
ESP-FAQ

2020 - 2023, Espressif Systems (Shanghai) Co., Ltd.

Aug 22, 2023

CONTENTS





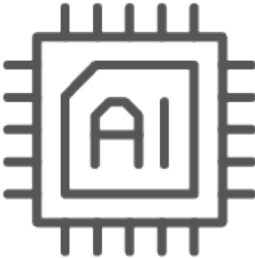

1	Instruction	3
1.1	Question search	3
1.2	Document contribution	4
2	Development environment	9
2.1	IDE plugins	9
2.2	Debugging	10
2.3	Environment setup	15
2.4	Firmware update	19
3	application solution	25
3.1	Android application	25
3.2	Artificial intelligence	25
3.3	AT	27
3.4	Audio development framework	27
3.5	BLE Mesh development framework	30
3.6	Camera application	31
3.7	Community sw and platforms	35
3.8	ESP Matter	35
3.9	ESP-NOW	38
3.10	ESP RainMaker cloud service	42
3.11	ios application	42
3.12	Third party cloud service	42
3.13	ESP-WIFI-MESH development framework	43
4	Software framework	47
4.1	Bluetooth LE & Bluetooth	47
4.2	Ethernet	63
4.3	coexistence	67
4.4	Peripherals	68
4.5	Protocols	101
4.6	Provisioning	118
4.7	Security	119
4.8	Storage	124
4.9	System	133
4.10	Wi-Fi	142
5	Hardware related	173
5.1	Chip Comparison	173
5.2	Development board	176

5.3	Hardware design	177
5.4	RF Related	190
5.5	Process and ESD Protection	192
5.6	Production Test	193
6	Test verification	195
6.1	Power Consumption Verification	195
7	Commercial FAQ	197
7.1	Which certificates have your products passed?	197
7.2	Does your company have the ISO Quality Management System Certification?	197
7.3	Do your chips and modules have environmental certificates such as REACH, ROHS, etc?	197
7.4	Do you have distributors in China, Europe, the United States and Canada?	197
7.5	How can I start a distribution business with Espressif?	198
7.6	Where can I find your product information? Which of your products are in mass production?	198
7.7	Do your products have a longevity commitment?	198
7.8	Where can I find the SPQ (Standard Pack Quantity) and MOQ (Minimum Order Quantity) for your products?	198
7.9	What is your recommended purchasing method?	198
7.10	What's the price for bulk purchasing? How can I purchase in bulk?	198
7.11	Where can I find all the differences between your products (e.g. in terms of series and types)?	199
7.12	Do your products have firmware? Can I customize my module/chip flash before the product leaves the factory? How much does this process cost? How long does it take? How can you help me do this?	199
7.13	Which of your products support HomeKit? Where can I get the Espressif HomeKit SDK?	199
7.14	What is your company's address?	199
7.15	How can I contact your technical team?	200
7.16	How can I get in touch with your company?	200
7.17	How can I tell if an Espressif module is in mass production or an NPI product?	200



[Coming Soon ...]

ESP-FAQ is a summary document of frequently asked questions released by Espressif. This repository aims to help our users to quickly locate those questions and get answers through simple explanations. The current categories of FAQ cover: development environment, application solution, software framework, hardware related and test verification.

		
Instruction	Development environment	Application solution
		
Software framework	Hardware related	Test verification

INSTRUCTION



This section provides instructions for using ESP-FAQ. The purpose of “Question search” is to help you quickly understand the search methods and categories of this repository, so as to save much time for searching. In the meantime, you are welcomed to make contributions directly to ESP-FAQ, such as fixing bugs and adding new documents. For detailed information about this process, please go to “Document contribution”.

1.1 Question search



This instruction includes the following two parts

- Question search techniques
- Question classification framework

1.1.1 Question Search Techniques

Currently, there are mainly two searching techniques:

- Search keywords
- Exclude a specific keyword

Search key words

Extract keywords from your question and search for them, then the search results should list the best matches.

For example, if you expect to ask a question as What is the Bluetooth LE Throughput for ESP32?

Then just searching keywords such as ESP32, BLE and throughput should give you the result.

Exclude a Specific Keyword

Add a tag `-` into the search content in the format: `keyword -excluded keyword`. By doing so, the search results will not show the specific keyword you excluded.

For example, if you search `ESP32 -ble`, then any results with `ble` inside will not be shown.

1.1.2 Question Classification Categories

Once you have mastered the above mentioned `question search techniques`, you can use the categories in ESP-FAQ for reference to extract keywords for the questions you expect to ask and then search for them. The framework of ESP-FAQ categories is shown as follows:

- *Development environment*
- *Application solution*
- *Software framework*
- *Hardware related*
- *Test verification*
- *Commercial FAQ*

1.2 Document contribution

□

We welcome all the contributions to `esp-faq` project, such as fixing bugs, adding new documents and etc. We will accept new requests via [Github Pull Requests](#).

1.2.1 Commit process

This section provides a brief overview of the `Add new items` and `Modify contents` processes. For the specific requirements during the processes, please refer to the links provided.

Here, we do not provide further operational instructions on `git`, please see [Git learning material](#).

Add new items

1. *Create a new branch* following the *Branch naming conventions*;
2. Find the corresponding `*.rst` file locally or on web IDE, then add new items according to the template formats;
3. After your writing finished, you can check the document in the preview interface and build it using *Local build environment* to see if it has any formatting issues;
4. Push your branch to github and commit a pull request following the *Commit message standards*;
5. If all the abovementioned steps are finished following requirements, then you can *Submit a merge request*;
6. After all the review comments resolved and new pull requests updated, then this process is fully completed.

Modify contents

1. *Create a new branch* locally following the *Branch naming conventions*;
2. Find the corresponding *.rst file locally or on web IDE, then modify the desired contents;
3. After the modification finished, you can check the document in the preview interface and build it using the *Local build environment* to see if it has any formatting issues;
4. Push your branch to github and commit a pull request following the *Commit message standards*;
5. If all the abovementioned steps are finished following requirements, then you can *Submit a merge request*;
6. After all the review comments resolved and new pull requests updated, this process is fully completed.

1.2.2 Create a new branch

All the new branches are based on the **master branch**, so please make sure your current branch is the one you expect to merge.

For example:

```
git status #To see the status of your current branch
git checkout -b add/artificial-intelligence_camera_model #To add new contents about
↪ "artificial-intelligence camera model"
```

1.2.3 Branch naming conventions

- Add a new item: add/artificial-intelligence_{q&a}, {q&a} is the brief English name of the file. For example, if you expect to add a new item as artificial intelligence camera model, then the branch name should be: add/artificial-intelligence_camera_model.
- Modify contents: mod/artificial-intelligence_q&a, q&a is the brief English name of the file. For example, if you expect to modify the contents about artificial intelligence camera model, then the branch name should be: mod/artificial-intelligence_camera_model.

1.2.4 Q&A Guidelines

Please add new Q&A items and do updates according to the guidelines as follows:

General guideline:

- If you are going to add a new Q&A item, always remember to add a separate line after the previous one as “_____”.

For questions:

- Illustrate questions clearly, for example:
 - When flashing firmware to ESP32-S2, an error occurred as “A fatal error occurred: Invalid head of packet (0x50)”? (**NOT Clear**)
 - When flashing firmware to ESP32-S2, an error occurred as “A fatal error occurred: Invalid head of packet (0x50)”. How to resolve such issue? (**Preferable**)
- Do not make the question too long. If this is the case, extract the main question as the title and describe the background and details below.

For answers:

- If there will be code in the text, use code box to separate it with the main text.
- If an answer only includes one sentence, there is no need to write a list, just use a regular paragraph.
- Use lists to separate items or to enumerate sequential items:
 - Use numbered lists for items that are in a required order (such as step-by-step instructions) or for items that are referred to by item number.
 - Use bulleted lists for items that are in no required order.
- Provide introductory phrase before a list to indicate the meaning or purpose of the list, and place a colon “:” at the end of it.
- If two items are alternatives, use a bullet list (not numbered list) and indicate their relationship in the introductory phrase.
- Always add two spaces before an listed item or paragraph in the answer.
- When a separate notice or explanatory paragraph follows a item, indent that separate material to the text of the parent list item.
- Follow list punctuation rules described in [Espressif Manual of Style](#), Section Punctuation in Lists.

For additional guidance regarding list please refer to [Bulleted and Numbered Lists](#). Please see the example template for text and figures below.

Q&A example

```
-----  
  
Does Espressif have any AI image recognition products?  
-----  
  
    Yes, we already have the ESP-EYE development board. With ESP32 as its main_  
↪control chip, ESP-EYE supports various types of cameras, such as 0v2640,_  
↪3660, 5640 and etc.
```

Q&A figure example

```
-----  
  
curses.h: No such file or directory  
-----  
  
Screenshot: support ESP8266 chip, but ESP8266_RT  
  
.. figure:: _static/application-solution/android-application/case_two_  
↪kconfig_error.png  
    :align: center  
    :width: 900  
    :height: 100  
  
Solution: sudo apt-get install libncurses5-dev
```

1.2.5 Local build environment

- Use ubuntu or Debian system as test environment, and configure your python version to 3.7.
- It is recommended to use python virtual environment or docker environment.

```
# Install python3.7 and virtual environment
sudo apt-get install python3.7 python3.7-venv

# Create virtual environment
python3.7 -m venv ~/.pyenv3_7

# Activate virtual environment
source ~/.pyenv3_7/bin/activate

# Upgrade pip
pip install --upgrade pip

# Install pip component
pip install -r docs/requirements.txt

# build the Chinese version
cd docs/cn/ && make html && cd -

# Build the English version
cd docs/en/ && make html && cd -

# Exit virtual environment
deactivate
```

1.2.6 Commit message standards

Please add commit messages on your branch to explain what you have added/modified/deleted. Each commit has one message, for example:

```
artificial-intelligence: add esp-eye support those camera models

1. esp-eye support those camera models.
```

The first line of the commit message should be like “Q&A category: add/fix/modify/delete something”. And this line should be started with the file name you updated, for example:

```
artificial-intelligence: esp-eye support those camera models.
```

If more information should be added into the commit message, please add it in the later commits after the first line.

A good commit message should tell why this update came up, thus making others get to know about this project when reading these commit logs. It may seem like a waste of time to write a good commit message, but it will be useful for you when trying to know why something changed.

1.2.7 Submit a merge request

Once your updates finished, you can conduct the first commit of your branch. Please add more commits if you need to do further updates. After finishing all the commits on this branch, you are ready to submit a merge request.

We use the github “Merge Requests” function to merge your branch into the master, the steps include:

1. Push your branch to the github repository;
2. Go to [esp-faq](#) and click “New pull request”;
3. Select the branch that you created and waited for merge, and fill detailed information in the “Merge Request”.

Please see [IDF Contribution Guide](#).

Merge request specifications

- Title:

`add: a brief overview`

- Description:

Describe the updates of this merge request in points.

- For example

Title:

`artificial-intelligence: add esp-eye support those camera models.`

Description:

`1. add esp-eye support those camera models.`

DEVELOPMENT ENVIRONMENT

[]

2.1 IDE plugins

[]

2.1.1 How to add ESP32 development board on Arduino IDE?

- For installation instructions of Arduino-ESP32, please refer to [arduino-ide getting started](#).
- For instructions on how to add development boards on Arduino IDE, please refer to [arduino Cores](#).

2.1.2 When using the Arduino IDE development platform, how to read the MAC address of the Wi-Fi that comes with ESP32?

- Please refer to the [Arduino-ESP32 Development Framework](#).
- Use “WiFi.macAddress()” to obtain the MAC address of ESP32’s Wi-Fi.
- Please also refer to the [WiFiClientStaticIP example](#).

2.1.3 How to use the flash download tool to flash the bin file generated by Arduino to ESP32?

- Please go to File -> Preferences -> Show verbose output during and select compilation. After compilation succeeded, a Python flashing command will be printed with the bin file to be flashed and the corresponding flashing address.
- Download the flash download tool on the [tools page](#) on Espressif’s official website, select the bin file when using the flash download tool to flash, and enter the corresponding flashing address.

2.2 Debugging



2.2.1 What is the serial port name of ESP devices

The serial port name is usually assigned by the operating system, and different operating systems and devices may have different serial port names. Common ones are as follows:

- In Windows system: COM*
 - In Linux system: /dev/ttyUSB*
 - In macOS system: /dev/cu.usbserial-*
-

2.2.2 How to block debugging messages sent through UART0 by default in ESP32?

- For first-stage Bootloader log, you could block the logs by connecting GPIO15 to Ground.
 - For second-stage Bootloader log, go to menuconfig and configure the `Bootloader config` option.
 - For ESP-IDF log, go to menuconfig > `Component config` and configure the `Log output` option.
-

2.2.3 How to modify the default method of RF calibration in ESP32?

- During RF initialization, the partial calibration solution is used by default. Go to menuconfig and enable the `CONFIG_ESP32_PHY_CALIBRATION_AND_DATA_STORAGE` option.
- If the boot time is not critical, the full calibration solution can be used instead. Go to menuconfig and disable the `CONFIG_ESP32_PHY_CALIBRATION_AND_DATA_STORAGE` option.
- It is recommended to use the **partial calibration** solution, which ensures less boot time and enables you to add the function of erasing RF calibration information in NVS so as to trigger the full calibration operation.

For detailed information, please refer to the [RF Calibration documentation](#).

2.2.4 How to modify the default method of RF calibration in ESP8266?

During RF initialization, the partial calibration solution is used by default, in which the value of byte 115 in `esp_init_data_default.bin` is 0x01. The initialization only takes a short time. If the boot time is not critical, the full calibration solution can be used instead.

For NONOS SDK and earlier versions of RTOS SDK 3.0:

- Call `system_phy_set_powerup_option(3)` in function `user_pre_init` or `user_rf_pre_init`.
 - In `phy_init_data.bin`, modify the value of byte 115 to 0x03.
-

For RTOS SDK 3.0 and later versions:

- Go to menuconfig and disable CONFIG_ESP_PHY_CALIBRATION_AND_DATA_STORAGE.
- If CONFIG_ESP_PHY_INIT_DATA_IN_PARTITION is enabled in menuconfig, please modify the value of byte 115 in phy_init_data.bin to 0x03. If CONFIG_ESP_PHY_INIT_DATA_IN_PARTITION is disabled, please modify the value of byte 115 in phy_init_data.h to 0x03.

If you use the default partial calibration solution, and want to add the function of triggering the full calibration operation:

- For NONOS SDK and earlier versions of RTOS SDK 3.0, please erase the RF parameters to trigger the full calibration operation.
- For RTOS SDK 3.0 and later versions, please erase the NVS partition to trigger the full calibration operation.

2.2.5 How to troubleshoot in ESP32 Boot mode

- The ESP32-WROVER uses 1.8 V flash and PSRAM, which is 0x33 by default in boot status and 0x23 in download mode.
- Other modules use 3.3 V flash and PSRAM, which are 0x13 by default in boot status and 0x03 in download mode.
- For detailed information, please refer to Section Strapping Pins in [ESP32 Series Datasheet](#). Taking 0x13 as an example, the pins are as follows:

Pins	GPIO12	GPIO0	GPIO2	GPIO4	GPIO15	GPIO5
Level	0	1	0	0	1	1

You can also refer to the [Boot Mode Selection](#) documentation directly.

2.2.6 When debugging with ESP32 JLINK, an ERROR occurs as: No Symbols For Freertos. How can I resolve such issue?

This issue will not affect actual operations. For solutions, please go to the [ST Community](#).

