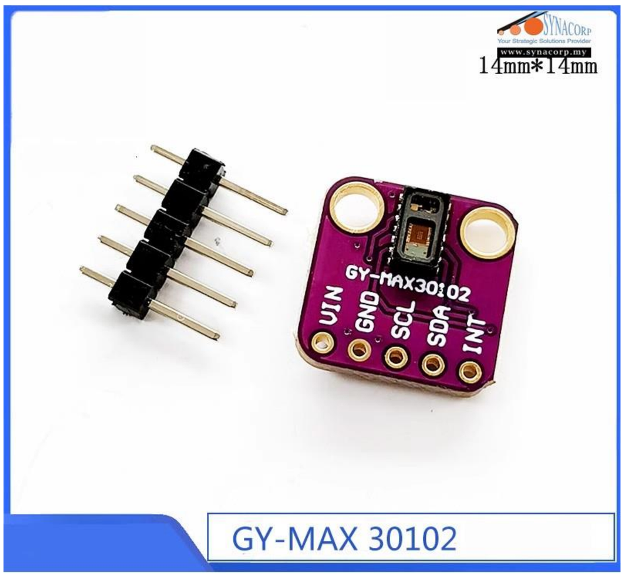
REF: B31-MAX30102

MAX30102 Heart Rate Sensor Module Pulse Detection Blood Oxygen Concentration For Rduino 1.8V 3.3V 5V Ultra-Low Power



Description

The module is based on the MAX30102 chip and communicates via the I2C interface. MAX30102 is a complete system solution that facilitates the design process of mobile devices and wearable devices. The MAX30102 is integrated pulse oximetry and a heart-rate monitor module. It includes internal LEDs, photodetectors, optical elements, and low-noise electronics with ambient light rejection. The MAX30102 operates on a single 1.8V power supply and a separate 3.3V power supply for the internal LEDs. Communication is through a standard I2C-compatible interface. The module can be shut down through software with zero standby current, allowing the power rails to remain powered at all times

Specifications

- **Power supply**: 3.3V to 5.5V
- **Type**: Optical heart rate and pulse oximeter sensor
- **Functionality**: Measures heart rate (HR) and blood oxygen saturation (SpO2) levels
- **Red LED Wavelength**: 660nm
- **IR LED Wavelength**: 880nm
- **Interface**: I2C (TWI) communication interface
- **Operating Temperature Range**: -40°C to +85°C
- **Temperature Accuracy:** ±1 ˚C

Pin connection

The MAX30102 module brings out the following connections.

VIN - is the power pin. You can connect it to 3.3V or 5V output from your Arduino.

GND - is the ground.

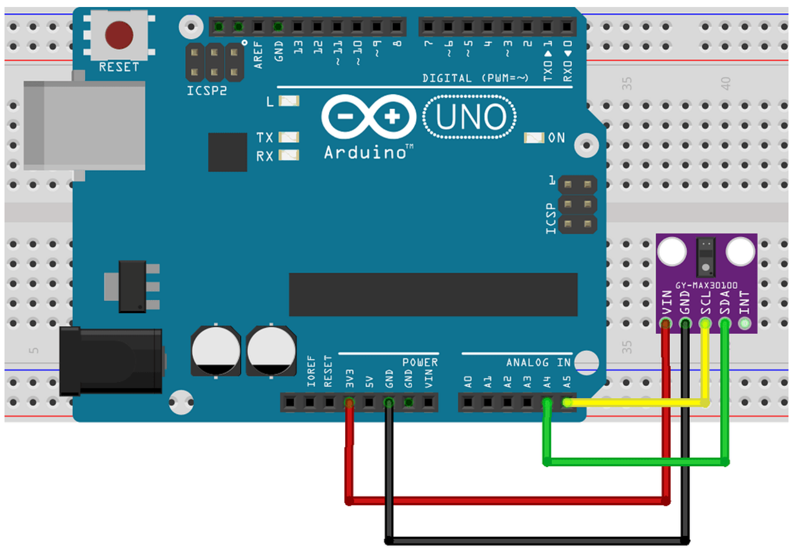
SDA - is the I2C data pin, connect to your Arduino's I2C data line

SCL - is the I2C clock pin, connect to your Arduino's I2C clock line.

