Espressif 32

Configuration: platform = `espressif32`

Espressif Systems is a privately held fabless semiconductor company. They provide wireless communications and Wi-Fi chips which are widely used in mobile devices and the Internet of Things applications.

For more detailed information please visit vendor site.

Contents

Tutorials

- Get started with Arduino and ESP32-DevKitC: debugging and unit testing
- Get started with ESP-IDF and ESP32-DevKitC: debugging, unit testing, project analysis
- Video: Free Inline Debugging for ESP32 and Arduino Sketches

Configuration

- CPU Frequency
- FLASH Frequency
- FLASH Mode
- External RAM (PSRAM)
- Debug Level
- Upload Speed

- Erase Flash
- Partition Tables
- Embedding Binary Data
- Uploading files to file system SPIFFS
- Over-the-Air (OTA) update
  - Using JFrog Bintray (free and secure Cloud solution)
  - Using built-in Local solution
    - Authentication and upload options
- Using Arduino Framework with Staging version
- Arduino Core Wiki

CPU Frequency

See board_build.f_cpu option from "platformio.ini" (Project Configuration File)

```
[env:myenv]
; set frequency to 160MHz
board_build.f_cpu = 160000000L
```

FLASH Frequency

Please use `board_build.f_flash` option from "platformio.ini" (Project Configuration File) to change a value. Possible values:

- `40000000L` (default)
- `80000000L`

```
[env:myenv]
; set frequency to 80MHz
board_build.f_flash = 80000000L
```

FLASH Mode

Flash chip interface mode. This parameter is stored in the binary image header, along with the flash size and flash frequency. The ROM bootloader in the ESP chip uses the value of these parameters in order to know how to talk to the flash chip.

Please use `board_build.flash_mode` option from "platformio.ini" (Project Configuration File) to change a value. Possible values:

- `qio`
- `qout`
- `dio`
- `dout`

```
[env:myenv]
board_build.flash_mode = qio
```

External RAM (PSRAM)

You can enable external RAM using the next extra build_flags in "platformio.ini" (Project Configuration File) depending on a framework type.

Framework Arduino:

```
[env:myenv]
platform = espressif32
framework = arduino
board = ...
build_flags =
    -DBOARD_HAS_PSRAM
    -mfix-esp32-psram-cache-issue
```

Framework Espressif IoT Development Framework:

```
[env:myenv]
platform = espressif32
framework = espidf
board = ...
build_flags =
    -DCONFIG_SPIRAM_CACHE_WORKAROUND
```

More details are located in the official ESP-IDF documentation - Support for external RAM.

Debug Level

Please use one of the next build_flags to change debug level. A build_flags option could be used only the one time per build environment. If you need to specify more flags, please separate them with a new line or space.

Actual information is available in Arduino for ESP32 Board Manifest. Please scroll to `esp32.menu.DebugLevel` section.

```
[env:myenv]
platform = ...
board = ...
framework = arduino

;;;;; Possible options ;;;;;;

; None
build_flags = -DCORE_DEBUG_LEVEL=0

; Error
build_flags = -DCORE_DEBUG_LEVEL=1

; Warn
build_flags = -DCORE_DEBUG_LEVEL=2

; Info
build_flags = -DCORE_DEBUG_LEVEL=3

; Debug
build_flags = -DCORE_DEBUG_LEVEL=4

; Verbose
build_flags = -DCORE_DEBUG_LEVEL=5
```

Upload Speed

You can set custom upload speed using upload_speed option from "platformio.ini" (Project Configuration File)

```
[env:myenv]
upload_speed = 9600
```

Erase Flash

Please `pio run --target` the next command to erase the entire flash chip (all data replaced with 0xFF bytes):

```
> pio run --target erase

# or short version

> pio run -t erase
```

Partition Tables

You can create a custom partitions table (CSV) following ESP32 Partition Tables documentation. PlatformIO uses **default partition tables** depending on a framework type:

- `default.csv` for Arduino (show pre-configured partition tables)
- `partitions_singleapp.csv` for Espressif IoT Development Framework (show pre-configured partition tables)

To override default table please use `board_build.partitions` option in "platformio.ini" (Project Configuration File).

