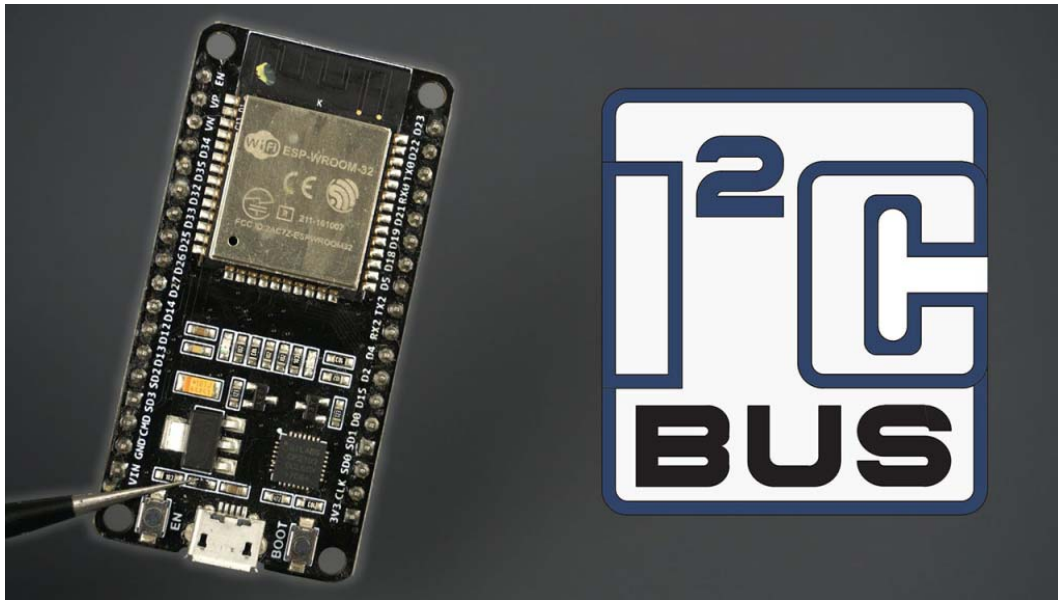


ESP32 I2C Communication: Set Pins, Multiple Bus Interfaces and Peripherals (Arduino IDE)

The ESP32 has two I2C bus interfaces that can serve as I2C master or slave. In this tutorial we'll take a look at the I2C communication protocol with the ESP32 using Arduino IDE: how to choose I2C pins, connect multiple I2C devices to the same bus and how to use the two I2C bus interfaces.



In this tutorial, we'll cover the following concepts:

- [Connecting I2C Devices with ESP32](#)
- [Scan I2C Address with ESP32](#)
- [Use Different I2C Pins with ESP32 \(change default I2C pins\)](#)
- [ESP32 with Multiple I2C Devices](#)
 - [same bus, different addresses](#)
 - [same address](#)
- [ESP32 Using Two I2C Bus Interfaces](#)

We'll program the ESP32 using Arduino IDE, so before proceeding with this tutorial you should have the ESP32 add-on installed in your Arduino IDE. Follow the next tutorial to install the ESP32 on the Arduino IDE, if you haven't already.

- [Installing the ESP32 Board in Arduino IDE \(Windows, Mac OS X, and Linux instructions\)](#)

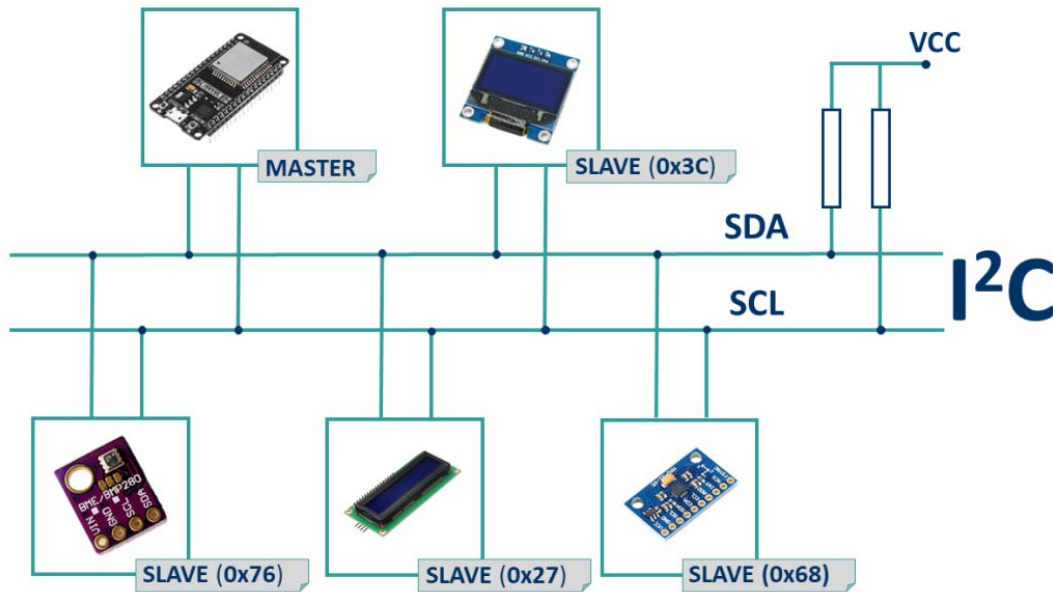
Introducing ESP32 I2C Communication Protocol

I²C means Inter Integrated Circuit (it's pronounced I-squared-C), and it is a synchronous, multi-master, multi-slave communication protocol. You can connect :

- **multiple slaves to one master:** for example, your ESP32 reads from a BME280 sensor using I2C and writes the sensor readings in an I2C OLED display.
- **multiple masters controlling the same slave:** for example, two ESP32 boards writing data to the same I2C OLED display.



We use this protocol many times with the ESP32 to communicate with external devices like sensors and displays. In these cases, the ESP32 is the master chip and the external devices are the slaves.



We have several tutorials with the ESP32 interfacing with I2C devices:

- [0.96 inch I2C OLED display with ESP32](#)
- [ESP32 Built-in OLED Board](#)
- [I2C LCD Display with ESP32](#)
- [BMP180 with ESP32](#)
- [BME280 with ESP32](#)

ESP32 I2C Bus Interfaces

The ESP32 supports I2C communication through its two I2C bus interfaces that can serve as I2C master or slave, depending on the user's configuration. Accordingly to the ESP32 datasheet, the I2C interfaces of the ESP32 supports:

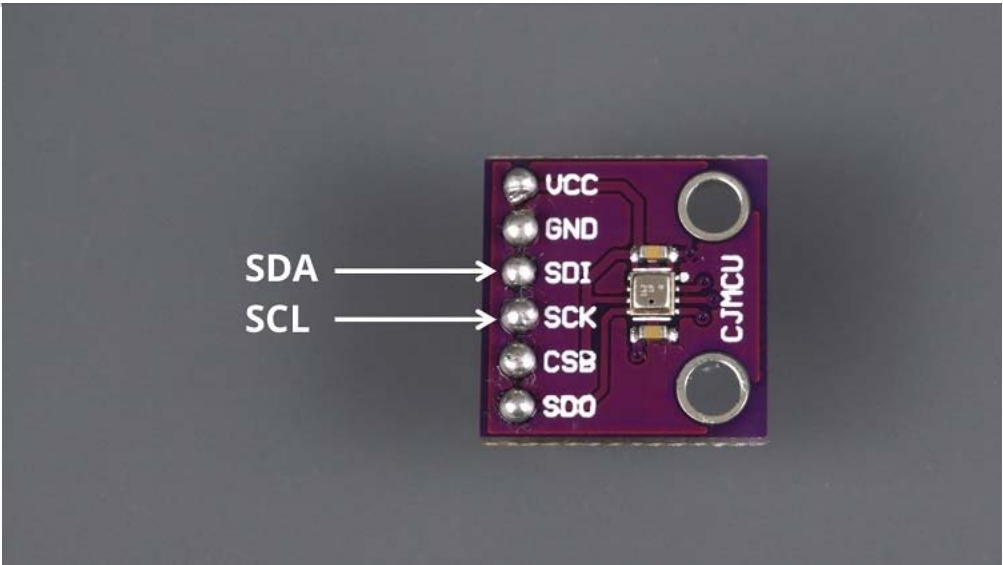
- Standard mode (100 Kbit/s)
- Fast mode (400 Kbit/s)
- Up to 5 MHz, yet constrained by SDA pull-up strength
- 7-bit/10-bit addressing mode
- Dual addressing mode. Users can program command registers to control I²C interfaces, so that they have more flexibility

Connecting I2C Devices with ESP32

I2C communication protocol uses two wires to share information. One is used for the clock signal (**SCL**) and the other is used to send and receive data (**SDA**).

Note: in many breakout boards, the SDA line may also be labeled as SDI and the SCL line as SCK.





The SDA and SCL lines are active low, so they should be pulled up with resistors. Typical values are 4.7k Ohm for 5V devices and 2.4k Ohm for 3.3V devices.

Most sensors we use in our projects are breakout boards that already have the resistors built-in. So, usually, when you're dealing with this type of electronics components you don't need to worry about this.

Connecting an I2C device to an ESP32 is normally as simple as connecting GND to GND, SDA to SDA, SCL to SCL and a positive power supply to a peripheral, usually 3.3V (but it depends on the module you're using).

I2C Device	ESP32
SDA	SDA (default is GPIO 21)
SCL	SCL (default is GPIO 22)
GND	GND
VCC	usually 3.3V or 5V

When using the ESP32 with Arduino IDE, the default I2C pins are GPIO 22 (SCL) and GPIO 21 (SDA) but you can configure your code to use any other pins.

Recommended reading: [ESP32 GPIO Reference Guide](#)

Scan I2C Address with ESP32

With I2C communication, each slave on the bus has its own address, a hexadecimal number that allows the ESP32 to communicate with each device.

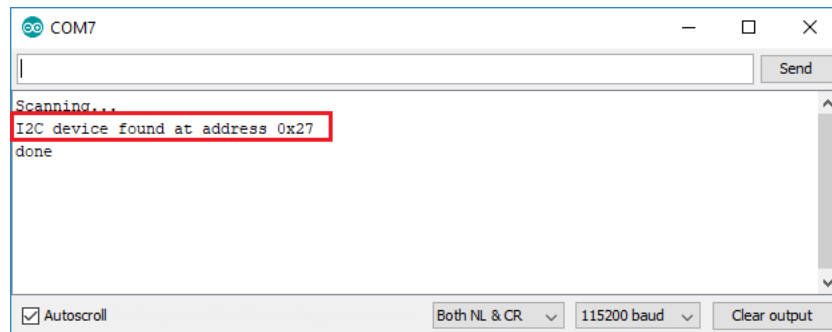
The I2C address can be usually found on the component's datasheet. However, if it is difficult to find out, you may need to run an I2C scanner sketch to find out the I2C address.