2.2.7 How to monitor the free space of the task stack?

The function `vTaskList()` can be used to print the available space of the task stack regularly. For detailed information, please refer to [CSDN Blog](#).

2.2.8 Is it possible to use JTAG to debug with ESP32-S2

Yes. For detailed information, please refer to [ESP32-S2 JTAG Debugging](#).

2.2.9 How to modify the log output without changing the output level of menuconfig

To modify the log output without changing the output level of menuconfig, you can use the `esp_log_level_set()` function. This function allows you to set the log level for a specific module or subsystem, rather than changing the global log level.

For example, to set the log level for the network module to `ESP_LOG_DEBUG`, you can use the following code:

```
esp_log_level_set("network", ESP_LOG_DEBUG);
```

For more information about this functionality, please refer to [Logging library](#).

2.2.10 ESP8266 enters boot mode (2,7) and hits a watchdog reset. What could be wrong?

- Please make sure that when ESP8266 boots, the strapping pins are held in the required logic levels. If externally connected peripherals drive the strapping pins to an inappropriate logic level, ESP8266 may boot into a wrong mode of operation. With the absence of a valid program, the WDT may then reset the chip.
 - Thus, in design practices, it is recommended to only use the strapping pins for input to high resistive external devices so that the strapping pin is not forced high/low at power-up. For more information, please refer to [ESP8266 Boot Mode Selection](#).
-

2.2.11 When using the ESP-WROVER-KIT board with OpenOCD, an error occurred as: Can't find board/esp32-wrover-kit-3.3v.cfg. How can I resolve such issue?

- With 20190313 and 20190708 versions of OpenOCD, please use instruction `openocd -f board/esp32-wrover.cfg`.
 - With 20191114 and 20200420 (2020 later versions) versions of OpenOCD, please use instruction `openocd -f board/esp32-wrover-kit-3.3v.cfg`.
-

2.2.12 The RTC_watch_dog keeps resetting during ESP32 SPI boot. What could be the reason?

- Reason: The flash has a requirement for time interval between VDD_SDIO power-up and the first access. For example, GD's 1.8 V flash requires 5 ms of time interval, while the time interval of ESP32 is about 1 ms (XTAL frequency is 40 MHz). Under such condition, the flash access will fail and either timer watchdog reset or RTC watchdog reset is triggered, depending on which one is triggered first. The threshold for RTC watchdog reset is 128 KB cycle, while the threshold for timer watchdog reset is 26 MB cycle. Taking the 40 MHz XTAL clock as an example, when the frequency of RTC slow clock is greater than 192 KHz, an RTC watchdog reset will be triggered first, otherwise a timer watchdog reset will be triggered. VDD_SDIO will be continuously powered when the timer watchdog is reset, so there will be no problem in accessing flash and the chip will work normally. When the RTC watchdog is reset, the VDD_SDIO power supply will be disabled and the access to flash will fail, resetting the RTC_watch_dog continuously.
- Solution: When an RTC watchdog reset occurs, the power supply to VDD_SDIO is disabled. You can add a capacitor to VDD_SDIO to ensure that the voltage of VDD_SDIO does not drop below the voltage that the flash can tolerate during this period.

2.2.13 How to obtain and parse coredump with ESP32?

- To obtain the 64 KB coredump file from the firmware, you need to know its offset from the partition table. Assuming the offset is 0x3F0000, run the following command to read the firmware:

```
python esp-idf/components/esptool_py/esptool/esptool.py -p /dev/ttyUSB* read_
↪flash 0x3f0000 0x10000 coredump.bin
```

- Use the coredump reading script to convert the file obtained at the first step into readable messages. Assuming the coredump file is coredump.bin and the elf file is hello_world.elf, run the following command to convert the file:

```
python esp-idf/components/espcoredump/espcoredump.py info_corefile -t raw -c_
↪coredump.bin hello_world.elf
```

For more information, please refer to the [Core Dump documentation](#).

2.2.14 How to do RF performance test with ESP32, ESP8266, and ESP32S2?

- Please refer to [ESP RF Test Guide](#).

2.2.15 My PC cannot recognize the device connected in Win10 system. What could be the reason?

- Check if the device is identified in the Linux virtual subsystem of Win10.
- If the device cannot be identified only in Win10 system, go to Device Manager to see whether such device exists (e.g., COM x). If the answer is still no, please check your cable and driver.
- If the device cannot be identified only in Linux virtual subsystem, taken VMWare as an example, please go to Settings > USB Controller and select Show all USB input devices.

2.2.16 One error occurred with ESP32 as: Core 1 panicked (Cache disabled but cache memory region accessed). What could be the reason?

Reasons:

- During the time when cache is disabled (e.g., when using the API `spi_flash` to read/write/erase/map the SPI flash), an interrupt is generated and the interrupt program accesses the flash resources.
- It is usually because the processor called programs from the flash and used its constants. One important thing is that since the Double variable is implemented through software, thus when this kind of variable is used in the interrupt programs, it is also implemented in the flash (e.g., forced type conversion operation).

Solution:

- Add an `IRAM_ATTR` modifier to the accessed function during interrupt
- Add an `DRAM_ATTR` modifier to the accessed constant during interrupt
- Do not use Double variable in the interrupt programs

For more information, please refer to the [Fatal error documentation](#).

2.2.17 How to read the flash model information of the modules?

- Please use the python script `esptool` to read information of Espressif's chips and modules.

```
esptool.py --port /dev/ttyUSB* flash_id
```

2.2.18 When debugging the Ethernet Example in ESP-IDF, the following exception log appears. How can I resolve such issue?

```
emac: Timed out waiting for PHY register 0x2 to have value 0x0243 (mask_↵  
↵0xffff). Current value:
```

You can refer to the following configurations of the development board. Please see the schematics for details:

- `CONFIG_PHY_USE_POWER_PIN=y`
 - `CONFIG_PHY_POWER_PIN=5`
-

2.2.19 I found a “Brownout detector was triggered” failure on my ESP32. How to resolve such issue?

- ESP32 has a built-in brownout detector which can detect if the voltage is lower than a specific value. If it happens, the detector will reset the chip to prevent unintended behavior.
- This message may be reported in various scenarios, while the root cause should always be that the chip with a power supply has momentarily or permanently dropped below the brownout threshold. Please try replacing power supply, USB cable, or installing capacitor on power supply terminals of your module.

- Apart from the above solution, you can also try to configure the reset threshold value or disable the brownout detector. For more information, please refer to [config-esp32-brownout-det](#).
- For ESP32 power-up and reset timing descriptions, see [ESP32 Series Datasheet](#).

2.2.20 After imported the `protocol_examples_common.h` header file, how come it cannot be found while compiling?

CHIP: ESP32

- Please add “`set(EXTRA_COMPONENT_DIRS $ENV{IDF_PATH}/examples/common_components/protocol_examples_co`” in `CMakeLists.txt` under the project.
- For more information, please refer to the [Build system documentation](#).

2.2.21 When using ESP8266 NonOS v3.0 SDK, the following error occurred. What could be the reason?

```
E:M 536      E:M 1528
```

Any error logs beginning with `E:M` indicates insufficient memory.

2.3 Environment setup

□

2.3.1 When setting up ESP32-S2 environment using command `idf.py set-target esp32s2`, an error occurred as “Error: No such command ‘set-target’”. What could be the reason?

- The ESP-IDF is adapted to ESP32-S2 from release/v4.2, thus setting up ESP32-S2 environment in previous versions will cause errors. In this case, when using command `idf.py set-target esp32s2`, there will be error as “Error: No such command ‘set-target’”. It is recommended to perform tests and development on ESP32-S2 using ESP-IDF release/v4.2 and later versions. For more information, please refer to [ESP32-S2 Get Started](#).
- To check which ESP chips are supported by different ESP-IDF versions, please refer to [ESP-IDF Release and SoC Compatibility](#).

2.3.2 When installing ESP-IDF version master using ESP-IDF Tools 2.3 in Windows system, an error occurred as: Installation has failed with exit code 2. What could be the reason?

This is related to the bad network environment. The Github repository cannot be downloaded smoothly under such network environment, causing SDK download failure on your PC. If you encounter Github access problems, it is recommended to use the **offline** version of the latest [ESP-IDF Windows Installer](#).

2.3.3 When setting up environment using esp-idf-tools-setup-2.3.exe on Windows, errors occurred when make menuconfig is executed:

```
-- Warning: Did not find file Compiler/-ASM Configure
-- Configuring incomplete, errors occurred!
```

This is because the system could not find the project to be compiled. You need to change directory to the ESP-IDF project before running commands to configure and compile the project. For example, to build the project hello world, go to esp-idf/examples/get-started/hello_world before running the commands.

2.3.4 When using esp-idf-tools-setup-2.2.exe in Windows system, a python error occurred during the installation:

```
Installation has failed with exit code 1
```

1. Update your tool chain: <https://dl.espressif.com/dl/esp-idf-tools-setup-2.3.exe>
 2. Remove the obsolete option “--no-site-packages” from idf_tools.py
-

2.3.5 What should I do if I get Download failed: security channel support error when installing build environment in the Windows system?

This is because the Windows system has disabled the default support for SSL3.0.

Solution: Go to *Control Panel* and find *Internet* option, select *Advanced*, and check the use *SSL 3.0* option.

2.3.6 When executing `export.bat` in Windows system, what should I do if I get CMake and gdbgui version errors?

```
C:\Users\xxxx\.espressif\tools\cmake\3.16.4\bin
The following Python requirements are not satisfied:
gdbgui>=0.13.2.0
```

This is because the upstream gdbgui has been updated, thus it is not compatible with the low version of python. The current solution is to manually modify the root file `requirements.txt` in ESP-IDF by changing the description of gdbgui version to `gdbgui==0.13.2.0`.

2.3.7 Errors occurred when using `idf.menuconfig` and `idf.build` after updating the ESP-IDF version from v3.3 to the latest one:

- Rebuild the environment following [Get Started](#).
- Remove build directory `build` and configuration file `sdkconfig` under the `hello_world` directory.

2.3.8 How to configure `PATH` and `IDF_PATH` when developing ESP32 and ESP8266 simultaneously?

- For `PATH`, there is no need to do extra configurations. You can put them together as: `export PATH="$HOME/esp/xtensa-esp32-elf/bin:$HOME/esp/xtensa-lx106-elf/bin:$PATH"`.
- For `IDF_PATH`, you can specify it for separate chips as:

In ESP32 related projects, use `IDF_PATH = $(HOME)/esp/esp-idf`. In ESP8266 related projects, use `IDF_PATH = $(HOME)/esp/ESP8266_RTOS_SDK`.

2.3.9 Do I need to use command `idf.py set-target` every time when switching to another project?

When building the project with `idf.py build`, the target is determined as follows:

1. If the build directory `build` already exists, the system will use the target the project was previously built for. It is stored in `CMakeCache.txt` file in the `build` directory.
2. Alternatively, if the build directory doesn't exist, the system will check if the `sdkconfig` file exists, and use the target specified there.
3. If both the build directory and `sdkconfig` file exist with different targets specified, the system will report an error. This shouldn't happen normally, unless `sdkconfig` was changed manually without deleting the build directory.
4. If neither `sdkconfig` file nor build directory exists, it can be considered to use `IDF_TARGET` to set the target as a CMake variable or environment variable. If this variable is set and is different from the target specified in `sdkconfig` or in the build directory, the system will also report an error.

5. Finally, if `sdkconfig` doesn't exist, build directory doesn't exist, and the target is not set via `IDF_TARGET`, then the system will use the default value. The default value can be set in `sdkconfig.defaults`.
6. If the target isn't set using any of the above methods, then the system will build for ESP32 target.

To answer your question:

- `idf.py set-target` stores the selected target in the project's build directory and `sdkconfig` file, not in the terminal environment. So, once the project is configured and built once for a certain target, if you switch to a different directory and build another project, then come back, the target will not change, and will be the same as previously set for this project. And it's not necessary to run `idf.py set-target` again other than to switch to a different target.
 - If you want to make the project built for certain target by default, add `CONFIG_IDF_TARGET="esp32s2"` to the `sdkconfig.defaults` file of the project. After this, if `sdkconfig` file doesn't exist and build directory doesn't exist, `idf.py build` command will build for that target specified in `sdkconfig.defaults`.
 - `idf.py set-target` command can still be used to override the default target set in `sdkconfig.defaults`.
-

2.3.10 How to know the version of ESP-IDF, is it recorded in a certain document?

- Command line: You can obtain the version number by inputting `idf.py --version` in the terminal with an IDF environment.
 - CMake script: You can obtain the version number through the variable `${IDF_VERSION_MAJOR}.${IDF_VERSION_MINOR}.${IDF_VERSION_PATCH}`.
 - Code compilation: You can obtain the version number by calling `esp_get_idf_version` during code compilation or directly using the macro definition of version in "components/esp_common/include/esp_idf_version.h".
-

2.3.11 How to optimize ESP-IDF compilation in Windows environment?

- Please add the directories of ESP-IDF source code and compiler `.espressif` to the exclusions of anti-virus program.
-

2.3.12 Is there an esptool that can be used directly on Windows?

- You can go to [esptool -> Releases](#) and download the Windows version of the esptool from the Asset column on the drop-down page.

2.4 Firmware update



2.4.1 How does the host MCU flash ESP32 via serial interfaces?

- For the related protocol, please refer to [ESP32 Serial Protocol](#). For the corresponding documentation, please refer to [Serial Protocol](#).
- For code examples, please refer to [esp-serial-flasher](#).

2.4.2 How to download firmware for ESP32 series modules using the USB-Serial tool?

The methods are as follows:

Modules	3V3	GND	TXD	RXD	IO0	EN
Serial tool	3V3	GND	RXD	TXD	DTR	RTS

Note: For ESP8266 modules, IO15 should be specially connected to ground.

2.4.3 How to flash firmware in macOS and Linux systems?

- For Apple system (macOS), you can use [esptool](#) downloaded via brew or git to flash firmware.
- For Linux system (e.g., ubuntu), you can use [esptool](#) downloaded via apt-get or git to flash firmware.

2.4.4 Does ESP32 support programming using JTAG pins directly?

Yes, ESP32 supports using [JTAG Pins](#) to flash directly. Please refer to [Upload application for debugging](#).

2.4.5 Does ESP_Flash_Downloader_Tool support customized programming control

- The GUI tool is not open-sourced and does not support embedded executive script.
- The low-level component [esptool](#) is open-sourced and can be used to perform all functions such as flashing and encryption. It is recommended to conduct secondary development based on this component.

2.4.6 Can I enable the Security Boot function for ESP32 via OTA?

- It is not recommended to enable Security Boot function through OTA, as it poses operational risks and requires multiple OTA firmware updates.
 - Since the Security Boot function is in Bootloader, please update Bootloader first to enable this function.
 1. First, check the partition table of your current device to see if it can store the Bootloader with Security Boot enabled.
 2. Then, update an intermediate firmware which can be written in Bootloader partition. By default, the Bootloader partition cannot be erased or written, you need to enable them via *make menuconfig*.
 3. Sign the intermediate firmware and upgrade it to the target device through OTA. Then upgrade the Bootloader of this firmware and the signed new firmware through OTA.
 4. If there are situations as powered-down or network break-down and restart during the Bootloader OTA process, the device would not be started and needs to be re-flashed.
-

2.4.7 How to resolve the following error occurred when flashing firmware to ESP32-S2 based on ESP-IDF v4.1?

```
esptool.py v2.9-dev
Serial port /dev/ttyUSB0
Connecting....
Chip is ESP32S2 Beta
Features: Engineering Sample
Crystal is 40MHz
MAC: 7c:df:a1:01:b7:64
Uploading stub...
Running stub...

A fatal error occurred: Invalid head of packet (0x50)
esptool.py failed with exit code 2
```

Solution

If you are using ESP32-S2 instead of ESP32-S2 Beta, please update ESP-IDF to v4.2 or later versions.