> **Warning**
>
> SPIFFS partition **MUST** have configured "Type" as "data" and "SubType" as "spiffs". For example, `spiffs, data, spiffs, 0x291000, 1M,`

Examples:

```
; 1) A "partitions_custom.csv" in the root of project directory
[env:custom_table]
board_build.partitions = partitions_custom.csv

; 2) Switch between built-in tables
; https://github.com/espressif/arduino-esp32/tree/master/tools/partitions
; https://github.com/espressif/esp-idf/tree/master/components/partition_table
[env:custom_builtin_table]
board_build.partitions = no_ota.csv
```

Embedding Binary Data

Sometimes you have a file with some binary or text data that you'd like to make available to your program - but you don't want to reformat the file as C source.

There are two options `board_build.embed_txtfiles` and `board_build.embed_files` which can be used for embedding data. The only difference is that files specified in `board_build.embed_txtfiles` option are null-terminated in the final binary.

```
[env:myenv]
platform = espressif32
board = ...
board_build.embed_txtfiles =
  src/private.pem.key
  src/certificate.pem.crt
  src/aws-root-ca.pem
```

The file contents will be added to the `.rodata` section in flash, and are available via symbol names as follows:

```
extern const uint8_t aws_root_ca_pem_start[] asm("_binary_src_aws_root_ca_pem_start");
extern const uint8_t aws_root_ca_pem_end[] asm("_binary_src_aws_root_ca_pem_end");
extern const uint8_t certificate_pem_crt_start[] asm("_binary_src_certificate_pem_crt_start");
extern const uint8_t certificate_pem_crt_end[] asm("_binary_src_certificate_pem_crt_end");
extern const uint8_t private_pem_key_start[] asm("_binary_src_private_pem_key_start");
extern const uint8_t private_pem_key_end[] asm("_binary_src_private_pem_key_end");
```

The names are generated from the full name of the file. Characters `/, .`, etc. are replaced with underscores. The `_binary` + `_nested_folder` prefix in the symbol name is added by “objcopy” and is the same for both text and binary files.

> **Note**
>
> With the ESP-IDF framework symbol names should not contain path to the files, for example `_binary_private_pem_key_start` instead of `_binary_src_private_pem_key_start`.

See full example with embedding Amazon AWS certificates:

- https://github.com/platformio/platform-espressif32/tree/develop/examples/espidf-aws-iot

Uploading files to file system SPIFFS

1. Create new project using PlatformIO IDE or initialize project using PlatformIO Core (CLI) and pio project init (if you have not initialized it yet)
2. Create `data` folder (it should be on the same level as `src` folder) and put files here. Also, you can specify own location for data_dir
3. Run “Upload File System image” task in PlatformIO IDE or use PlatformIO Core (CLI) and `pio run --target` command with `uploadfs` target.

To upload SPIFFS image using OTA update please specify `upload_port` / `--upload-port` as IP address or mDNS host name (ending with the `*.local`).

Examples:

- SPIFFS for Arduino
- SPIFFS for ESP-IDF

Over-the-Air (OTA) update

Using JFrog Bintray (free and secure Cloud solution)

- Video and presentation - swampUP: Over-The-Air (OTA) firmware upgrades for Internet of Things devices with PlatformIO and JFrog Bintray
- Demo source code: https://github.com/platformio/bintray-secure-ota

Using built-in Local solution

Demo code for:

- Arduino
- ESP-IDF

There are 2 options:

- Directly specify `pio run --upload-port` in command line

```
pio run --target upload --upload-port IP_ADDRESS_HERE or mDNS_NAME.local
```

- Specify `upload_port` option in "platformio.ini" (Project Configuration File)

You also need to set upload_protocol to `espota`.

```
[env:myenv]
upload_protocol = espota
upload_port = IP_ADDRESS_HERE or mDNS_NAME.local
```

For example,

- `pio run -t upload --upload-port 192.168.0.255`
- `pio run -t upload --upload-port myesp8266.local`