You can use the following sketch to find your devices' I2C address.

```
if (error == 0) {
  Serial.print("I2C device found at address 0x");
  if (address < 16) {
    Serial.print("0");
  }
  Serial.println(address, HEX);
  nDevices++;
}
else if (error == 4) {
  Serial.print("Unknown error at address 0x");
  if (address < 16) {
    Serial.print("0");
  }
  Serial.println(address, HEX);
}
}
if (nDevices == 0) {
  Serial.println("No I2C devices found\n");
}
else {
  Serial.println("done\n");
}
delay(5000);
}
```

[View raw code](#)

You'll get something similar in your Serial Monitor. This specific example is for an [I2C LCD Display](#).



Use Different I2C Pins with ESP32 (change default I2C pins)

With the ESP32 you can set almost any pin to have I2C capabilities, you just need to set that in your code.

When using the ESP32 with the Arduino IDE, use the `Wire.h` library to communicate with devices using I2C. With this library, you initialize the I2C as follows:

```
Wire.begin(I2C_SDA, I2C_SCL);
```

So, you just need to set your desired SDA and SCL GPIOs on the `I2C_SDA` and `I2C_SCL` variables.



and it might be a bit tricky to select other
your own `Wire` instance when initializing the

library.

In those cases, you need to take a closer look at the `.cpp` library files and see how to pass your own `TwoWire` parameters.

For example, if you take a closer look at the [Adafruit BME280 library](#), you'll find out that you can pass your own `TwoWire` to the `begin()` method.

```

    /*!
    * @brief Initialise sensor with given parameters / settings
    * @param addr the I2C address the device can be found on
    * @param theWire the I2C object to use
    * @returns true on success, false otherwise
    */
    bool Adafruit_BME280::begin(uint8_t addr, TwoWire *theWire) {
        _i2caddr = addr;
        _wire = theWire;
        return init();
    }

```

So, the example sketch to read from the BME280 using other pins, for example `GPIO 33` as SDA and `GPIO 32` as SCL is as follows.

```

    //-----\,
    delay(delayTime);
}

void printValues() {
    Serial.print("Temperature = ");
    Serial.print(bme.readTemperature());
    Serial.println(" *C");

    // Convert temperature to Fahrenheit
    /*Serial.print("Temperature = ");
    Serial.print(1.8 * bme.readTemperature() + 32);
    Serial.println(" *F");*/

    Serial.print("Pressure = ");
    Serial.print(bme.readPressure() / 100.0F);
    Serial.println(" hPa");

    Serial.print("Approx. Altitude = ");
    Serial.print(bme.readAltitude(SEALEVELPRESSURE_HPA));
    Serial.println(" m");

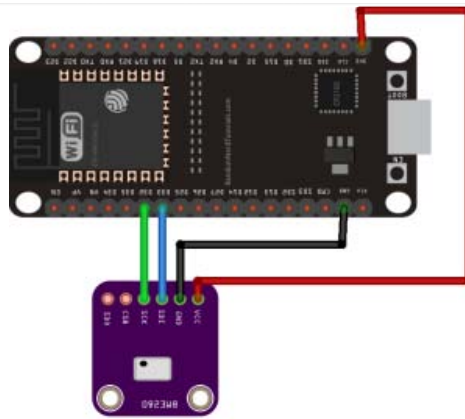
    Serial.print("Humidity = ");
    Serial.print(bme.readHumidity());
    Serial.println(" %");

    Serial.println();
}

```

View raw code





Click image to enlarge

Let's take a look at the relevant parts to use other I2C pins.

First, define your new I2C pins on the `I2C_SDA` and `I2C_SCL` variables. In this case, we're using `GPIO 33` and `GPIO 32`.

```
#define I2C_SDA 33
#define I2C_SCL 32
```

Create a new `TwoWire` instance. In this case, it's called `I2CBME`. This simply creates an I2C bus.

```
TwoWire I2CBME = TwoWire(0);
```

In the `setup()`, initialize the I2C communication with the pins you've defined earlier. The third parameter is the clock frequency.

```
I2CBME.begin(I2C_SDA, I2C_SCL, 400000);
```

Finally, initialize a `BME280` object with your sensor address and your `TwoWire` object.

```
status = bme.begin(0x76, &I2CBME);
```

After this, you can use the usual methods on your `bme` object to request temperature, humidity and pressure.

Note: if the library you're using uses a statement like `wire.begin()` in its file, you may need to comment that line, so that you can use your own pins.

ESP32 with Multiple I2C Devices

As we've mentioned previously, each I2C device has its own address, so it is possible to have multiple I2C devices on the same bus.

Multiple I2C devices (same bus, different addresses)

When we have multiple devices with different addresses, it is trivial how to set them up:



- in the code, refer to each peripheral by its address;

Take a look at the following example that gets sensor readings from a BME280 sensor (via I2C) and displays the results on an I2C OLED display.

```

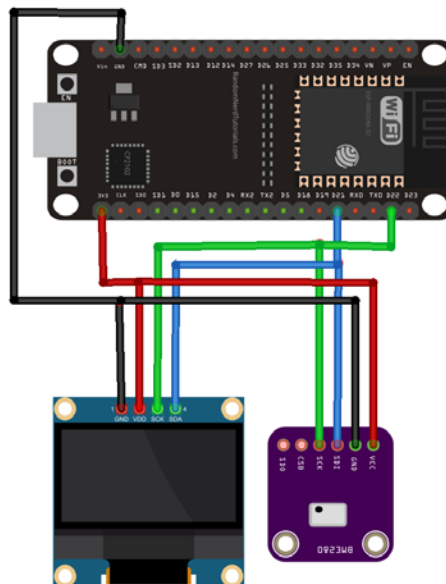
// display temperature
display.setTextSize(1);
display.setCursor(0,0);
display.print("Temperature: ");
display.setTextSize(2);
display.setCursor(0,10);
display.print(String(bme.readTemperature()));
display.print(" ");
display.setTextSize(1);
display.cp437(true);
display.write(167);
display.setTextSize(2);
display.print("C");

// display humidity
display.setTextSize(1);
display.setCursor(0, 35);
display.print("Humidity: ");
display.setTextSize(2);
display.setCursor(0, 45);
display.print(String(bme.readHumidity()));
display.print(" %");

display.display();

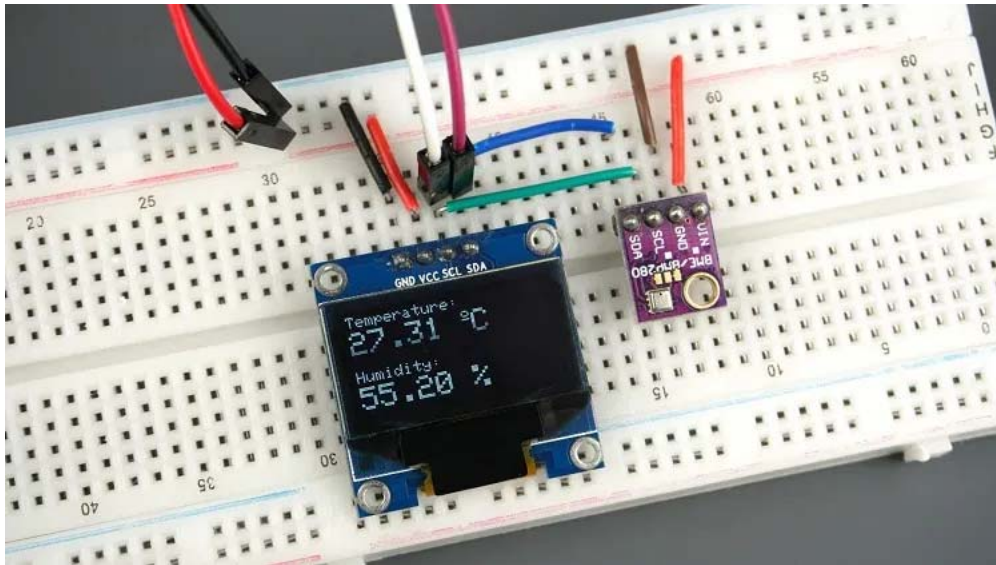
delay(1000);
}

```

[View raw code](#)

Because the OLED and the BME280 have different addresses, we can use the same SDA and SCL lines without any problem. The OLED display address is `0x3C` and the BME280 address is `0x76`.

```
if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {  
  Serial.println(F("SSD1306 allocation failed"));  
  for(;;);  
}  
  
bool status = bme.begin(0x76);  
if (!status) {  
  Serial.println("Could not find a valid BME280 sensor, check wiring!");  
  while (1);  
}
```



Recommended reading: [ESP32 OLED Display with Arduino IDE](#)

Multiple I2C devices (same address)

But, what if you have multiple peripherals with the same address? For example, multiple OLED displays or multiple BME280 sensors? There are several solutions.

- change the device I2C address;
- use an I2C multiplexer.

Changing the I2C address

Many breakout boards have the option to change the I2C address depending on its circuitry. For example, that a look at the following OLED display.



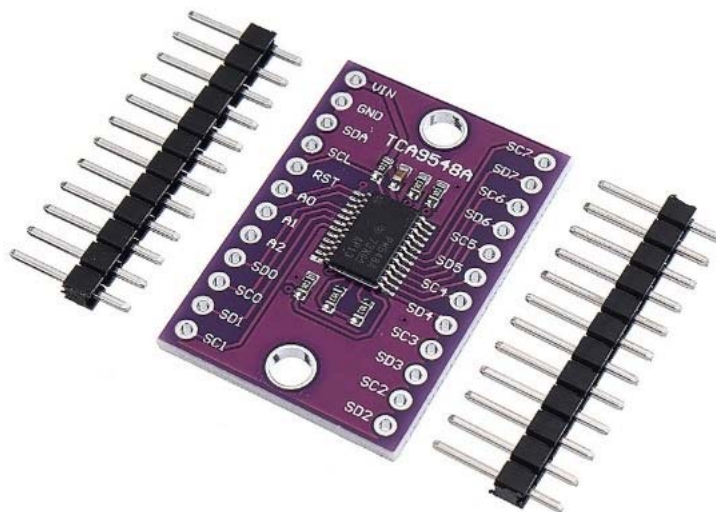


By placing the resistor on one side or the other, you can select different I2C addresses. This also happens with other components.