Notes:

- ESP-IDF v4.1 only supports ESP32-S2 Beta, which is not compatible with ESP32-S2.
 - The version of esptool came with ESP-IDF v4.1 is v2.9-dev, which only supports ESP32-S2 Beta as well.
 - Both ESP-IDF v4.2 and its esptool v3.0-dev support ESP32-S2 series chips.
-

2.4.8 How to download firmware based on ESP-IDF using flash_download_tool?

- Please refer to [get-started-guide](#) when building an ESP-IDF project for the first time.
- Taken hello-world example for instance, run `idf.py build` (Only for ESP-IDF v4.0 or later versions. Please use `make` for previous versions). After building, the following flash command for the bin file will be generated:

```
#Project build complete. To flash, run this command:
../../../../components/esptool_py/esptool/esptool.py -p (PORT) -b 921600 write_
↪flash --flash_mode dio --flash_size detect --flash_freq 40m 0x10000 build/
↪hello-world.bin build 0x1000 build/bootloader/bootloader.bin 0x8000 build/
↪partition_table/partition-table.bin
or run 'idf.py -p PORT flash'
```

You can use `flash_download_tool` to flash according to the bin file and flash address prompted by this command.

2.4.9 What is the communication protocol for flashing ESP chips?

- ESP Serial Protocol: [Serial Protocol](#).
 - Python-based implementation: [esptool](#).
 - C-language based implementation: [esp-serial-flasher](#).
-

2.4.10 How to program ESP32-C3's firmware offline?

- Currently, there is no tool that supports the offline programming of ESP32-C3's firmware. However, the [Flash Download Tools](#) we released can directly download binary firmware and support mass production download mode for up to eight ESP32-C3 devices at the same time.
 - In addition, we also offer the [Test Fixture](#) for mass production, which supports up to four ESP32-C3 modules to download firmware simultaneously.
-

2.4.11 How does ESP32 set Flash SPI to QIO mode?

- It can be set in menuconfig through `Serial flasher config -> Flash SPI mode`, the corresponding API is `esp_image_spi_mode_t()`.
-

2.4.12 After downloading program and powering on EPS8266, the serial port printed the following log. What is the reason?

```
ets Jan 8 2013,rst cause:1, boot mode:(7,7)
waiting for host
```

- *waiting for host* means the Boot is in SDIO mode, indicating that GPIO15 (MTDO) is pulled up (HIGH). Please refer to [ESP8266 Boot Mode Description](#).
-

2.4.13 What are the Espressif module programming tools?

- For Espressif programming software, please go to [Flash Download Tools](#). Installation-free GUI tools for Windows environment only.
 - Espressif programming tool `esptool` is written based on *Python* with open-source code, supporting secondary development.
-

2.4.14 What is the difference between the Factory mode and Developer mode of the flash download tool?

- Factory mode supports multi-channel downloads, while Developer mode only supports single channel.
 - The path of bin files under the Factory mode is relative, while under Developer is absolute.
-

2.4.15 Why does the programming failed for the jig with a 4-port hub in factory mode ?

:CHIP: ESP32 | ESP8266 :

- It is because Espressif products complete the calibration operation through transmitting some packets when starting up. This operation requires 3.3 V voltage and a guaranteed peak current of 500 mA. Therefore, when it comes to more than one ports, there will be situations where the computer cannot program or the programming is interrupted due to the insufficient power supply of the computer's USB when programming via connecting to a computer's USB. It is recommended to use the hub for programming and supply power to the hub in the meantime.
-

2.4.16 I'm using an ESP32-WROVER-B module to download the AT firmware via the flash download tool. However, an error occurred after writing to flash. But the same operation succeeded when replacing the module with ESP32-WROVER-E, what is the reason?

- The ESP32-WROVER-B module leads out the SPI Flash pin, but the ESP32-WROVER-E module does not. Please check whether the SPI Flash pin of the ESP32-WROVER-B module is re-used by other external application circuits.

- Connecting the CMD pin of the SPI Flash in ESP32-WROVER-B to GND will cause the flash failing to start. And the following error log will be printed:

```
rst:0x10 (RTCWDT_RTC_RESET),boot:0x1b (SPI_FAST_FLASH_BOOT)
flash read err, 1000
ets_main.c 371
ets Jun 8 2016 00:22:57
```

2.4.17 The encrypted device cannot be re-flashed via the flash download tool. What could be the reason?

CHIP: ESP32 | ESP32-S2

- Currently, an encrypted device cannot be flashed again using the [flash download tool](#). It only supports one-time encryption of plaintext.

2.4.18 When updating ESP32 firmware through UART interface based on esptool serial port protocol, can I add a new app partition?

- The partitions in flash depend on the data in partition_table.bin. If partition_table.bin can be updated, the storage space of other data, such as bootloader.bin and app.bin, can be redivided to create an app partition.

2.4.19 I am using ESP8266 to download the firmware via flash download tool. After downloading the firmware, there is no programming output log, and the serial port printed the following messages. What could be the reason?

```
ets Jan 8
2013,rst cause:1, boot mode:(3,7)
ets_main.c
```

- Please check whether the hardware wiring is correct. See [Boot mode wiring instructions](#).
- Please check whether the download offset address of bootloader.bin is correct. The offset address downloaded from bootloader.bin of ESP8266 is "0x0". If the offset address is wrong, the flash cannot be started.

2.4.20 Why does my USB driver failed to be recognized by the Windows7 system?

- Please download and install the *USB Serial JTAG driver* <<https://dl.espressif.com/dl/idf-driver/idf-driver-esp32-usb-jtag-2021-07-15.zip>> manually for the Windows7 system.

2.4.21 After using the ESP32-WROVER-E module to download the program, the following log is printed after powered on. What could be the reason?

```
rst0x10 RTCWDT_RTC_RESETboot:0x37SPI_FLASH_BOOT
2020-12-11 15:51:42 049invalrd header0xffffffff
invalrd header0xffffffff
invalrd header0xffffffff
```

- Generally, it is because the GPIO12 was pulled high. It is recommended to pull it low and see the results. Please see [ESP32 Boot Log Guide](#).

2.4.22 When using the Flash Download Tools to flash ESP32-C3 via USB, 8-download data fail occurs repeatedly. How can I solve it?

- Please erase the chip completely first before flashing
- This problem has been solved in V3.9.4 and above versions

APPLICATION SOLUTION

□

3.1 Android application

□

3.2 Artificial intelligence

□

3.2.1 What types of cameras are supported on AI image recognition products?

With ESP32 as its main control chip, ESP-EYE supports various types of cameras, such as OV2640, OV3660, OV5640, OV7725, etc. Please see [esp32-camera Github](#).

3.2.2 Which versions of ESP-IDF are supported by ESP-WHO?

The subsequent supported versions will be updated on [ESP-WHO Github](#).

3.2.3 Is there any information about the WeChat mini program of ESP-EYE?

For open source resources of ESP-EYE mini program demo, please check [EspEyeForWeChat](#).

3.2.4 What languages are supported by the esp-skainet demo?

Only Chinese and English currently.

3.2.5 What model frameworks does ESP-DL support?

Currently, [ESP-DL](#) supports models from mxnet, pytorch, and tensorflow.

3.2.6 Does ESP-DL support all models of the three platforms mentioned above (mxnet, pytorch, and tensorflow)?

ESP-DL supports models in which all the operators are supported by ESP-DL. Please check [layer](#) for the supported operators.

3.2.7 Can the model files of ESP-SKAINET be stored in the SD card?

Yes.

3.2.8 How to customize command words in ESP-SKAINET?

To customize command words, please see [ESP-SR GitHub](#).

3.2.9 How to reduce the system footprint of AI speech models?

You can choose to turn off the three functions, namely AEC, AE, and VAD.

3.2.10 What is the difference between a 16-bit quantization model and an 8-bit quantization model?

The 16-bit quantization model has higher precision and more accurate results, while the 8-bit quantization model is more lightweight.

3.2.11 How does the AI voice model modify the number of microphone channels?

The number of microphone channels and the number of playback channels can be configured in the AFE.

3.2.12 How do I get the actual audio captured in the development board?

To obtain the actual audio, an SD card interface is required to store the audio files to the SD card.

3.2.13 Do you have relevant study materials for ESP-SR GitHub?

Please refer to [ESP-SR User Guide](#).

3.2.14 Do you have relevant study materials for ESP-DL?

Please refer to [How to deploy hand gesture recognition with ESP-DL](#).

3.2.15 How does ESP32-S3 customize English command words for recognition?

- For MultiNet6, you need to prepare `commands_en.txt` to customize English command words. For MultiNet5, you need to use the `multinet_g2p.py` script to convert English command words into phonemes that can be recognized by multinet. For details, please refer to [esp-sr/tool](#).

3.3 AT

□

For ESP-AT FAQ, please go to [ESP-AT User Guide](#).

3.4 Audio development framework

□

3.4.1 What is the maximum power of supported speakers for ESP32 series audio development board?

- ESP32 development board uses NS4150 PA by default, and its maximum power is 3 W according to its datasheet.
-

3.4.2 Does Alexa solution have certain requirements for environmental noise?

- The current Espressif voice solution can meet the environmental requirements of a signal-to-noise ratio of less than 5dB, and for some fixed noise scenarios, it can also be less than 0dB (need to be optimized for the actual product).
-

3.4.3 There is an AUX input on the ESP32 AI development board, can MIC be used to pick up the sound?

- The ESP-ADF development framework can choose a variety of ways to pick up sound, including MIC input and Line-in.
- The pick-up method is as follows:

```
typedef enum {  
    AUDIO_HAL_CODEC_MODE_ENCODE = 1, /*! <select adc */           // MIC pickup  
    AUDIO_HAL_CODEC_MODE_DECODE, /*! <select dac*/  
    AUDIO_HAL_CODEC_MODE_BOTH, /*! <select both adc and dac */    // MIC +  
↪speaker  
    AUDIO_HAL_CODEC_MODE_LINE_IN, /*! <set adc channel *//,        //  
↪microphone pickup  
} Audio_hal_codec_mode_t;
```

- The configuration of the pickup method is as follows:

```
audio_board_handle_t board_handle = audio_board_init();  
audio_hal_ctrl_codec(board_handle->audio_hal, AUDIO_HAL_CODEC_MODE_DECODE,   
↪AUDIO_HAL_CTRL_START); //To MIC pickup, please modify this configuration  
↪option.
```

3.4.4 When using ESP32-WROVER-B module + ES8311 to design audio development board, which pins can be selected for MCLK clock?

- On the hardware side, MCLK can only use GPIO0, GPIO1, and GPIO3 pins. Other pins cannot be used. You can read [ESP32 Datasheet](#) about CLK_OUT* pins in IO_MUX table. GPIO0 is used by default.
 - Please refer to the [schamatic of ESP32-LyraT-Mini](#).
 - For allocation of pins, please refer to [ESP32-LyraT-Mini V1.2 Hardware Reference](#).
-

3.4.5 Can ESP32-WROVER-E module use one I2S line to realize simultaneous broadcasting and recording?

- Yes, you can refer to [ESP32-LyraT Development Board](#).
-

3.4.6 Do Espressif modules support Spotify Connect?

:CHIP: ESP32 | ESP32-S2 | ESP32-S3 :

- Not supported yet. It is suggested to try [dlna](#), which has similar functions.
-

3.4.7 When running the korvo_du1906 example on an ESP32-Korvo-DU1906 board, a reboot caused the following error message: *Guru Meditation Error: Core 0 panic'ed (IllegalInstruction). Exception was unhandled.* How to resolve such issue?

- Please check the power supply.
 - it is recommended that the system be connected to an at least 5 V/2 A power adapter for sufficient current supply.
-

3.4.8 Can ESP-DSP fft run 4096, 8192 and more samples?

- Yes, up to 32 K samples are supported. The maximum number can be configured in menuconfig, e.g., for [fft demo](#), go to `idf.py menuconfig-->Component config-->DSP Library-->Maximum FFT length-->(*) 32768`.
-

3.4.9 How to connect a microphone with ESP32?

- You can connect I2S peripheral if it is a digital microphone.
 - You can connect ADC peripheral if it is an analog microphone.
-

3.4.10 Does ESP32 support analog audio output or digital audio output?

- ESP32 supports DAC analog audio output for simple outputs such as tones. But if you use it for music playing, the effect will not be so desirable.
 - ESP32 supports PWM analog audio output, which has slightly better effect than DAC. The demo code is at [esp-iot-solution](#).
 - ESP32 also supports I2S digital audio output. For I2S configurable pins, please see [ESP32 Datasheet](#) > Chapter Peripherals and Sensors.
-

3.4.11 What audio formats does the ESP32 chip support?

The ESP32 chip supports audio formats such as MP3, AAC, FLAC, WAV, OGG, OPUS, AMR, G.711, etc. Please refer to the [ESP-ADF](#) SDK for instructions.

3.4.12 How to use the ESP32 chip to decode compressed audio?

- Application examples that use the ESP32 chip to decode compressed audio can be found under the [esp-adf/examples/recorder](#) folder.
-

3.4.13 Where is the code example for ESP-LED-Strip?

- Code examples for ESP-LED-Strip are provided in the ESP-ADF repo, please refer to [led_pixels](#) example.
-

3.4.14 Does ESP32 support online voice recognition?

- Yes. Please refer to [esp-adf/examples/dueros](#).

3.5 BLE Mesh development framework

□

3.5.1 What is the maximum data transmission load for Bluetooth® LE (BLE) mesh?

- Up to 384 bytes for the single packet in application layer, up to 11 bytes in the bottom layer with no sub-packages.
-

3.5.2 Could you provide an example of networking through ESP32 BLE-Mesh? What APP can be used for BLE-Mesh networking?

- Please use example [onoff_server](#) and use nRF Mesh APP for mobile phones.
 - For the network configuration process, please refer to [Getting Started with ESP-BLE-MESH](#).
-

3.5.3 For unprovisioned device in BLE-MESH, the default name is ESP-BLE-MESH, how to modify this name?

- You can use API `esp_ble_mesh_set_unprovisioned_device_name()`, it is suggested to call it after `esp_ble_mesh_init()`, otherwise the device name is still ESP-BLE-MESH.
-

3.5.4 How many node devices can ESP32's BLE-MESH application connect to?

- Theoretically, the ESP32 BLE-MESH application supports 32767 node devices. The number of connections supported in actual application depends on the memory usage.
-

3.5.5 How can I manually reset a BLE mesh device without the mobile provisioning app or provisioning device?

- You can call the `esp_ble_mesh_node_local_reset` interface to reset the BLE Mesh Node, erase all the provisioning information, and wait until the reset event arrives to confirm a successful reset. After the call, the device needs to be provisioned again.
-

3.5.6 After running the BLE MESH program for a long time on ESP32, it found that a segmentation fault occurs when the client sends a message to the server and printed NO multi-segment message contexts available. How to handle the fault?

