

Modular tester Programming instructions ESP32

Version 31/10/2024 © Jef Collin

The ESP32 needs to be programmed before use, the supplied .INO files hold the source code. The Arduino environment is used to program the processor module directly, therefore you need to install it, compile the source code and write the compiled code to the ESP32 module. These instructions will guide you through the process:

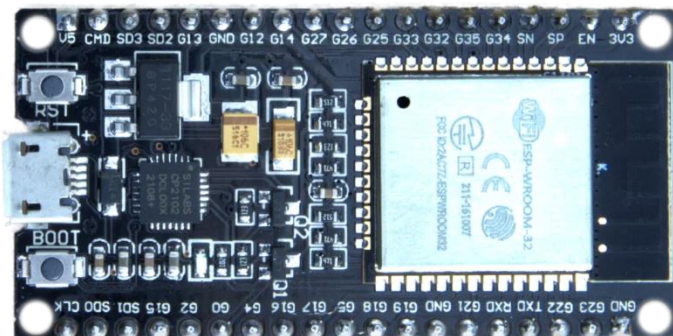
- Installing the Arduino IDE environment
- Copying the source code
- Installing the additional libraries
- Setup the IDE and libraries
- Programming the ESP32

Note: the Arduino IDE is a moving target, constant development is done on the IDE and libraries, often there are breaking changes in the libraries and the code no longer compiles.

To avoid these pitfalls I recommend using Arduino IDE version 1.8.19 in “portable mode”, this allows you to store all you need in one folder and “freeze” the entire setup in a known working state. You could even run the entire setup from an USB stick. This is not (yet) possible with version 2.

One such portable setup will be used for all modular tester source codes, this means that only one set of libraries needs to be maintained.


All tester modules use the same type of ESP32 devkit, this 38 pin version, other types will not work because of different pins assignments.



A well-documented modification to the module is recommended, it removes the need to press buttons to enter code download mode. Install a 10uF electrolytic capacitor (10V or higher), negative terminal to the shield of the CPU, positive terminal to the EN pin.



Make a new folder.

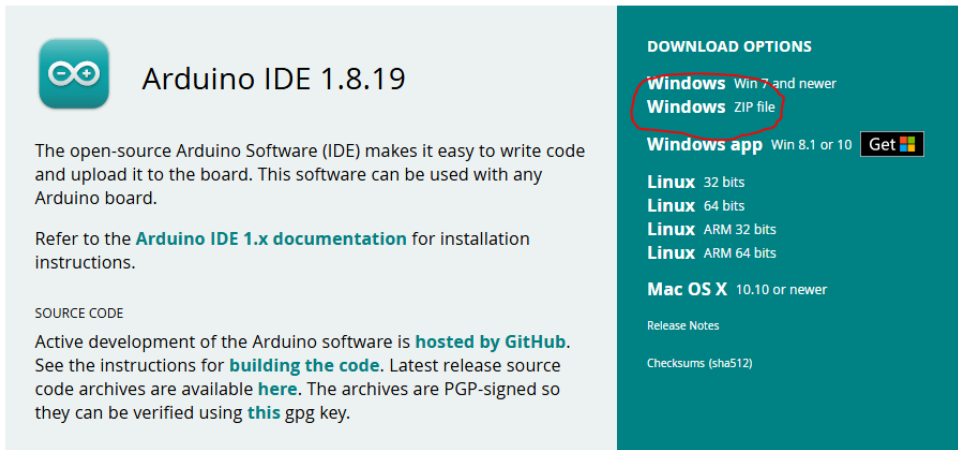
▼  modular_tester

Go to the Arduino download page.

<https://www.arduino.cc/en/software>

Download the ZIP file for version 1.8.19 (final version of IDE 1), DO NOT download the installer.

Legacy IDE (1.8.X)



Arduino IDE 1.8.19

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board.

Refer to the [Arduino IDE 1.x documentation](#) for installation instructions.

SOURCE CODE

Active development of the Arduino software is [hosted by GitHub](#). See the instructions for [building the code](#). Latest release source code archives are available [here](#). The archives are PGP-signed so they can be verified using [this](#) gpg key.