Using an I2C Multiplexer

However, in this previous example, this only allows you to have two I2C displays on the same bus: one with 0x3C address and another with 0x3D address.

Additionally, sometimes it is not trivial changing the I2C address. So, in order to have multiple devices with the same address in the same I2C bus, you can use an [I2C multiplexer like the TCA9548A](#) that allows you to communicate with up to 8 devices with the same address.



We have a detailed tutorial explaining how to use an I2C multiplexer to connect multiple devices with the same address to the ESP32: [Guide for TCA9548A I2C Multiplexer: ESP32, ESP8266, Arduino](#).

ESP32 Using Two I2C Bus Interfaces

To use the two I2C bus interfaces of the ESP32, you need to create two `TwoWire` instances.

```
TwoWire I2Cone = TwoWire(0);
```



Then, initialize I2C communication on your desired pins with a defined frequency.

```
void setup() {  
  I2Cone.begin(SDA_1, SCL_1, freq1);  
  I2Ctwo.begin(SDA_2, SCL_2, freq2);  
}
```

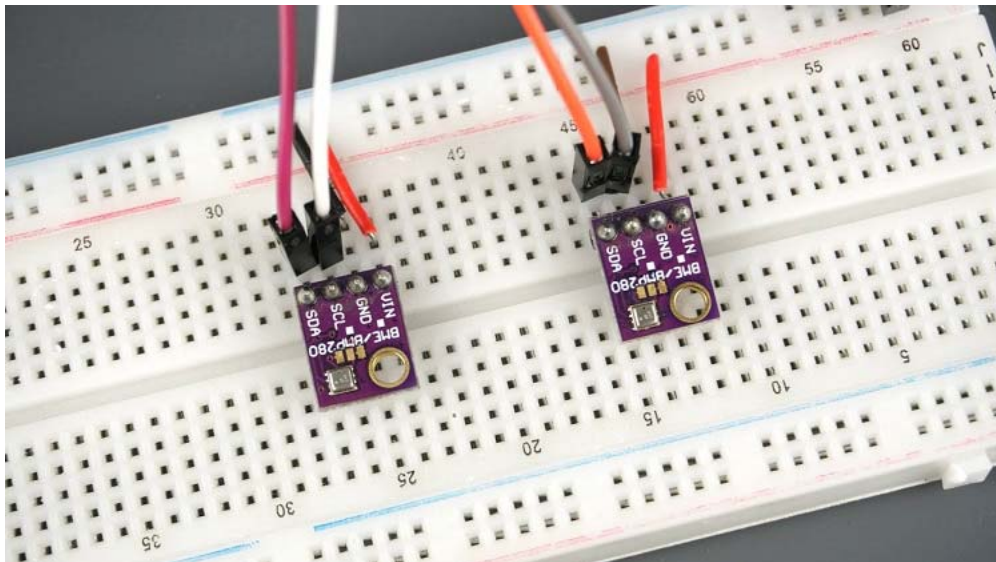
Then, you can use the methods from the `Wire.h` library to interact with the I2C bus interfaces.

A simpler alternative is using the predefined `Wire()` and `Wire1()` objects. `Wire().begin()` creates an I2C communication on the first I2C bus using the default pins and default frequency. For the `Wire1.begin()` you should pass your desired SDA and SCL pins as well as the frequency.

```
setup(){  
  Wire.begin(); //uses default SDA and SCL and 1000000HZ freq  
  Wire1.begin(SDA_2, SCL_2, freq);  
}
```

This method allows you to use two I2C buses, one of them uses the default parameters.

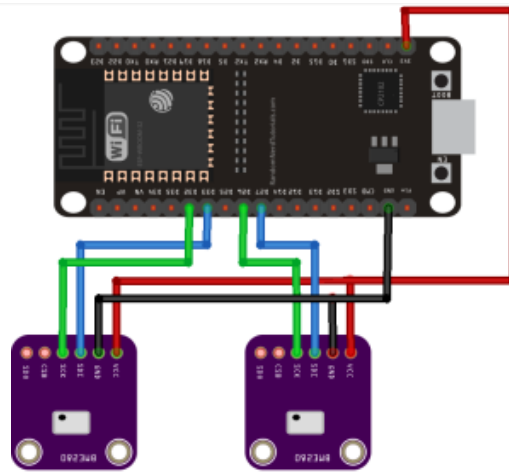
To better understand how this works, we'll take a look at a simple example that reads temperature, humidity and pressure from two BME280 sensors.



Each sensor is connected to a different I2C bus.

- I2C Bus 1: uses GPIO 27 (SDA) and GPIO 26 (SCL);
- I2C Bus 2: uses GPIO 33 (SDA) and GPIO 32 (SCL);





Click image to enlarge

```

Serial.print("Humidity from BME1 = ");
Serial.print(bme1.readHumidity());
Serial.println(" %");

Serial.print("Pressure from BME1 = ");
Serial.print(bme1.readPressure() / 100.0F);
Serial.println(" hPa");

Serial.println("-----");

// Read from bme2
Serial.print("Temperature from BME2 = ");
Serial.print(bme2.readTemperature());
Serial.println(" *C");

Serial.print("Humidity from BME2 = ");
Serial.print(bme2.readHumidity());
Serial.println(" %");

Serial.print("Pressure from BME2 = ");
Serial.print(bme2.readPressure() / 100.0F);
Serial.println(" hPa");

Serial.println("-----");

delay(5000);
}

```

[View raw code](#)

Let's take a look at the relevant parts to use the two I2C bus interfaces.

Define the SDA and SCL pins you want to use:

```

#define SDA_1 27
#define SCL_1 26

```



```
#define SDA_2 33  
#define SCL_2 32
```

Create two `TwoWire` objects (two I2C bus interfaces):

```
TwoWire I2Cone = TwoWire(0);  
TwoWire I2Ctwo = TwoWire(1);
```

Create two instances of the `Adafruit_BME280` library to interact with your sensors: `bme1` and `bme2`.

```
Adafruit_BME280 bme1;  
Adafruit_BME280 bme2;
```

Initialize an I2C communication on the defined pins and frequency:

```
I2Cone.begin(SDA_1, SCL_1, 100000);  
I2Ctwo.begin(SDA_2, SCL_2, 100000);
```

Then, you should initialize the `bme1` and `bme2` objects with the right address and I2C bus. `bme1` uses `I2Cone`:

```
bool status = bme1.begin(0x76, &I2Cone);
```

And `bme2` uses `I2Ctwo`:

```
bool status1 = bme2.begin(0x76, &I2Ctwo);
```

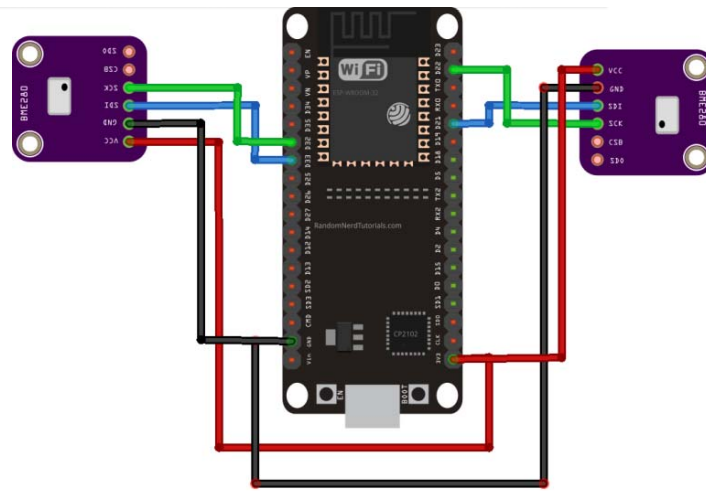
Now, you can use the methods from the `Adafruit_BME280` library on your `bme1` and `bme2` objects to read temperature, humidity and pressure.

Another alternative

For simplicity, you can use the predefined `Wire()` and `Wire1()` objects:

- `Wire()` : creates an I2C bus on the default pins `GPIO 21` (SDA) and `GPIO 22` (SCL)
- `Wire1(SDA_2, SCL_2, freq)` : creates an I2C bus on the defined `SDA_2` and `SCL_2` pins with the desired frequency.





Click image to enlarge

Here's the same example but using this method. Now, one of your sensors uses the default pins, and the other uses GPIO 32 and GPIO 33.

```

/*
  Rui Santos
  Complete project details at https://RandomNerdTutorials.com/esp32-i2c-communication-arduino-ide/

  Permission is hereby granted, free of charge, to any person obtaining a copy
  of this software and associated documentation files.

  The above copyright notice and this permission notice shall be included in all
  copies or substantial portions of the Software.
  */

#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BME280.h>

#define SDA_2 33
#define SCL_2 32

Adafruit_BME280 bme1;
Adafruit_BME280 bme2;

void setup() {
  Serial.begin(115200);
  Serial.println(F("BME280 test"));

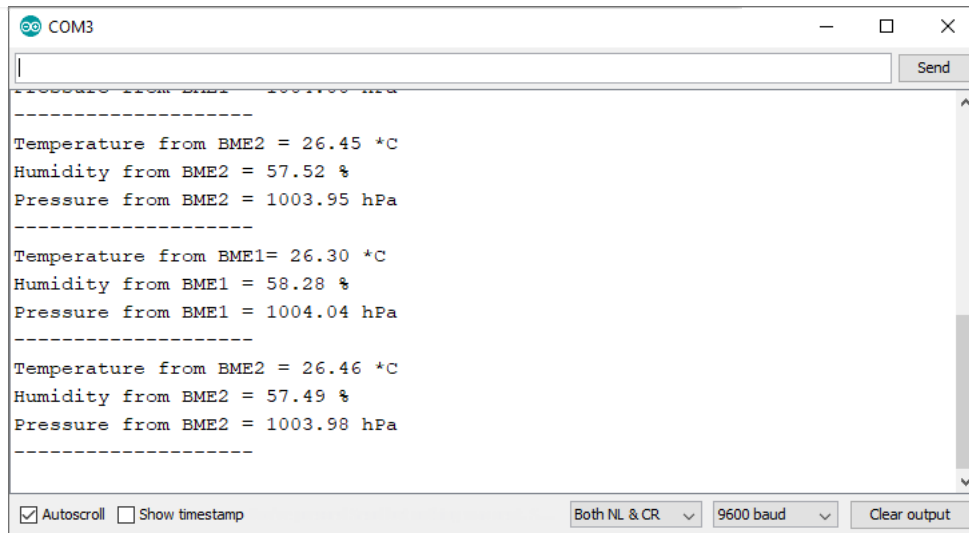
  Wire.begin();
  Wire1.begin(SDA_2, SCL_2);

```

View raw code

You should get both sensor readings on your Serial Monitor.





```
-----  
Temperature from BME2 = 26.45 *C  
Humidity from BME2 = 57.52 %  
Pressure from BME2 = 1003.95 hPa  
-----  
Temperature from BME1= 26.30 *C  
Humidity from BME1 = 58.28 %  
Pressure from BME1 = 1004.04 hPa  
-----  
Temperature from BME2 = 26.46 *C  
Humidity from BME2 = 57.49 %  
Pressure from BME2 = 1003.98 hPa  
-----
```

COM3

Send

☒ Autoscroll ☐ Show timestamp

Both NL & CR 9600 baud Clear output

Wrapping Up

In this tutorial you learned more about I2C communication protocol with the ESP32. We hope you've found the information in this article useful.

