



WHADDA WPB109 ESP32 Development Board User Manual

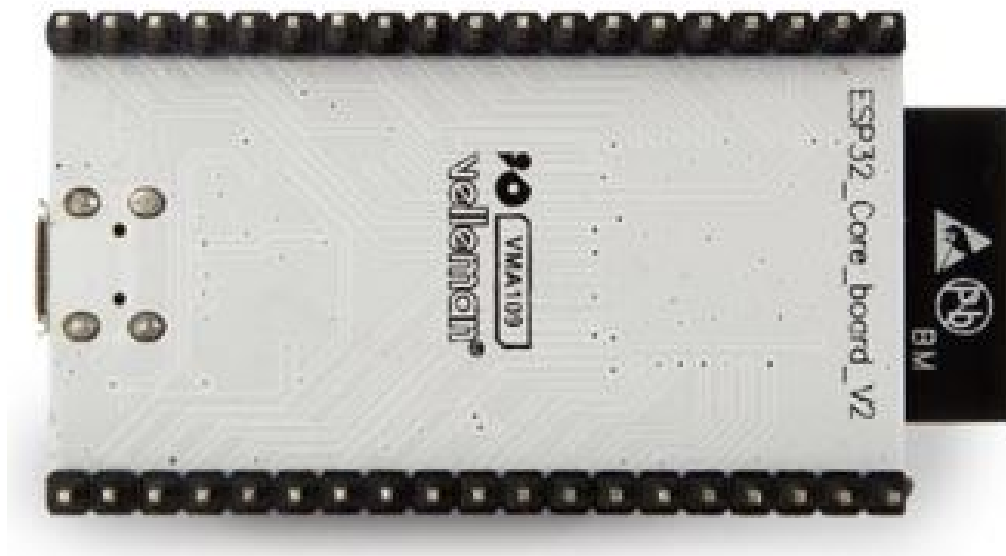
[Home](#) » [WHADDA](#) » WHADDA WPB109 ESP32 Development Board User Manual 

Contents

- [1 WHADDA WPB109 ESP32 Development Board](#)
- [2 Introduction](#)
- [3 Safety Instructions](#)
- [4 General Guidelines](#)
- [5 What is Arduino®](#)
- [6 Product overview](#)
- [7 Specifications](#)
- [8 Functional overview](#)
- [9 Getting started](#)
- [10 WiFi connection example](#)
- [11 Documents / Resources](#)
 - [11.1 References](#)

WHADDA

WHADDA WPB109 ESP32 Development Board



Introduction

To all residents of the European Union Important environmental information about this product This symbol on the device or the package indicates that disposal of the device after its lifecycle could harm the environment. Do not dispose of the unit (or batteries) as unsorted municipal waste; it should be taken to a specialized company for recycling. This device should be returned to your distributor or to a local recycling service. Respect the local environmental rules. If in doubt, contact your local waste disposal authorities. Thank you for choosing Whadda! Please read the manual thoroughly before bringing this device into service. If the device was damaged in transit, do not install or use it and contact your dealer.

Safety Instructions

- Read and understand this manual and all safety signs before using this appliance.
- For indoor use only.
- This device can be used by children aged from 8 years and above, and persons with reduced physical, sensory or mental capabilities or lack of experience and knowledge if they have been given supervision or instruction concerning the use of the device in a safe way and understand the hazards involved. Children shall not play with the device. Cleaning and user maintenance shall not be made by children without supervision.

General Guidelines

- Refer to the Velleman® Service and Quality Warranty on the last pages of this manual.
- All modifications of the device are forbidden for safety reasons. Damage caused by user modifications to the device is not covered by the warranty.
- Only use the device for its intended purpose. Using the device in an unauthorised way will void the warranty.
- Damage caused by disregard of certain guidelines in this manual is not covered by the warranty and the dealer will not accept responsibility for any ensuing defects or problems.
- Nor Velleman nv nor its dealers can be held responsible for any damage (extraordinary, incidental or indirect) – of any nature (financial, physical...) arising from the possession, use or failure of this product.
- Keep this manual for future reference.

What is Arduino®

Arduino® is an open-source prototyping platform based on easy-to-use hardware and software. Arduino® boards are able to read inputs – light-on sensor, a finger on a button or a Twitter message –and turn it into an output – activating of a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so, you use the Arduino programming language (based on Wiring) and the Arduino® software IDE (based on Processing). Additional shields/modules/components are required for reading a twitter message or publishing online. Surf to www.arduino.cc for more information

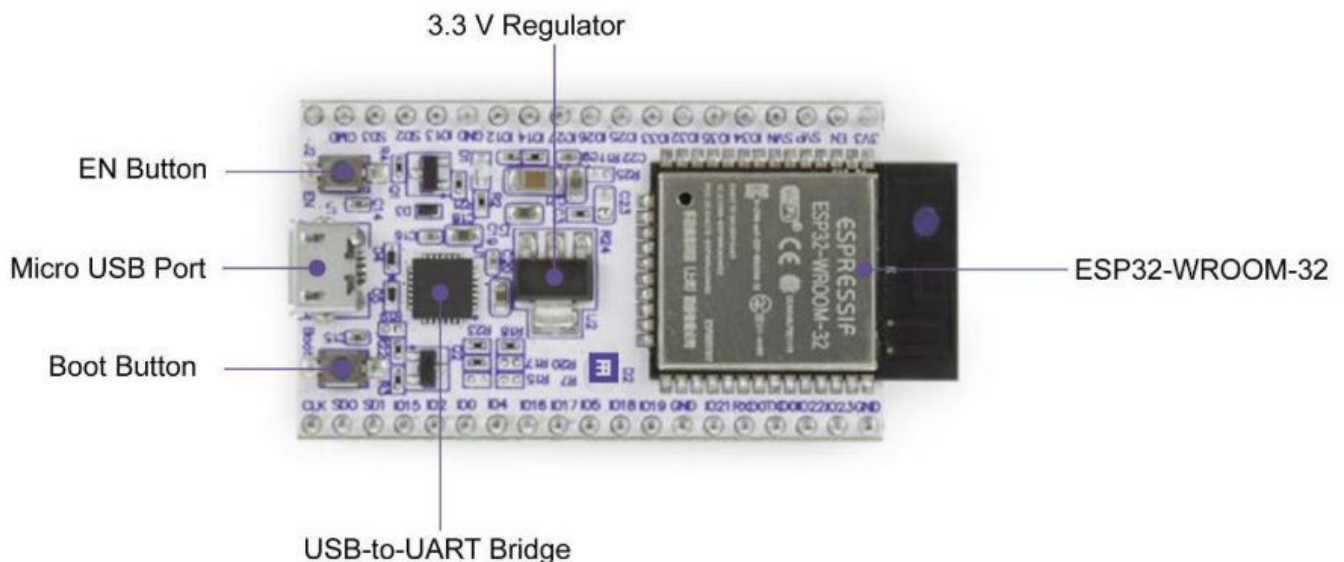
Product overview

The Whadda WPB109 ESP32 development board is a comprehensive development platform for Espressif's ESP32, the upgraded cousin of the popular ESP8266. Like the ESP8266, the ESP32 is a WiFi-enabled microcontroller, but to that it adds support for Bluetooth low-energy (i.e BLE, BT4.0, Bluetooth Smart), and 28 I/O pins. The ESP32's power and versatility makes it the ideal candidate to serve as the brains of your next IoT project.

