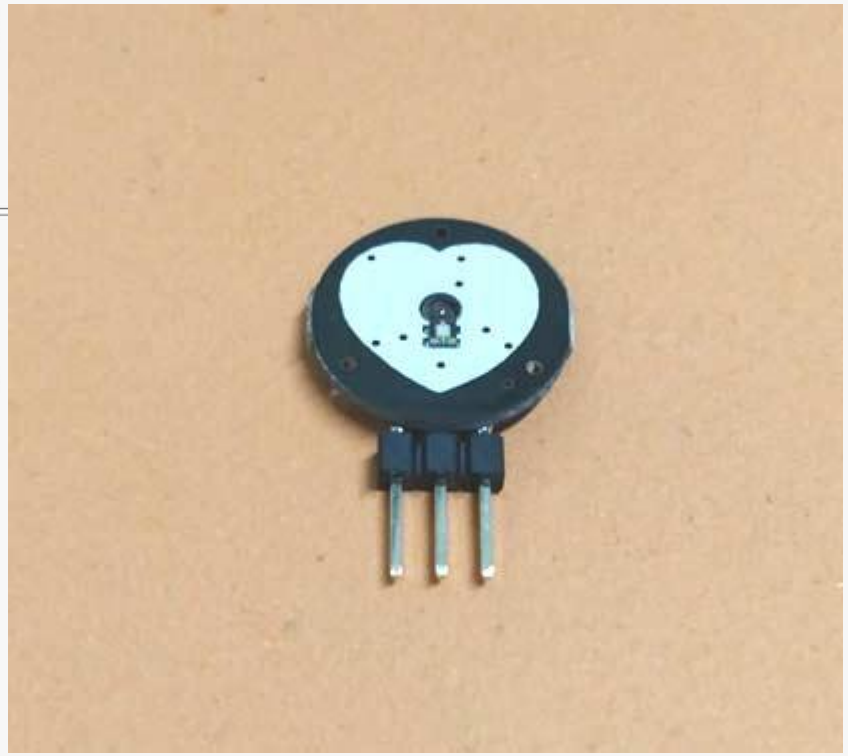
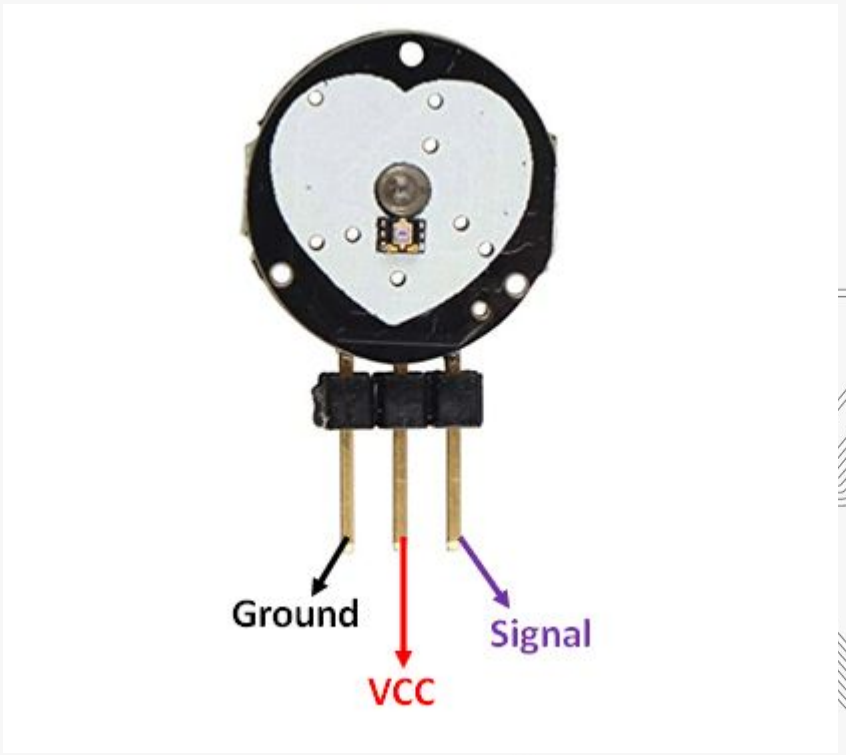
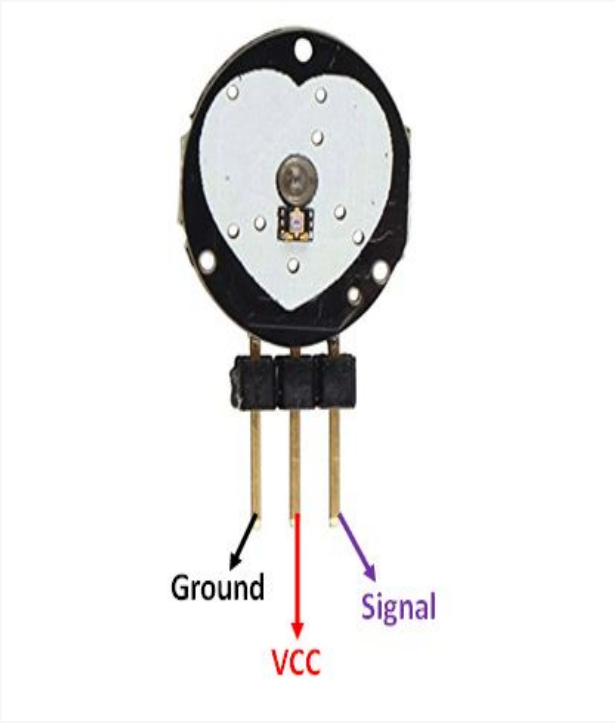
Interfacing of Pulse rate sensor





