rate-Measurement.jpg)

the readings may not be immediately accurate. To get reliable results, try to keep your finger as steady as possible while waiting.

1. **Literature survey**
2. **Methodology**
3. **Observation Done**

Placed our finger on Pulse Sensor and we have seen the BPM Value displayed on LCD. Screen.

1. **Photographs taken in company.**

**NA**

1. **Complete analysis of Project done**

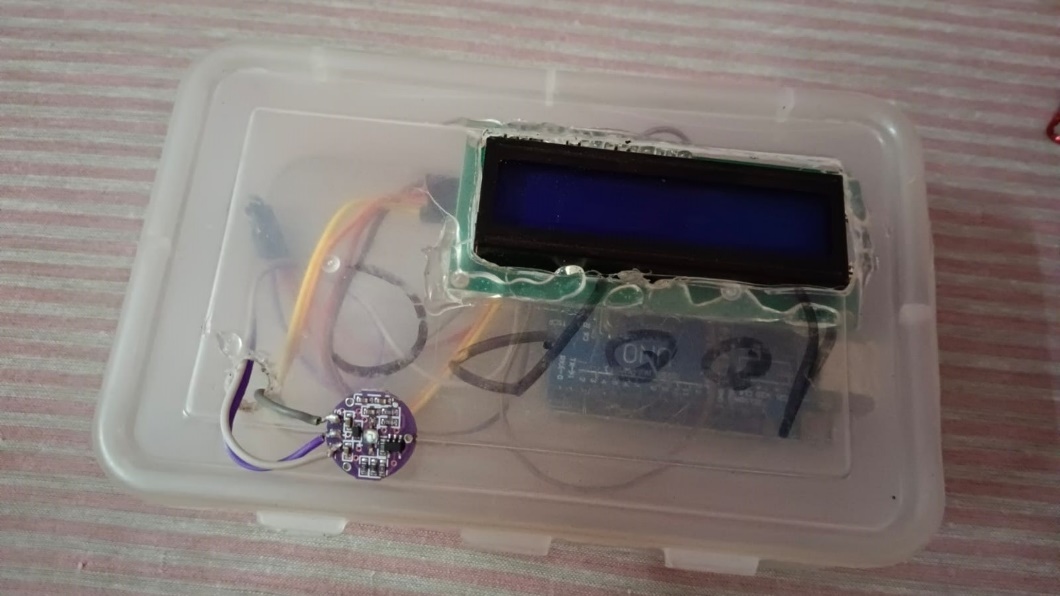
Placed our finger on Pulse Sensor and we have seen the BPM Value displayed on LCD. Screen.

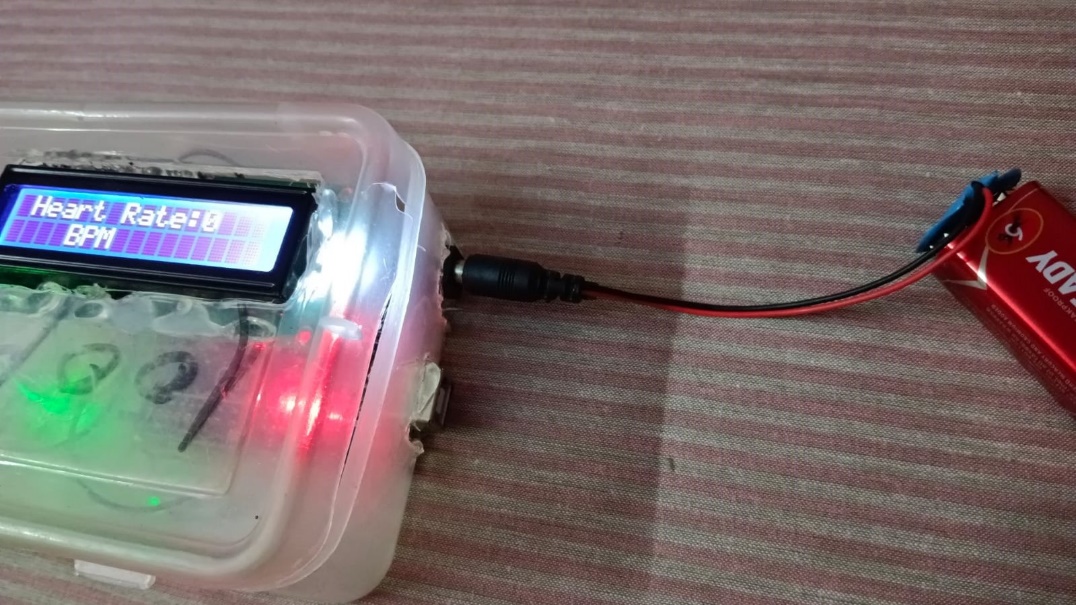
1. **Technology Used**

Hardware, we have used Arduino & Pulse Sensor, LCD & I2C Module. Arduino standard library coding we have used & uploaded from examples.

1. **Project photos**

**Completed Project Images:**







1. **Conclusion**

In this tutorial, we’ve walked you through the step-by-step process of creating a **Pulse Rate Monitor** using an **Arduino** and a **Pulse Sensor**. This project is not only educational but also highly practical, offering a cost-effective solution for **monitoring your heart rate** in real-time. Whether you’re a healthcare professional, a fitness enthusiast, or simply curious about electronics, this project provides valuable insights into both **health monitoring** and **Arduino programming**.

1. **References**
   * <https://techatronic.com/heart-beat-sensor-using-arduino-bpm-monitor/>
   * <https://www.technolabcreation.com/heartbeat-sensor-using-arduino-heart-rate-monitor/>
   * <https://how2electronics.com/pulse-rate-bpm-monitor-arduino-pulse-sensor/>

**PO & PSO Attainment**

| **PO.No** | **Graduate Attribute** | **Attained** | **Justification** |
| --- | --- | --- | --- |
| **PO 1** | **Engineering knowledge** | Yes | Gained knowledge in the field Electronics basic components & assembly. |
| **PO 2** | **Problem analysis** | Yes | Analysis of various problems were done. |
| **PO 3** | **Design/Development of solutions** | Yes | The solution for the problems were designed and developed. |
| **PO 4** | **Conduct investigations of complex problems** | Yes | Complex problems were investigated and solved during project. |
| **PO 5** | **Modern Tool usage** | Yes | Modern tools such as Arduino, sensors were used in this project. |
| **PO 6** | **The Engineer and society** | No | **-** |
| **PO 7** | **Environment and Sustainability** | No | **-** |
| **PO 8** | **Ethics** | Yes | Transparency and honesty in handling data. |
| **PO 9** | **Individual and team work** | Yes | This is a team project. Discussed and completed as team work. |
| **PO 10** | **Communication** | Yes | During project work, each and every steps, issues discussed withing team and solved. |
| **PO 11** | **Project management and finance** | Yes | As a team project, everything planned and executed with planned manner. |
| **PO 12** | **Life-long learning** | Yes | Learned knowledge of designing a electronic project from scratch |

| **PSO.No** | **Graduate Attribute** | **Attained** | **Justification** |
| --- | --- | --- | --- |
| **PSO 1** | To analyze, design and develop solutions by applying the concepts of Robotics for societal and industrial needs. | **Yes** | To create a solution for societal needs. |
| **PSO 2** | To create innovative ideas and solutions for real time problems in Manufacturing sector by adapting the automation tools and technologies. | **Yes** | To create solutions for societal needs. |