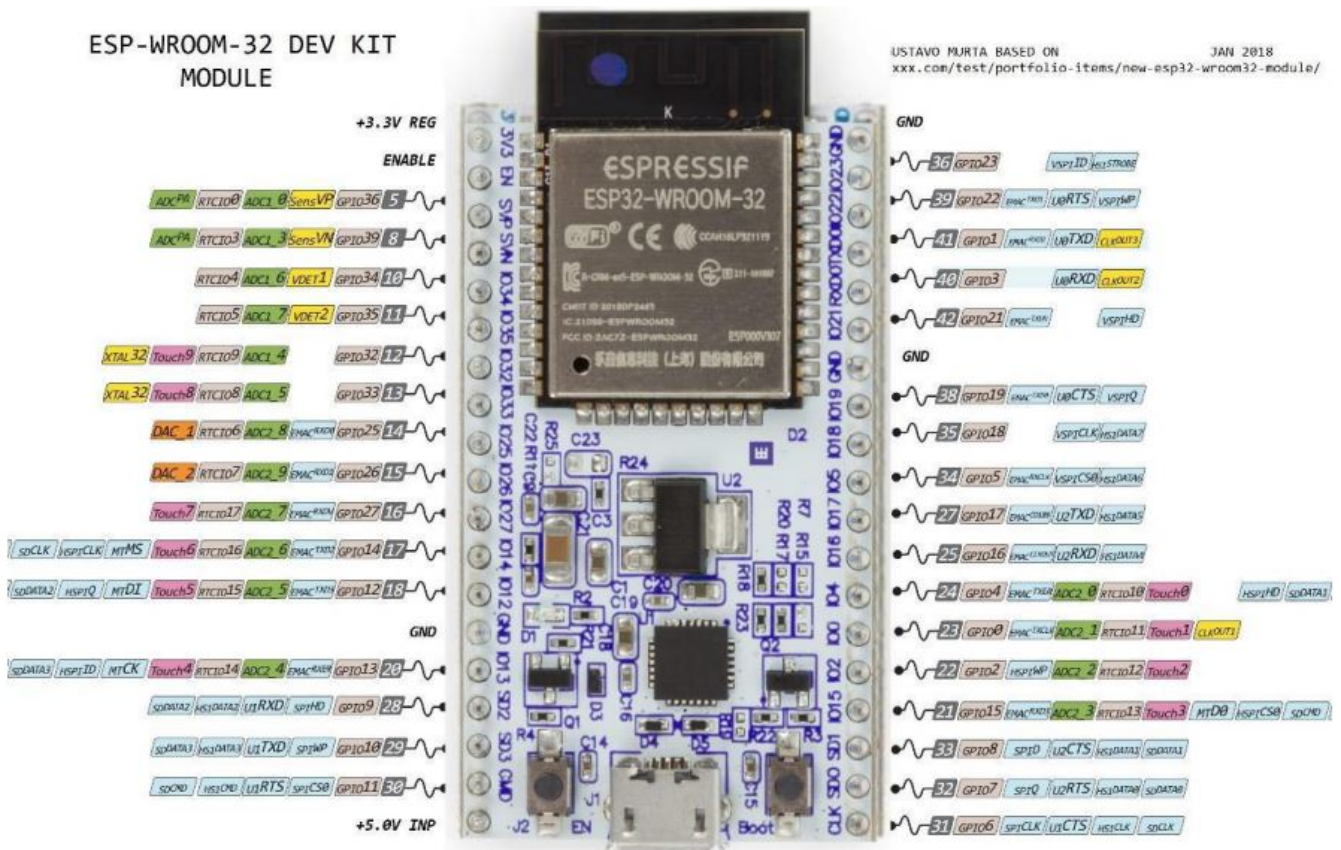
Specifications

- Chipset: ESPRESSIF ESP-WROOM-32 CPU: Xtensa dual-core (or single-core) 32-bit LX6 microprocessor
- Co-CPU: ultra low power (ULP) co-processor GPIO Pins 28
- Memory:
 - RAM: 520 KB of SRAM ROM: 448 KB
- Wireless connectivity:
 - WiFi: 802.11 b/g/n
 - Bluetooth®: v4.2 BR/EDR and BLE
- **Power management:**
 - max. current consumption: 300 mA
 - Deep sleep power consumption: 10 μ A
 - max. battery input voltage: 6 V
 - max. battery charge current: 450 mA
 - Dimensions (W x L x H): 27.9 x 54.4.9 x 19mm

Functional overview



Key Component	Description
ESP32-WROOM-32	A module with ESP32 at its core.
EN Button	Reset button
Boot Button	Download button. Holding down Boot and then pressing EN initiates Firm ware Download mode for downloading firmware through the serial port.
USB-to-UART Bridge	Converts USB into UART serial in order to facilitate the communication between the ESP32 and pc
Micro USB Port	USB interface. Power supply for the board as well as the communication interface between a computer and the ESP32 module.
3.3 V Regulator	Converts 5 V from USB to 3.3 V needed to supply the ESP32 module