Pulse rate sensor

Pulse rate sensor is a well-designed plug-and-play **heart-rate sensor** for Arduino. The **sensor** clips onto a fingertip or earlobe and plugs right into Arduino with some jumper cables. It also includes an open-source monitoring app that graphs your **pulse** in real time. It can be used by students, artists, athletes, makers, and game & mobile developers who want to easily incorporate live heartrate data into their projects.



Working of Pulse rate sensor

- The working of the **Pulse/Heart beat sensor** is very simple. The sensor has two sides, on one side the LED is placed along with an ambient light sensor and on the other side we have some circuitry. This circuitry is responsible for the amplification and noise cancellation work. The LED on the front side of the sensor is placed over a vein in our human body. This can either be your Finger tip or you ear tips, but it should be placed directly on top of a vein.
- Now the LED emits light which will fall on the vein directly. The veins will have blood flow inside them only when the heart is pumping, so if we monitor the flow of blood we can monitor the heart beats as well. If the flow of blood is detected then the ambient light sensor will pick up more light since they will be reflect ted by the blood, this minor change in received light is analysed over time to determine our heart beats.

How to use Pulse rate sensor

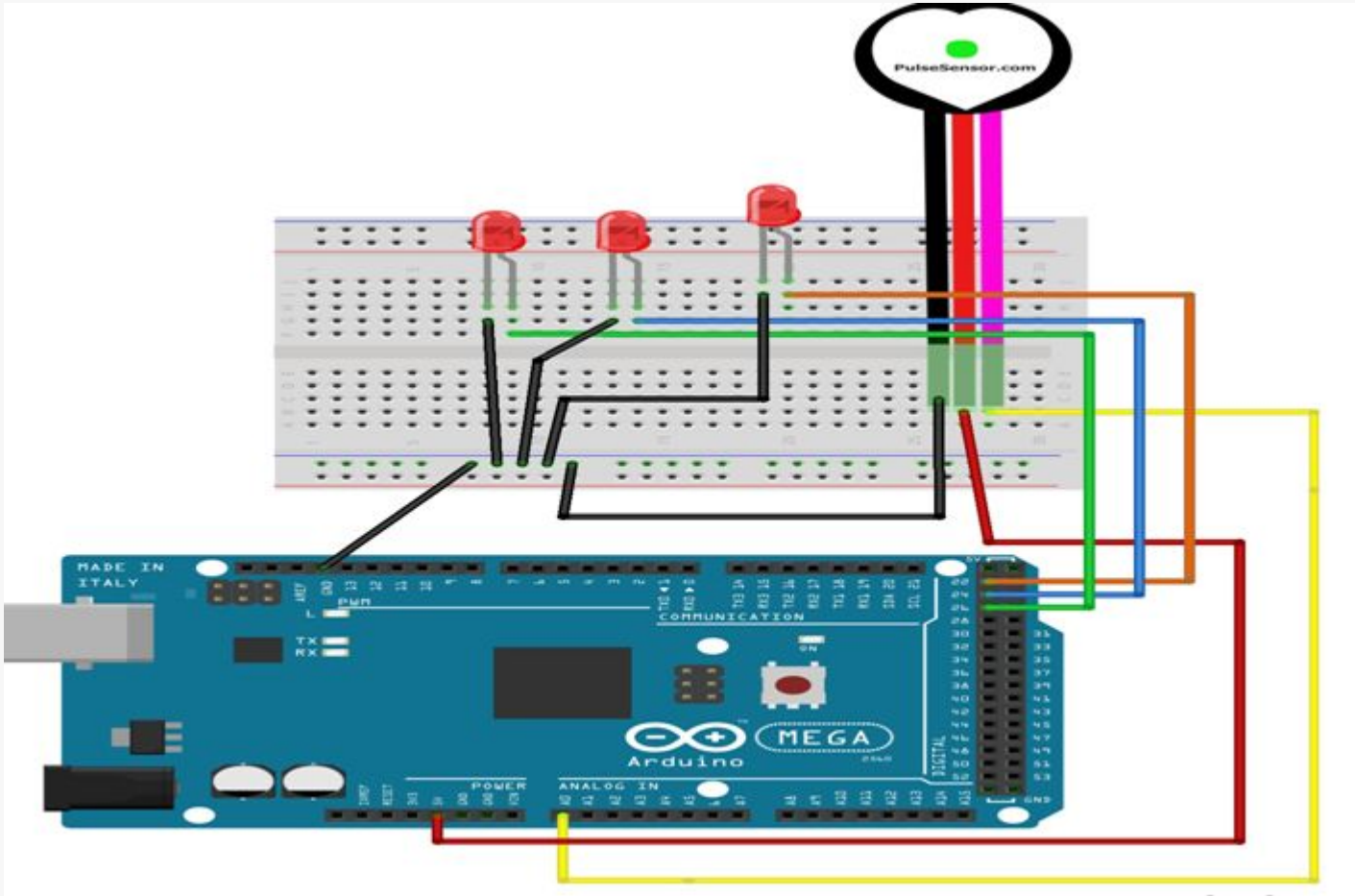
- **Using the Pulse sensor** is straight forward, but positioning it in the right way matters. Since all the electronics on the sensor are directly exposed it is also recommended to cover the sensor with hot glue, vinyl tape or other non conductive materials. Also it is not recommended to handle these sensors with wet hands. The flat side of the sensor should be placed on top of the vein and a slightly press should be applied on top of it, normally clips or Velcro tapes are used to attain this pressure.