To learn more about the ESP32 GPIOs, read our reference guide: [ESP32 Pinout Reference: Which GPIO pins should you use?](#)

Learn more about the ESP32 with our resources:

- [Learn ESP32 with Arduino IDE \(Video course + eBook\)](#)
- [MicroPython Programming with ESP32 and ESP8266 \(eBook\)](#)
- [More ESP32 tutorials...](#)

Thanks for reading.



PCBWay PCB Fabrication & Assembly

ONLY \$5 for 10 PCBs

- ✓ 24-hour Build Time
- ✓ Quality Guaranteed
- ✓ Most Soldermask Colors:

Order now

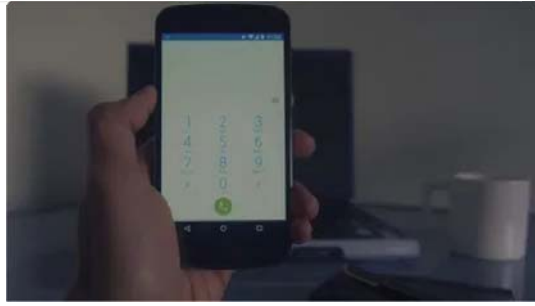
www.pcbway.com



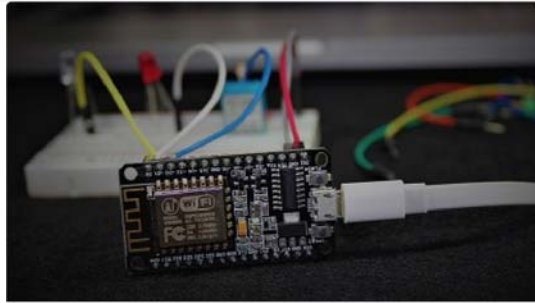
SMART HOME with Raspberry Pi, ESP32, ESP8266 [eBook]

Learn how to build a home automation system and we'll cover the following main subjects: Node-RED, Node-RED Dashboard, Raspberry Pi, ESP32, ESP8266, MQTT, and InfluxDB database [DOWNLOAD »](#)

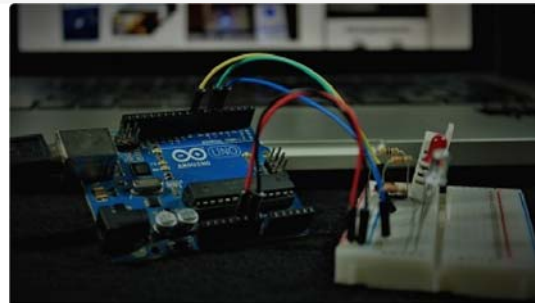




[Build a Home Automation System from Scratch »](#) With Raspberry Pi, ESP8266, Arduino, and Node-RED.



[Home Automation using ESP8266 eBook and video course »](#) Build IoT and home automation projects.



[Arduino Step-by-Step Projects »](#) Build 25 Arduino projects with our course, even with no prior experience!

What to Read Next...



[ESP8266 and Node-RED with MQTT \(Publish and Subscribe\)](#)[ESP8266 DS18B20 Temperature Sensor with Arduino IDE \(Single, Multiple, Web Server\)](#)[ESP32 LoRa Sensor Monitoring with Web Server \(Long Range Communication\)](#)

Enjoyed this project? Stay updated by subscribing our newsletter!



✉ SUBSCRIBE

128 thoughts on “ESP32 I2C Communication: Set Pins, Multiple Bus Interfaces and Peripherals (Arduino IDE)”



Francisco Teloken

October 2, 2019 at 4:37 pm



[Reply](#)**Sara Santos**

October 2, 2019 at 10:13 pm

Thank you for following our work 😊

Regards,
Sara

[Reply](#)**朱盼**

October 4, 2019 at 1:40 am

Thank you for your tutorial, can you post an example about using multiple spi sensors, thank you

[Reply](#)**Sara Santos**

October 4, 2019 at 9:37 am

Hi.

Thank you for your suggestion.

Yes, we'll probably write a tutorial about that. But I still don't know when.

Regards,
Sara

[Reply](#)**Simon**

October 9, 2022 at 7:30 am

Hi

Can I use ESP32 (3.3V) and other I2C device on 5V?

[Reply](#)**Jairo**

October 2, 2019 at 5:28 pm



Thanks for this valuable information, I'm getting started on ESP32 all your web site information is pure gold, thanks!

[Reply](#)



Sara Santos

October 2, 2019 at 10:12 pm

Thank you.

I'm glad you find our information useful 😊

Regards,

Sara

[Reply](#)



Salvador Hernández

October 2, 2019 at 6:51 pm

Thanks so much, Your tutorials are perfect.

[Reply](#)



Sara Santos

October 2, 2019 at 10:07 pm

Thank you 😊

[Reply](#)



Rajiv Shankar

October 3, 2019 at 4:35 am

Hi Sara,

Very good tutorial. You have mentioned that multiple devices can be looped PI, let me know how many devices we can connect on same bus. What will be the maximum distance between devices and the ES32. Which type and size of cable is required to connect various devices since different types of cable have different resistance per meter of length.

Thanks

[Reply](#)



**Friedhelm**

October 3, 2019 at 7:25 am

Congratulation!

That's really useful!

[Reply](#)**Duncan Amos**

October 3, 2019 at 7:52 am

Hi Rui and Sara

I can see the advantage of having two I2C busses when you want to use two sensors with the same fixed addresses (although the two BME280s in your example produce different results when located next to each other) but what other advantages are there in running two separate busses?

Apart from when I need BlueTooth, I tend to stick to ESP8266, but maybe I'm missing out on something...

[Reply](#)**Bernard Lheureux**

October 3, 2019 at 7:57 am

Sara,
WONDEFULL as USEFULL.

Bernard

[Reply](#)**Roger Moseley**

October 3, 2019 at 6:23 pm

Rui, great tutorial. Thanks.

Looking forward to the ESP8266 version, since that's the kind of hardware I have, mostly.



**Raynal**

October 19, 2019 at 9:33 pm

Hello, i would like to use my ESP32 as a Slave, to do this, i use de Wire.h library, but it's look like the .onReceive don't working...
Do you have some solution ?

[Reply](#)**Sara Santos**

October 23, 2019 at 6:15 pm

Hi.
At the moment, we don't have any tutorial to set the ESP32 as a slave.
Regards,
Sara

[Reply](#)**Mattias**

March 4, 2021 at 10:23 pm

I also wonder how to set up the ESP32 as an I2C-Slave?
As I understand it, it's not supported by the wire.h for ESP32.
Could you suggest a workaround?
Br
/Mattias

[Reply](#)**Douglas Spencer Basberg**

March 28, 2023 at 7:37 pm

"ESPHome" with "home-assistant" does this.
In general, ESPHome can be used over wifi to accomplish this.

[Reply](#)

**Janis**

February 26, 2022 at 8:49 am

Looks like ESP32 as a slave issue is resolved (<https://github.com/espressif/arduino-esp32/pull/5746>), but I cannot figure out how to get it.

In any case cannot found I2C tutorial where several ESP32s, Arduinos and Sensors are used. I would be fine to see smt like ESP32 as as as slave/master talking wit Unos master/slave through logic-level shifter, mixed with several i2C sensors to properly illustrate application of i2C buss

[Reply](#)**Umar Muhammad**

December 26, 2019 at 10:45 pm

Thank you for the great tutorials. Could this example also work with esp8266?

[Reply](#)**Heshan**

October 28, 2019 at 5:10 am

hi is good for i2c.

can you do for the these kind of example for INA3221 sensor its really helpful

[Reply](#)**Sara Santos**

October 29, 2019 at 11:12 am

Hi.

At the moment, we don't have any tutorials about that subject.

Regards,

Sara

[Reply](#)**Darrin Martin**

December 21, 2019 at 8:22 am



Excellent article! Thank you so much, I was pulling my hair out until I read it!

[Reply](#)



Umar Muhammad

December 26, 2019 at 10:46 pm

Thank you for the great tutorials. Could this example also work with esp8266?

[Reply](#)



Mckinley

January 20, 2020 at 9:41 pm

Me ha gustado mucho el tutorial. Gracias!

[Reply](#)



Sara Santos

January 22, 2020 at 10:41 am

Thank you 😊

[Reply](#)



Gustavo

February 19, 2020 at 7:17 pm

Hi! Thanks for the tutorial!

I'm trying to connect a gpio extender (PCF8574) to the esp32 CAM, with no luck. Do you know how can I set up a new i2c bus for the extender? I've tried using ports I2C_SDA 0 , I2C_SCL 2 with no luck.

[Reply](#)



Sara Santos

February 20, 2020 at 11:02 am



Hi Gustavo.
Why did you choose those GPIOs?

[Reply](#)



Nikolay Usatyy

March 2, 2020 at 6:33 pm

Hello friends.
Could you please explain how this works when using the Adafruit ADS1115 library?

How to tell her that it is necessary to use the second woWire bus (1)?

I tried pointing as Adafruit_ADS1115 ads1115 (& I2CSensors);
This does not work.