Authentication and upload options

You can pass additional options/flags to OTA uploader using `upload_flags` option in "platformio.ini" (Project Configuration File)

```
[env:myenv]
upload_protocol = espota
; each flag in a new line
upload_flags =
    --port=3232
```

Available flags

- `--port=ESP_PORT` ESP32 OTA Port. **Default 8266**
- `--auth=AUTH` Set authentication password
- `--spiffs` Use this option to transmit a SPIFFS image and do not flash the module

For the full list with available options please run

```
~/.platformio/packages/framework-arduinoespressif32/tools/espota.py --help

Usage: espota.py [options]

Transmit image over the air to the esp32 module with OTA support.

Options:
  -h, --help            show this help message and exit

  Destination:
    -i ESP_IP, --ip=ESP_IP
                        ESP32 IP Address.
    -I HOST_IP, --host_ip=HOST_IP
                        Host IP Address.
    -p ESP_PORT, --port=ESP_PORT
                        ESP32 ota Port. Default 3232
    -P HOST_PORT, --host_port=HOST_PORT
                        Host server ota Port. Default random 10000-60000

  Authentication:
    -a AUTH, --auth=AUTH
                        Set authentication password.

  Image:
    -f FILE, --file=FILE
                        Image file.
    -s, --spiffs        Use this option to transmit a SPIFFS image and do not
                        flash the module.

  Output:
    -d, --debug         Show debug output. And override loglevel with debug.
    -r, --progress      Show progress output. Does not work for ArduinoIDE
    -t TIMEOUT, --timeout=TIMEOUT
                        Timeout to wait for the ESP32 to accept invitation
```

Warning

For windows users. To manage OTA check the ESP wifi network profile isn't checked on public be sure it's on private mode``

Using Arduino Framework with Staging version

PlatformIO will install the latest Arduino Core for ESP32 from https://github.com/espressif/arduino-esp32. The Git should be installed in a system. To update Arduino Core to the latest revision, please open PlatformIO IDE and navigate to `PlatformIO Home > Platforms > Updates`.

1. Please install PlatformIO IDE
2. Initialize a new project, open "platformio.ini" (Project Configuration File) and specify the link to the framework repository in platform_packages section. For example,

```
[env:esp32dev]
platform = espressif32
board = esp32dev
framework = arduino
platform_packages =
    platformio/framework-arduinoespressif32 @ https://github.com/espressif/arduino-esp32.git
```

3. Try to build the project
4. If you see build errors, then try to build this project using the same `stage` with Arduino IDE
5. If it works with Arduino IDE but doesn't work with PlatformIO, then please file a new issue with attached information:
    - test project/files
    - detailed log of build process from Arduino IDE (please copy it from console to https://hastebin.com)
    - detailed log of build process from PlatformIO Build System (please copy it from console to https://hastebin.com)

Arduino Core Wiki

Tips, tricks and common problems: http://desire.giesecke.tk/index.php/2018/01/30/esp32-wiki-entries/

Examples

Examples are listed from Espressif 32 development platform repository:

- espidf-storage-sdcard

- espidf-blink
- espidf-coap-server
- espidf-exceptions
- arduino-blink
- simba-blink
- espidf-ble-eddystone
- espidf-ulp-adc
- espidf-ulp-pulse
- espidf-arduino-wifiscan
- espidf-http-request
- espidf-arduino-blink
- pumbaa-blink
- espidf-hello-world
- espidf-aws-iot
- arduino-briki-internal-libs
- espidf-peripherals-uart
- arduino-wifiscan