ARDUINO PIN	MAX30102 PIN
3.3V	VIN
GND	GND
A4	SDA
A5	SCL

Circuit diagram

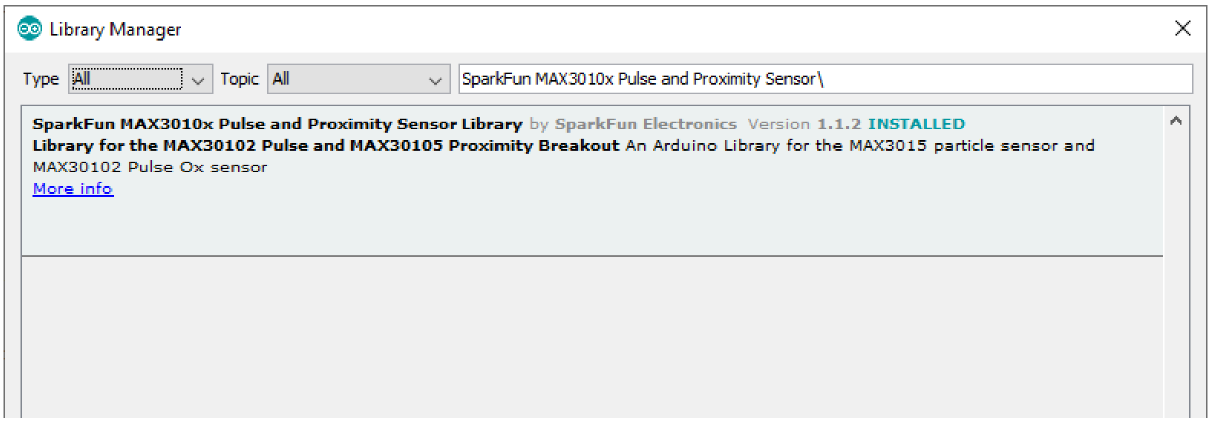
Start by connecting the VCC pin to the power supply, 3V-5V is fine. Use the same voltage that your microcontroller logic is based off of. For most Arduinos, that is 5V. For 3.3V logic devices, use 3.3V. Now connect GND to common ground. Connect the SCL pin to the I2C clock pin and the SDA pin to the I2C data pin on your Arduino. Note that each Arduino Board has different I2C pins which should be connected accordingly. On the Arduino boards with the R3 layout, the SDA (data line) and SCL (clock line) are on the pin headers close to the AREF pin. They are also known as A5 (SCL) and A4 (SDA).



Libraries

There are several libraries available for the MAX30102 sensor. However in our example, we are using the one by SparkFun Electronics. This library exposes most of the features of the MAX30102 and offers simple and easy to use functions to calculate pulse rate and SpO2. You can download this library from within the Arduino IDE Library Manager. To install the library navigate to the Sketch > Include Library > Manage Libraries… Wait for Library Manager to download libraries index and update list of installed libraries.