[Reply](#)



Sara Santos

March 3, 2020 at 10:52 am

Hi.
I think you need to go to the library file: https://github.com/adafruit/Adafruit_ADS1X15/blob/master/Adafruit_ADS1015.cpp
and in line 138, you need to insert your I2C pins, otherwise it will use the default pins.
`void Adafruit_ADS1015::begin() { Wire.begin(SDA,SCL); }`
I'm not sure if this works.
Regards,
Sara

[Reply](#)



Nico

February 6, 2021 at 9:10 am

Hello Nikolay.

Did you manage to get the ADS1115 working on the second I2C bus?

I already tried to modify the SDA, SCL Pins in the library, what Sara suggested.
Did not work for me.
Returns "no matching function to call" when I want to start the ads.begin like this:
`ads.begin(0x48, &I2COne);`



The standard I2C bus is already blocked by OLED and LoRa, so I have to use Pins SDA 21 and SCL 13 for BME280, BH1750 and ADS1115.

[Reply](#)**Radago**

November 3, 2022 at 7:53 pm

Hey Nico. Did you ever get I2C working on an ESP32-CAM? I am having some trouble connecting an ADS1115 (Since I could not find any ADC pin to use on ESP32-CAM).

Hoping you have some insight!

[Reply](#)**fotosettore**

April 1, 2020 at 11:07 pm

hi !
really great tutorial!!! many many thanks !!
My question is : is it available a tutorial to connect ESP32 as master and one or more d1-mini (esp8266) as slaves ?
Many thanks
peppe

[Reply](#)**Sara Santos**

April 3, 2020 at 10:32 am

Hi.
At the moment, we don't have any tutorial about that.
Regards,
Sara

[Reply](#)**Dave K**

June 12, 2020 at 4:58 pm

You may find a wireless protocol such as ESP-NOW would be a better solution than I2C. I am pretty sure ESP32 and
sion. It all depends what you want to



[Reply](#)**Dave K**

June 12, 2020 at 4:54 pm

Having worked a bit with I2C sensors in my past, I can say they are not very good for long distance (i.e, more than a meter) separation. One wire is a much better protocol for that. I2C was designed for use on the same board with distances of a few centimeters of bus length.

That said, you can get more distance with lower bus frequencies, The maximum distance for your sensor would need to be found experimentally, and many factors affect it. Things such as the type of cable, external interference/noise, and the voltage used to drive the bus. The longer the distance the more critical these become.

Personally, I prefer other methods to connect over longer distances. It is easier and more time efficient to use a protocol that was designed for distance such as RS232 or RS485 with appropriate driver ICs. For short distances such as to a display on on board sensor, I2C is great though!

It is important in this hobby to pay attention to what things were designed to do, and use them appropriately for the greatest success.

[Reply](#)**Nikolay Usatyy**

July 30, 2020 at 9:30 pm

Hello friends.

Has anyone managed to combine this board with a solar battery for autonomous power supply?

Share the schematic pls!

[Reply](#)**Lockie Cresswell**

August 31, 2020 at 9:51 am

I have found your tutorials most enlightening. I am trying to log a BMP280 with ESP32 CAM on alternate I2C, but keep getting compiler errors with your code, which was written for BME280. I have delved into the .h and .cpp codes but cannot find why the BMP280 is not being found. Your code works fine with ESP12E and BMP280, if I use the library BMP280_DEV instead of the Adafruit library, but that library code does not support ESP32 with I2C, only SPI. I need to use the SD card facility on the ESP32CAM to log temperature and pressure. I tried mapping defaults 21 & 22 which are not accessible, to 3 & 16. Could it be that the Adafruit BMP280 library does not support alternate I2C ?



**Sara Santos**

August 31, 2020 at 4:23 pm

Hi.

The BME280 library from adafruit and the camera library for the ESP32-CAM conflict with each other.

I have used the Saprkun BME280 library with the ESP32-CAM and it works well.

But I don't know if it works with the BMP280.

I hope this helps.

Regards,

Sara

[Reply](#)**Lockie Cresswell**

September 1, 2020 at 6:57 am

Hi Sara,

Thanks so much for the quick reply. I tried the Sparkfun Library you suggested but I get a compiler error "Error compiling for board AiThinker ESP32 CAM", which is what I get with Adafruit as well.

I do not think there has been much written for ESP32 Cam that uses alternate I2C, which is needed because of the scarcity of GPIO pins. I will try the TTGO next up.

Regards

Lockie

[Reply](#)**Sara Santos**

September 2, 2020 at 9:23 am

Hi.

Can you provide more details about your error?

Regards,

Sara

[Reply](#)**Lockie Cresswell**

September 2, 2020 at 1:24 pm



Hi Sara,

I ran the example sketch "softI2C" from the Sparkfun BME library. I set compiler preferences to verbose and picked out examples of the noted errors. See below:

```
C:\Users\marts\Documents\Arduino\libraries\SoftwareWire\SoftwareWire.cpp:134:16: error: cannot convert 'volatile
uint32_t* {aka volatile unsigned int}' to 'volatile uint8_t {aka volatile unsigned char}' in assignment
_sdaPinReg = portInputRegister(port); // PinReg is the input register, not the Arduino pin.
C:\Users\marts\Documents\Arduino\libraries\SoftwareWire\SoftwareWire.cpp: In member function 'void
SoftwareWire::printStatus(Print&)':
C:\Users\marts\Documents\Arduino\libraries\SoftwareWire\SoftwareWire.cpp:533:27: error: cast from 'volatile uint8_t {aka
volatile unsigned char*}' to 'uint16_t {aka short unsigned int}' loses precision [-fpermissive]
Ser.println( (uint16_t) _sdaPortReg, HEX);
Using library SoftwareWire at version 1.5.1 in folder: C:\Users\marts\Documents\Arduino\libraries\SoftwareWire
Using library Wire at version 1.0.1 in folder:
C:\Users\marts\AppData\Local\Arduino15\packages\esp32\hardware\esp32\1.0.4\libraries\Wire
Using library SparkFun_BME280 at version 2.0.8 in folder: C:\Users\marts\Documents\Arduino\libraries\SparkFun_BME280
Using library SPI at version 1.0 in folder:
C:\Users\marts\AppData\Local\Arduino15\packages\esp32\hardware\esp32\1.0.4\libraries\SPI
exit status 1
Error compiling for board AI Thinker ESP32-CAM.
```

I haven't found anything yet that works with alternate I2C on ESP32-CAM.

Cheers,

lockie

[Reply](#)



Sara Santos

September 4, 2020 at 10:22 am

Hi.

We used the BME280 successfully with the ESP32-CAM using the BME280 Sparkfun library.

This is the project: <https://randomnerdtutorials.com/esp32-cam-shield-pcb-telegram/>

Here's the code. Take a look at the parts that refer to the BME280 sensor and ignore the rest:

https://raw.githubusercontent.com/RuiSantosdotme/ESP32-CAM-Shield-Telegram/master/ESP32_CAM_Shield_PCB_Telegram/ESP32_CAM_Shield_PCB_Telegram.ino

Regards,

Sara

[Reply](#)



Lockie Cresswell

September 5, 2020 at 7:41 am



What to do is include

“SparkFunBME280.h” to get compiler errors which relate to “SoftwareWire.h” by Testato, which is called by the former if you enable software I2C as per the instructions.

Does your example (noted in your last post) enable software i2c?

I have tried it with TTGO and get compiler errors as well, although if I use BMP280_Dev.h instead all works OK (But not for software I2C, which BMP280_Dev does not support for ESP32 devices)

Regards

lockie

[Reply](#)



Sara Santos

September 6, 2020 at 10:39 am

Hi.

We use the library as shown in the post.

include the following libraries:

```
#include <Wire.h>
```

```
#include "SparkFunBME280.h"
```

Define the I2C pins:

```
#define I2C_SDA 14
```

```
#define I2C_SCL 15
```

Initialize the sensor as follows:

```
Wire.begin(I2C_SDA, I2C_SCL);
```

```
bme.settings.commInterface = I2C_MODE;
```

```
bme.settings.I2CAddress = 0x76;
```

```
bme.settings.runMode = 3;
```

```
bme.settings.tStandby = 0;
```

```
bme.settings.filter = 0;
```

```
bme.settings.tempOverSample = 1;
```

```
bme.settings.pressOverSample = 1;
```

```
bme.settings.humidOverSample = 1;
```

```
bme.begin();
```

Get readings:

```
float temperature, humidity;
```

```
temperature = bme.readTempC();
```

```
//temperature = bme.readTempF();
```

```
humidity = bme.readFloatHumidity();
```

No need for any other libraries or other configurations.

Regards,

Sara

[Reply](#)



**Lockie Cresswell**

September 7, 2020 at 12:47 pm

Hi Sara,

I have included the code you posted in my sketch, but I still get the same errors in the SoftwareWire.cpp library, when using software I2C with my TTGO as "ESPDEV Module" in Arduino as suggested by TTGO. So that makes me think that the source of the error is in the SoftwareWire library.

So I checked the Github for Testato's SoftwareWire and it appears there are issues when using MCU's with 32 bit registers. Maybe you are not using the software I2C feature – that might explain why the code works for you.

Anyway, I appreciated your suggestions.

Regards

Lockie

[Reply](#)**Lockie Cresswell**

September 7, 2020 at 3:19 pm

Hello again,

I tried to compile some example sketches from SparkfunBME280 and I get compiler errors. Then I tried to compile a sketch from SoftwareWire and that also gives a compile error. So then I reverted to the SparkfunBME280.h that doesn't call SoftwareWire and tried compiling your code and it compiled OK.

That suggests my argument that SoftwareWire library does not support ESP32 devices may be correct. My original intention was to use a BMP280 with ESP32-Cam using software defined I2C as there is a shortage of GPIO's available. The hardware GPIO pins are 21 & 22, but when using the SD card on ESP32-CAM using one of Rui's sketches only GPIO 0, 3, & 16 are accessible. That is why I wanted to use software defined I2C pins.

I have tried numerous BMP280 libraries and the only one that seems to work well is BMP280_DEV, but as I said previously it does not support ESP32.

I then changed my board to AiThinker ESP32CAM and recompiled and got an out of memory error. So I thought that was more positive and decided to power down, then power-up and re-compile. This time it worked!!

So to re-cap for other newbies, SoftwareWire library does not appear to work with ESP32 devices, so do not call it in SparkFunBME280 for software I2C.

Once again, Thanks.

