**How to Use I2C LCD with ESP32 on Arduino IDE (ESP8266 compatible)**

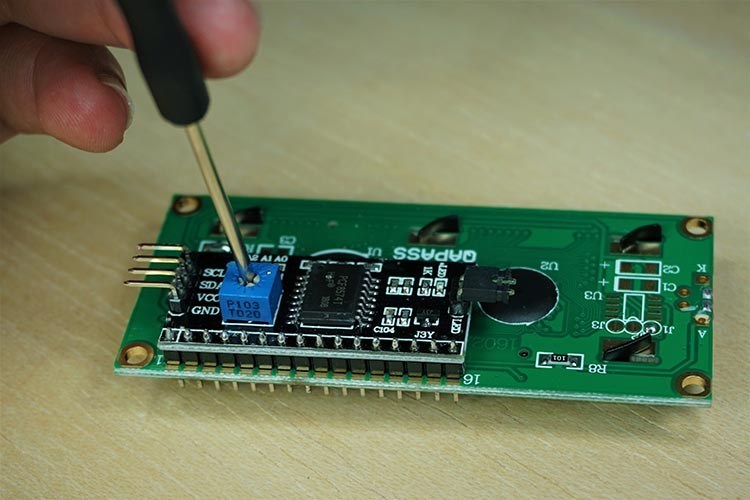
This tutorial shows how to use the I2C LCD (Liquid Crystal Display) with the ESP32 using Arduino IDE. We’ll show you how to wire the display, install the library and try sample code to write text on the LCD: static text, and scroll long messages.



For this tutorial we’ll be using a 16×2 I2C LCD display, but LCDs with other sizes should also work.

The advantage of using an I2C LCD is that the wiring is really simple. You just need to wire the SDA and SCL pins.

Additionally, it comes with a built-in potentiometer you can use to adjust the contrast between the background and the characters on the LCD. On a “regular” LCD you need to add a potentiometer to the circuit to adjust the contrast.



To follow this tutorial you need these parts:

* [ESP32 DOIT DEVKIT V1 Board](https://makeradvisor.com/tools/esp32-dev-board-wi-fi-bluetooth/) – [read ESP32 Development Boards Review and Comparison](https://makeradvisor.com/esp32-development-boards-review-comparison/)
* Optional – [ESP8266 12-E](https://makeradvisor.com/tools/esp8266-esp-12e-nodemcu-wi-fi-development-board/) – read [Best ESP8266 Wi-Fi Development Boards](https://makeradvisor.com/best-esp8266-wi-fi-development-board/)
* [16×2 I2C Liquid Crystal Display (LCD)](https://makeradvisor.com/i2c-lcd-16x2/)
* [Female to female jumper wires](https://makeradvisor.com/tools/jumper-wires-kit-120-pieces/)

You can use the preceding links or go directly to [MakerAdvisor.com/tools](https://makeradvisor.com/tools/?utm_source=rnt&utm_medium=post&utm_campaign=post) to find all the parts for your projects at the best price!

**IT IS IMPORTANT, TO DEFINE IF YOU ARE USING A ESP32 OR ESP3266**

|  |  |  |
| --- | --- | --- |
| I2C LCD | ESP32 | ESP3266 |
| GRD | GND | GND |
| VCC | VIN | VIN |
| SDA | GPIO 21 | GPIO 4 (D2) |
| SCL | GPIO 22 | GPIO 5 (D1) |

|  |  |
| --- | --- |
| **ESP32** | **ESP3266** |
|  |  |

**Installing the LiquidCrystal\_I2C Library**

There are several libraries that work with the I2C LCD. We’re using [this library by Marco Schwartz](https://github.com/marcoschwartz/LiquidCrystal_I2C). Follow the next steps to install the library:

Se baja de internet la librería “LiquidCrystal\_I2C library” o de link

<https://www.arduinolibraries.info/libraries/liquid-crystal-i2-c>

Descomprima el archive zip y se creara una carpeta

Renombre carpeta a LiquidCrystal\_I2C

Mueva la nueva carpeta a la carpeta de librerías de Arduino

Abra nuevamente el Arduino IDE

1. [Click here to download the LiquidCrystal\_I2C library](https://github.com/marcoschwartz/LiquidCrystal_I2C/archive/master.zip). You should have a .zip folder in your Downloads
2. Unzip the *.zip* folder and you should get **LiquidCrystal\_I2C-master**folder
3. Rename your folder from  to**LiquidCrystal\_I2C**
4. Move the **LiquidCrystal\_I2C**folder to your Arduino IDE installation **libraries**folder
5. Finally, re-open your Arduino IDE

**Getting the LCD Address**

Before displaying text on the LCD, you need to find the LCD I2C address. With the LCD properly wired to the ESP32, upload the following I2C Scanner sketch.