DOWNLOAD OPTIONS


- Windows** Win 7 and newer
- Windows** ZIP file
- Windows app** Win 8.1 or 10 [Get](#)
- Linux** 32 bits
- Linux** 64 bits
- Linux** ARM 32 bits
- Linux** ARM 64 bits
- Mac OS X** 10.10 or newer


[Release Notes](#)

[Checksums \(sha512\)](#)

Copy the ZIP file to the new folder and unpack, do not run the arduino.exe software yet (you can delete the ZIP file when done unpacking).


This will make a new folder with the Arduino IDE.


▼  modular_tester










>  arduino-1.8.19

Open this folder, depending on how you unpacked there might be extra folder levels, open until you get the tree structure below.

In this structure create a new folder called “portable”.

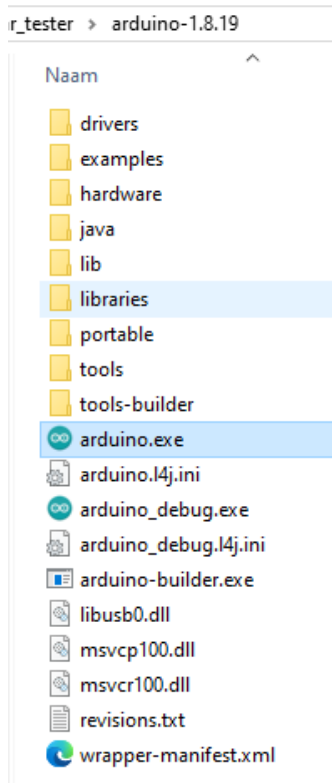
▼  modular_tester

▼  arduino-1.8.19

- >  drivers
- >  examples
- >  hardware
- >  java
- >  lib
- >  libraries
- >  portable
- >  tools
- >  tools-builder

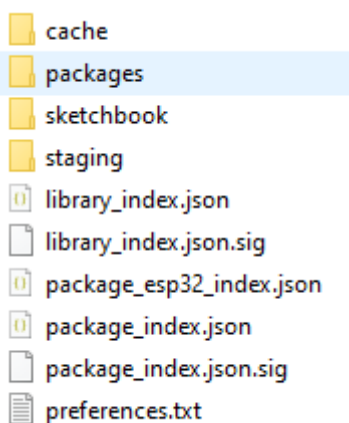
Start the Arduino IDE by clicking *arduino.exe* in folder Arduino_1.8.19.

Important: this will be the method of starting the Arduino IDE for this development, do not try to open it by clicking on the INO source file since Windows will try to open any Arduino IDE you have installed with the standard installer, since this one is not “installed” Windows does not know the correct one to open. You can make multiple folders with other development setups for other projects.



The IDE will detect the portable folder and will create additional folders within it. Wait a moment until the IDE is fully loaded and done configuring itself and then close it.

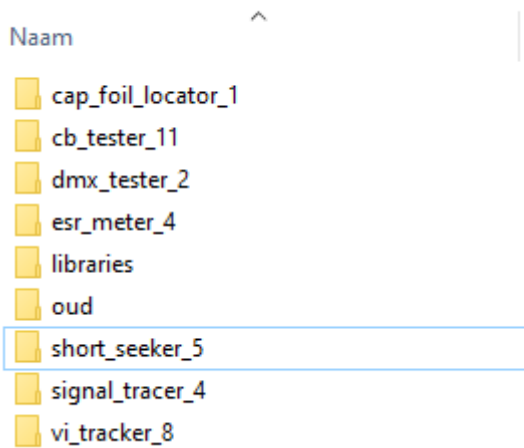
Open the portable folder, there will be folders and files created by the IDE.



Open the sketchbook folder.