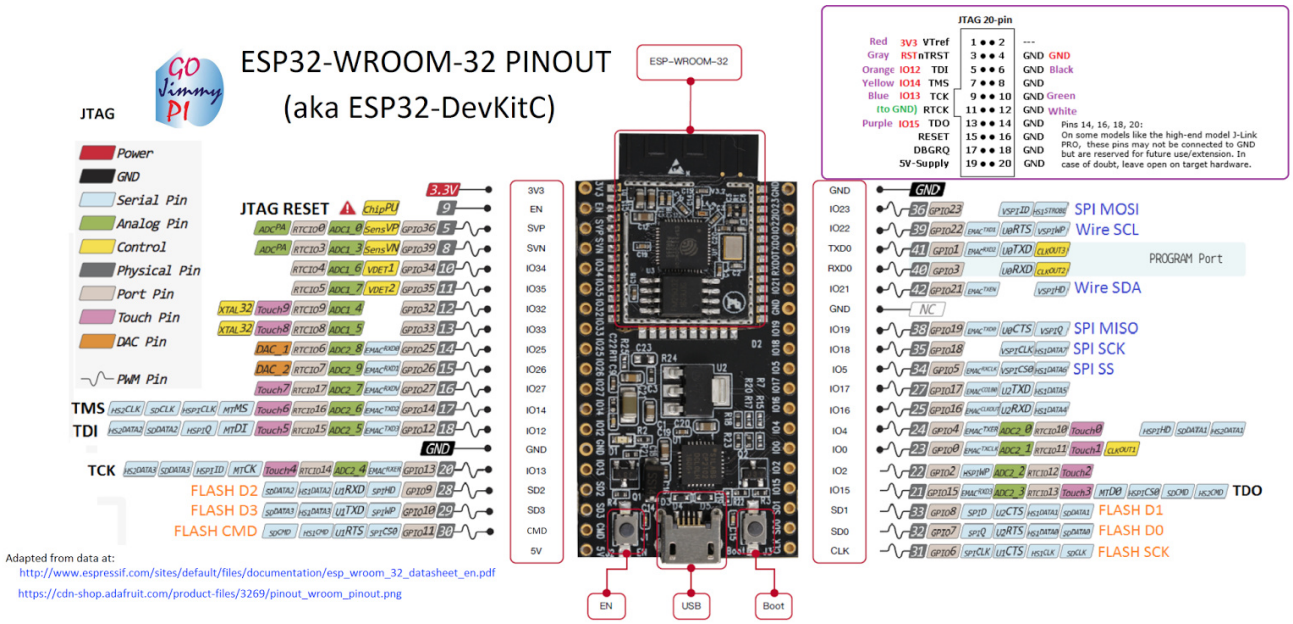
Debugging

Debugging - “1-click” solution for debugging with a zero configuration.

Pinout Diagram

JTAG Wiring Connections

Board Pin	JTAG Tool Pin
IO13	TCK
IO12	TDI
IO15	TDO
IO14	TMS
EN	RST
GND	GND



Tools & Debug Probes

Supported debugging tools are listed in “Debug” column. For more detailed information, please scroll table by horizontal. You can switch between debugging Tools & Debug Probes using debug_tool option in “platformio.ini” (Project Configuration File).

> **Warning**
>
> You will need to install debug tool drivers depending on your system. Please click on compatible debug tool below for the further instructions.

On-Board Debug Tools

Boards listed below have on-board debug probe and **ARE READY** for debugging! You do not need to use/buy external debug probe.

Name	MCU	Frequency	Flash	RAM
Espressif ESP-WROVER-KIT	ESP32	240MHz	4MB	320KB

External Debug Tools

Boards listed below are compatible with Debugging but **DEPEND ON** external debug probe. They **ARE NOT READY** for debugging. Please click on board name for the further details.

Name	MCU	Frequency	Flash	RAM
AI Thinker ESP32-CAM	ESP32	240MHz	4MB	320KB
ALKS ESP32	ESP32	240MHz	4MB	320KB
AZ-Delivery ESP-32 Dev Kit C V4	ESP32	240MHz	16MB	520KB
Adafruit ESP32 Feather	ESP32	240MHz	4MB	320KB
Briki ABC (MBC-WB) - ESP32	ESP32	240MHz	3.25MB	320KB
Briki MBC-WB - ESP32	ESP32	240MHz	3.25MB	320KB
D-duino-32	ESP32	240MHz	4MB	320KB
DOIT ESP32 DEVKIT V1	ESP32	240MHz	4MB	320KB
Dongsen Tech Pocket 32	ESP32	240MHz	4MB	320KB
ESP32 FM DevKit	ESP32	240MHz	4MB	320KB
ESP32 Pico Kit	ESP32	240MHz	4MB	320KB
ESP32vn IoT Uno	ESP32	240MHz	4MB	320KB
ESPectro32	ESP32	240MHz	4MB	320KB
ESPino32	ESP32	240MHz	4MB	320KB
Espressif ESP32 Dev Module	ESP32	240MHz	4MB	320KB
FireBeetle-ESP32	ESP32	240MHz	4MB	320KB
Frog Board ESP32	ESP32	240MHz	4MB	320KB
Heltec WiFi LoRa 32	ESP32	240MHz	4MB	320KB
Heltec WiFi LoRa 32 (V2)	ESP32	240MHz	8MB	320KB