#include <Wire.h>

void setup() {

Wire.begin();

Serial.begin(115200);

Serial.println("\nI2C Scanner");

}

void loop() {

byte error, address;

int nDevices;

Serial.println("Scanning...");

nDevices = 0;

for(address = 1; address < 127; address++ ) {

Wire.beginTransmission(address);

error = Wire.endTransmission();

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("Unknow 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 {

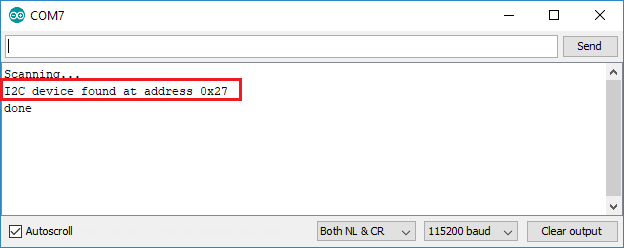
Serial.println("done\n");

}

delay(5000);

}

After uploading the code, open the Serial Monitor at a baud rate of 115200. Press the ESP32 EN button. The I2C address should be displayed in the Serial Monitor.



In this case the address is **0x27**. If you’re using a similar 16×2 display, you’ll probably get the same address.

**Display Static Text on the LCD**

Here’s a very simple sketch example that displays “**Hello, World!**“.

#include <LiquidCrystal\_I2C.h>

// set the LCD number of columns and rows

int lcdColumns = 16;

int lcdRows = 2;

// set LCD address, number of columns and rows

// if you don't know your display address, run an I2C scanner sketch

LiquidCrystal\_I2C lcd(0x27, lcdColumns, lcdRows);

void setup(){

// initialize LCD

lcd.init();

// turn on LCD backlight

lcd.backlight();

}

void loop(){

// set cursor to first column, first row

lcd.setCursor(0, 0); lcd.print("Hello, World!");

delay(1000);

// clears the display to print new message

lcd.clear();

// set cursor to first column, second row

lcd.setCursor(0,1); lcd.print("Hello, World!");

delay(1000);

lcd.clear();

}

It displays the message in the first row, and then in the second row.

**How the code works**

First, you need to include theLiquidCrystal\_I2C library.

#include <LiquidCrystal\_I2C.h>

The next two lines set the number of columns and rows of your LCD display. If you’re using a display with another size, you should modify those variables.

int lcdColumns = 16;

int lcdRows = 2;

Then, you need to set the display address, the number of columns and number of rows. You should use the display address you’ve found in the previous step.

LiquidCrystal\_I2C lcd(0x27, lcdColumns, lcdRows);

In the setup(), first initialize the display with the init() method.

lcd.init();

Then, turn on the LCD backlight, so that you’re able to read the characters on the display.

lcd.backlight();

To display a message on the screen, first you need to set the cursor to where you want your message to be written. The following line sets the cursor to the first column, first row.

lcd.setCursor(0, 0);

**Note**: 0 corresponds to the first column, 1 to the second column, and so on…

Then, you can finally print your message on the display using the print()method.

lcd.print("Hello, World!");

Wait one second, and then clean the display with the clear() method.

lcd.clear();

After that, set the cursor to a new position: first column, second row.

lcd.setCursor(0,1);

Then, the process is repeated.

So, here’s a summary of the functions to manipulate and write on the display:

* lcd.init(): initializes the display
* lcd.backlight(): turns the LCD backlight on
* lcd.setCursor(int column, int row): sets the cursor to the specified column and row
* lcd.print(String message): displays the message on the display
* lcd.clear(): clears the display

This example works well to display static text no longer than 16 characters.

**Display Scrolling Text on the LCD**

Scrolling text on the LCD is specially useful when you want to display messages longer than 16 characters. The library comes with built-in functions that allows you to scroll text. However, many people experience problems with those functions because:

* The function scrolls text on both rows. So, you can’t have a fixed row and a scrolling row;
* It doesn’t work properly if you try to display messages longer than 16 characters.

So, we’ve created a sample sketch with a function you can use in your projects to scroll longer messages.

The following sketch displays a static message in the first row and a scrolling message longer than 16 characters in the second row.

#include <LiquidCrystal\_I2C.h>

// set the LCD number of columns and rows

int lcdColumns = 16;

int lcdRows = 2;

// set LCD address, number of columns and rows

// if you don't know your display address, run an I2C scanner sketch

LiquidCrystal\_I2C lcd(0x27, lcdColumns, lcdRows);

String messageStatic = "Static message";

String messageToScroll = "This is a scrolling message with more than 16 characters";

// Function to scroll text

// The function acepts the following arguments:

// row: row number where the text will be displayed

// message: message to scroll

// delayTime: delay between each character shifting

// lcdColumns: number of columns of your LCD

void scrollText(int row, String message, int delayTime, int lcdColumns) {

for (int i=0; i < lcdColumns; i++) {

message = " " + message;

