# Monitor the Heart Rate using Pulse Sensor and Arduino

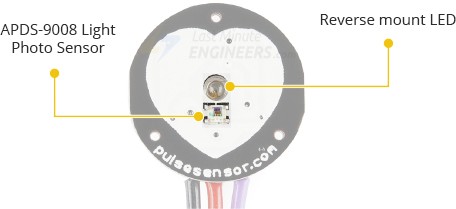
The Pulse Sensor is a well-designed low-power plug-and-play heart-rate sensor for the Arduino. Anyone who wants to incorporate real-time heart-rate data into their work—students, artists, athletes, makers, and game and mobile developers—can benefit from it.

The best part is that this sensor plugs right into Arduino and easily clips onto a fingertip or earlobe. It is also super small (button-shaped) and has holes for sewing into fabric.



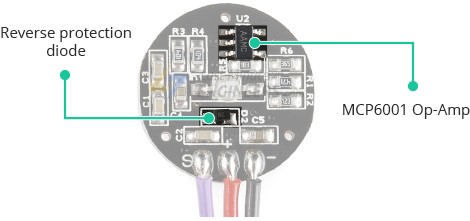
## Hardware Overview

The front of the sensor, with the heart logo, is where you put your finger. You’ll also notice a tiny circular opening through which the Kingbright’s reverse mounted green LED shines.



Just beneath the circular opening is a small ambient light photo sensor – [APDS-9008](https://docs.broadcom.com/doc/AV02-1169EN) from Avago. This sensor is similar to the ones used in cell phones, tablets, and laptops to adjust the screen’s brightness based on the ambient lighting conditions.

On the back of the module are an MCP6001 Op-Amp from Microchip and a few resistors and capacitors that make up the R/C filter network. Additionally, there is a reverse protection diode to prevent damage in the event that the power leads are accidentally reversed.



The module requires a DC power supply ranging from 3.3 to 5V and draws less than 4mA of current.

### Technical Specifications

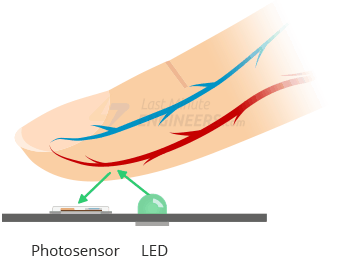
Here are the technical specifications:

| **Specification** | **Value** |
| --- | --- |
| Sensor Type | Pulse Sensor |
| Measurement Method | Photoplethysmography (PPG) |
| Output Type | Analog Voltage or Digital |
| Operating Voltage | 3.3V - 5V |
| Output Signal Range | 0 - 5V (Analog) or High/Low |
| Sampling Rate | Typically 100 Hz - 1000 Hz |
| Compatibility | Arduino, Raspberry Pi, etc. |

|  |  |  |
| --- | --- | --- |
|  |  |
|  |  |
|  |  |  |
|  |  |
|  |  |  |
|  |  |

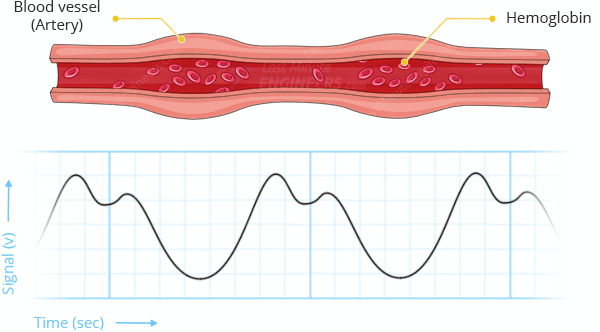
## How Does a Pulse Sensor Work?

The theory behind optical heart-rate sensors is very simple. If you’ve ever shined a flashlight through your fingers and observed your heartbeat pulsing, the concept of optical heart-rate pulse sensors can be easily grasped.