- User can go to Component config -> ESP BLE Mesh Support -> Maximum number of simultaneous outgoing segmented messages and expand the space by configuring `BLE_MESH_TX_SEG_MSG_COUNT`.
-

3.5.7 Can I turn off network keys and IV updates when using the ESP32 BLE Mesh application?

- No, please don't. Network keys and IV updates are required for using the application.

3.6 Camera application

[]

3.6.1 What type of camera does the ESP32 series chips support?

- Please refer to [Camera models supported by ESP32 series](#).
-

3.6.2 Where is the factory firmware of ESP-EYE?

- Please refer to [ESP-EYE's factory firmware](#).
-

3.6.3 Does ESP32 support the camera with a 12-bit DVP interface?

No, the driver currently only supports an 8-bit DVP interface.

3.6.4 Does ESP32 support acquiring JPEG images using a camera without JPEG encoding?

- If the camera itself does not support JPEG encoding, you can refer to the [esp-iot-solution/examples/camera/pic_server](#) example provided by us to achieve software JPEG encoding on the ESP32 devices. This method encodes YUV422 or RGB565 data by software to obtain JPEG images.
-

3.6.5 Can the 2-megapixel OV2640 camera on the ESP-EYE be changed to only output 0.3-megapixel images?

Yes, you can specify the output resolution of the camera by configuring the value of `frame_size` during initialization.

3.6.6 Does ESP32 support a global shutter camera?

Yes. Currently, the camera models supported by ESP32 are SC031GS and SC132GS, while other cameras need additional driver support.

3.6.7 What is the frame rate when ESP32 transfers 1080P video via RTSP using the DVP camera?

We haven't conducted the test for 1080P yet. Currently, 720P can reach 20 FPS.

3.6.8 ESP32-S3 only supports MJPEG encoding, but H264/H265 format encoding is needed when implementing rtsp/rtmp streaming. Is there any encoding that supports H264/H265 format?

Currently, encoding in H264/H265 format is not supported.

3.6.9 Does ESP32/ESP32-S3 have a camera that supports wide-angle?

Yes, you can refer to the two cameras modules: BF3005 and OV5640.

3.6.10 It takes 5 seconds for ESP32-S2 to display the camera image from power-on. Can it be improved?

Yes, please refer to the following:

- Try to remove some delay functions in `esp_camera_init()`.
 - Change the clock frequency of `sccb` in `menuconfig` -> `component config` -> `camera configuration` to 400000.
-

3.6.11 Can ESP32 directly provide 24 MHz frequency to GC0308 camera?

I'm afraid not. The XCLK provided by ESP32 to GC0308 has been tested, with a maximum stable test value of 20 MHz.

3.6.12 Does ESP32/ESP32-S3 support MMS streaming protocol?

No, ESP32 and ESP32-S3 do not support the Microsoft Media Server (MMS) streaming protocol directly. The streaming media protocols supported by ESP32 and ESP32-S3 are RTSP and SIP. If you need to use ESP32 or ESP32-S3 for scenarios that require MMS protocol support, you can consider using middleware or converters that support the MMS protocol.

3.6.13 When debugging the GC2145 camera with ESP32-S3, the maximum supported resolution seems to be 1024x768. If it is adjusted to a larger resolution, such as 1280x720, it will print *cam_hal: EV-EOF-OVF* error. How to solve this issue?

In this case, it is necessary to reduce the PCLK of GC2145. For specific methods, try to configure a smaller XCLK and debug the PLL clock coefficient of the camera.

3.6.14 Does ESP32-S3 support GB28181 protocol?

Not yet supported.

3.6.15 Is there any reference for ESP32/ESP32-S2/ESP32-S3 to recognize the QR code through the camera?

Yes, please refer to the [code recognition](#) in ESP-WHO.

3.6.16 When adding the SD-card interface and camera interface for OV5640 sensor, we found that some pins of different ESP32 drivers conflicted with each other. Please suggest pins for the camera interface and SD-card interface.

ESP-WROVER-KIT development board includes the camera and SD card circuits, so you can refer to pins configuration of [ESP-WROVER-KIT V3 Getting Started Guide](#).

3.6.17 Can a driver for a specific camera model be added if the currently supported camera sensors do not meet my requirements?

Yes. Please confirm your requirements and select the camera sensor model with our engineers through [technical support](#). We can then provide the corresponding driver for your camera sensor.

3.6.18 How to add a custom resolution?

Suppose you need a resolution of 640x240. You can add the custom resolution in two ways:

- Configure the sensor to work at the typical resolution of 640x480 and only use the upper half of the data (640x240).
 - Add the identifier `FRAMESIZE_640*240` in `esp32-camera/driver/include/sensor.h`, and define the length and width of that resolution in `esp32-camera/driver/sensor.c` [<https://github.com/espressif/esp32-camera/blob/master/driver/sensor.c#L31>](https://github.com/espressif/esp32-camera/blob/master/driver/sensor.c#L31) as `{640, 240, ASPECT_RATIO_16X9}`. This method requires support for custom resolutions in the sensor's driver to work properly.
-

3.6.19 How to modify the register configuration of a camera sensor?

Suppose you want to change the register configuration of the OV5640 sensor. This can be achieved in two ways:

- Directly configure the relevant registers using `write_reg()` in the `reset()` function of `esp32-camera/sensors/ov5640.c`.
- Configure the relevant registers using `set_reg()` at the application level:

```
//Initialize the camera
esp_err_t ret = esp_camera_init(&camera_config);
sensor_t *s = esp_camera_sensor_get();
s->set_reg(s, 0xFFFFA, 0xFF, 0xA1);
```

3.6.20 What triggers “cam_hal: EV-VSYNC-OVF” in esp32-camera?

This issue occurs when the frame synchronization signal triggered by the sensor is too fast. You can troubleshoot it following the steps below:

- Run the `esp-iot-solution/examples/camera/pic_server` example. If the example runs correctly, it indicates that the issue is not hardware-related.
- Check the XCLK and resolution specified during sensor initialization. A smaller resolution or a larger XCLK can cause the frame synchronization signal triggered by the sensor to be too fast. Note that the XCLK used by the sensor should match the specified resolution.

3.7 Community sw and platforms

□

3.8 ESP Matter

□

3.8.1 Which ESP modules can be connected to Matter?

- Please refer to [Espressif Matter Platforms](#).

3.8.2 What learning materials are available to get started with ESP Matter?

You can refer to:

- [ESP Matter Github](#)
 - [Espressif Matter series blog](#)
 - [Espressif Matter solution video](#)
 - [Espressif Matter Demo video](#)
-

3.8.3 Can I compile and develop ESP Matter on Windows?

- It is currently not possible to compile on Windows. Also, it is not recommended to use a virtual machine for ESP Matter development, as the Matter Controller will use Bluetooth hardware where unknown errors may occur with the virtual machine. Using Linux or macOS systems directly are recommended.
-

3.8.4 Where can I find the Matter protocol documentation?

Currently, Matter's standard protocol has been made public, you can [submit a request and download the documentation](#) on the [CSA Alliance official website](#).

3.8.5 How to register an ESP Matter product?

- To register a Matter product requires a CSA membership. After the product passes the Matter certification test, it can be registered with the CSA.
-

3.8.6 When using Ubuntu virtual machine to develop ESP Matter on ESP32-C3, network provisioning failed when I followed the Matter official tutorial. What could be the reason?

It is not recommended to use a virtual machine for ESP Matter development, as the Matter Controller will use Bluetooth hardware where unknown errors may occur with the virtual machine. It is recommended to use Ubuntu 20.04 LTS and above hosts directly.

3.8.7 How to apply for a Matter DAC?

There are three ways to apply for a Matter DAC:

- Cooperate with an established PKI provider: Many organizations already have a certificate authority that they rely on to obtain their public key infrastructure certificates. In some cases, a Device Attestation Certificate (DAC) can be obtained from such a PKI provider.
 - Use your own PKI: Many organizations already have public key infrastructures for code signing, existing device authentication requirements, or other tasks that rely on asymmetric encryption. And public keys are distributed via digital certificates.
 - Cooperate with your platform vendor (Espressif): The platform vendor may embed a DAC in a chip or platform module using their VID/PID. CD is used to remap VID/PID using the `dac_origin_vid/dac_origin_pid` fields.
-

3.8.8 Does ESP Matter have a test tool/test app?

- Yes, it is recommended to use [chip-tool](#). You can also refer to [Configuration test chip tool](#).
-

3.8.9 Matter needs to use DCL in the network provision process. What is the specific function of DCL?

- Matter DCL is a secure and encrypted distributed storage network based on blockchain technology, which allows the Connection Standards Alliance (CSA) and authorized suppliers to publish their Matter device information. It also allows the Matter ecosystem to query related information through DCL clients.
 - For simplicity, Matter DCL will be used for device verification and OTA.
-

3.8.10 How to connect our Zigbee-based products with Matter through ESP chip?

- The device based on ZigBee technology is not a Matter standard device. At this time, you need to bridge the ZigBee device through the Matter Bridge device to access the Matter network.
 - Matter Bridge devices can be implemented using an Espressif Wi-Fi chip + 802.15.4 chip. Matter Bridge For BLE Mesh devices can be implemented with one Espressif Wi-Fi chip + BLE chip, or only one Wi-Fi + BLE combo chip.
-

3.8.11 Does Matter work with Samsung's smartthings?

- Yes, please refer to [Configuration test smartthings official blog](#).
-

3.8.12 Can Matter-enabled ESP devices be remotely controlled using Amazon/Google/Apple voice devices? Do these voice devices need to support the Matter protocol? (For example: Saying “Turn off the light” to turn off the lights in the house)

- Using Amazon/Google/Apple voice devices that support Matter protocol, it is possible to remotely control Matter ESP devices. In addition, if other ecosystems also support the ecosystem of the Matter protocol, then the home hub devices such as speakers in this ecosystem can also control the Matter devices remotely.
 - The specific practical steps are: build a Matter application scenario for testing based on the [esp-matter SDK](#). - [Google Matter Test Method](#) - [Apple Matter Test Method](#)
-

3.8.13 Does the product need to pass WiFi authentication and Bluetooth BQB authentication before submitting the Matter authentication?

- Yes. Matter is a protocol that runs on other technologies such as Wi-Fi, Ethernet, Thread, and Bluetooth. Before the Matter authentication, the device must be pass the transport layer protocol authenticated. This requires not only the original Wi-Fi or Thread authentication, but also the BQB authentication of the Bluetooth SIG, given that Matter requires the use of Bluetooth for provisioning.
-

3.8.14 Where is the DAC (Device Attestation Certificate) pre-imported by ESP Matter module stored?

- The DAC (Device Attestation Certificate) pre-imported by the ESP Matter module is stored in flash. In the Matter Pre-Provisioning service, the Matter DAC certificate is pre-flashed in the esp_secure_cert partition. An example of adding this partition to a partition table is as follows:

```
# ESP-IDF Partition Table
# Name,           Type, SubType, Offset,  Size, Flags
esp_secure_cert, 0x3F,      ,      0xd000, 0x2000, , # Never mark this as an
→encrypted partition
```

3.8.15 Can I configure Wi-Fi of ESP32 Matter devices by BLE

- Yes. For all the application examples in the [esp-matter SDK](#), Wi-Fi is configured by BLE. You can refer to [Section 2.2 Commissioning and Control](#).

3.9 ESP-NOW

□

3.9.1 What is the one-to-one bit rate for ESP32 in ESP-NOW mode

Test result:

- Test board: [ESP32-DevKitC V4](#).
 - Wi-Fi mode: station.
 - PHY rate is 1 Mbps by default.
 - Around 214 Kbps in opened environment.
 - Around 555 Kbps in shielding box.
 - If you require a higher rate, it's feasible to configure the rate through `esp_wifi_config_espnw_rate`.
-

3.9.2 What is ESP-NOW? What are its advantages and application scenarios?

- [ESP-NOW](#) is a connectionless communication protocol defined by Espressif.
 - In ESP-NOW, application data is encapsulated in action frames from different vendors and then transmitted from one Wi-Fi device to another without a connection.
 - ESP-NOW is ideal for smart lights, remote control devices, sensors, and other applications.
-

3.9.3 Can Wi-Fi be used with ESP-NOW at the same time?

- Yes, but it should be noted that the channel of ESP-NOW must be the same as that of the connected AP.
-

3.9.4 How do I set the rate at which ESP-NOW data is sent?

You may use the `esp_wifi_config_espnw_rate()` function to configure the rate, such as `esp_wifi_config_espnw_rate(WIFI_IF_STA, WIFI_PHY_RATE_MCS0_LGI)`.

3.9.5 ESP-NOW allows pairing with a maximum of 20 devices. Is there a way to control more devices?

You can use broadcast packets and provide the destination addresses in the payload. The number of addresses would then not be affected by the limited number. You only need to configure the correct broadcast address.

3.9.6 What is the maximum number of devices that can be controlled by ESP-NOW?

This depends on the specific communication method:

- If unicast packets are used, up to 20 devices can be paired and controlled at the same time.
 - If ESP-NOW encrypted mode is used, up to 6 devices can be paired and controlled at the same time.
 - If broadcast packets are used, theoretically there is no limitation in the number of devices that can be controlled. You only need to configure the correct broadcast address and take care of the interference issue when too many devices are paired.
-

3.9.7 Do I need to connect a router for communication between ESP-NOW devices?

ESP-NOW interacts directly from device to device and does not require a router to forward data.

3.9.8 Why does ESP-NOW limit the data length of each packet to 250 bytes? Can it be configured?

- The maximum length does not support configuration. ESP-NOW uses one vendor-specific element field of action frame to transmit ESP-NOW data, whose length field is only 1 byte (0xff = 255) as defined by IEEE 802.11. Thus, the maximum length of ESP-NOW data is limited to 250 bytes.
 - Alternatively, you may try with API `esp_wifi_80211_tx()` to send and use sniffer mode to receive. This way could fulfill the need of working only base on Wi-Fi stack without involving TCP/IP stack.
-

3.9.9 What should I pay attention to when using ESP-NOW applications?

- The device cannot switch channels after connecting to Wi-Fi. It can only transmit and receive data on the current Wi-Fi channel.
 - After the device enters Modem-sleep mode, it cannot receive data from ESP-NOW.
-

3.9.10 How can I reduce power consumption when using ESP-NOW?

You can use the following methods to reduce power consumption:

- If you use ESP-IDF in versions earlier than v5.0, when the AP is not connected, you can configure the wake-up window size and interval using the `esp_now_set_wake_window()` and `esp_wifi_set_connectionless_wake_interval()` functions respectively to save power.
- If you use ESP-IDF v5.0 or the latest master version, the functions are different from the other versions. Whether the AP is connected or not, you can use the `esp_now_set_wake_window()` and `esp_wifi_connectionless_module_set_wake_interval()` functions to set the wake-up window size and interval, respectively.

- Note that the issue of window synchronization between the sending end and receiving end needs to be considered in the application layer design. In this way, the chip will wake up at every “interval” and work for a period of time equalling the value of “window size”. Under this situation, you also need to configure `CONFIG_ESP_WIFI_STA_DISCONNECTED_PM_ENABLE=y` in `sdkconfig.defaults`.
-

3.9.11 In addition to wireless communication through ESP-NOW, is there any other better way to realize one-to-one and one-to-many communication?

One-to-one and one-to-many communication can also be realized by using SoftAP + Station. The master device applies Wi-Fi SoftAP mode to establish connections with multiple slave devices (Wi-Fi Station) at the same time.