For each of the test modules you need to create a new folder within the sketchbook folder.

ester > arduino-1.8.19 > portable > sketchboo

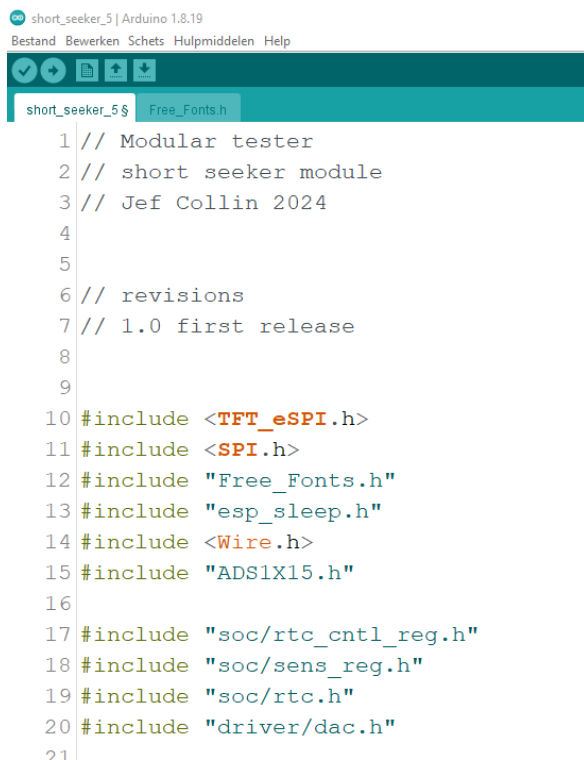


Look at the .INO files in the src folder on github.

Create a new folder for each test module source code file (and associated files) with the exact same name as the .INO file in the src folder on github, note that the sequential number can change, a new number is used for every development change of the source code regardless of what the final version number will be (as displayed at power on).

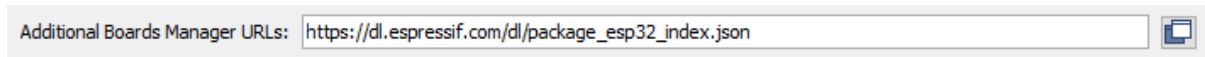
Create the matching folders and copy the .INO file and all other files in each github folder to these folders.

Start the Arduino IDE as shown above, open the folder for the tester module you want to compile and open the .INO file.



We now have to install extra libraries that are not installed by default such as the ESP32 board libraries and display driver libraries (those listed in the #include section).

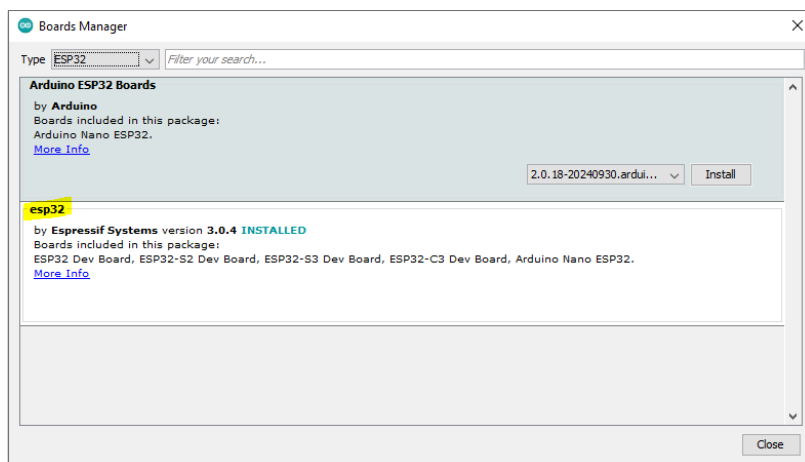
Go to *File->Preferences*, edit *Additional Boards Manager URLs* as follows:



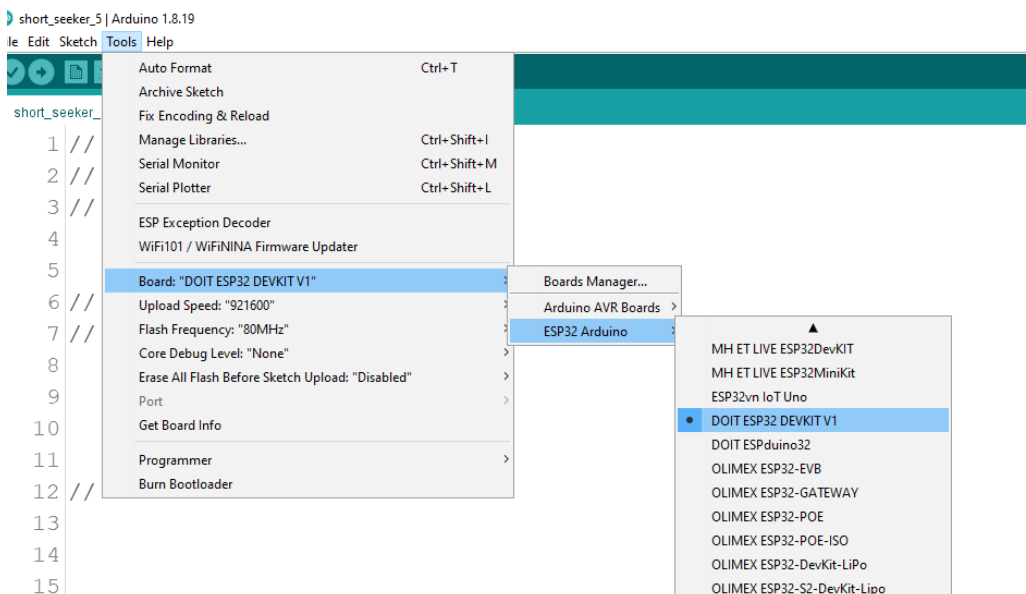
Important: to ensure the development environment is setup with the library versions which are known to work, install the version numbers listed. Do not update the libraries, ignore automatic messages from the IDE to update. When new versions of the source are released, re-check this document for the latest used library versions and update when required.

Go to *Tools->Board:...->Board Manager*.

In the search field type “esp32”, install *esp32 by Espressif Systems* version 3.0.4.



Go to *Tools-> Board:...->ESP32 Arduino* and select *DOIT ESP32 DEVKIT V1*.

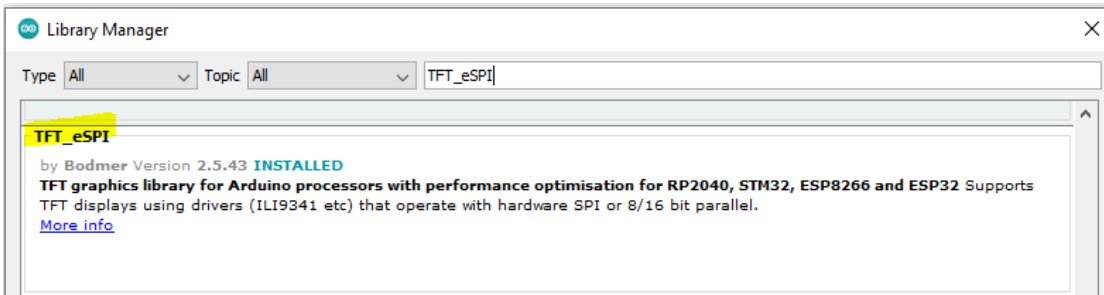


Go to *Tools-> Manage Libraries*, wait until the list is loaded, note that the library manager is very slow so be patient. Note that there might be multiple libraries with similar names but from different developers, use the ones listed below. Not all listed libraries are used with every module.

They will be installed automatically in the libraries folder of the sketchbook folder.

In the search field type “TFT_eSPI”.

Install this library version 2.5.43, this installs the graphics system.



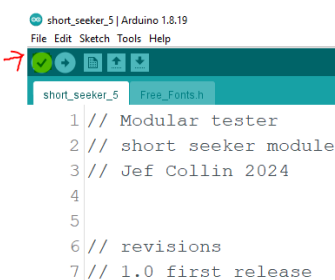
Repeat for following libraries:

- ADS1X15 by Rob Tillaart Version 0.4.3.
- Adafruit MCP23017 Arduino Library by Adafruit Version 2.3.2.
- Esp_dmx by Mitch Weisbrod Version 4.1.0.
- DAC8560 by Rob Tillaart Version 0.1.1.
- Time by PaulStoffregen Version 1.6.1.