}

message = message + " ";

for (int pos = 0; pos < message.length(); pos++) {

lcd.setCursor(0, row);

lcd.print(message.substring(pos, pos + lcdColumns));

delay(delayTime);

}

}

void setup(){

// initialize LCD

lcd.init();

// turn on LCD backlight

lcd.backlight();

}

void loop(){

// set cursor to first column, first row

lcd.setCursor(0, 0);

// print static message

lcd.print(messageStatic);

// print scrolling message

scrollText(1, messageToScroll, 250, lcdColumns);

}

After reading the previous section, you should be familiar on how this sketch works, so we’ll just take a look at the newly created function: scrollText()

void scrollText(int row, String message, int delayTime, int lcdColumns) {

for (int i=0; i < lcdColumns; i++) {

message = " " + message;

}

message = message + " ";

for (int pos = 0; pos < message.length(); pos++) {

lcd.setCursor(0, row);

lcd.print(message.substring(pos, pos + lcdColumns));

delay(delayTime);

}

}

To use this function you should pass four arguments:

* row: row number where the text will be display
* message: message to scroll
* delayTime: delay between each character shifting. Higher delay times will result in slower text shifting, and lower delay times will result in faster text shifting.
* lcdColumns: number of columns of your LCD

In our code, here’s how we use the scrollText() function:

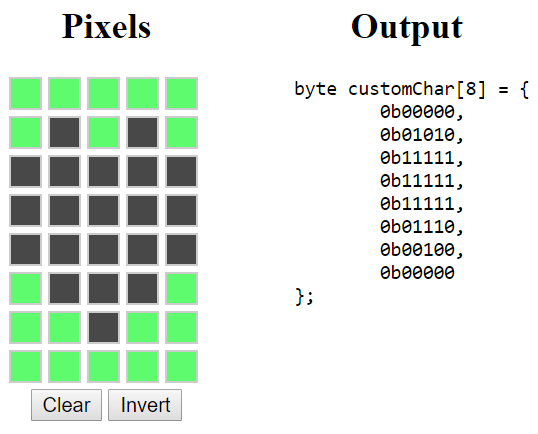
scrollText(1, messageToScroll, 250, lcdColumns);

The messageToScroll variable is displayed in the second row (1 corresponds to the second row), with a delay time of 250 ms (the GIF image is speed up 1.5x).

**Display Custom Characters**

In a 16×2 LCD there are 32 blocks where you can display characters. Each block is made out of 5×8 tiny pixels. You can display custom characters by defining the state of each tiny pixel. For that, you can create a byte variable to hold  the state of each pixel.

To create your custom character, you can go [here](https://omerk.github.io/lcdchargen/) to generate the byte variable for your character. For example, a heart:



Copy the byte variable to your code (before the setup()). You can call it heart:

byte heart[8] = {

0b00000,

0b01010,

0b11111,

0b11111,

0b11111,

0b01110,

0b00100,

0b00000

};

Then, in the setup(), create a custom character using the createChar()function. This function accepts as arguments a location to allocate the char and the char variable as follows:

lcd.createChar(0, heart);

Then, in the loop(), set the cursor to where you want the character to be displayed:

lcd.setCursor(0, 0);

Use the write() method to display the character. Pass the location where the character is allocated, as follows:

lcd.write(0);

**Wrapping Up**

In summary, in this tutorial we’ve shown you how to use an I2C LCD display with the ESP32/ESP8266 with Arduino IDE: how to display static text, scrolling text and custom characters. This tutorial also works with the Arduino board, you just need to change the pin assignment to use the Arduino I2C pins.

We have other tutorials with ESP32 that you may find useful:

* [ESP32 with Multiple DS18B20 Temperature Sensors](https://randomnerdtutorials.com/esp32-with-multiple-ds18b20-temperature-sensors/)
* [ESP32 Data Logging Temperature to MicroSD Card](https://randomnerdtutorials.com/esp32-data-logging-temperature-to-microsd-card/)
* [ESP32 with DC Motor and L298N Motor Driver – Control Speed and Direction](https://randomnerdtutorials.com/esp32-dc-motor-l298n-motor-driver-control-speed-direction/)
* [More ESP32 tutorials](https://randomnerdtutorials.com/category/esp32/)

We hope you’ve found this tutorial useful. If you like ESP32 and you want to learn more, we recommend enrolling in [**Learn ESP32 with Arduino IDE** course](https://randomnerdtutorials.com/learn-esp32-with-arduino-ide/).

Thanks for reading.

[**[eBook] MicroPython Programming with ESP32 and ESP8266**](https://randomnerdtutorials.com/micropython-programming-with-esp32-and-esp8266/)

Learn how to program and build projects with the ESP32 and ESP8266 using MicroPython firmware [**DOWNLOAD »**](https://randomnerdtutorials.com/micropython-programming-with-esp32-and-esp8266/)