3.9.12 Do ESP-NOW applications support sending packets over each Wi-Fi channel?

Yes, please refer to [ESP-NOW documentation](#).

3.9.13 Are there any special procedures required if I intend to use ESP-NOW for commercial purposes? Could you provide technical documentation on ESP-NOW? To evaluate the quality of wireless communication, I would like to know relevant information about parameters including CSMA/CA, modulation method, and bit rate.

- The application for ESP-NOW does not require any special procedures.
 - For technical documentation, please refer to [ESP-NOW User Guide](#). You can use examples in [ESP-NOW SDK](#) for testing.
 - The default bit rate of ESP-NOW is 1 Mbps.
-

3.9.14 I tested the application `esp-idf/examples/wifi/espnow` using ESP32. Does it only support connecting to 7 encrypted devices at the maximum?

- In the `esp-now` application, ESP32 supports connecting to no more than 17 encrypted devices, and the default value is 7. For more details, please refer to “[Add Paired Devices](#)”.
- If you want to change the number of paired encryption devices, set `CONFIG_ESP_WIFI_ESPNOW_MAX_ENCRYPT_NUM` in WiFi component configuration menu.

3.10 ESP RainMaker cloud service

[]

3.11 ios application

[]

3.12 Third party cloud service

[]

3.12.1 Are there any examples for OTA upgrading?

- For ESP8266, please refer to [ESP8266 OTA](#).
 - For ESP32 series, please refer to [ESP32 series OTA](#).
-

3.12.2 Does ESP Azure library support Azure IoT Central? Is there an example?

- ESP Azure already supports Azure IoT Central. But there is no relevant example on the master.
 - The PnP example on the ESP Azure's [preview/pnp_example](#) branch will report some actual data from sensors, you can refer to the operation of Azure IoT Central for the data management.
-

3.12.3 What should I do to connect ESP32 to Alibaba Cloud via ESP32 + Ethernet + MQTT?

- Use [esp-aliyun](#) but replace the Wi-Fi initialization code with Ethernet initialization. You can refer to the Ethernet example under [ESP_IDF](#).
-

3.12.4 what do Alexa LED states indicate?

- You can refer to [Alexa LEDs](#).

3.13 ESP-WIFI-MESH development framework



Note: If you have new requirements for Wi-Fi Mesh related application scenarios, we recommend that you directly use our newly launched [ESP-Mesh-Lite solution](#), instead of Wi-Fi Mesh.

3.13.1 What is the maximum data transmission load for Wi-Fi mesh?

- Up to 1456 bytes.
-

3.13.2 Does ESP32's Wi-Fi Mesh supports No Router self-networking?

- Yes, please refer to example [esp-mdf/examples/function_demo/mwifi/no_router](#).
-

3.13.3 What is the maximum number of node layers allowed when ESP32 uses Wi-Fi Mesh?

- In the Wi-Fi Mesh network, you can set the maximum number of layers via [esp_mesh_set_max_layer\(\)](#).
 - For tree topology structure, the maximum number is 25; while for chain topology structure, the maximum number is 1000.
-

3.13.4 When using an ESP32 development board to test the [esp-mdf/examples/function_demo/mwifi/router](#) example, After ESP32 is connected to the router, the device name in connection is “espressif”. How to modify this name?

- Please modify the “menuconfig → Component config → LWIP → (espressif) Local netif hostname” setting.
-

3.13.5 Can Wi-Fi Mesh send messages to specific nodes via TCP Server?

- Wi-Fi Mesh network can send data to the specified node or group in the TCP server, please refer to the [demo](#).
-

3.13.6 During the operation of the ESP32 Wi-Fi Mesh network, if the Root node is lost, what events will the system report back?

- If the Root node is lost, all nodes will trigger 'MDF_EVENT_MWIFI_PARENT_DISCONNECTED (MESH_EVENT_PARENT_DISCONNECTED)', and then start rescanning and re-election until a new Root node is elected.
-

3.13.7 I'm using ESP32 for Wi-Fi Mesh application with the `esp_mesh_send()` function, but the server did not receive any data. How to transfer data from leaf nodes to external servers?

- `esp_mesh_send()` can only be used for data communication within the Wi-Fi Mesh network.
 - If leaf nodes want to send data to an external server, the data needs to be forwarded through the root node.
 - The correct approach is: the leaf node first sends the data to the root node, and the root node then sends the data to the external server.
-

3.13.8 How do I upgrade my ESP-MESH device via OTA after networking?

- The ROOT node can connect to the server to get the upgrade bin file and then send the firmware to the corresponding module via MAC address for OTA upgrade.
 - For more information, please refer to [mupgrade demo](#).
-

3.13.9 Can you provide ESP-MESH light reference design?

- The overall design of the lamp is done by a third-party factory and we do not have a schematic or PCB layout. But from the module level, we only need to supply power to the chip and the chip outputs PWM to control the color or color temperature change of the lamp, which does not involve complicated design.
 - Please refer to [ESP-MDF](#) for more information on MESH.
-

3.13.10 What is the default mode for ESP-MESH nodes without any configuration?

- The default is IDLE mode.
-

3.13.11 ESP-MESH starts with AP+STA mode enabled, can the phone search for APs?

- No, ESP-MESH is a private protocol of Espressif, please refer to [WIFI-MESH Introduction](#) .
-

3.13.12 Do I need to rescan for all the newly added devices when the original device has already been networked?

- No, just scan through the current child nodes and find the one with the strongest signal as its parent node.
-

3.13.13 When using an ESP32 as a master device to synchronize time for multiple slave devices, can the time error be less than 2 ms?

- For this application scenario, it is recommended to develop based on esp-mdf, please refer to [esp-mdf/examples/development_kit/light](#) example.
 - Please use `esp_mesh_get_tsf_time()`, whose accuracy can meet your demand.
-

3.13.14 How do I get the type of the node in ESP-MESH?

- You can call `esp_mesh_get_type` interface to get it.
-

3.13.15 Is there any demo of ESP-Mesh root node sending messages to a service via ethernet?

- Please see [root_on_ethnernet](#) demo.
-

3.13.16 Does the esp-mesh-lite solution support the applications without routers?

- Yes, it supports. For the applications supported by esp-mesh-lite, please refer to [esp-mesh-lite features](#).
- You can conduct tests by enabling Component config > ESP Wi-Fi Mesh Lite > Enable Mesh-Lite > Mesh-Lite info configuration > [*] Join Mesh no matter whether the node is connected to router in the [esp-mesh-lite/examples/mesh_local_control](#) example.
- Please pay attention to the following tips if you want to use esp-mesh-lite without routers:
 - Identify a root node if possible, which can be set via `esp_mesh_lite_set_allow_level(1)`.

- It is recommended to use the `esp_mesh_lite_set_disallow_level(1)` function to prohibit the other nodes from being the root node.
 - In the applications of Mesh-Lite, a mesh network should be established based on some factors such as the distance of devices and the quality of Wi-Fi signal. As a result, you should test and debug the mesh network to ensure its performance and stability.
-

3.13.17 When `esp-wifi-mesh` is already networked, does the root or node device can also enable Wi-Fi Scan to scan surrounding available AP information?

- The Wi-Fi Scan feature is not supported on any node device when `esp-wifi-mesh` is already networked.
-

3.13.18 How can I switch to a new router for networking when using the `esp-wifi-mesh` router solution?

- You can modify the following code after the `MESH_EVENT_PARENT_DISCONNECTED` event:

```
mesh_router_t change_router = {
    .ssid = "TP-LINK_CSW",
    .password = "12345678",
    .ssid_len = strlen("TP-LINK_CSW"),
};
esp_mesh_set_self_organized(false, false);
esp_mesh_set_router(&change_router);
esp_mesh_set_self_organized(true, true);
```

SOFTWARE FRAMEWORK

[]

4.1 Bluetooth LE & Bluetooth

[]

4.1.1 When porting example gatt_server, an error occurred indicating head file does not exist. What could be the reasons?

When porting example gatt_server, an error occurred as fatal error: `esp_gap_ble_api.h: No such file or directory`, but this file is already included in the head file.

- Check sdkconfig to see whether sdkconfig.defaults is ported from the example or not. By default, Bluetooth® is disabled in SDK and needs to be enabled manually.
- If you are using cmake, the link configurations in the CMakeLists.txt file should be copied from the example too.

4.1.2 Does ESP32 support Bluetooth 5.0?

No, the ESP32 hardware only supports Bluetooth LE 4.2.

The ESP32 has passed the Bluetooth LE 5.0 certification, but some of its functions are still not supported on ESP32 (there will be a future chip which supports all functions in Bluetooth LE 5.0).

4.1.3 After the Bluetooth® LE starts advertising, why some mobile phones cannot successfully scan them?

- Please check whether your mobile phone supports Bluetooth LE function. Some mobile phones, such as iPhones, display Classic Bluetooth only in “Settings” -> “Bluetooth” (by default), and the Bluetooth LE advertisement will be filtered out.
 - It is recommended to use a dedicated Bluetooth LE application to debug the Bluetooth LE function. For example, LightBlue application can be used on iPhone.
 - Please check whether the advertising packet conforms to the specified format. Mobile phones tend to filter out packets that do not conform to the specified format and display only the correct ones.
-

4.1.4 Can I process OTA through Bluetooth® on ESP32?

Yes, please operate basing on `bt_spp_acceptor` and `bt_spp_initiator`.

If using Bluetooth LE, please operate basing on `ble_spp_server` and `ble_spp_client`.

4.1.5 How do ESP32 Bluetooth® and Bluetooth® LE dual-mode coexist and how can I use this coexistence mode?

The ESP32 Bluetooth and Bluetooth LE dual-mode does not require complex configurations. For developers, it is simple as calling Bluetooth LE API for Bluetooth LE, and calling Classic Bluetooth API for Classic Bluetooth.

For specifications on Classic Bluetooth and Bluetooth LE coexistence, please refer to [ESP32 BT&BLE Dual-mode Bluetooth](#).

4.1.6 What is the throughput of ESP32 Bluetooth® LE?

- The throughput of ESP32 Bluetooth LE depends on various factors such as environmental interference, connection interval, MTU size, and the performance of peer devices.
 - The maximum throughput of Bluetooth LE communication between ESP32 boards can reach up to 700 Kbps, which is about 90 KB/s. For details, please refer to example `ble_throughput` in ESP-IDF.
-

4.1.7 Does ESP32 support Bluetooth® 4.2 DLE (Data Length Extension)

Yes, Bluetooth 4.2 DLE is supported in all versions of ESP-IDF. There is no sample code provided currently. You can implement this by calling corresponding APIs directly. Please refer to `esp_ble_gap_set_pkt_data_len`.

4.1.8 How do ESP32 Bluetooth® and Wi-Fi coexist?

In the menuconfig, there is a special option called `Software controls WiFi/Bluetooth coexistence`, which is used to control the coexistence of Bluetooth and Wi-Fi for ESP32 using software, thus balancing the coexistence requirement for controlling the RF module by both the Wi-Fi and Bluetooth modules.

- Please note that if `Software controls WiFi/Bluetooth coexistence` is enabled, the Bluetooth LE scan interval shall not exceed `0x100 slots` (about 160 ms). If the Bluetooth LE and Wi-Fi coexistence is required, this option can be enabled or disabled. However, if this option is not enabled, please note that the Bluetooth LE scan window should be larger than 150 ms, and the Bluetooth LE scan interval should be less than 500 ms.
- If the Classic Bluetooth and Wi-Fi coexistence is required, it is recommended to enable this option.

4.1.9 How can I get ESP32 Bluetooth® Compatibility Test Report?

Please contact sales@espressif.com.

4.1.10 What is the transmit power of ESP32 Bluetooth®?

The ESP32 Bluetooth has 8 transmit power levels, corresponding to -12 ~ 9 dBm of transmit power, with a 3 dBm interval. The controller software limits the transmit power and selects the power level according to the corresponding power level declared by the product.

4.1.11 Could ESP32 realize bridging between Wi-Fi and Bluetooth® LE?

Yes, this function is developed on the application layer. You can retrieve data through Bluetooth LE and send them out via Wi-Fi. For detailed information, please refer to [Wi-Fi and Bluetooth LE Coexist demo](#).

4.1.12 What is the operating current of ESP32 Bluetooth® LE?

Current	MAX (mA)	Min (mA)	Average (mA)
Advertising: Adv Interval = 40 ms	142.1	32	42.67
Scanning: Scan Interval = 160 ms, Window = 20 ms	142.1	32	44.4
Connection(Slave): Connection Interval = 20 ms, latency = 0	142.1	32	42.75
Connection(Slave): Connection Interval = 80 ms, latency = 0	142.1	32	35.33

4.1.13 What kinds of Bluetooth® LE profiles does ESP32 support?

Currently, ESP32 Bluetooth LE fully supports some basic profiles, such as GATT/SMP/GAP, as well as some self-defined profiles. The ones that have already been implemented include Bluetooth LE HID (receiving side), Bluetooth LE SPP-Like, Battery, DIS, BluFi (Bluetooth Network Configuration-transmitting side), and so on.

4.1.14 How do I connect mobile phones and play music using ESP32 Bluetooth®?

ESP32 is used as an A2DP receiver when connected to a cell phone to play music. Please note that the A2DP Sink Demo uses a mobile phone to obtain SBC encoded data stream only. In order to play sounds, you will also need to decode the data and some peripherals, including codec modules, D/A converter, and speaker.

4.1.15 How is the ESP32 SPP performance?

When we use two ESP32 boards to run SPP, one-way throughput can reach up to 1900 Kbps (about 235 KB/s), which is close to the theoretical value in the specifications.

4.1.16 What is the maximum transmission rate for ESP32 Bluetooth® LE?

The transmission rate of ESP32 Bluetooth LE can reach 700 Kbps when it is tested in a shielded box.

4.1.17 How does ESP32 Bluetooth® LE enter Light-sleep mode?

On the hardware level, a 32 kHz external crystal should be added, or the Light-sleep mode will not take effect.

On the software level (SDK4.0 and later versions), the following configurations should be enabled in menuconfig:

- **Power Management:** | menuconfig > Component config > Power management > [*] Support for power management
- **Tickless Idle:** | menuconfig > Component config > FreeRTOS > [*] Tickless idle support (3) Minimum number of ticks to enter sleep mode for (NEW)

Note: Tickless idle needs to be enabled to allow automatic light-sleep mode. FreeRTOS will enter Light-sleep mode if no tasks need to run for 3 ticks (by default), that is 30 ms if tick rate is 100 Hz. Configure the FreeRTOS tick rate to be higher if you want to allow shorter duration of light-sleep mode, for example: menuconfig > ``Component config > FreeRTOS > (1000) Tick rate (Hz).

- **Configure external 32.768 kHz crystal as RTC clock source :** | menuconfig > Component config > ESP32-specific > RTC clock source (External 32 kHz crystal) [*] Additional current for external 32 kHz crystal

Note: The “additional current” option is a workaround for a hardware issue on ESP32 that the crystal can fail in oscillating. Please enable this option when you use external 32 kHz crystal. This hardware issue will be resolved in the next chip revision.

- Enable Bluetooth modem sleep with external 32.768kHz crystal as low power clock :!
menuconfig>Component config>Bluetooth>Bluetooth controller>
MODEM SLEEP Options>[*] Bluetooth modem sleep
-

4.1.18 Are there any documentation references for ESP32 BluFi networking?

For BluFi networking, please refer to [ESP32 BluFi](#). For BluFi networking examples, please refer to [BluFi](#).