Name	MCU	Frequency	Flash	RAM
Heltec Wireless Stick	ESP32	240MHz	8MB	320KB
Hornbill ESP32 Dev	ESP32	240MHz	4MB	320KB
Hornbill ESP32 Minima	ESP32	240MHz	4MB	320KB
IoTaaP Magnolia	ESP32	240MHz	4MB	320KB
MH ET LIVE ESP32DevKIT	ESP32	240MHz	4MB	320KB
MH ET LIVE ESP32MiniKit	ESP32	240MHz	4MB	320KB
Node32s	ESP32	240MHz	4MB	320KB
NodeMCU-32S	ESP32	240MHz	4MB	320KB
OLIMEX ESP32-DevKit-LiPo	ESP32	240MHz	4MB	320KB
OLIMEX ESP32-EVB	ESP32	240MHz	4MB	320KB
OLIMEX ESP32-GATEWAY	ESP32	240MHz	4MB	320KB
Pycom LoPy	ESP32	240MHz	4MB	320KB
Pycom LoPy4	ESP32	240MHz	4MB	1.25MB
SG-O AirMon	ESP32	240MHz	4MB	320KB
Silicognition wESP32	ESP32	240MHz	4MB	320KB
SparkFun ESP32 Thing	ESP32	240MHz	4MB	320KB
SparkFun LoRa Gateway 1-Channel	ESP32	240MHz	4MB	320KB
TTGO LoRa32-OLED V1	ESP32	240MHz	4MB	320KB
TTGO LoRa32-OLED V2	ESP32	240MHz	4MB	320KB

Name	MCU	Frequency	Flash	RAM
TTGO T-Beam	ESP32	240MHz	4MB	1.25MB
TTGO T1	ESP32	240MHz	4MB	320KB
VintLabs ESP32 Devkit	ESP32	240MHz	4MB	320KB
WEMOS LOLIN D32	ESP32	240MHz	4MB	320KB
WEMOS LOLIN D32 PRO	ESP32	240MHz	4MB	320KB
WEMOS LOLIN32	ESP32	240MHz	4MB	320KB
WeMos D1 MINI ESP32	ESP32	240MHz	4MB	320KB
WeMos WiFi and Bluetooth Battery	ESP32	240MHz	4MB	320KB
XinaBox CW02	ESP32	240MHz	4MB	320KB
oddWires IoT-Bus Io	ESP32	240MHz	4MB	320KB
oddWires IoT-Bus Proteus	ESP32	240MHz	4MB	320KB

Stable and upstream versions

You can switch between stable releases of Espressif 32 development platform and the latest upstream version using platform option in "platformio.ini" (Project Configuration File) as described below.

Stable

```
; Latest stable version
[env:latest_stable]
platform = espressif32
board = ...

; Custom stable version
[env:custom_stable]
platform = espressif32@x.y.z
board = ...
```

Upstream

```
[env:upstream_develop]
platform = https://github.com/platformio/platform-espressif32.git
board = ...
```