[Reply](#)**Lockie Cresswell**

September 8, 2020 at 3:31 pm

Whilst I can now print temperature and pressure to the serial monitor, I am struggling to print it to the SD card. I am using a version of Rui's logSD which successfully writes the date and time to the card, but leaves the temperature blank. The code for the routine is:



```
void logSDCard() {
  dataMessage = String(readingID) + "," + String(dayStamp) + "," + String(timeStamp) + "," + String(temperature) + "\r\n";
  Serial.print("Save data: ");
  Serial.println(dataMessage);
  appendFile(SD_MMC, "/data.txt", dataMessage.c_str());
}
```

I have not been able to pass the temperature variable to this routine.

Any ideas?

Thanks, Lockie

[Reply](#)



Dave K.

September 9, 2020 at 12:29 am

Maybe this will help:

You may need to convert the data to a string first. A welcomed utility function I found is dtostrf. (decimal to string).

Remember that serial comms know nothing about data types, they simply send characters.

```
char * dtostrf (double __val, signed char __width, unsigned char __prec, char *__s)
```

The dtostrf() function converts the double value passed in val into an ASCII representation that will be stored under s. The caller is responsible for providing sufficient storage in s.

Conversion is done in the format "[-]d.ddd". The minimum field width of the output string (including the '.' and the possible sign for negative values) is given in width, and prec determines the number of digits after the decimal sign. width is signed value, negative for left adjustment.

The dtostrf() function returns the pointer to the converted string s.

```
char buffer[10];
dtostrf(float or double Var, 5, 1, buffer);
```

//now you can print or send the buffer to a text display

Below is how I used it with a couple of one wire probes. As per above

templnC is a float, I expect a max of 6 chars, with 1 after decimal point. I have two char buffers named testInC and testOutC respectively, each 7 chars.

```
dtostrf(templnC, 6, 1, testInC);
dtostrf(tempOutC, 6, 1, testOutC);
```

Best of luck!

Dave K.

[Reply](#)



Lockie Cresswell



Thanks so much, Dave.

As a newbie what you told me was DD, but I perservered and after dozens of compiles I got the declarations correct, and so now it prints both to the screen and the SD card.

Here is the code snippit:

```
static char tempc[8];
static char pressr[9];
static float temperature;
static float pressure;
// Function to get temperature
String Getreadings(){
//float temperature, pressure;
temperature = bme.readTempC();
pressure = bme.readFloatPressure()/100;
dtostrf(temperature,6,2,tempc);
dtostrf(pressure,7,2,pressr);
String message = "Temperature: " + String (temperature) + " °C \n";
message += "Pressure: " + String (pressure) + " hPa \n";
return message;
```

So now I am logging BMP280 temperature and pressure every minute and writing the data to the onboard SD card on the ESP32 CAM together with time and date off the internet. (I started Arduino mid-August)

Thanks to both you and Sara for getting me there, and to Rui (for the ideas).

I have been at this project for some weeks now, and I have been frustrated by the lack of documentation of the various libraries and their foibles.

[Reply](#)



Dave K

September 9, 2020 at 12:48 pm

Sounds like it helped! GREAT!
Congratulations on your success!

I know for sure that info helped me get something to display on an OLED, and in a log, so I was happy to share it.

Dave

[Reply](#)



Raffaele Zuccarini

September 22, 2020 at 2:03 pm

Ottimo tutorial!
Complimenti



**Sara Santos**

September 23, 2020 at 4:22 pm

Thank you 😊

[Reply](#)**Ajay Raj**

November 22, 2020 at 6:10 am

As a beginner to the IOT area I find your tutorials really helpful.
Could you please elaborate on using multiplexer for connecting two devices, probably with a schematic.

[Reply](#)**Sara Santos**

November 23, 2020 at 3:23 pm

Hi.
Unfortunately, at the moment, we don't have any tutorials about I2C multiplexer.
Regards,
Sara

[Reply](#)**Luberth Dijkman**

November 29, 2020 at 8:46 pm

Thank You!
usefull for my thermostat project
<https://oshwlab.com/l.dijkman/esp32-dev-kit-38-pin-to-spi-touch-tft>

[Reply](#)**Osama Riyad**

December 21, 2020 at 2:28 pm



I want to ask how can I connect the esp32(as a master) with the arduino uno (as a slave) which is controlling a motor driver

[Reply](#)



Sara Santos

December 22, 2020 at 11:34 am

Hi.

At the moment we don't have any tutorials about that subject.

Regards,

Sara

[Reply](#)



Asmaa

January 25, 2021 at 11:33 am

Hi Sara,

Tanks very much for this very useful article.

I'm ask if I can use two devices in the same pins of SPI interface (MISO,MOSI,CLK,SS)?
and if yes, this is accurate or not?

[Reply](#)



Sara Santos

January 25, 2021 at 3:18 pm

Hi.

You can use more than one SPI device on the same pins as long as each device has its own chip select (SS) pin.

Regards,

Sara

[Reply](#)



Hugo

February 1, 2021 at 1:19 am

Hi, Thank you very much for the information, Do you have an example of how to configure the ESP32 in Slave mode for example at address 10 or 0x0A? I trv to make a chat between two modules FSP32



Best Regards

Hugo

[Reply](#)



CEP

February 4, 2021 at 8:57 am

Hi Rui,

I will want to know how can multiple masters controlling the same slave. I have tried to use two ESP8266 to read one MCP23017 module but not successful. After googling people commented that ESP8266 has no slave feature. I am not sure about ESP32. Please advise.

Regards

CEP

[Reply](#)



Asmaa

February 14, 2021 at 9:09 pm

Hi,

I want to connect 3 modules to the I2C protocols
if it is easy to use it by giving it a different addresses ??
or use i2c mux like TCA9548A better ??

[Reply](#)



Marco Loaiza

April 2, 2021 at 5:14 am

Hi Sara,

Thank you very much for the information.

I tried to connect the sensor getting 0.0 values in both Temperature and Humidity using the following code:

```
#include <Wire.h>
```

```
#include "SparkFunBME280.h"
```

```
//CAMERA_MODEL_AI_THINKER
```

```
#define PWDN_GPIO_NUM 32
```

```
#define RESET_GPIO_NUM -1
```

```
#define YCLK_GPIO_NUM 0
```



```
#define SIOD_GPIO_NUM 26
#define SIOC_GPIO_NUM 27

#define Y9_GPIO_NUM 35
#define Y8_GPIO_NUM 34
#define Y7_GPIO_NUM 39
#define Y6_GPIO_NUM 36
#define Y5_GPIO_NUM 21
#define Y4_GPIO_NUM 19
#define Y3_GPIO_NUM 18
#define Y2_GPIO_NUM 5
#define VSYNC_GPIO_NUM 25
#define HREF_GPIO_NUM 23
#define PCLK_GPIO_NUM 22

//Define the I2C pins:
#define I2C_SDA 14
#define I2C_SCL 15
BME280 bme;
void setup(){
  Serial.begin(115200);
  delay(5000);
  Serial.println("Empezamos");
}

void loop(){

  //Initialize the sensor as follows:
  Wire.begin(I2C_SDA, I2C_SCL);
  bme.settings.commInterface = I2C_MODE;
  bme.settings.I2CAddress = 0x76;
  bme.settings.runMode = 3;
  bme.settings.tStandby = 0;
  bme.settings.filter = 0;
  bme.settings.tempOverSample = 1;
  bme.settings.pressOverSample = 1;
  bme.settings.humidOverSample = 1;
  bme.begin();
  //Get readings:
  float temperature, humidity;
  temperature = bme.readTempC();
  //temperature = bme.readTempF();
  humidity = bme.readFloatHumidity();

  Serial.println(temperature);
  Serial.println(humidity);
  delay(5000);
}
```

Is there any option or code to check if the sensor is correctly wired ?
Thanks in advance for your help

[Reply](#)



**Sara Santos**

April 2, 2021 at 2:34 pm

Hi.

We have this project that interfaces the ESP32-CAM the BME280 sensor.

You can take a look at the parts of code that interface with the sensor to see if you're missing something.

Regards,

Sara

[Reply](#)**piter**

April 18, 2021 at 7:37 pm

Dear Sara ad Rui,

Thank you for this explanation on using IIC on other than the "standard" GPIO pins. The BME280 works perfect on an ESP32-CAM, using GPIO 12 and GPIO15. Connecting SCL on GPIO13 won't work -> No proper boot.

When adding other IIC component(s) though, I run into trouble.

Basically since I can not discover the proper libraries where "off standard" GPIO-pins are supported or can be specified.

More specifically for i.e. a RTC (DS1307). The goal is to make a temp/humidity/pressure logger, writing the data on a micro SD card, without internet access, hence the RTC. Time is set once, RTC keeps time.

No Cam needed, I use the ESP32-Cam since it has a built in CF-card, an because of the size. (ESP32-CAM and BME280 do fit in one LEGO-Block 😊)

Where for the BME28 your example allows for "off standard" GPIO-pins by means of:

```
I2CBME.begin(I2C_SDA, I2C_SCL, 100000);
```

```
status = bme.begin(0x76, &I2CBME);
```

This is not continued in your example where two IIC devices are used – the display and the BME280:

```
display.begin(SSD1306_SWITCHCAPVCC, 0x3C) // init the display
```

```
bool status = bme.begin(0x76); // init the BME280
```

When I check the RTCLib on the Adafruit repository, also no SCL and SCA can be specified.

<https://github.com/adafruit/RTCLib/blob/master/RTCLib.cpp>

```
boolean RTC_DS1307::begin(void) {
```

```
Wire.begin();
```

```
Wire.beginTransmission(DS1307_ADDRESS)
```

```
....etc...
```

Question:

On what way can I program around this and use pin , or is it required to (re)program the C++ methods?

Thank you very much for any help



**Sara Santos**

April 19, 2021 at 9:21 am

Hi.

It seems that library doesn't support TwoWire to set up different pins: <https://github.com/adafruit/RTClib/issues/172>However, there is this modified library that supports different pins: <https://forum.arduino.cc/t/software-i2c-rtc-lib-h/662222>

I hope this helps.

Regards,

Sara

[Reply](#)**piter**

April 19, 2021 at 9:35 pm

Hi Sara,

Thank you for the lightning fast reply.

The "test" library for the RTC works like a charm with the software set i2c pins. (i2c_rtc.h)

The BME280 works perfect with the Adafruit_BME280.h lib

Each perfect when going without the other,..... but using them in one sketch provides a spin-crash-and-burn reboot (causes by a int. div-zero)

Output below: Wifi connected, print temp, pressure and humidity,
next line: print day and time....->> Crash.