4.1.19 What is the maximum transmission rate for ESP32 Classic Bluetooth® SPP?

In an open environment, the transmission rate for ESP32 Classic Bluetooth SPP can reach 1400+ Kbps ~ 1590 Kbps (only for reference, please do tests based on your actual application environment) with bi-directional transmitting and receiving simultaneously.

4.1.20 Is ESP32 Bluetooth® compatible to Bluetooth® ver2.1 + EDR protocol?

Yes. The ESP32 Bluetooth is downward-compatible, you can do tests using our official [Bluetooth examples](#).

4.1.21 How many Bluetooth® clients can be connected to ESP32?

The Bluetooth LE server supports up to nine client connections, please check the configuration of parameter `ble_max_conn` for applications. For stable connection, three clients should be good.

4.1.22 How can I get the MAC address of Bluetooth® devices for ESP32?

You can get the MAC address configured by Bluetooth via API `esp_bt_dev_get_address(void)`;; also the system pre-defined MAC address types via API `esp_err_t esp_read_mac(uint8_t* mac, esp_mac_type_t type);`.

4.1.23 What is the default Bluetooth® transmit power for ESP32 SDK?

- By default, the power level of ESP32 SDK is 5, and the corresponding transmit power is +3 dBm.
 - The power level of ESP32 Bluetooth ranges from 0 to 7, with the corresponding transmit power ranges from -12 dBm to 9 dBm. Each time the power level increases by 1, the corresponding transmit power will increase by 3 dBm.
-

4.1.24 Is it possible to use Wi-Fi Smartconfig and Bluetooth® LE Mesh for ESP32 simultaneously?

It is not recommended to use them simultaneously.

- The Smartconfig will need to receive the networking data, thus occupying the antenna all the time. If it is used together with Bluetooth LE Mesh, there will be an extremely high rate of failure.
 - The Bluetooth LE Mesh can be used together with BluFi. So it is recommended to use BluFi for networking.
-

4.1.25 What is the operating current for ESP32 Classic Bluetooth®

A2DP (Single core CPU 160 MhzDFS = falsecommit a7a90f)

Current	Maximum (mA)	Minimum (mA)	Average (mA)
Scanning	106.4	30.8	37.8
Sniff	107.6	31.1	32.2
Play Music	123	90.1	100.4

4.1.26 How can I modify the transmit power for ESP32 Bluetooth®

The Bluetooth transmit power can be configured via function `esp_ble_tx_power_set()`; Please refer to [esp_bt.h](#).

4.1.27 How is the networking compatibility of ESP32 Bluetooth® LE? Is it open-sourced?

- ESP32 Bluetooth networking, BluFi networking for short, has a good compatibility as Bluetooth LE and is compatible with many mainstream mobile phones such as Apple, HUAWEI, Mi, OPPO, MEIZU, OnePlus, ZTE and etc.
 - Currently, the BluFi protocol and phone application code is open-sourced.
-

4.1.28 When I execute example `bt_spp_acceptor` on ESP32, the IOS device cannot find the ESP32 device during scanning. What could be the reasons?

- Apple has opened Bluetooth® as: A2DP, HID's keyboard, avrcp, SPP (need MFI), high-level Bluetooth LE and ANCS for Bluetooth LE.
 - If the IOS device expects to communicate with the end device via SPP, the SPP of the end device should have the MFI certificate. However, ESP32 SPP does not have the MFI certificate, thus the IOS device cannot find ESP32 during scanning.
-

4.1.29 How is the security of ESP32 Bluetooth® LE/Bluetooth® Secure Simple Pairing (SSP) compared to legacy pairing?

- Secure Simple Pairing (SSP) is more secure than legacy pairing.
 - The legacy pairing uses symmetric encryption algorithm, while Secure Simple Pairing (SSP) uses asymmetric cryptography algorithm.
-

4.1.30 How can I confirm the MTU size of ESP32 Bluetooth® LE?

- By default, the MTU size of ESP32 Bluetooth LE is 23 bytes, and can be configured to reach 517 bytes.
 - For phones, the MTU size can be self-defined. Then, the end device with a smaller MTU will be chosen for communication.
-

4.1.31 When advertising in ESP32 Bluetooth® LE mode, an error occurred as “W (17370) BT_BTMT: data exceed max adv packet length”. How can I resolve such issue?

- This is because the advertising data has exceeded the maximum advertising packet length.
 - The maximum data length of advertising payload is 31 bytes. If the actual data length exceeds 31 bytes, the Bluetooth protocol stack will drop some data and generate an error warning.
 - If the data to be advertised exceeds the maximum packet length, the extra data can be put in the scan response packet.
-

4.1.32 Does ESP32 Bluetooth® LE support Client-Server mode, in which gatt server and gatt client can coexist?

- Yes, please refer to example `gattc_gatts_coex`.
-

4.1.33 What are the risks if there are over six devices connected to ESP32 Bluetooth® LE?

- Usually it depends on the specific application scenario. In general, the ESP32 Bluetooth LE can communicate stably with three devices connected.
 - There is no exact number for maximum Bluetooth LE connections. When there are multiple devices connected to Bluetooth LE simultaneously, the RF is time-multiplexed, thus requiring the designer to ensure that each device is not overly occupied, causing other devices to timeout and disconnected.
 - The connection parameters include: connection interval, connection window, latency and timeout. It is ok for devices to not respond within the latency, but if the responding time exceeds timeout threshold, the device will be disconnected.
 - If the interval is configured to 100 and window to 5, the Bluetooth LE will be able to connect to more devices with Wi-Fi disconnected. However, If Wi-Fi is connected and the value of interval is too small, only a few devices can be connected.
 - When the Bluetooth LE supports multiple devices connected simultaneously, there will be bigger possibility for RF slot management to generate error. So when there are multiple connections for Bluetooth LE, it is necessary to debug for different scenarios.
-

4.1.34 When using ESP32 device as the server of Bluetooth® LE, how many client devices can be connected?

- The ESP32 Bluetooth LE supports up to nine client devices for connection. It is recommended to hold this number within three.
 - Please make configurations via `menuconfig > Component config > Bluetooth > Bluetooth controller > BLE MAX Connections`.
-

4.1.35 How can I send files via Bluetooth® BR/EDR for ESP32?

- Please refer to example `bt_spp_acceptor` or `bt_spp_initiator` in [classic bt](#).
-

4.1.36 When I download example ESP_SPP_SERVER for ESP32, how can I modify the name of the Bluetooth® device?

- The name of the Bluetooth device can be modified via `adv` parameter:

```
static const uint8_t spp_adv_data[23] = {
    0x02, 0x01, 0x06,
    0x03, 0x03, 0xF0, 0xAB,
    0x0F, 0x09, 0x45, 0x53, 0x50, 0x5f, 0x53, 0x50, 0x50, 0x5f, 0x53, 0x45, 0x52, 0x56, 0x45,
    ↪ 0x52};
```

- The “0x0F” in the third line means the length of the following data is 15, “0x09” stands for data type (fixed) and data from “0x45” indicates the corresponding ASCII code of the device names (BLE_SPP_SERVER by default).
-

4.1.37 When I use the “BluFi” example to configure network for ESP32, the Wi-Fi cannot be connected during the distribution process via the EspBluFi application since a wrong Wi-Fi has been configured. Then the device is restarted after sending a SCAN command from the application. What is the reason?

- The “BluFi” example stipulates that Wi-Fi “SCAN” commands cannot be sent when Wi-Fi is connected.
- To solve this issue, you can add `ESP_ERROR_CHECK(esp_wifi_disconnect());` to the first line of the `ESP_BLUFI_EVENT_GET_WIFI_LIST: {};` function under the `blufi_example_main.c` file.

4.1.38 How can I specify a BLE connection/transmit operation to run on core 0 when I use ESP32?

- Currently, ESP32’s BLE connection/transmit operation only can be run on core 1. You can enable this via `menuconfig>Component config>FreeRTOS>Run FreeRTOS only on first core.`
- According to this application requirement, you can distribute tasks to a certain core using the “`xTaskCreatePinnedToCore()`” or “`xTaskCreateStaticPinnedToCore()`” API. For specific instructions, please see [core assignment](#).

4.1.39 When I set name for the bluetooth of an ESP32 device using Chinese characters, messy code shows instead. What is the reason

- This is because the Chinese encoding format of the editor is not UTF-8 at this time, and the encoding format of the editor needs to be changed to UTF-8.

4.1.40 When I upload sub-packages to the Bluetooth channel using ESP32, the maximum transmission data length of a packet is 253 (MTU is set to 263). This results in slower transmission when a large number of data packets are transmitted for multi-packet reading. Is there a BluFi extension protocol that can support the transmission of a larger length of data in one packet, or are there other solutions to increase the transmission rate?

- The transmission is slow When a large number of data packets on the Bluetooth channel are transmitted for multi-packet reading. You can improve the transmission speed by adjusting the Bluetooth connection parameters.
- The BLE packet length setting depends on the `ESP_GATT_MAX_MTU_SIZE` setting, please refer to the [Description](#).
- The configured MTU size will affect the data transmission rate. The effective MTU length needs to be changed by MTU exchange to change the default MTU size. The MTU size used in the final MTU exchange is used as the MTU size for the communication between the two devices. You can check the value of the MTU after exchange, such as the follows:

```
case ESP_GATTS_MTU_EVT:
    ESP_LOGI(GATTS_TAG, "ESP_GATTS_MTU_EVT, MTU%d", param->mtu.mtu);
```

4.1.41 What profile does ESP32's classic Bluetooth® support?

- Currently, it supports A2DP, AVRCP, SPP, HFP, and HID.
-

4.1.42 How many stable connections can be reached for ESP32-C3's Bluetooth® LE (BLE)?

- We recommend the connection number does not exceed four.
-

4.1.43 How can I adjust the BLE advertising interval?

- The advertising interval is decided by `adv_int_min` and `adv_int_max` parameters in BLE advertising struct, which configures the minimum and maximum advertising interval respectively.
 - The advertising interval ranges from 0x0020 to 0x4000 and the default value is 0x0800. The interval time is the value * 0.625 ms, i.e., 20 ms to 10.24 sec.
 - If the values of `adv_int_min` and `adv_int_max` are different, the advertising interval is within the range of the two values. If the values are the same, the interval will be this fixed value.
-

4.1.44 How can I input the PIN code via mobile phone during ESP32's Classic Bluetooth Pairing mode?

You can disable Secure Simple Pairing to support only Legacy Pairing.

- From esp-idf v3.3 to v4.0 (not include v4.0): `Component config > Bluetooth > Bluedroid Enable > [*] Classic Bluetooth > [] Secure Simple Pairing`
 - esp-idf v4.0 and above: `Component config > Bluetooth > Bluedroid Options > [] Secure Simple Pairing`
-

4.1.45 How much memory does ESP32 Bluetooth occupy?

- Controller:
 - BLE single mode: 40 KB
 - BR/EDR single mode: 65 KB
 - Dual mode: 120 KB
- Main equipment:
 - BLE GATT Client (Gatt Client demo): 24 KB (.bss+.data) + 23 KB (heap) = 47 KB
 - BLE GATT Server (GATT Server demo): 23 KB (.bss+.data) + 23 KB (heap) = 46 KB
 - BLE GATT Client & GATT Server: 24 KB (.bss+.data) + 24 KB (heap) = 48 KB
 - SMP: 5 KB
 - Classic Bluetooth (Classic Bluetooth A2DP_SINK demo, including SMP/SDP/A2DP/AVRCP): 48 KB (.bss+.data) + 24 KB (heap) = 72 KB (an additional 13 KB is added when the example is running)

Note: The above heap (Heap) all include the task stack (Task Stack), because the task stack is allocated from the heap and considered as a heap.

- Optimized PSRAM version:

In ESP-IDF v3.0 and later versions, if you open the PSRAM related options of the Bluetooth menu in `menuconfig`, and put part of the .bss/.data section and heap of Bluedroid (Host) into PSRAM, almost 50 KB memory space can be saved.

4.1.46 When I use the “gattc_gatts_coex.c” example on ESP32 to test BLE multi-connection, it can only connect to four devices even after I set the BLE Max connections in menuconfig to five. What is the reason?

- Please set the BT/BLE MAX_ACL_CONNECTION in `menuconfig` to five.
-

4.1.47 Does ESP32-C3 BLE support master and slave mode at the same time? What is the number of connections in master mode and slave mode?

IDF: release/v4.3, master

- ESP32-C3 supports master and slave mode at the same time, which share 8 connections. For example, if ESP32-C3 connects to 4 slave devices, it can be connected by 8 - 4 = 4 master devices.
 - In addition, when ESP32-C3 is used as a slave, it can be connected by 8 master devices; when used as a master, it can connect to 8 slave devices.
-

4.1.48 What is the maximum MTU Size of ESP32 Classic Bluetooth?

- ESP32 Classic Bluetooth has two protocols, namely A2DP and SPP. The maximum MTU Size setting of BT A2DP (default) is 1008 bytes, of which the header occupies 12 bytes and the actual amount of data transmitted by the application layer is $1008 - 12 = 996$ (bytes); the maximum MTU Size of BT SPP (default) Set to 990 bytes.
-

4.1.49 How can I resolve the frequently occurred ELxXX error (such as ELx200) when Wi-Fi and Ble co-exit

CHIP: ESP32

- It has been fixed in commit 386a8e37f19fecc9ef62e72441e6e1272fa985b9. Please switch to the corresponding commit to test.
-

4.1.50 How does BLE capture packets?

- There are many available tools, such as:
 - TI Packet sniffer
 - NRF Packet sniffer
-

4.1.51 When I use an ESP32 development board to test several versions of bluefi example under ESP-IDF for networking, the following error kept printing. What is the reason?

```
E (117198) BT_L2CAP: l2ble_update_att_acl_pkt_num not found p_tcb
W (117198) BT_BTC: btc_blufi_send_encap wait to send blufi custom data
```

- When this error occurs, please modify the `esp_ble_get_cur_sendable_packets_num(blufi_env.conn_id)` to `esp_ble_get_sendable_packets_num()` in the `components/bt/host/bluedroid/btc/profile/esp/blufi/blufi_prf.c` file.
 - This bug has been fixed in all branches, you can update ESP-IDF to the latest release version.
-

4.1.52 When I use ESP32, can Light-sleep mode be enabled for Bluetooth and can Bluetooth be kept connected in Light-sleep mode?

- To use Light-sleep mode for ESP32, release/4.0 or above versions of ESP-IDF and a 32.768 kHz crystal are needed.
 - Bluetooth can be kept connected in Light-sleep mode. Please refer to [Bluetooth modem sleep with external 32.768 kHz xtal under light sleep](#).
-

4.1.53 How can I modify the Bluetooth device name of ESP32?

- The structure to be modified is as follows:

```
static uint8_t raw_adv_data[] = {

/* flags*/

0x02, 0x01, 0x06,

Tx power*/

0x02, 0x0a, 0xeb,

/* service uuid*/

0x03, 0x03, 0xFF, 0x00,

/* device name*/

0x0f, 0x09, 'E', 'S', 'P', '_', 'G', 'A', 'T', 'T', 'S', '_', 'D', 'E', ' ', 'M', 'O'

};
```

- The above `/* device name*/` is the modified item. Among them, `0x0f` is the total length of the field type plus specific content, and `0x09` indicates that this type refers to the device name. Subsequent `'E','S','P','_','G','A','T','T','S','_','D','E',' ','M','O'` are the ASCII code of the broadcast device name.