Packages

Name	Description
framework-arduino-mbcwb	Fork of Arduino Framework for briki MBC-WB boards
framework-arduinoespressif32	Arduino Wiring-based Framework for Espressif ESP32 microcontrollers
framework-espidf	Espressif IoT Development Framework. Official development framework for ESP32 chip
framework-pumbaa	Pumbaa Framework - a port of MicroPython, designed for embedded devices with limited amount of RAM and code memory
framework-simba	Simba is an Embedded Programming Platform. It aims to make embedded programming easy and portable
tool-cmake	CMake is an open-source, cross-platform family of tools designed to build, test and package software
tool-esptoolpy	Espressif ESP8266 and ESP32 serial bootloader utility
tool-idf	idf is a top-level config/build command line tool for ESP-IDF
tool-mbctool	MBC-WB Uploader Application
tool-mconf	Fork of kconfig-frontends project with some modifications for use with ESP-IDF
tool-mkspiffs	Tool to build and unpack SPIFFS images
tool-ninja	Ninja is a small build system with a focus on speed
tool-openocd-esp32	Open On-Chip Debugger for Espressif ESP32
toolchain-esp32s2ulp	Binutils fork with support for the ESP32-S2 ULP co-processor

Name	Description
toolchain-esp32ulp	Binutils fork with support for the Espressif ESP32 ULP co-processor
toolchain-xtensa32	GCC Toolchain for Xtensa32 processor
toolchain-xtensa32s2	GCC Toolchain for Xtensa32-S2 processor

Warning

Linux Users:

- Install “udev” rules 99-platformio-udev.rules
- Raspberry Pi users, please read this article Enable serial port on Raspberry Pi.

Windows Users:

Please check that you have a correctly installed USB driver from board manufacturer

Frameworks

Name	Description
Arduino	Arduino Wiring-based Framework allows writing cross-platform software to control devices attached to a wide range of Arduino boards to create all kinds of creative coding, interactive objects, spaces or physical experiences
Espressif IoT Development Framework	ESP-IDF is the official development framework for the ESP32 and ESP32-S Series SoCs.
Pumbaa	Pumbaa is Python on top of Simba. The implementation is a port of MicroPython, designed for embedded devices with limited amount of RAM and code memory
Simba	Simba is an RTOS and build framework with aims to make embedded programming easy and portable

Boards

Note

- You can list pre-configured boards by pio boards command or PlatformIO Boards Explorer
- For more detailed `board` information please scroll the tables below by horizontally.

AI Thinker

Name	Debug	MCU	Frequency	Flash	RAM
AI Thinker ESP32-CAM	External	ESP32	240MHz	4MB	320KB

AZ-Delivery

Name	Debug	MCU	Frequency	Flash	RAM
AZ-Delivery ESP-32 Dev Kit C V4	External	ESP32	240MHz	16MB	520KB

Adafruit

Name	Debug	MCU	Frequency	Flash	RAM
Adafruit ESP32 Feather	External	ESP32	240MHz	4MB	320KB

Aiyarafun

Name	Debug	MCU	Frequency	Flash	RAM
Node32s	External	ESP32	240MHz	4MB	320KB

April Brother

Name	Debug	MCU	Frequency	Flash	RAM
April Brother ESPea32	No	ESP32	240MHz	4MB	320KB

BPI Tech

Name	Debug	MCU	Frequency	Flash	RAM
BPI-Bit	No	ESP32	160MHz	4MB	320KB

DFRobot

Name	Debug	MCU	Frequency	Flash	RAM
FireBeetle-ESP32	External	ESP32	240MHz	4MB	320KB

DOIT

Name	Debug	MCU	Frequency	Flash	RAM
DOIT ESP32 DEVKIT V1	External	ESP32	240MHz	4MB	320KB

DSTIKE

Name	Debug	MCU	Frequency	Flash	RAM
D-duino-32	External	ESP32	240MHz	4MB	320KB

Dongsen Technology

Name	Debug	MCU	Frequency	Flash	RAM
Dongsen Tech Pocket 32	External	ESP32	240MHz	4MB	320KB

DycodeX

Name	Debug	MCU	Frequency	Flash	RAM
ESPectro32	External	ESP32	240MHz	4MB	320KB

ESP32vn

Name	Debug	MCU	Frequency	Flash	RAM
ESP32vn IoT Uno	External	ESP32	240MHz	4MB	320KB

Electronic SweetPeas

Name	Debug	MCU	Frequency	Flash	RAM
Electronic SweetPeas ESP320	No	ESP32	240MHz	4MB	320KB