A pulse sensor, like any other optical heart-rate sensor, works by shining a green light (~ 550nm) on the finger and measuring the amount of reflected light with a photosensor.

This optical pulse detection technique is known as a [Photoplethysmogram](https://en.wikipedia.org/wiki/Photoplethysmogram).



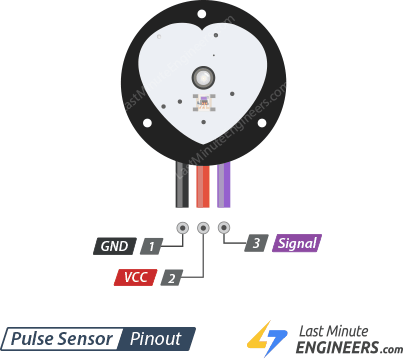
The oxygenated hemoglobin in arterial blood has the property of absorbing green light. The redder the blood (the higher the hemoglobin), the greater the absorption of green light. With each heartbeat, blood is pumped through the finger, causing a change in the amount of reflected light, which in turn produces a waveform at the photosensor’s output.

As you keep shining light and taking photosensor readings, you quickly begin to obtain a heart-beat pulse reading.

This signal from the photosensor is typically small and noisy; therefore, it is passed through an R/C filter network and then amplified with an Op-Amp to create a signal that is significantly larger, cleaner, and easier to detect.

## Pulse Sensor Pinout

The sensor comes with a 24″ flat ribbon cable with three male header connectors. The pinout is shown in the figure below.



S (Signal) is the signal output. Connects to analog input of an Arduino.

+ (VCC) is the VCC pin. Connects to 3.3 or 5V.

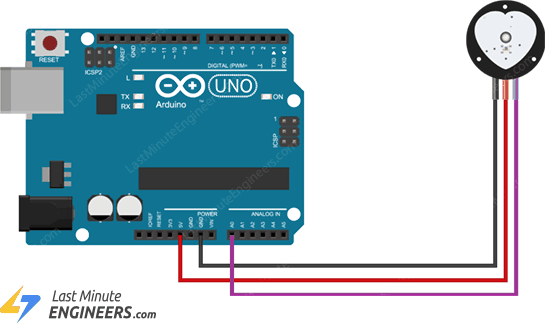
– (GND) is the Ground pin.

## Wiring a Pulse Sensor to an Arduino

Connecting the Pulse Sensor to an Arduino is a breeze. You only need to connect three wires: two for power and one for reading the sensor value.

The module can be supplied with either 3.3V or 5V. Positive voltage is connected to ‘+,’ while ground is connected to ‘-.’ The third ‘S’ wire is the analog signal output from the sensor, which will be connected to the Arduino’s A0 analog input.

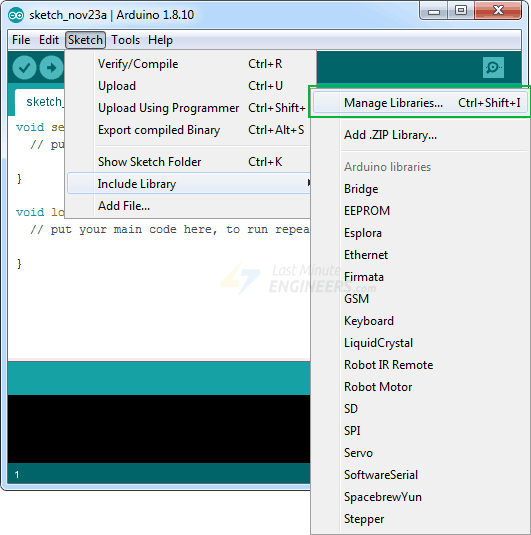
The following is the wiring diagram for the Pulse Sensor experiments:



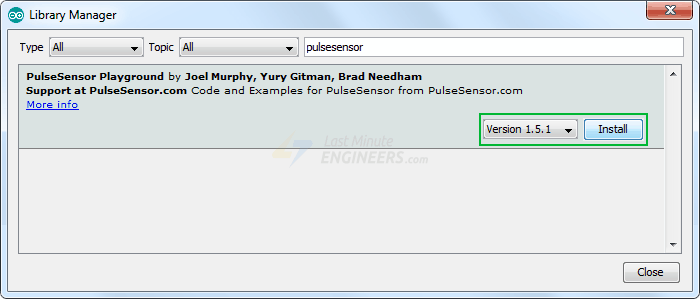
## Library Installation

To run the following sketches, you must first install the ‘PulseSensor Playground’ library.

To install the library, navigate to Sketch > Include Library > Manage Libraries… Wait for the Library Manager to download the libraries index and update the list of installed libraries.



Filter your search by entering ‘pulsesensor’.There should only be a single entry. Click on that and then choose Install.

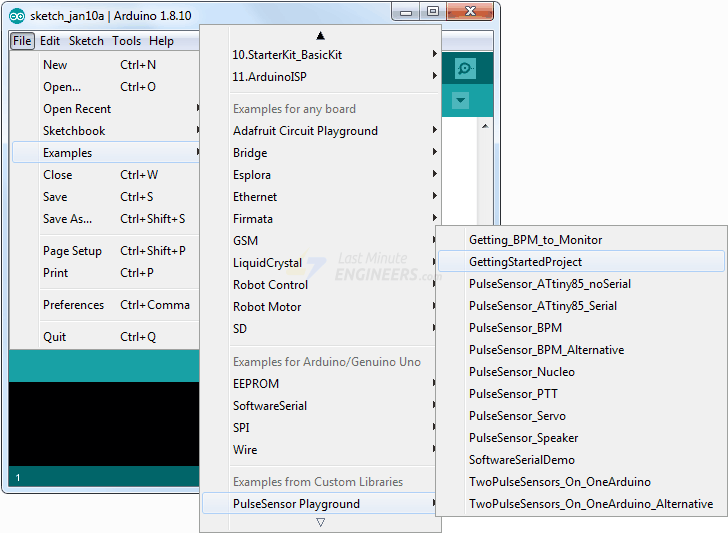


## PulseSensor Example Sketches

The PulseSensor library includes several example sketches. We’ll go over a few of them here, but you can also experiment with the others.

To access the example sketches, navigate to File > Examples > PulseSensor Playground.

You will see a selection of example sketches. You can choose any of them to load the sketch into your IDE. Let’s start off with the GettingStartedProject.



## Arduino Example 1 – Blink with the Heartbeat

Load the GettingStartedProject sketch from the example sketches into your Arduino IDE. This is a basic Arduino sketch. Upload the code to your Arduino and clip the sensor to your earlobe or fingertip. You should see the Arduino’s onboard LED blink in time with your heartbeat!

**CODE:**

int const PULSE\_SENSOR\_PIN = 0; // 'S' Signal pin connected to A0

int Signal; // Store incoming ADC data. Value can range from 0-1024

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

void setup()

{

pinMode(LED\_BUILTIN,OUTPUT); // Built-in LED will blink to your heartbeat

Serial.begin(9600); // Set comm speed for serial plotter window

}

void loop()

{

Signal = analogRead(PULSE\_SENSOR\_PIN); // Read the sensor value

Serial.println(Signal); // Send the signal value to serial plotter

if(Signal > Threshold){ // If the signal is above threshold, turn on the LED

digitalWrite(LED\_BUILTIN,HIGH);

} else

{

digitalWrite(LED\_BUILTIN,LOW); // Else turn off the LED

}

delay(10);

}

Where we connect the pulse sensor pin to A0 and assign it as INPUT in otherhand we assign the buld-in LED as OUTPUT.ans assign the threshold value to 550.if the output value is greater then threshold value then the build-in LED will blink(i.e:HIGH) else it will be LOW.