IP address: 192.168.2.18

Temp. C: 20.65 Pressure: 1020.47 Humidity: 42.46

(Mon) 23:11:7

Temp. C: Guru Meditation Error: Core 1 panic'ed (IntegerDivideByZero). Exception was unhandled.

Core 1 register dump:

.

.

Any Clues .. ?

I will investigate further also.

Greetings & Regards

[Reply](#)**Sara Santos**

Hi again.

I'm not sure what's the issue.

Are you using the same I2C pins or different for each module?

You can use the same pins for both modules. If that doesn't work, you can also try to use different pins for each of them.

Regards,

Sara

[Reply](#)



piter

April 20, 2021 at 1:15 pm

Hi Again....

I'm using the same pins for both IO devices.

Using other pins won't do for me on the ESP32. Not enough free GPIO's.

I need -that's the reason I took this dev-board- a small size board AND cf-card writing capabilities. Using an other dev-board or using more pins compromises either one. (soldering / tampering excluded)

Next try is 2 other pin's for both devices (BME280 and RTC DS1307), but initially 1 SCL/SDA set.

Q: Is it helpfull if I undress the code snippet to a minimum and post it here ?

Re-Regards,

Piter

[Reply](#)



Sara Santos

April 20, 2021 at 5:00 pm

Hi.

You mentioned that both modules worked individually. So, it may not be related to the code.

With both modules connected to the board, can you run an I2C scanner sketch? To see if it can find both modules when they are connected at the same time.

If it can't find the modules, it can be something related to the board...

I'm not sure what can be the problem 🙄

Regards,

Sara

[Reply](#)



piter

April 20, 2021 at 9:32 pm



Hi Sara,
 To begin with : The problem is solved.
 It was code related (Examples in soft_RTCLib library caused the problem).

I could not run the I2C scanner sketch, (no overloaded methods for softpins available).. Any plans for this... ?

If you check the output from my posting April 19, 2021 at 9:35 pm, you see that the ESP32-CAM crashes when reading the BME280 temperature for the second time.

It prints Temp. C:... and then the DIV 0 occurs.

So, BME280 is read OK initially, also the RTC clock is read OK initially

The culprit was in the file "Test.ino" in the soft_RTCLib "test.zip"

I used / copied the following function readRTC():

```
//Reads and shows RTC/
void read_RTC(byte soft_SDA_pin, byte soft_SCL_pin) {
  RTC_DS3231 my_RTC; //address I2C: 0x68/
  TwoWire connection = TwoWire(0);
  /(SDA, SCL, freq);/
  connection.begin(soft_SDA_pin, soft_SCL_pin, 100000);
  my_RTC.begin(&connection);
  char week_day[7][4] = {"Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat"};
  DateTime now = my_RTC.now();
  Serial.print(" ..... etc.....")
}
```

Taking the declaration and initialisations out of this function, to global and setup() respectively solved the problem.

Sounds easy... took some tries... 😊

Having SCL and SDA on 13 and 15 the sketch output is now like:

..

(Tue) 23:26:42 Temp:20.41 C Pressure: 1018.82 Hpa Humidity: 44.31%

(Tue) 23:26:43 Temp:20.41 C Pressure: 1018.86 Hpa Humidity: 44.30%

(Tue) 23:26:44 Temp:20.41 C Pressure: 1018.82 Hpa Humidity: 44.30%

..

On with the next steps for the datalogger.

Thank you for your help in pointing me in the right direction for this library.

Regards,
 Piter

[Reply](#)



Sara Santos

April 21, 2021 at 10:03 am

Hi.



Keep up working on your project.

Regards,
Sara

[Reply](#)



piter

April 21, 2021 at 10:59 am

Hi,

So be aware... Declaring and initializing the soft_I2C variables / structs / methods inside a function to be called later, may cause havoc.

I guess somewhere wire() uses some global var's or global interaction with other 2wire comm. Doing the initial stuff inside a function (on the stack) does not perform well with this, since the function stack is invalid after function termination.

Regards,
Piter.

[Reply](#)



David Wreski

April 24, 2021 at 11:36 pm

Your information is pure GOLD ! Thank you so much for your hard work. I am sure you spent many hours doing this for us !
Your the greatest.

[Reply](#)



Sara Santos

April 25, 2021 at 10:25 am

Thank you! 😊

[Reply](#)



bendito

April 29, 2021 at 3:49 pm

I have an I2C sensor without a library (ADPD144RI sensor). Is there a tutorial or how to use an I2C sensor without a library?



**Sara Santos**

April 29, 2021 at 5:45 pm

Hi.

I answered your question here: <https://rntlab.com/question/is-there-a-tutorial-or-how-to-use-an-i2c-sensor-without-a-library/>

Regards.

Sara

[Reply](#)**Wolfgang Ewald**

May 14, 2021 at 1:01 pm

Just want to say thanks for this very clear article!

[Reply](#)**Sara Santos**

May 15, 2021 at 3:33 pm

Thank you 😊

[Reply](#)**Joko**

June 3, 2021 at 2:34 pm

Dear Sara,

I have problem to communicate esp32 via i2c to pcf8574 also AHT10, first of all I Scan my pc8574 with sketch that your provide but the result "No i2c device found" my wiring connection as simple as your explanation GND→GND, VCC→3.3V, SCL→SCL, SDA→SDA, But if I plugout the power of PCF8574 from ESP32 board , My serial Monitor Found 27 Device starting 0x01 till 0x43 this result are the same if i used separate supply to PCF8574. and then I try also To AHT10 Breakout Sensor and use example from adafruit library my serial monitor result "no device connected".. what's wrong with this situation? for your info I use esp32 with 38 pin dev board.

Your Help Highly Appreciate

Thanks

[Reply](#)

**Pieter**

June 5, 2021 at 1:27 pm

hmmm,

Maybe try shorter wires (max 10cm) and/or reduce clock speed in the initial 2wire call
. Solved some of my problems.

[Reply](#)**Paulo**

September 9, 2021 at 9:17 am

Great tutorial! Thank you.

Was wondering if it is possible to use five I2C Bus Interfaces, like you did with two. In case of multiple screens and sensors, it would be very useful.

Then I could set sda1, sda2, sda3, sda4, sda5.

[Reply](#)**Daer, Jg**

September 16, 2021 at 8:31 pm

Greetings Sara, in case you use, for example, one of the AD inputs and need to show the results on 2 different I2C displays (one Oled and the other being the TM1637) with the appropriate libraries in the sketch, how would the program look to control these two displays simultaneously? Thank you very much

[Reply](#)**Sara Santos**

September 17, 2021 at 9:54 am

Hi.

If the displays have different I2C addresses, which should be the case. You can use the same pins.

Then, you just need to put together the code for the OLED with the code for the TM1637 without major modifications.

I think they will work ok together.

Regards,

Sara



**Daer, Jg**

September 17, 2021 at 12:11 pm

Hi Sara thanks so much for your attention! I used the second display on the same bus only changing the address and in this second display I loaded an example library in which I have a fixed/stopped text; on the main display I have a reading that changes constantly, but I don't know what the command line should be so that this second display can become a mirror of the main one, having the same information in both; I'm working with esp32, would it help to upload my simple sketch? No way to thank.

[Reply](#)**Sara Santos**

September 17, 2021 at 2:05 pm

Hi.

You need to display the same data in both OLED displays. Use the same variable.

Get the sensor readings on the loop() every X number of seconds, and display them on both displays.

Every time the code goes through the loop(), it gets new readings and displays the new readings on the displays. So, they are updated.

Regards,

Sara

[Reply](#)**Hassan**

October 9, 2021 at 2:14 pm

Hi

do you know where MCLK ,SCLK, LRCLK, SDIN and PDN pin connect in Esp32 module ?

[Reply](#)**Sara Santos**

October 9, 2021 at 4:13 pm

Hi.

What is the module that has those pins?

Regards,

Sara



**Andy Z**

November 21, 2021 at 6:35 pm

I find Random Nerd Tutorials as the most useful site for learning ESP programming. Encouraged by this lesson I built a portable experiment to measure atmospheric pressure (and indirectly altitude differences) using a Firebeetle ESP32-E, LiPo battery, and for the first time connected two I2C peripherals (BME sensor and LCD 2 rows), using MicroPython. Thanks for all the tutorials!

[Reply](#)**Sara Santos**

November 22, 2021 at 11:15 am

Thank you!

[Reply](#)**SianEe Goh**

November 23, 2021 at 11:01 am

I am a newbie to Arduino and ESP32.
At the moment I am using the TOF VL53L1X and MCP23017 on my hobby project.
I am so grateful to your above tutorial on I2C because it helps me understand this subject matter much better. I just can't log off without expressing my gratitude to your great work.
Thank you very very much.

[Reply](#)**Sara Santos**

November 23, 2021 at 12:01 pm

Thank you.

[Reply](#)**Riaz Ahmed**

Hi, I have a SHT31 temp/humidity sensor and want to connect it thru IIC to Lora TTGO ESP32, I followed the instructions for bme thinking its just a replacement but somehow the ESP32 fails to read temp and humidity once the OLED is initialised. I am attaching the code which works with SDA/SCL at 21/22 pins and the OLED initialisation commented. Initializing OLED seems to put off the sensor. I guess its about the IIC already used by the OLED but in your example you use different sda/scl pins. Any leads or examples where SHT31 Lora ESP32 are used or any hint?

```
#include <Arduino.h>
#include <Wire.h>
#include "Adafruit_SHT31.h"
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

Adafruit_SHT31 sht31 = Adafruit_SHT31();

//OLED pins
#define OLED_SDA 4
#define OLED_SCL 15
#define OLED_RST 16
#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RST);

void setup()
{
  Serial.begin(9600);
  if (! sht31.begin(0x44))
  {
    Serial.println("Couldn't find SHT31");
    while (1) delay(1);
  }

  //reset OLED display via software
  // pinMode(OLED_RST, OUTPUT);
  // digitalWrite(OLED_RST, LOW);
  // delay(20);
  // digitalWrite(OLED_RST, HIGH);

  //initialize OLED
  // Wire.begin(OLED_SDA, OLED_SCL);
  // if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3c, false, false)) { // Address 0x3C for 128x32
  // Serial.println(F("SSD1306 allocation failed"));
  // for(;;); // Don't proceed, loop forever
  // }
}

void loop()
{
  float t = sht31.readTemperature();
  float h = sht31.readHumidity();

  if (! isnan(t))
  {
    Serial.print("Temp *C = "); Serial.println(t);
```



```
else
{
  Serial.println("Failed to read temperature");
}

if (!isnan(h))
{
  Serial.print("Hum. % = "); Serial.println(h);
}
else
{
  Serial.println("Failed to read humidity");
}
Serial.println();
delay(1000);
}
```

[Reply](#)**Yancho Kostov**

February 12, 2022 at 11:45 am

Thank you so much!!!