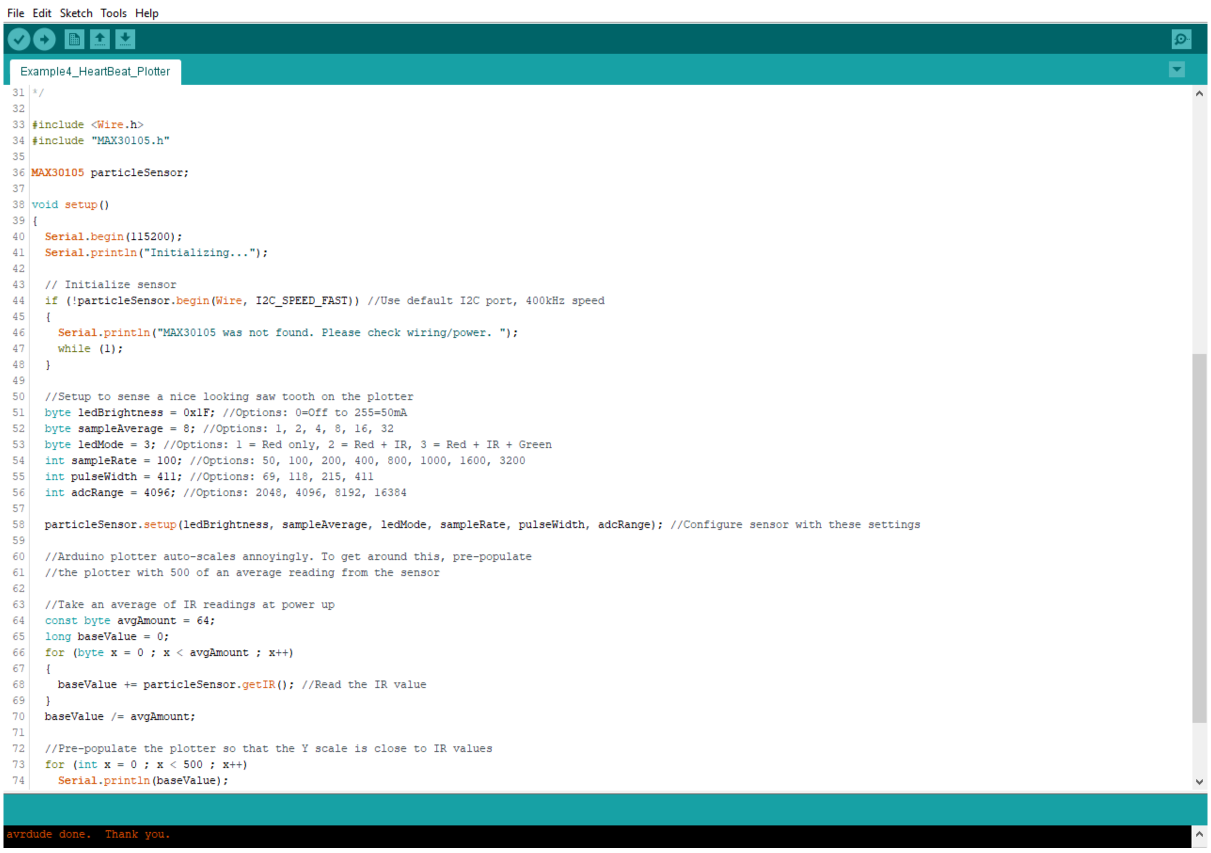
Filter your search by typing **MAX3010x**. Look for **SparkFun MAX3010x Pulse and Proximity Sensor Library**. Click on that entry, and then select Install.



SparkFun MAX3010x Pulse and Proximity Sensor\

SparkFun MAX3010x Pulse and Proximity Sensor Library by SparkFun Electronics Version 1.1.2 INSTALLED
Library for the MAX30102 Pulse and MAX30105 Proximity Breakout An Arduino Library for the MAX3015 particle sensor and MAX30102 Pulse Ox sensor
More info