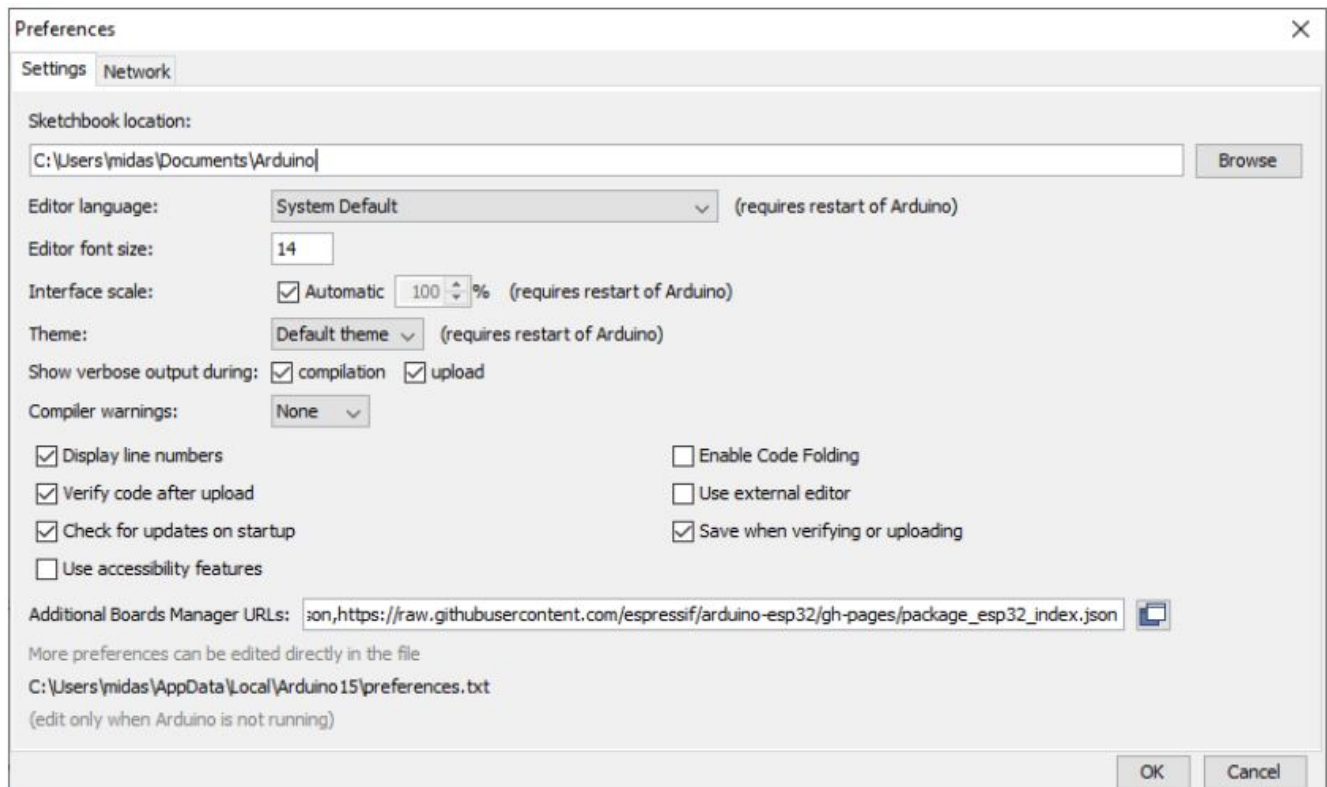


Getting started

Installing the required software

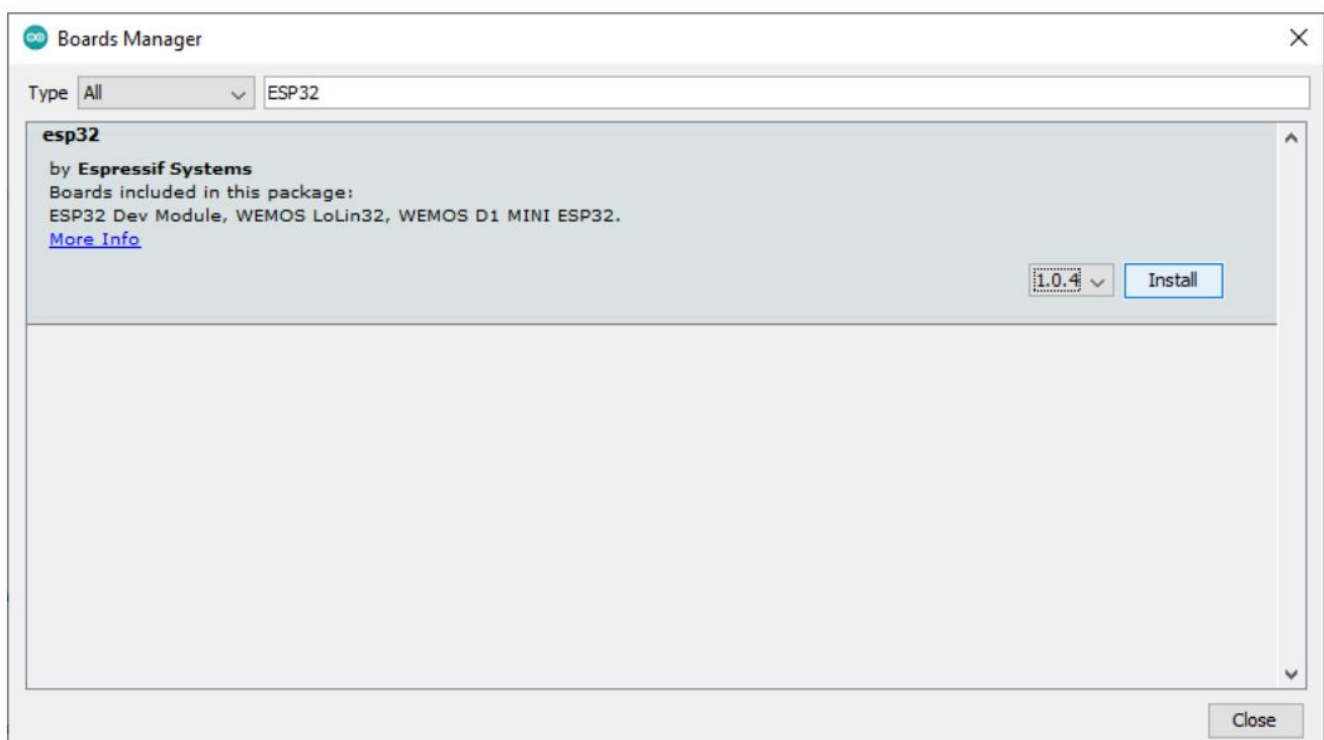
1. First, make sure you have the latest version of the Arduino IDE installed on your computer. You can download the latest version by going to www.arduino.cc/en/software.
2. Open the Arduino IDE, and open the preferences menu by going to File > Preferences. Enter the following URL into the “Additional Boards Manager URLs” field:

https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json , and



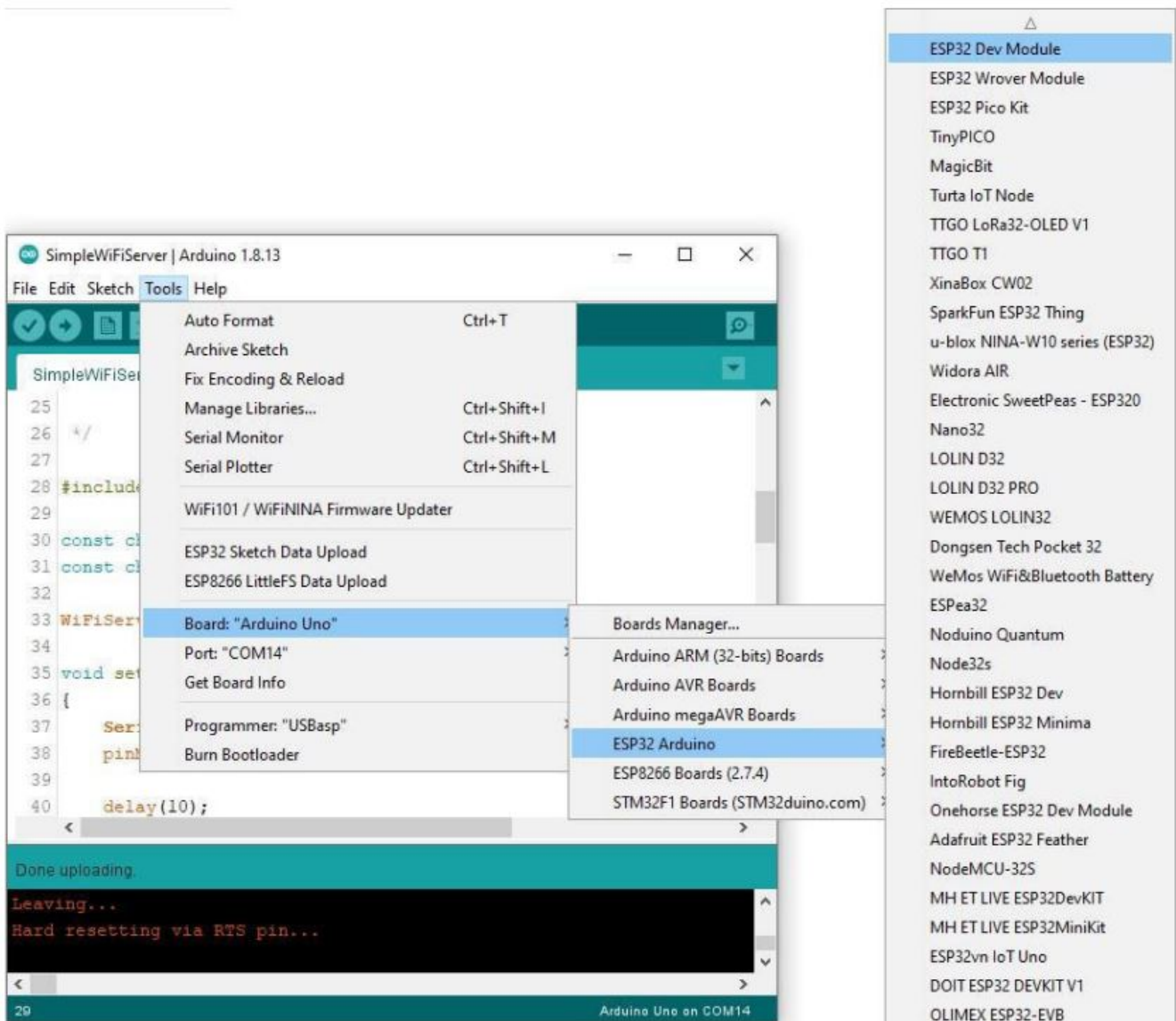
hit “OK”.