4.1.54 What is the maximum supported broadcast length of BLE 5.0 broadcast after it is set to legacy mode?

- The maximum supported length is 31-byte.

4.1.55 How can I set a BLE broadcast package as unconnectable package?

CHIP: ESP32

- please refer to the [gatt_server demo](#) and set `adv_type` as `ADV_TYPE_NONCONN_IND`.

```
static esp_ble_adv_params_t adv_params = {
    .adv_int_min      = 0x20,
    .adv_int_max      = 0x40,
    .adv_type          = ADV_TYPE_NONCONN_IND,
    .own_addr_type     = BLE_ADDR_TYPE_PUBLIC,
    //.peer_addr        =
    //.peer_addr_type   =
    .channel_map       = ADV_CHNL_ALL,
    .adv_filter_policy = ADV_FILTER_ALLOW_SCAN_ANY_CON_ANY,
}
```

4.1.56 How can I send Bluetooth HCI commands directly to ESP32-WROOM-32D module through the serial port?

- Please refer to `controller_hci_uart_esp32`.
 - When ESP32 is used as a controller, and the other device serves as a host, HCI commands can be sent to ESP32 via UART.
-

4.1.57 Does ESP32 support transmitting audio stream using A2DP?

Yes, please refer to example `a2dp_source`.

4.1.58 How many devices can be connected at the most as suggested by the White List of ESP32 Bluetooth LE?

- The maximum supported number is 12.
-

4.1.59 Can ESP32 Bluetooth LE use PSRAM?

To enable Bluetooth LE to use PSRAM, please go to Component config > Bluetooth > Bluedroid Options and enable BT/BLE will first malloc the memory from the PSRAM

4.1.60 When using ESP32-C3 BLE Scan, can I set it to only scan the Long Range devices?

- Yes, you can make tests based on `esp-idf/examples/bluetooth/bluedroid/ble_50/ble50_security_client`. By changing the configuration `.cfg_mask = ESP_BLE_GAP_EXT_SCAN_CFG_UNCODE_MASK | ESP_BLE_GAP_EXT_SCAN_CFG_CODE_MASK` in `ext_scan_params` to `.cfg_mask = ESP_BLE_GAP_EXT_SCAN_CFG_CODE_MASK`, you can scan the broadcast packets whose primary PHY type is LE CODED PHY.
-

4.1.61 Is there a limit to the name length of ESP32 as a Bluetooth device?

- The names should be no longer than 248 bytes. However, in practice, the name length is also limited by the length of Bluetooth advertising packets. For the description of configurations, please refer to `CONFIG_BT_MAX_DEVICE_NAME_LEN`.
-

4.1.62 How do I set the ESP32 BLE Scan to the permanent scan without generating a timeout?

- You can realize this by setting “duration” to 0 before using the `esp_ble_gap_start_scanning()` function to start BLE Scan.

4.1.63 How can I get RSSI of BLE devices through ESP32?

- You can use the `esp_ble_gap_read_rssi()` function to get RSSI of connected BLE devices.
- If you want to get RSSI of all scanned BLE devices around, please use the `ble_scan_result_evt_param` structure in the `ESP_GAP_BLE_SCAN_RESULT_EVT` event to enable the printing of RSSI.

4.1.64 How can I increase the transmission distance of BLE5.0? How can I set BLE5.0 to long-range mode?

- In practice, the transmission distance of BLE5.0 is about 200 m. It is recommended to refer to the actual test distance. ESP32-S3 supports the features of BLE5.0, and supports long-range communication through Coded PHY (125 Kbps and 500 Kbps) and broadcast extension.
- You can realize long-range communication by using 125 Kbps Coded PHY and increasing the transmit power (`tx_power`). Refer to the following settings:

```
esp_ble_gap_ext_adv_params_t ext_adv_params_coded = {
    .type = ESP_BLE_GAP_SET_EXT_ADV_PROP_SCANNABLE,
    .interval_min = 0x50,
    .interval_max = 0x50,
    .channel_map = ADV_CHNL_ALL,
    .filter_policy = ADV_FILTER_ALLOW_SCAN_ANY_CON_ANY,
    .primary_phy = ESP_BLE_GAP_PHY_CODED,
    .max_skip = 0,
    .secondary_phy = ESP_BLE_GAP_PHY_CODED,
    .sid = 0,
    .scan_req_notif = false,
    .own_addr_type = BLE_ADDR_TYPE_RANDOM,
    .tx_power = 18,
};
```

- For the BLE5.0 examples, please refer to `ble_50` examples in ESP-IDF.

4.1.65 I have changed the name of the Bluetooth device with `esp_ble_gap_set_device_name()` in ESP32-C3. It works for Android devices and the customized device name can be shown. However, it does not work on IOS devices. The device name is still the default Bluetooth name. How can I make it work on Apple devices as well?

- In this case, you need to use raw data to create BLE advertising packets. First, enable the `CONFIG_SET_RAW_ADV_DATA` option in `menuconfig` (`idf.py menuconfig` > Example 'GATT

SERVER' Config > Use raw data for advertising packets and scan response data), and then customize [Broadcast packet structure in the gatt server example](#).

- Please use nRF Connect APP to test. We have tested and it works on the nRF connect APP. This issue is related to IOS APPs.
-

4.1.66 I want to use two ESP32 development boards to test the Bluetooth connection. How can I set the specified key to automatically connect them with `gatt_security_client` and `gatt_security_server` examples?

- In `gatt_security_client` and `gatt_security_server` examples, the default key is 123456. For details, please refer to `uint32_t passkey = 123456`. You can also set other passwords.
 - Since the ESP32 device has no display or input keyboard by default, the example sets the IO capability to `No output No input`. For more details, please refer to [Gatt Security Server Example Walkthrough](#).
 - To manually input the key, please set `esp_ble_io_cap_t iocap` in the `gatt_security_server` example to `ESP_IO_CAP_OUTmode`, and then you can use the nRF Connect APP to establish a connection with the BLE Server.
-

4.1.67 After setting `gatt_security_server` to `ESP_IO_CAP_OUTmode` and setting `gatt_security_client` to `ESP_IO_CAP_OUT mode`, I deliberately set the wrong passkey. However, the two development boards can still be connected. What is the reason?

- When the server is set to `ESP_IO_CAP_OUTmode`, `gatt_security_client` should be set to `ESP_IO_CAP_IN mode`.
- To avoid such a situation, please add the following code into the `case ESP_GAP_BLE_PASSKEY_REQ_EVT` event on the `gatt_security_client` side:

```
esp_ble_passkey_reply(param->ble_security.ble_req.bd_addr,true,123457);
```

4.1.68 Does ESP32-C3/ESP32-C6/ESP32-S3 support Bluetooth AOA/AOD?

- ESP32-C3/ESP32-C6/ESP32-S3 does not support Bluetooth AOA/AOD. Currently, none of Espressif products support Bluetooth AOA/AOD.
-

4.1.69 What is the maximum length of data in a BLE advertising packet supported by ESP32-C3 with the BLE5.0 feature?

- The maximum length is 1650 bytes, which can be set via the `esp_ble_gap_config_ext_adv_data_raw()` API.

4.1.70 Does ESP32 have any API to check whether BLE advertising has started or stopped?

- For bluedroid stack, there is no such API currently.
- For Nimble stack (and using non-extended advertising of BLE 4.2), you can use the `ble_gap_adv_active` API.

4.2 Ethernet

□

4.2.1 When building an example on ESP32 Ethernet development board, an error occurred as “emac: Reset EMAC Timeout”. What could be the reasons?

This is because the EMAC initialization is timeout, and is possibly related to the RMII clock. It is recommended to check your hardware, e.g., see if the PHY crystal oscillator is a cold joint.

4.2.2 When ESP32 connected to LAN8720 externally, with GPIO0 providing CLK, the initialization of Ethernet example failed. How to resolve such issue?

```
I (229) cpu_start: App cpu up.
I (247) heap_init: Initializing. RAM available for dynamic allocation:
I (254) heap_init: At 3FFAE6E0 len 00001920 (6 KiB): DRAM
I (260) heap_init: At 3FFB40A8 len 0002BF58 (175 KiB): DRAM
I (266) heap_init: At 3FFE0440 len 00003AE0 (14 KiB): D/IRAM
I (273) heap_init: At 3FFE4350 len 0001BCB0 (111 KiB): D/IRAM
I (279) heap_init: At 400885D0 len 00017A30 (94 KiB): IRAM
I (285) cpu_start: Pro cpu start user code
I (303) cpu_start: Chip Revision: 1
W (303) cpu_start: Chip revision is higher than the one configured in
↳ menuconfig. Suggest to upgrade it.
I (307) cpu_start: Starting scheduler on PRO CPU.
I (0) cpu_start: Starting scheduler on APP CPU.
I (319) system_api: Base MAC address is not set, read default base MAC
↳ address from BLK0 of EFUSE
E (1329) emac: Timed out waiting for PHY register 0x2 to have value
↳ 0x0007(mask 0xffff). Current value 0xffff
E (2329) emac: Timed out waiting for PHY register 0x3 to have value
↳ 0xc0f0(mask 0xfff0). Current value 0xffff
E (2329) emac: Initialise PHY device Timeout
```

(continues on next page)

(continued from previous page)

```

ESP_ERROR_CHECK failed: esp_err_t 0xffffffff (ESP_FAIL) at 0x40084140
0x40084140: _esp_error_check_failed at /mnt/hgfs/workspace/esp32/IDF/esp-idf-
↳v3.3/components/esp32/panic.c:720

file: "/mnt/hgfs/workspace/esp32/project/ethernet/main/ethernet_example_main.
↳c" line 153
func: app_main
expression: esp_eth_enable()

ELF file SHA256:
↳`597d55ebf237c1cffa5f47c73148a159b22726d94a7b78100bd941d7d5fc906e`

Backtrace: 0x40083cdc:0x3ffb5e80 0x40084143:0x3ffb5ea0 0x400d32c1:0x3ffb5ec0
↳0x400d1742:0x3ffb5f20 0x40085d91:0x3ffb5f40
0x40083cdc: invoke_abort at /mnt/hgfs/workspace/esp32/IDF/esp-idf-v3.3/
↳components/esp32/panic.c:715

0x40084143: _esp_error_check_failed at /mnt/hgfs/workspace/esp32/IDF/esp-idf-
↳v3.3/components/esp32/panic.c:721

0x400d32c1: app_main at /mnt/hgfs/workspace/esp32/project/ethernet/main/
↳ethernet_example_main.c:153 (discriminator 1)

0x400d1742: main_task at /mnt/hgfs/workspace/esp32/IDF/esp-idf-v3.3/
↳components/esp32/cpu_start.c:542

0x40085d91: vPortTaskWrapper at /mnt/hgfs/workspace/esp32/IDF/esp-idf-v3.3/
↳components/freertos/port.c:403

Rebooting...

```

- Please check if there are capacitors on IO0. It is better to have no capacitors on IO0 when it is used as the CLK output pin, or it would affect the timing.
- When GPIO0 starts to output RMII clock, please remember to check the CONFIG_PHY_CLOCK_GPIO0_OUT option in Kconfig.
- For references about Ethernet, besides the README in example, you can also refer to [API Reference](#).

4.2.3 When using the Ethernet example in ESP-IDF, an error occurred as “Timed out waiting for PHY register 0x3 to have value 0xc0f0 (mask 0xfff0). Current value 0xffff”. How to resolve such issue?

- Please refer to [BBS issue](#) and [Github issue](#).
- When the value of the PHY register is 0xFFFF, please check the following:
 - a. If the wiring of MDIO and MDC is correct
 - b. If the 50 MHz clock required by RMII is normal
 - c. If the address of PHY (both software and hardware) is correctly configured

- It is strongly recommended to check the strapping pins that control the PHY address to make sure they are not floating and **not in the default state**. Please make sure they have already been pulled up or down by external resistance.
- If you are still not sure about the PHY address, you can try to set the PHY address from 0 to 31, and read the PHY ID register to see if you can get the correct data. If yes, then write down the current PHY address.

4.2.4 When using ESP-IDF v4.1, how to set the static IP for ESP32 Ethernet?

Since v4.1 and later versions of ESP-IDF will remove the tcp/ip interface, it is recommended to use the [ESP-NETIF](#) interface.

Code example:

```
{
    ...
    esp_netif_config_t cfg = ESP_NETIF_DEFAULT_ETH();
    esp_netif_t *eth_netif = esp_netif_new(&cfg);
    // Set default handlers to process TCP/IP stuffs
    ESP_ERROR_CHECK(esp_eth_set_default_handlers(eth_netif));
    ...
    char* ip= "192.168.5.241";
    char* gateway = "192.168.5.1";
    char* netmask = "255.255.255.0";
    esp_netif_ip_info_t info_t;
    memset(&info_t, 0, sizeof(esp_netif_ip_info_t));

    if (eth_netif)
    {
        ESP_ERROR_CHECK(esp_netif_dhcpc_stop(eth_netif));
        info_t.ip.addr = esp_ip4addr_aton((const char *)ip);
        info_t.netmask.addr = esp_ip4addr_aton((const char *)netmask);
        info_t.gw.addr = esp_ip4addr_aton((const char *)gateway);
        esp_netif_set_ip_info(eth_netif, &info_t);
    }
    ...
}
```

4.2.5 Is there any impact on Ethernet functionality if replacing the module of ESP32-Ethernet-Kit with ESP32-WROOM-32D?

- The ESP32-WROVER-B of ESP32-Ethernet-Kit can be replaced with ESP32-WROOM-32D, and its Ethernet functionality will not be affected.
- The main difference between ESP32-WROOM and ESP32-WROVER module series: ESP32-WROVER modules have a 4 MB PSRAM while ESP32-WROOM modules do not have any PSRAM by default. Please refer to:
 - [ESP32-WROOM-32D Datasheet](#)
 - [ESP32-WROVER-B Datasheet](#)
- The ESP32-WROOM and ESP32-WROVER modules all use the ESP32 chip as their core, which supports Ethernet. For more information, please refer to [ESP32 Datasheet](#).

- Related document: [ESP32-Ethernet-Kit Getting Started Guide](#).
-

4.2.6 When using ESP32 to design a self-developed Ethernet board, after downloaded the official esp-idf/examples/ethernet example, errors are reported as follows, what is the reason?

```
E (5556) emac: Timed out waiting for PHY rdgister 0x2 to have value 0x0022_
↪(mask 0xffff). Current value 0xffff
E (6556) emac: Timed out waiting for PHY register 0x3 to have value 0x1430_
↪(mask 0xffff0). Current value 0xffff
```

- This error indicates something is wrong with your hardware circuit. The RMI clock is not working normally with the PHY, causing the PHY failed to read registers. For the more information about RMI clock, please refer to [Instructions](#).
-

4.2.7 When Ethernet and Wi-Fi co-exist, is Ethernet prioritized to transfer data?

:CHIP: ESP32 :

- Please call `esp_netif_get_route_prio` interface to check the priority of Ethernet and Wi-Fi first. If Wi-Fi has a higher priority than Ethernet, you can modify the priority via `route_prio` in the structure `esp_netif_t`.
-

4.2.8 Does ESP32 Ethernet support MII interface?

- Supported on hardware level, software adaptation is in development. Please refer to [Ethernet doc](#) for self implementation.
-

4.2.9 Is it possible to connect ESP32-S2 to Ethernet externally?

- Yes, ESP-IDF currently provides drivers for the DM9051 module, which has integrated Ethernet MAC and PHY functionality and can communicate with the MCU via the SPI interface. The DM9051 has an integrated MAC+PHY module, please refer to [example reference <https://github.com/espressif/esp-idf/tree/master/examples/ethernet/>](https://github.com/espressif/esp-idf/tree/master/examples/ethernet/) and API reference.
-

4.2.10 Do the ESP32 series chips support to use EMAC and SPI-Ethernet modules simultaneously?

- Yes. ESP32 series chips support to use EMAC and one or two SPI-Ethernet modules simultaneously. You can start PHY and SPI-Ethernet modules simultaneously in menuconfig to test by referring to the example [esp-idf/examples/ethernet/basic](#).