- To use the sensor simply power it using the Vcc and ground pins, the sensor can operate both at +5V or 3.3V system.

Components required

- Arduino Mega
- Pulse sensor
- LEDs
- Breadboard
- Jumper wires

ABLEDUCATION™
ACTIVITY BASED LEARNING

Connection Diagram



ABL EDUCATION™
ACTIVITY BASED LEARNING

Connections

1. Connect S(signal) pin of pulse sensor with Ao pin of Arduino.
2. Connect Vcc pin of pulse sensor with (+5V) of Arduino.
3. Connect GND pin of pulse sensor with GND of Arduino.
4. Connect LED's positive to 22 pin of Arduino and LED's negative with GND of Arduino.
5. Connect another LED's positive to 24 pin of Arduino and LED's negative with GND of Arduino.
6. Connect third LED's positive to 26 pin of Arduino and LED's negative with GND of Arduino.

ABLEDUCATION™
ACTIVITY BASED LEARNING

Code

Interfacing_of_pulse_sensor | Arduino 1.8.19

File Edit Sketch Tools Help

Interfacing_of_pulse_sensor

```
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
pinMode(A0,INPUT);
pinMode(22,OUTPUT);
pinMode(24,OUTPUT);
pinMode(26,OUTPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
//
delay(200);
int data=analogRead(A0);
Serial.println(data);
if(data>700){
  digitalWrite(22,1);
  digitalWrite(24,0);
    digitalWrite(26,0);
 // delay(200);
}else if(data<250){
  digitalWrite(22,0);
  digitalWrite(24,1);
    digitalWrite(26,0);
```

ABL EDUCATION™
ACTIVITY BASED LEARNING

Code

Interfacing_of_pulse_sensor | Arduino 1.8.19

File Edit Sketch Tools Help

Interfacing_of_pulse_sensor

```
void loop() {
  // put your main code here, to run repeatedly:
//
delay(200);
int data=analogRead(A0);
Serial.println(data);
if(data>700){
  digitalWrite(22,1);
  digitalWrite(24,0);
    digitalWrite(26,0);
 // delay(200);
}else if(data<250){
  digitalWrite(22,0);
  digitalWrite(24,1);
    digitalWrite(26,0);
  //delay(200);
}
  else{
  digitalWrite(24,0);
  digitalWrite(22,0);
    digitalWrite(26,1);

}}
```

Project Link : https://youtu.be/SP9iHq79FcA