Espressif

Name	Debug	MCU	Frequency	Flash	RAM
ESP32 Pico Kit	External	ESP32	240MHz	4MB	320KB
Espressif ESP-WROVER-KIT	On-board	ESP32	240MHz	4MB	320KB
Espressif ESP32 Dev Module	External	ESP32	240MHz	4MB	320KB

Fred

Name	Debug	MCU	Frequency	Flash	RAM
Frog Board ESP32	External	ESP32	240MHz	4MB	320KB

Hardkernel

Name	Debug	MCU	Frequency	Flash	RAM
ODROID-GO	No	ESP32	240MHz	16MB	320KB

Heltec Automation

Name	Debug	MCU	Frequency	Flash	RAM
Heltec WiFi Kit 32	No	ESP32	240MHz	4MB	320KB
Heltec WiFi LoRa 32	External	ESP32	240MHz	4MB	320KB
Heltec WiFi LoRa 32 (V2)	External	ESP32	240MHz	8MB	320KB

Name	Debug	MCU	Frequency	Flash	RAM
Heltec Wireless Stick	External	ESP32	240MHz	8MB	320KB

Hornbill

Name	Debug	MCU	Frequency	Flash	RAM
Hornbill ESP32 Dev	External	ESP32	240MHz	4MB	320KB
Hornbill ESP32 Minima	External	ESP32	240MHz	4MB	320KB

IntoRobot

Name	Debug	MCU	Frequency	Flash	RAM
IntoRobot Fig	No	ESP32	240MHz	4MB	320KB

IoTaaP

Name	Debug	MCU	Frequency	Flash	RAM
IoTaaP Magnolia	External	ESP32	240MHz	4MB	320KB

M5Stack

Name	Debug	MCU	Frequency	Flash	RAM
M5Stack Core ESP32	No	ESP32	240MHz	4MB	320KB
M5Stack FIRE	No	ESP32	240MHz	16MB	6.25MB
M5Stack GREY ESP32	No	ESP32	240MHz	16MB	520KB
M5Stick-C	No	ESP32	240MHz	4MB	320KB

MH-ET Live

Name	Debug	MCU	Frequency	Flash	RAM
MH ET LIVE ESP32DevKIT	External	ESP32	240MHz	4MB	320KB
MH ET LIVE ESP32MiniKit	External	ESP32	240MHz	4MB	320KB

Magicblocks.io

Name	Debug	MCU	Frequency	Flash	RAM
MagicBit	No	ESP32	240MHz	4MB	320KB

MakerAsia

Name	Debug	MCU	Frequency	Flash	RAM
MakerAsia Nano32	No	ESP32	240MHz	4MB	320KB

Microduino

Name	Debug	MCU	Frequency	Flash	RAM
Microduino Core ESP32	No	ESP32	240MHz	4MB	320KB

NodeMCU

Name	Debug	MCU	Frequency	Flash	RAM
NodeMCU-32S	External	ESP32	240MHz	4MB	320KB

Noduino

Name	Debug	MCU	Frequency	Flash	RAM
Noduino Quantum	No	ESP32	240MHz	16MB	320KB

OLIMEX

Name	Debug	MCU	Frequency	Flash	RAM
OLIMEX ESP32-DevKit-LiPo	External	ESP32	240MHz	4MB	320KB
OLIMEX ESP32-EVB	External	ESP32	240MHz	4MB	320KB
OLIMEX ESP32-GATEWAY	External	ESP32	240MHz	4MB	320KB
OLIMEX ESP32-PRO	No	ESP32	240MHz	4MB	320KB
OLIMEX ESP32-PoE	No	ESP32	240MHz	4MB	320KB
OLIMEX ESP32-PoE-ISO	No	ESP32	240MHz	4MB	320KB

OROCA

Name	Debug	MCU	Frequency	Flash	RAM
OROCA EduBot	No	ESP32	240MHz	4MB	320KB

Onehorse

Name	Debug	MCU	Frequency	Flash	RAM
Onehorse ESP32 Dev Module	No	ESP32	240MHz	4MB	320KB

Pycom Ltd.