3. Open the Boards Manager from Tools > Board menu and install the esp32 platform by putting ESP32 into the search field, selecting the most recent version of the esp32 core (by Espressif Systems), and clicking “Install”.

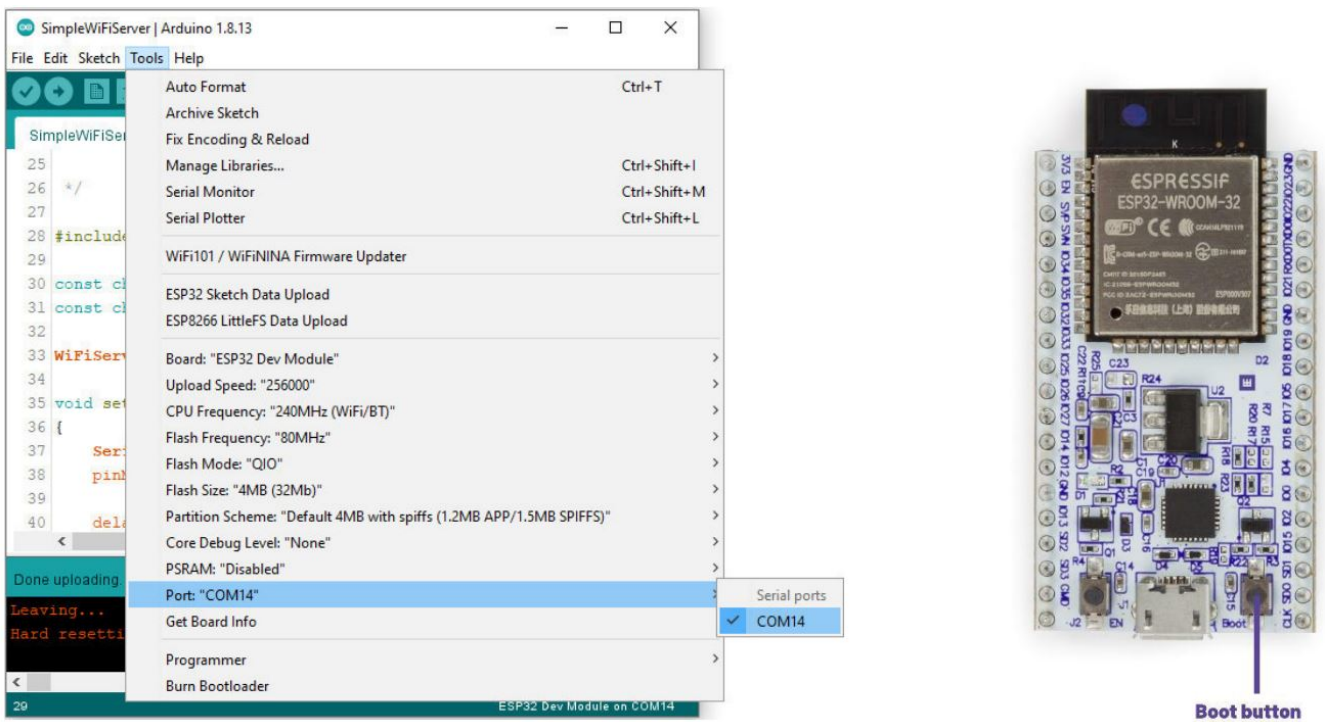


Uploading the first sketch to the board

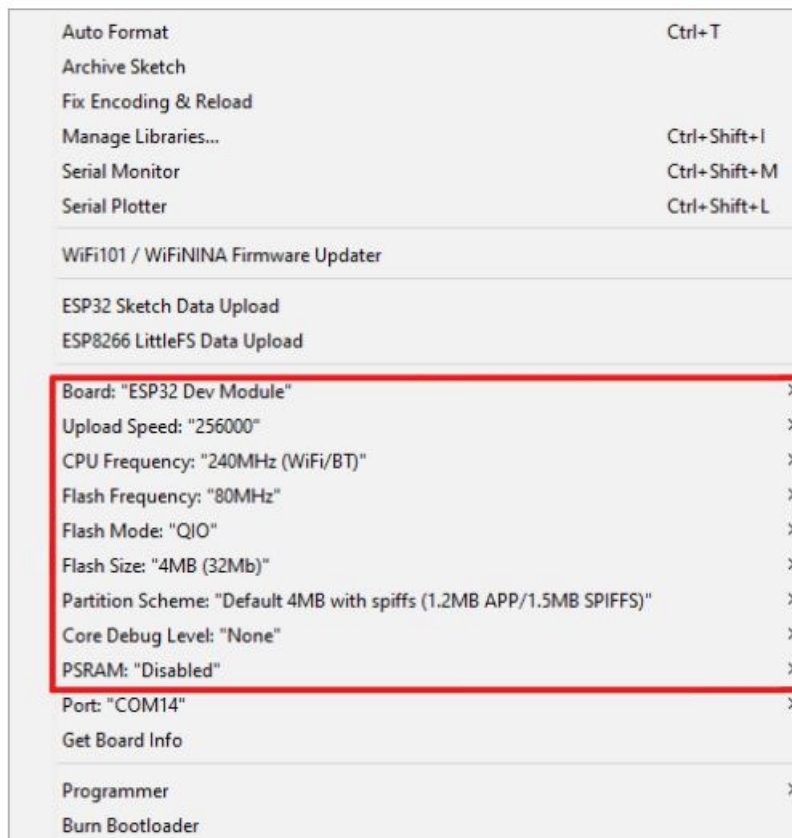
4. Once the ESP32 core has been installed, open the tools menu and select the ESP32 Dev module board by going to: Tools > Board:"..." > ESP32 Arduino > ESP32 Dev Module