Coding

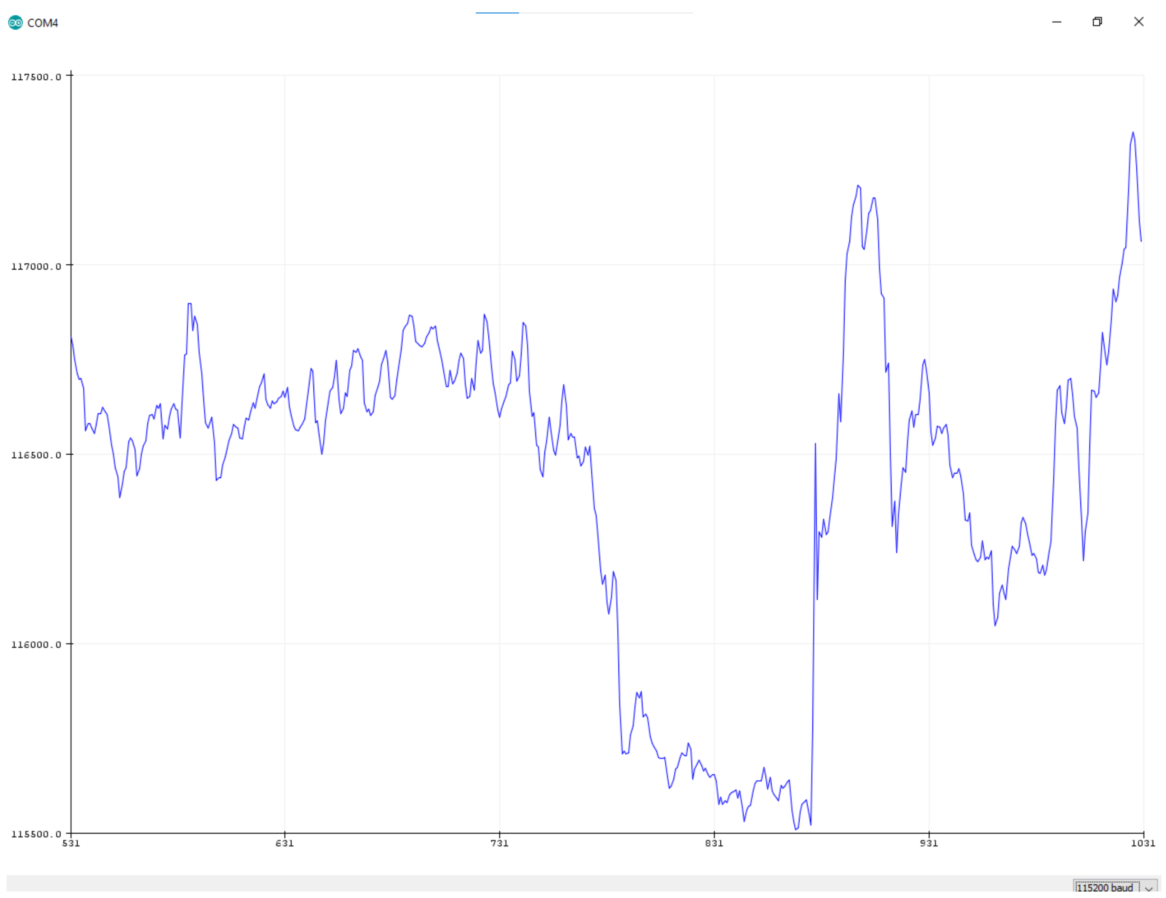


```
*/

#include <Wire.h>
#include "MAX30105.h"

MAX30105 particleSensor;

void setup()
{
  Serial.begin(115200);
  Serial.println("Initializing...");

  // Initialize sensor
  if (!particleSensor.begin(Wire, I2C_SPEED_FAST)) //Use default I2C port, 400kHz speed
  {
    Serial.println("MAX30105 was not found. Please check wiring/power. ");
    while (1);
  }

  //Setup to sense a nice looking saw tooth on the plotter
  byte ledBrightness = 0x1F; //Options: 0=Off to 255=50mA
  byte sampleAverage = 8; //Options: 1, 2, 4, 8, 16, 32
  byte ledMode = 3; //Options: 1 = Red only, 2 = Red + IR, 3 = Red + IR + Green
  int sampleRate = 100; //Options: 50, 100, 200, 400, 800, 1000, 1600, 3200
  int pulseWidth = 411; //Options: 69, 118, 215, 411
  int adcRange = 4096; //Options: 2048, 4096, 8192, 16384

  particleSensor.setup(ledBrightness, sampleAverage, ledMode, sampleRate, pulseWidth, adcRange); //Configure sensor with these settings

  //Arduino plotter auto-scales annoyingly. To get around this, pre-populate
  //the plotter with 500 of an average reading from the sensor

  //Take an average of IR readings at power up
  const byte avgAmount = 64;
  long baseValue = 0;
  for (byte x = 0 ; x < avgAmount ; x++)
  {
    baseValue += particleSensor.getIR(); //Read the IR value
  }
  baseValue /= avgAmount;

  //Pre-populate the plotter so that the Y scale is close to IR values
  for (int x = 0 ; x < 500 ; x++)
    Serial.println(baseValue);
```

```
avrdude done.  Thank you.
```

Result

In the Arduino IDE, choose Tools > Serial Plotter. You should see a wave similar to the image below, when you swipe your hand over the sensor.


COM4
115200 baud