[Reply](#)**Xylopyrographer**

March 11, 2022 at 4:43 am

Another great tutorial! Thanks.

I'm using an M5Stack ATOM Matrix in project. I need to use the I2C bus for part of the setup and then (because the Matrix has limited I/O pins exposed), reuse the SDA and SCL lines in a later part of the code as normal GPIO pins.

Using the `Object.begin(SDA, SCL, Frequency);` method works fine, but I need something akin to an `Object.end();` method that will release or detach the pins from being an I2C bus (and release the ESP32 I2C peripheral).

Is there a way to do that?

I did find in the ESP32 `esp32-hal-i2c.c` file (which `wire.cpp` brings in via an `#include`) the function:

```
void i2cRelease(i2c_t *i2c) // release all resources, power down peripheral
```

which is exactly what I'm looking for but I am at a loss as to how to call it from within the sketch (and/or if the bus needs to be set up in a particular way to use it?). Running arduino-esp32 core v1.06.



[Reply](#)**Sara Santos**

March 11, 2022 at 2:00 pm

Hi.
Did you try the `end()` method on the `i2c` object?
Regards,
Sara

[Reply](#)**Xylopyrographer**

March 11, 2022 at 6:21 pm

Oddly, `Wire.h` does not have a `.end()` method.

[Reply](#)**Sara Santos**

March 12, 2022 at 11:59 pm

It seems the `TwoWire` has an `end` method.
See here: <https://github.com/espressif/arduino-esp32/blob/master/libraries/Wire/src/Wire.h>
Regards,
Sara

[Reply](#)**Xylopyrographer**

March 13, 2022 at 7:08 pm

Interesting. Sometime in the `v2.x.y` series of the `arduino-esp32` core a `.end()` method was indeed added. It's not there in core `v1.0.6`.

The `Wire` library in the `v2` series core looks to be a complete rewrite. Can't quite figure out what `.end()` is doing there. The `.flush()` method in that version doesn't look to do what it does in `v1.0.6` either.

Guess I'll send a note to the dev team over there for clarification.



**Sara Santos**

March 13, 2022 at 7:35 pm

Great! Then, let us know your findings.

Regards,

Sara

**Xylopyrographer**

March 11, 2022 at 7:40 pm

But, after more digging, there is a `.flush()` method which does call `i2cRelease(i2c_t *i2c)` so that is the answer. 😊

[Reply](#)**Sara Santos**

March 13, 2022 at 12:03 am

Great!

Thanks for sharing it.

Regards,

Sara

[Reply](#)**Tony**

March 16, 2022 at 12:33 pm

Thank you for the contribution. This tutorial saved my day! It helped me figure out the addressing problem with my projects using I2C. Great work!

[Reply](#)**Sara Santos**

March 18, 2022 at 4:05 pm

That's great!



**Nik**

March 20, 2022 at 10:02 pm

Hi, thank you for the article

You are the best to study the ESP32!!

But if I wanted to replace one of the bmp280 with an oled SSD1306 , how to change the code?

Thanks 😊

Nik

[Reply](#)**Alex**

July 12, 2022 at 8:23 am

Hi all,

Good article.

Is it possible to use 4 separate I2C buses on the ESP32?

If the wire.h library is adjusted, or is this hardware technically not possible?

Thx Alex

[Reply](#)**Sara Santos**

July 13, 2022 at 7:45 pm

Hi.

Yes. It is possible.

Consider using an I2c multiplexer: <https://randomnerdtutorials.com/tca9548a-i2c-multiplexer-esp32-esp8266-arduino/>

Regards,

Sara

[Reply](#)**Alex**

July 16, 2022 at 8:43 am



Thanks for your reply. But is it possible without an I2C multiplexer.
I mean connected directly to the ESP32.

Thx Alex

[Reply](#)



Sara Santos

July 16, 2022 at 5:24 pm

Hi.
Yes. I think it should be possible using software I2C, but I never tried it.
Regards,
Sara

[Reply](#)



Alex

July 19, 2022 at 5:22 am

Dear Sara Santos,

Thank you for the Information.

Regards
Alex



mohammad Taghi

August 5, 2022 at 6:53 pm

Dear Sara Santos
Very very thanks for your software and Documentation.
They are really useful
I used a lot
Regards
mohammad Taghi

[Reply](#)



Sara Santos

August 5, 2022 at 8:54 pm



[Reply](#)**Rafael Freese Guerreiro**

October 6, 2022 at 11:51 am

Hello Sara and Rui, congratulations for the contents, they are excellent.

In my project, I'm using an ESP32 + 08 PCF8574, everything is working perfectly. The only point in question is that I selected the addresses of the PCF's as the datasheet says, but when I run the I2C_Scanner program, the answer I get is the following:

35 -- 38 --

56 57 58 -- 60 61 -- 63

Strange, as they should be 32, 33, 34, 35, 36, 37, 38 and 39.

[Reply](#)**Gonzalo**

November 13, 2022 at 1:06 am

Amazing tutorial! Thanks!

[Reply](#)**tomas**

November 21, 2022 at 7:47 am

Amazing tutorial! great work! Thanks!

[Reply](#)**puterboy**

December 2, 2022 at 2:47 am

Hi Sara, Great Tutorial!



or MCP9808 sensor.

All works fine for cable lengths < ~2m using 4-wire 22 gauge alarm cable.

I tried to lower the i2C clock frequency from the default ~100kHz to a lower value like ~25kHz but when I look at the signal on my scope the clock frequency is unchanged with a pulse duration of ~10uSec.

Since I use the default SCL/SDA pins on the esp32, I tried using the Wire() built-ins as follows:

```
Wire.begin();  
Wire.setClock(10000L);
```

I do this *before* initializing the sensors themselves. I can confirm using Wire.getClock that the frequency is changed at least in the software but as noted above the HW frequency is unchanged.

Why am I not able to change the frequency?

Are the Adafruit libraries overloading the default Wire() variables and thus resetting the bus back to 100kHz?

This seems unlikely since I tried passing &Wire to the initialization code (.begin) for the sensors and the clock frequency was still unchanged...

Any thoughts?

[Reply](#)



Sara Santos

December 2, 2022 at 3:57 pm

Hi.

Yes, I think the Adafruit libraries override the wire library settings.

You need to check the libraries' source code and maybe you might need to change its code.

Regards,

Sara

[Reply](#)



summer

January 18, 2023 at 7:56 pm

Hi Sara,

good post here.

I wonder do you have tested how to set custom I2C for DS3231 rtc module please?

Thanks

Summer

[Reply](#)



Sara Santos

January 18, 2023 at 10:27 pm



Hi.
No, I didn't test that.
Regards,
Sara

[Reply](#)



Martyn Griffin

January 20, 2023 at 5:52 pm

I copied your I2C scanner code and ran it on my esp32 and RTC_DS3231 from my breadboard (thanks for that). When running it gives me the following:

12:50:47.030 -> Scanning...

12:50:47.030 -> I2C device found at address 0x50

12:50:47.068 -> I2C device found at address 0x68

12:50:47.068 -> done

I understand the 0x68, but where does the 0x50 come from?

[Reply](#)



alessio

April 5, 2023 at 2:22 pm

Hi Sara,

if I have a slave device pulling up the SDA and SCL lines to 5V with internal resistors, is it a problem for the esp32 which works at 3.3V?

thank you very much,
Alessio

[Reply](#)



Mat

June 9, 2023 at 1:54 pm

You're such a boss! Admirable tutorial!!

Many thanks to you, I'll use my six BME280 5V I2C with my ESP32 without any multiplexer and with a great start with your code.



**Sara Santos**

June 9, 2023 at 8:51 pm

That's great!

Thank you.

Regards,

Sara

[Reply](#)**neko**

July 28, 2023 at 6:48 pm

Hi, there. I'm trying to interface mpu6050 with esp32 cam ai thinker. How can I do that as the camera already occupies the sda and scl pins?

[Reply](#)**adam**

August 7, 2023 at 5:47 pm

Hi Sara.

I tested bme1/bme2 two I2C bus code and got:

Could not find a valid BME280_1 sensor, check wiring!

checked wiring OK.

what can be the reason?

Thanks

Adam

[Reply](#)**Sara Santos**

August 9, 2023 at 4:51 pm

Hi.

Double-check the I2C bus, and double-check that you actually have a BME280 sensor and not a BMP280:

<https://randomnerdtutorials.com/solved-could-not-find-a-valid-bme280-sensor/>

Regards,

Sara



**JosephN**

August 30, 2023 at 7:22 pm

Hello,

it is possible to have 4 independant i2c software buses? for 4 sensors with same address?

Or how many i2c buses can be handle with this way?

Thanks

[Reply](#)

Leave a Comment

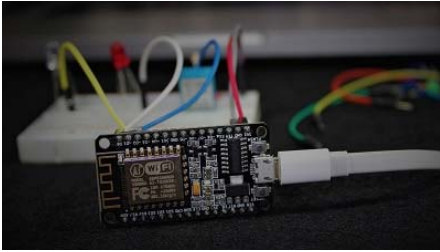
Name *



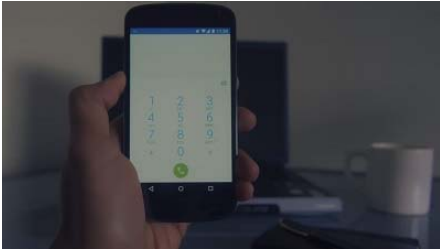
Email *

Website

☒ Notify me of follow-up comments by email.☐ Notify me of new posts by email.[Post Comment](#)



[Home Automation using ESP8266 eBook and video course »](#) Build IoT and home automation projects.



[Build Web Servers with ESP32 and ESP8266 »](#) boards to control outputs and monitor sensors remotely.





































[About](#) [Support](#) [Terms and Conditions](#) [Privacy Policy](#) [Refunds](#) [Complaints' Book](#) [MakerAdvisor.com](#) [Join the Lab](#)

Copyright © 2013-2023 · RandomNerdTutorials.com · All Rights Reserved

[Update Privacy Settings](#)