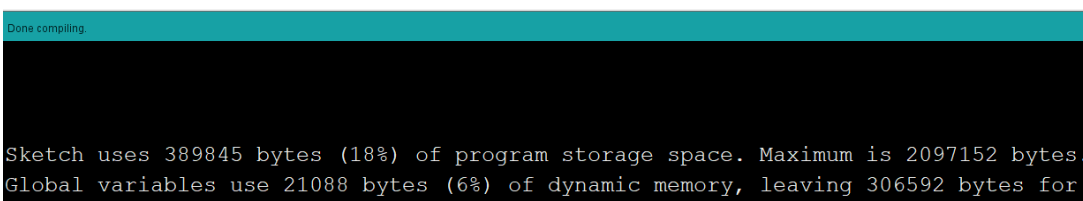
After the installation of the libraries you need to configure the display driver for the display used, a configuration file has been prepared, note that the file could be overwritten when a new version of the library is installed, in that case repeat this process.

On github, in folder tft_espi, copy file “User_Setup.h” to folder TFT_eSPI in the libraries folder of the sketchbook folder.

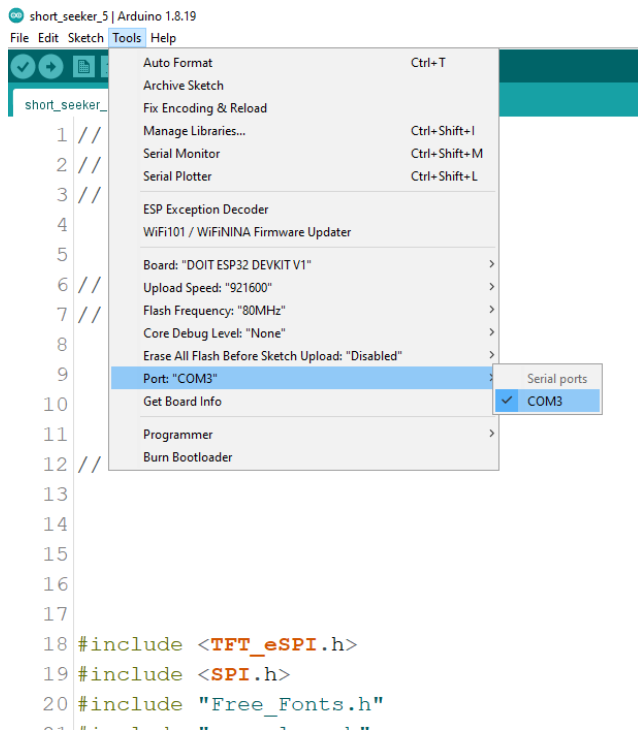
This completes the configuration, do a dry run first to verify the development environment, click the compile button and wait....



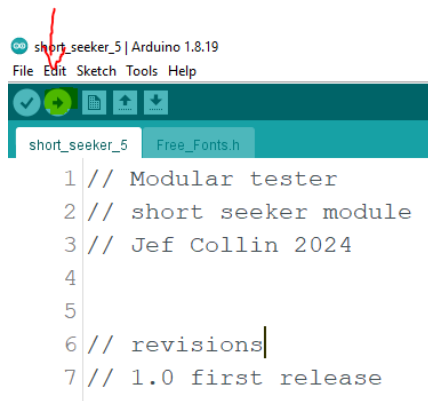
It should compile without errors.



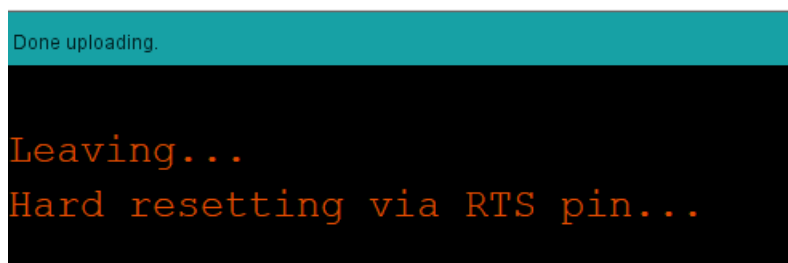
Connect the ESP32 board with the USB cable, Windows should detect it (if not install the correct USB drivers for the board in Windows), go to *Tools->Port* and select the port assigned to it.



Click the download button.



It should compile and download without errors, the ESP32 will be restarted automatically.



This completes the programming.