Name	Debug	MCU	Frequency	Flash	RAM
Pycom GPy	No	ESP32	240MHz	4MB	320KB
Pycom LoPy	External	ESP32	240MHz	4MB	320KB
Pycom LoPy4	External	ESP32	240MHz	4MB	1.25MB

Qmobot LLP

Name	Debug	MCU	Frequency	Flash	RAM
Qchip	No	ESP32	240MHz	4MB	320KB

RoboticsBrno

Name	Debug	MCU	Frequency	Flash	RAM
ALKS ESP32	External	ESP32	240MHz	4MB	320KB

SG-O

Name	Debug	MCU	Frequency	Flash	RAM
SG-O AirMon	External	ESP32	240MHz	4MB	320KB

Silicognition

Name	Debug	MCU	Frequency	Flash	RAM
Silicognition wESP32	External	ESP32	240MHz	4MB	320KB

SparkFun

Name	Debug	MCU	Frequency	Flash	RAM
SparkFun LoRa Gateway 1-Channel	External	ESP32	240MHz	4MB	320KB

SparkFun Electronics

Name	Debug	MCU	Frequency	Flash	RAM
SparkFun ESP32 Thing	External	ESP32	240MHz	4MB	320KB

TTGO

Name	Debug	MCU	Frequency	Flash	RAM
TTGO LoRa32-OLED V1	External	ESP32	240MHz	4MB	320KB
TTGO LoRa32-OLED V2	External	ESP32	240MHz	4MB	320KB
TTGO T-Beam	External	ESP32	240MHz	4MB	1.25MB
TTGO T-Watch	No	ESP32	240MHz	16MB	320KB
TTGO T1	External	ESP32	240MHz	4MB	320KB

ThaiEasyElec

Name	Debug	MCU	Frequency	Flash	RAM
ESPino32	External	ESP32	240MHz	4MB	320KB

TinyPICO

Name	Debug	MCU	Frequency	Flash	RAM
TinyPICO	No	ESP32	240MHz	4MB	320KB

Turta

Name	Debug	MCU	Frequency	Flash	RAM
Turta IoT Node	No	ESP32	240MHz	4MB	320KB

Unknown

Name	Debug	MCU	Frequency	Flash	RAM
ESP32 FM DevKit	External	ESP32	240MHz	4MB	320KB

VintLabs

Name	Debug	MCU	Frequency	Flash	RAM
VintLabs ESP32 Devkit	External	ESP32	240MHz	4MB	320KB

WEMOS

Name	Debug	MCU	Frequency	Flash	RAM
WEMOS LOLIN D32	External	ESP32	240MHz	4MB	320KB
WEMOS LOLIN D32 PRO	External	ESP32	240MHz	4MB	320KB
WEMOS LOLIN32	External	ESP32	240MHz	4MB	320KB
WeMos D1 MINI ESP32	External	ESP32	240MHz	4MB	320KB
WeMos WiFi and Bluetooth Battery	External	ESP32	240MHz	4MB	320KB

Widora

Name	Debug	MCU	Frequency	Flash	RAM
Widora AIR	No	ESP32	240MHz	16MB	320KB

XinaBox

Name	Debug	MCU	Frequency	Flash	RAM
XinaBox CW02	External	ESP32	240MHz	4MB	320KB

meteca

Name	Debug	MCU	Frequency	Flash	RAM
Briki ABC (MBC-WB) - ESP32	External	ESP32	240MHz	3.25MB	320KB

Name	Debug	MCU	Frequency	Flash	RAM
Briki MBC-WB - ESP32	External	ESP32	240MHz	3.25MB	320KB

oddWires

Name	Debug	MCU	Frequency	Flash	RAM
oddWires IoT-Bus Io	External	ESP32	240MHz	4MB	320KB
oddWires IoT-Bus Proteus	External	ESP32	240MHz	4MB	320KB

u-blox

Name	Debug	MCU	Frequency	Flash	RAM
u-blox NINA-W10 series	No	ESP32	240MHz	2MB	320KB