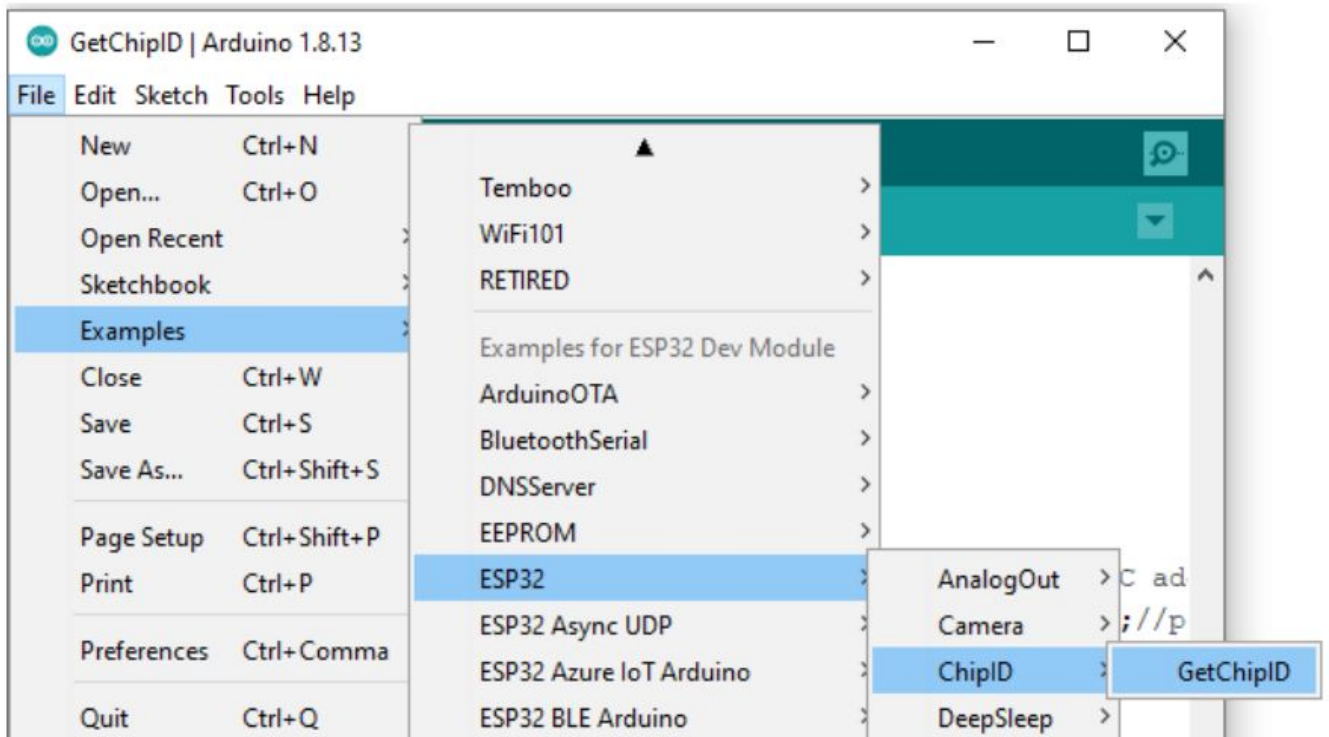
5. Connect the Whadda ESP32 module to your pc using a micro USB cable. Open the tools menu again and check if a new serial port has been added to the port list and select it (Tools > Port:"..." >). If this is not the case, you may need to install a new driver to enable the ESP32 to properly connect to your computer. Go to <https://www.silabs.com/developers/usb-to-uart-bridge-vcp-drivers> to download and install the driver. Reconnect the ESP32 and restart the Arduino IDE once the process has finished.




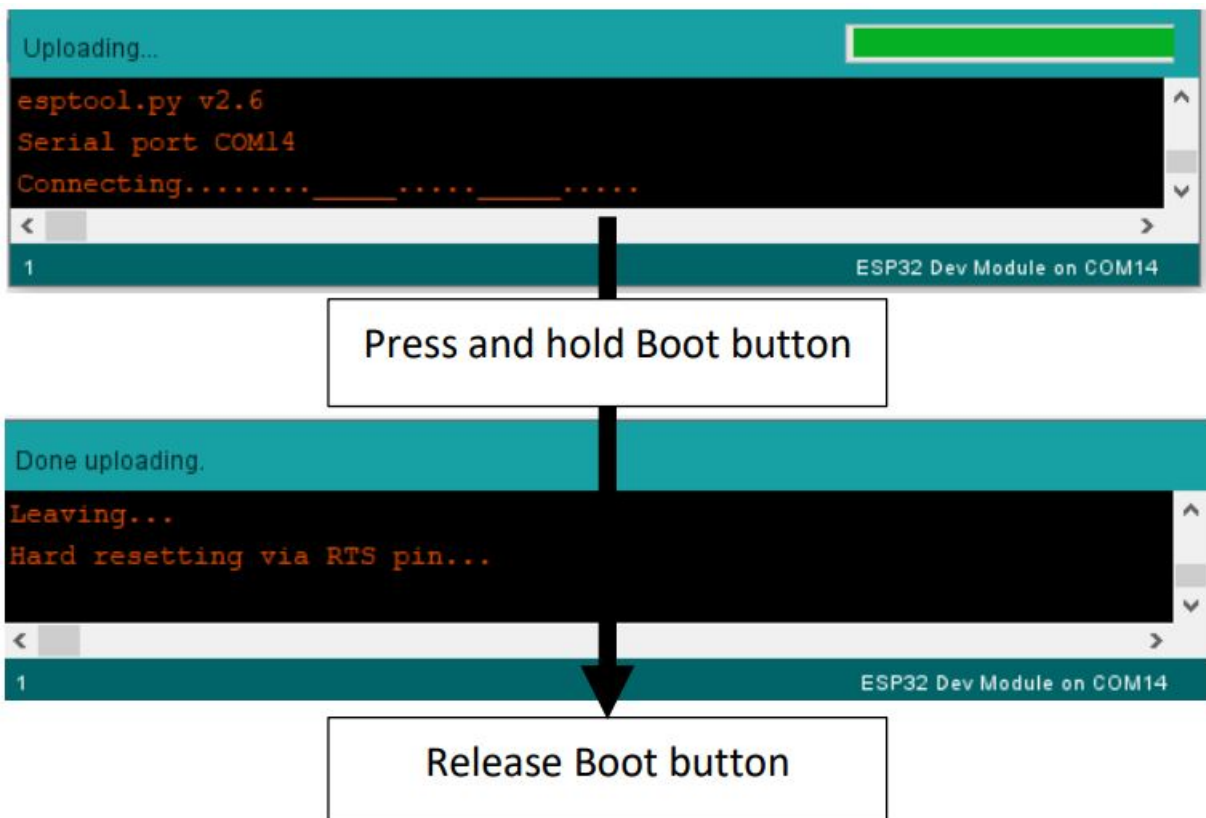
6. Check that the following settings have been selected in the tools board menu:



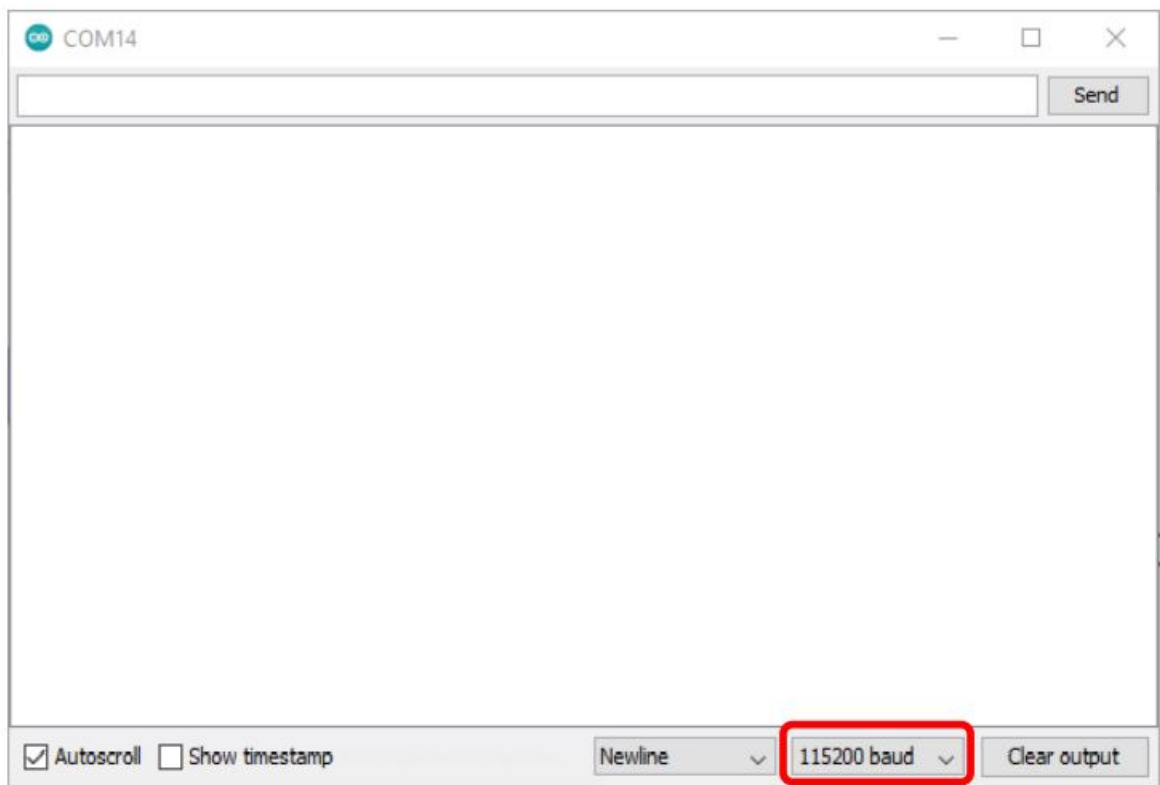
7. Select an example sketch from the "Examples for ESP32 Dev Module" in File > Examples. We recommend to run the example called "GetChipID" as a starting point, which can be found under File > Examples > ESP32 > ChipID.



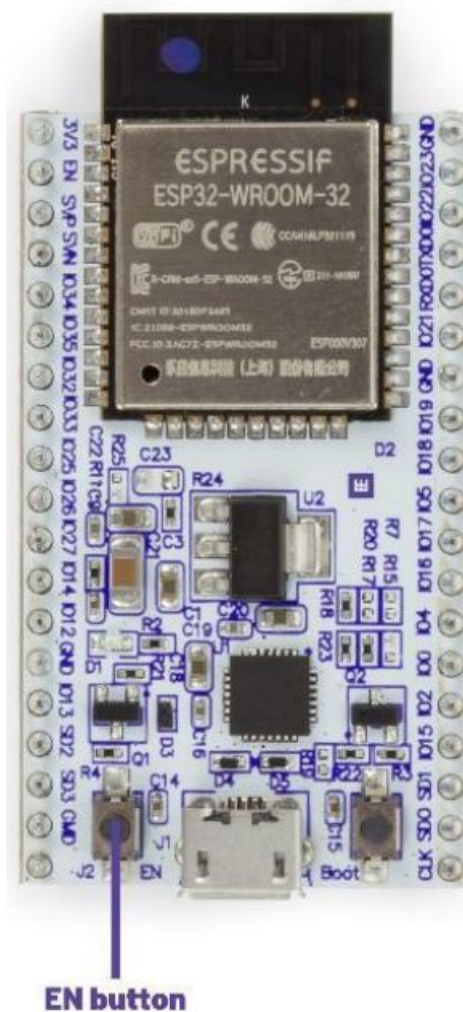
8. Click the Upload button (), and monitor the info messages at the bottom. Once the message "Connecting..." appears, press and hold the Boot button on the ESP32 until the uploading process has finished.



9. Open the serial monitor (), and check that the baudrate is set to 115200 baud:



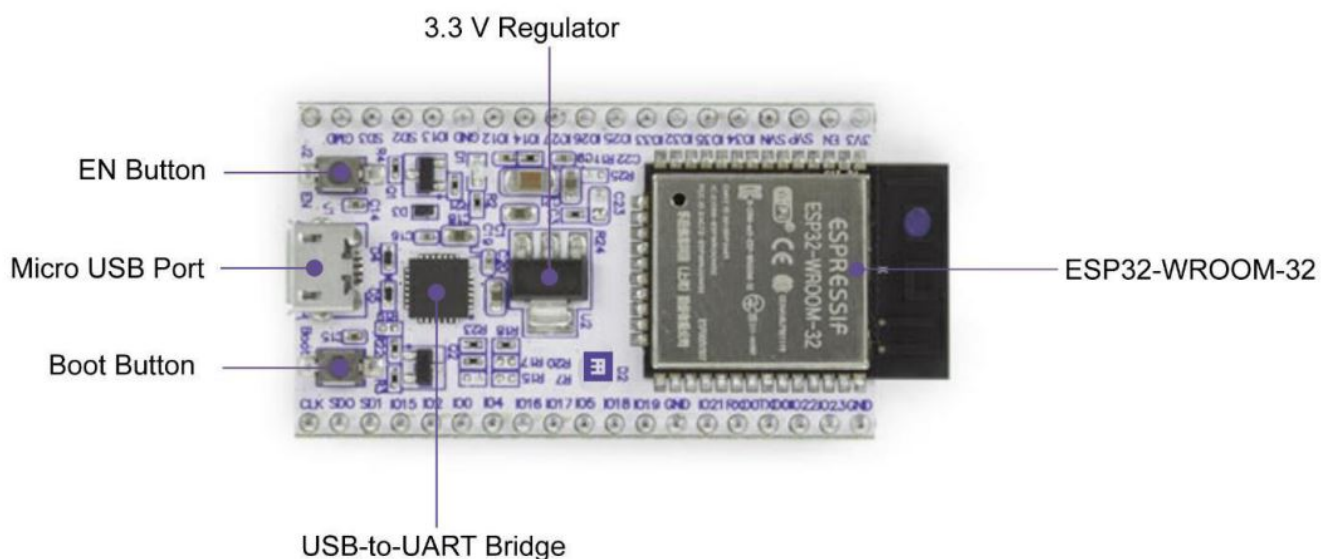
10. Press the Reset/EN button, debug messages should start appearing on the serial monitor, together with the Chip ID (If the GetChipID example was uploaded).



```
COM14

ets Jun  8 2016 00:22:57

rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:1
load:0x3fff0018,len:4
load:0x3fff001c,len:1216
ho 0 tail 12 room 4
load:0x40078000,len:9720
ho 0 tail 12 room 4
load:0x40080400,len:6352
entry 0x400806b8
ESP32 Chip ID = 60E71312CFA4
ESP32 Chip ID = 60E71312CFA4
ESP32 Chip ID = 60E71312CFA4
```



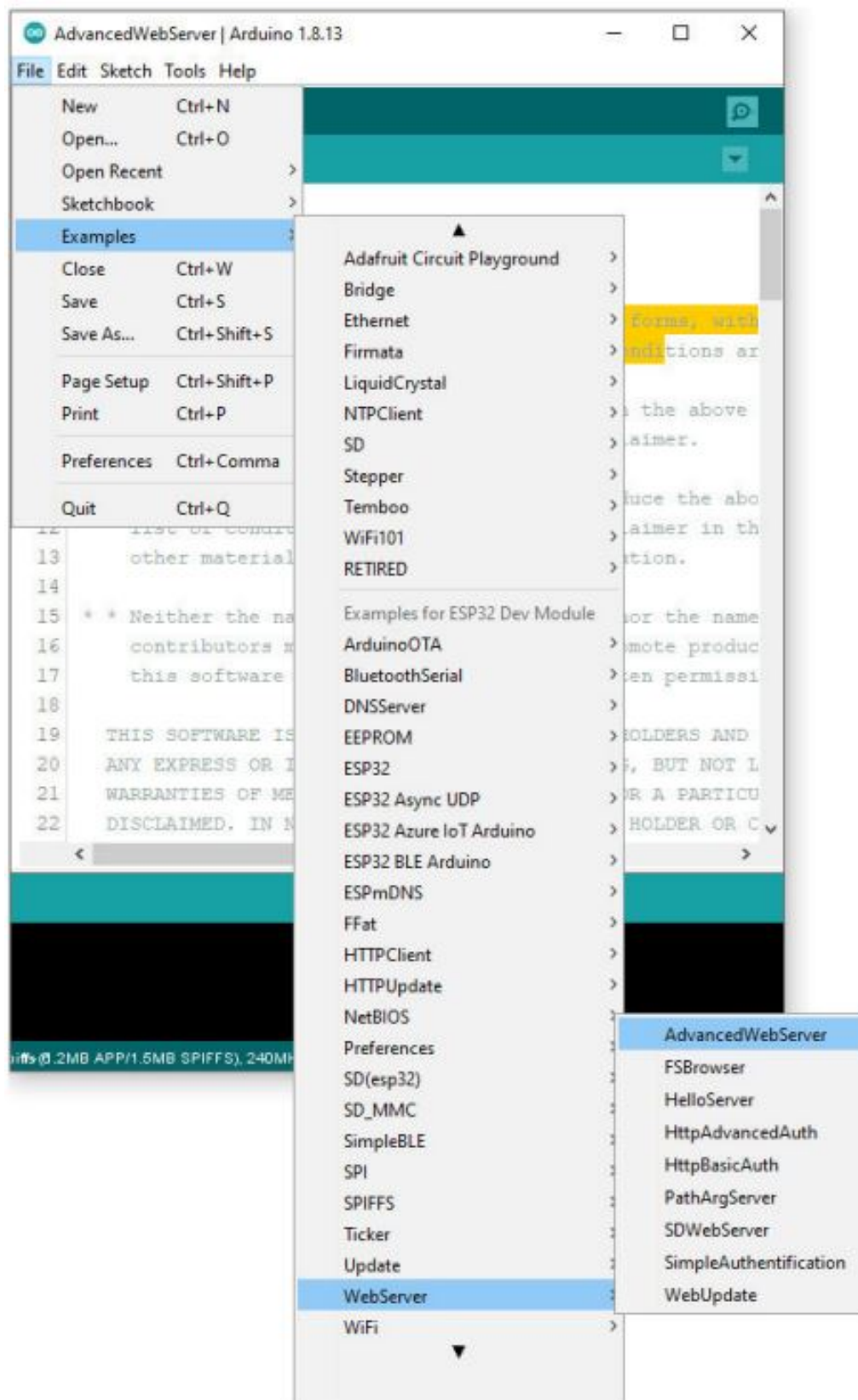
Having trouble?

Restart the Arduino IDE and reconnect the ESP32 board. You can check if the driver has been properly installed by checking Device manager on Windows under COM Ports to see if a Silicon Labs CP210x device is recognized. Under Mac OS you can run the command `ls /dev/{tty,cu}.*` in the terminal to check this.

WiFi connection example

The ESP32 really shines in applications where WiFi connectivity is required. The following example will harness this extra functionality by having the ESP module function as a basic webserver.

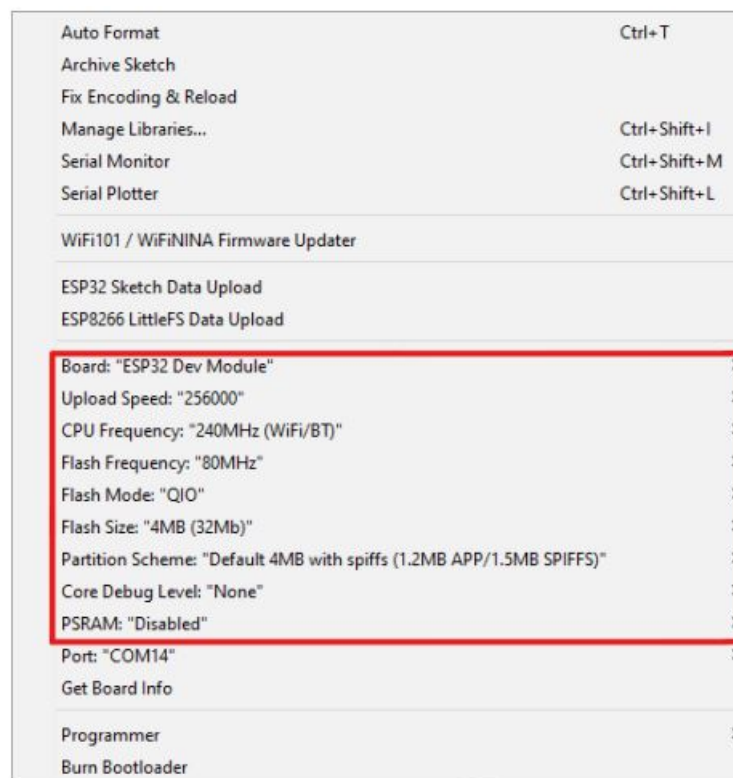
1. Open the Arduino IDE, and open the AdvancedWebServer example by going to File > Examples > WebServer > AdvancedWebServer




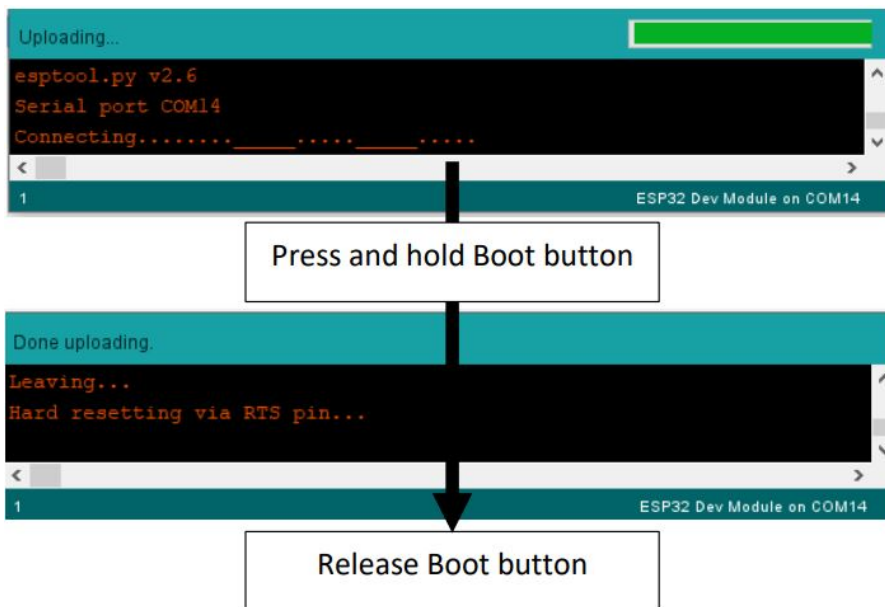
2. Replace YourSSIDHere with your own WiFi network name, and replace YourPSKHere with your WiFi network password.


```
AdvancedWebServer
29 */
30
31 #include <WiFi.h>
32 #include <WiFiClient.h>
33 #include <WebServer.h>
34 #include <ESPmDNS.h>
35
36 const char *ssid = "YourSSIDHere";
37 const char *password = "YourPSKHere";
38
```

3. Connect your ESP32 to your pc (if you haven't already), and make sure that the correct board settings in the Tools menu are set and that the proper serial communication port has been selected.



4. Click the Upload button (), and monitor the info messages at the bottom. Once the message "Connecting..." appears, press and hold the Boot button on the ESP32 until the uploading process has finished.

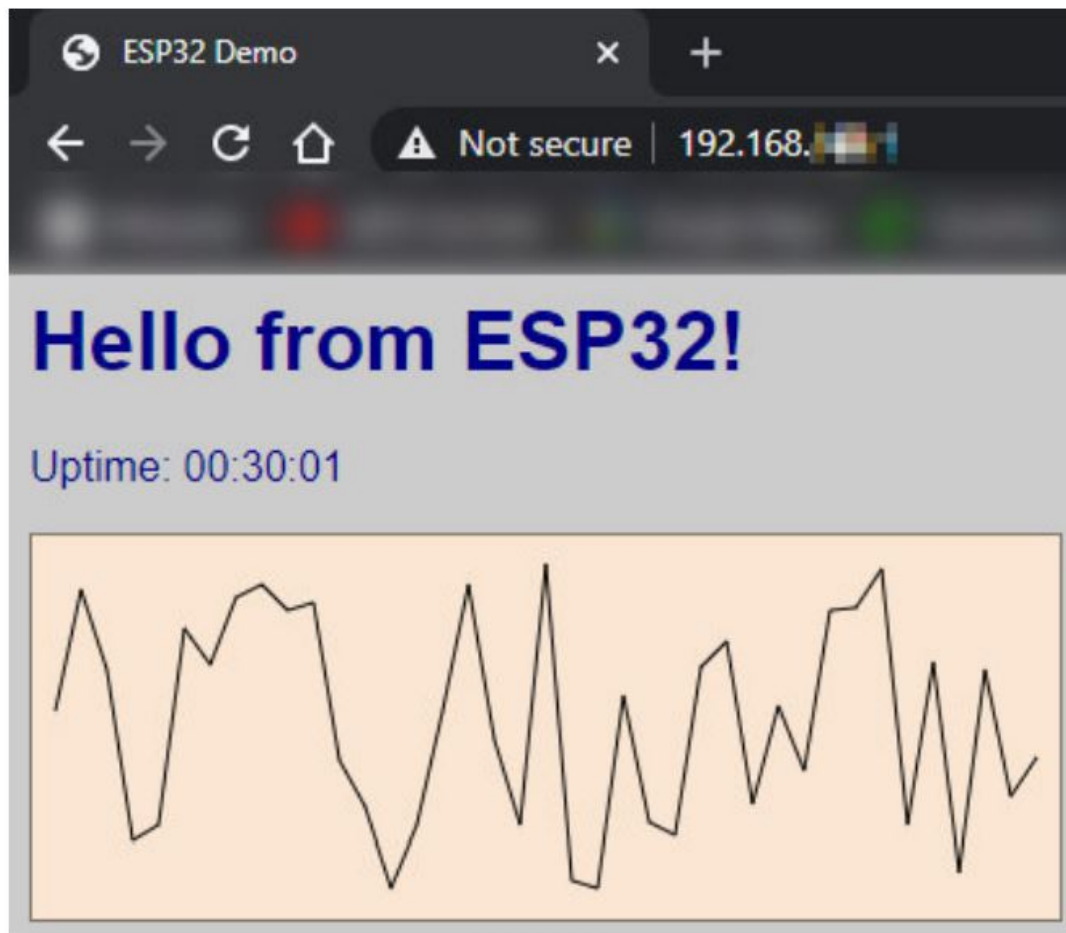


5. Open the serial monitor (), and check that the baudrate is set to 115200 baud:
6. Press the Reset/EN button, debug messages should start appearing on the serial monitor, together with status info about the network connection and the IP-address. Take a note of the IP address:

Is the ESP32 having trouble to connect to your WiFi network?

Check that the WiFi network name and password have been set up correctly, and that the ESP32 is in range of your WiFi access point. The ESP32 has a relatively small antenna so it might have more difficulties to pick up the WiFi signal at a certain location than your PC.

7. Open our web browser and try to connect to the ESP32 by entering it's ip addresses in the address bar. You should get a webpage that shows a randomly generated graph from the ESP32




What to do next with my Whadda ESP32 board?

Check out some of the other ESP32 examples that come preloaded in the Arduino IDE. You could try out the Bluetooth functionality by trying the example sketches in the ESP32 BLE Arduino folder, or try out the internal magnetic (hall) sensor test sketch (ESP32 > HallSensor). Once you tried out a few different examples you can try to edit the code to your liking, and combine the various examples to come up with your own unique projects! Also check out these tutorials made by our friends at last minute engineers:

lastminuteengineers.com/electronics/esp32-projects/

Modifications and typographical errors reserved – © Velleman Group nv, Legen Heirweg 33 – 9890 Gavere WPB109-26082021.

Documents / Resources

	<p>WHADDA WPB109 ESP32 Development Board [pdf] User Manual WPB109 ESP32 Development Board, WPB109, ESP32 Development Board, Development Board, Board</p>
---	--

References

- [ESP32 Projects - Last Minute Engineers](#)
- [Whadda - Exciting Electronics](#)
- [ESP32 Projects - Last Minute Engineers](#)
- raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json

-  [CP210x USB to UART Bridge VCP Drivers - Silicon Labs](#)

Manuals+.