4.3 coexistence

□

4.3.1 When Wi-Fi coexists with Bluetooth, what mode does it support?

For supported coexistence scenarios, please refer to [coexistence documentation](#).

4.3.2 When Wi-Fi coexists with ESP-BLE-MESH, the Wi-Fi throughput is low, why?

For ESP32-DevKitC boards without PSRAM, Wi-Fi can coexist with ESP-BLE-MESH but with a relatively low throughput. For ESP32-DevKitC boards with PSRAM, the transmit rate can stabilize at over 1 Mbps.

To support PSRAM, the following configurations in menuconfig should be enabled accordingly:

- ESP32-specific --> Support for external, SPI-connected RAM --> Try to allocate memories of Wi-Fi and LWIP...
 - Bluetooth --> Bluedriod Enable --> BT/BLE will first malloc the memory from the PSRAM
 - Bluetooth --> Bluedriod Enable --> Use dynamic memory allocation in BT/BLE stack.
 - Bluetooth --> Bluetooth controller --> BLE full scan feature supported.
 - Wi-Fi --> Software controls Wi-Fi/Bluetooth coexistence --> Wi-Fi
-

4.3.3 Does ESP32 support coexistence between ESP-WIFI-MESH and Bluetooth® LE Mesh?

No.

However, the ESP32 supports coexistence between ESP-WIFI-MESH and Bluetooth LE, or Wi-Fi STA and Bluetooth LE Mesh.

4.3.4 Does ESP32 support coexistence between Bluetooth® and Wi-Fi?

Yes, but time-sharing control is required for ESP32's coexistence between Wi-Fi and Bluetooth. Please go to menuconfig to enable the Wi-Fi/Bluetooth coexistence, shown as follows:

```
menuconfig -> Component config -> Wi-Fi -> Software controls WiFi/  
Bluetooth coexistence (Enable)
```

4.3.5 When Wi-Fi, Bluetooth® LE, and A2DP sink coexist, audio data reception is lost and lagged while entering Bluetooth LE scanning. How to resolve such issue?

- Use RingBuf to cache audio data
 - Pause music and add a tone, such as “scanning for devices”.
-

4.3.6 Can BLE advertising (Connectable) and iBeacon sending (advertising) coexist?

IDF: release/v4.0 and later versions | CHIP ESP32:

- Not supported yet on hardware level, but can be realized on application layer by polling and sending broadcast packets at regular intervals.

IDF: release/v4.3 and later versions | CHIP: ESP32-C3|ESP32-S3

- Yes.

4.4 Peripherals

□

4.4.1 Analog-to-Digital Converter (ADC)

□

What is the resolution of ESP8266 ADC?

- The 10-bit ESP8266 ADC has a theoretical resolution of $2^{10} = 1024$.
 - After connected to a router, the ESP8266 will enter Modem-sleep mode from STA mode, causing the change of the reference value inside the chip. Therefore, the ADC could measure the data change.
 - If you expect an accurate result, please read the ADC value using function `system_adc_fast_read` after turning off Wi-Fi.
-

How to get the Bitmap information of the ADC register?

Since the ADC of ESP8266 is highly integrated with the internal RF circuit, the Bitmap and register information is not opened. Please contact sales@espressif.com if you have any special needs.

How many channels does ESP32 ADC have? What is the sampling rate and significant digit?

- The ESP32 ADC has 18 channels.
 - If you stop Wi-Fi and use ADC DMA, the sampling rate does not exceed 2 MHz theoretically. However, we recommend you to use a smaller sampling rate in practice.
 - Its sampling rate can reach 1000 times per second with Wi-Fi.
 - The internal significant digit of ADC is 12 bits.
-

When calling the API `adc_read_fast()` with ESP8266, will it cause a Wi-Fi disconnection?

- Please turn off Wi-Fi and interrupts first before calling `adc_read_fast()`. Please refer to the [Specification](#) of this API.
 - Since the API `adc_read_fast()` performs continuous acquisition and the ADC is partially coupled internally with Wi-Fi RF, so it is not possible to call this function with Wi-Fi turned on.
 - Please use `adc_read()` for ADC acquisition when Wi-Fi is on. To ensure data stability, you need to use function `esp_wifi_set_ps(WIFI_PS_NONE)` to turn off Wi-Fi Modem-sleep mode.
-

Note: ADC sampling rate: can reach 100000 times per second with Wi-Fi turned off, and 1000 times per second with Wi-Fi turned on.

If I float the ADC pin and print out VDD3P3 value (65535), then the voltage of VDD3P3 should be 65535/1024 63 V. Why this is not the correct voltage value?

The input of ADC should be in the range of 0 V to 3.3 V (the upper limit varies in different chips). The floating measurement is an undefined state.

What is the input resistance of ESP32 ADC?

ADC is capacitive and can be considered as a large resistance.

When using ESP32's ADC to detect the power supply voltage, is it necessary to divide the voltage?

The ADC reference voltage of ESP32 is 1100 mV, but the ADC measurable range can be increased by configuring the internal attenuation. For more information on the measurable range, please refer to [ADC Section](#) in the chip datasheet. If the measurable range cannot satisfy your requirement, please add an external voltage division circuit.

What is ESP32's highest sampling rate in ADC DMA mode?

ESP32 supports up to 2 MHz of sampling rate theoretically.

When an ESP32 calling `adc2_get_raw()` between `esp_wifi_start()` and `esp_wifi_stop()`, the read operation fails. What is the reason?

Since Wi-Fi also uses ADC2, and the Wi-Fi driver has a higher priority, the application may fail to read using `adc2_get_raw()` during the operation period of Wi-Fi. It is recommended to check the return value of this function and re-measure it after failure.

Does ESP32 support using ADC2 and Bluetooth simultaneously?

Yes.

What is the sampling rate range supported by the ADC DMA mode of the ESP32-S2 chip?

Frequency limit : 611 Hz ~ 83333 Hz.

Does the ADC of ESP32 support simultaneous sampling of multiple channels?

No, If you are using ADC to do multi-channel sampling, please implement it via ADC polling scheme.

When using the ESP32-WROVER-B module with release/v4.2 version of ESP-IDF, I set the GPIO as an ADC interface, and then set GPIO to other IO mode while with IO mode not effective without any hardware reset, this GPIO does not respond. How do I release the corresponding GPIO mode?

- Please do not set the ADC interface as input-only GPIO.
 - When disabling the ADC interface mode, please use `adc_digi_stop()` to disable the ADC.
-

What is the measurement error between the ADCs of the ESP32 chip?

By default, the measurement error between ESP32 ADCs is $\pm 6\%$, please refer to [ESP32 datasheet](#) for details.

Can ESP32 measure different data from two ADC channels at the same time, such as current and voltage?

It is not possible to read multiple ADC channels at the same time using one ADC, but you can poll the data of both ADC channels in turn.

When ESP32-S3 ADC is configured as ADC_ATTEN_DB_11, why does the measured voltage not reach 3100 mV?

When ESP32-S3 ADC1 or ADC2 is configured as ADC_ATTEN_DB_11, the voltage measurement should be in the range of 0 ~ 3100 mV. However, the maximum voltage measurement of some chips may be less than 3100 mV due to consistency issues. You may use the following two solutions to fix this issue:

- Solution 1: Try to avoid using the boundary voltage values. You can use a divider circuit to reduce the input voltage to an intermediate value for higher accuracy and consistency.
 - Solution 2: Use the software [ADC Range Extension Solution](#) to increase the maximum voltage measurement to 3300 mV.
-

Can we use GPIO0 as the ADC pin when using ESP32 as a Wi-Fi access point?

- The ESP32 ADC2 pins cannot be used when you are using Wi-Fi. So, if you are having trouble getting the value from an ADC2 GPIO while using Wi-Fi, you may consider using an ADC1 GPIO instead. For more details, please refer to [Hardware Limitations of ADC Continuous Mode](#) and [Hardware Limitations of ADC Oneshot Mode](#).
 - The GPIO0, GPIO2, GPIO5, GPIO12 (MTDI), and GPIO15 (MTDO) are strapping pins. When using GPIO0 for other functions, you need to pay attention to the GPIO level during power-up. If the GPIO0 level is low during power-up, the chip can enter the download mode. For more information, please refer to [ESP32 datasheet](#).
-

I tried to test the functionality of ADC2 using GPIO19 and GPIO20 of ESP32-S3 based on “[esp-idf/examples/peripherals/adc/oneshot_read](#)” example and set the attenuation parameter of ADC2 to 11 dB. When the input voltage is 0.6 V, why are the test results 1.1 V and 2.8 V?

- Please check whether both two ADC2 channels have been configured according to [adc_oneshot_config_channel\(\)](#).

4.4.2 Digital-to-Analog Converter (DAC)



When using DAC output for ESP32-S2-Saola-1, the power supply is 3.3 V. But the actual tested voltage is only 3.1 V. Why?

Due to the internal voltage drop, even when using 3.3 V power supply, the actual maximum output is only about 3.2 V.

4.4.3 GPIO & RTC GPIO



What should I pay attention to for ESP32 pin configurations?

The ESP32 has ESP32-WROOM and ESP32-WROVER series modules. Please pay attention to the following configurations with GPIOs.

The WROOM-32/32D/32U series have 26 pins available for customers. Please note:

- GPIO6 ~ GPIO11 are used by the internal flash and cannot be used elsewhere;
- GPIO34, 35, 36 and 39 are input-only pins and cannot be used for outputs;
- The ESP32 has a built-in GPIO Matrix, and some peripheral interfaces can be connected to any free pins. That is, for hardware designs, there is no need to strictly distribute some functions on certain pins.

For detailed information, please refer to Table 9 in [ESP32 Datasheet](#).

The WROVERWROVER-IWROVER-BWROVER-IB series have 24 pins available for customers. Please note:

- GPIO6 ~ GPIO11 are used by the internal flash and cannot be used elsewhere;
- GPIO34, 35, 36 and 39 are input-only pins and cannot be used for outputs;
- For WROVER series, it is not recommended to use GPIO12 for Touch Sensor functions since it has been pulled up in the module;
- The ESP32 has a built-in Matrix, and some peripheral interfaces can be connected to any free pins. That is, for hardware designs, there is no need to strictly distribute some functions on certain pins.

For detailed information, please refer to Table 9 in [ESP32 Datasheet](#).

There are three sets of UARTs in ESP32, but only UART0 can be used for downloading with fixed pins.

Some ESP8266 GPIOs are high level. What could be the reasons?

- According to the hardware design, some GPIOs are pulled up or down by default. Thus the level of these pins are not controlled by the program during system initialization, causing some incorrect levels of GPIOs during the boot process.
 - If you expect to use these GPIOs, it is recommended to keep the hardware peripherals be consistent with the default level status, or adjust level status in software during bootloader process. When using the later method, you may also encounter temporary level exception.
-

Can I disable the thread scheduling and use a single CPU for ESP32 to realize real-time control of GPIO?

- For now, we do not have any related configurations for SDK to support the single operation of CPU1. Both cores of ESP32 support SMP only, but not AMP.
 - The following solutions can be used to resolve the issue of output waveform being interrupted:
 - Use hardware signal outputs, and choose related digital protocols to realize SPI, I2C, I2S and etc. For special usage with SPI, you can generate waveform using signal output lines.
 - See if the hardware RMT can generate the desired waveform with enough length.
 - When the hardware interrupt generated corresponding waveform, all callbacks need to be put in IRAM.
 - Use the co-processor in the chip as a single chip without an operation system.
-

What is the turning speed of ESP32 GPIO levels?

It takes around 300 ns.

When certain RTC peripherals (SARADC1, SARADC2, AMP, HALL) are powered on, the inputs of GPIO36 and GPIO39 will be pulled down for approximately 80 ns. How to solve the issue?

For applications that require accurate timing and detecting digital input status, the above problems can be avoided by software.

- Ignore the inputs from GPIO36 and GPIO39 when turning on the power domain of the above sensors.
 - Debounce digital inputs through software. When reading the input states of GPIO36 and GPIO39, debouncing can be implemented through software by sampling and filtering the inputs for multiple times, thus reducing misjudgments caused by short voltage drops.
-

The ESP32 GPIO peripheral may not trigger interrupts correctly if multiple GPIO pads are configured with edge-triggered interrupts. How to resolve such issue?

- Please search for this question and its answer in [ESP32 Series SoC Errata](#).
-

Using ESP-WROOM-02D module, can GPIO0, GPIO15, GPIO1 and GPIO3 be used as normal GPIOs?

- Strapping pins (GPIO0 and GPIO15) and download pins (GPIO1 and GPIO3) can be used as normal GPIOs.
 - When using the strapping pin as a normal GPIO, you need to pay attention to the level of the strapping pin in the Flash download mode.
-

After configuring the GPIO19 for ESP32-C3 as pulled-down input, the level of this pin still stays high. However other pins in ESP32-C3 does not have this issue. What is the reason?

- In ESP32-C3, GPIO19 is a USB D+ pin, whose pull-up resistor is controlled by the pin's pull-up value together with USB's pull-up value. If any of the two pull-up values is 1, the pin's pull-up resistor will be enabled.
 - The USB pull-up value of GPIO19 is 1 by default, so when the pin is pulled down, GPIO19 still keeps high level.
 - This issue has been fixed in the GPIO driver in ESP-IDF v4.4.3 and later versions. For other versions, please write the register `USB_SERIAL_JTAG_DP_PULLUP` to 0 for configuration.
-

When using the release/v4.2 version of ESP-IDF, how to set a single GPIO as input/output mode simultaneously for ESP32?

You can set via the `esp_err_t gpio_set_direction(gpio_num_t gpio_num, gpio_mode_t mode)` API.

Is it possible to set the drive capability of the GPIO in ESP-IDF?

Yes. Please use API `gpio_set_drive_capability` to set the GPIO drive capability.

When ESP32 uses `gpio_install_isr_service()` to attach a new interrupt service routine on GPIO, why does it return `ESP_ERR_NOT_FOUND`?

Generally, this error means that ESP32 does not have enough available interrupt sources. In this case, there are multiple peripherals occupying the interrupt sources at the same time. You can try to reduce the interrupt sources used by other components to attach new GPIO interrupts.

How do I get the input level of the ESP32 RTC_GPIO?

- You can obtain the input level of RTC_GPIO by reading the macro of the register address corresponding to RTC GPIO. Please refer to “esp-idf/components/soc/esp32/include/socrtc_io_reg.h”.
- The related code is as follows :

```
uint8_t level = (uint8_t)((REG_GET_FIELD(RTC_GPIO_IN_REG, RTC_GPIO_IN_NEXT) &
↪ BIT(gpio_num)) ? 1 : 0);
```

4.4.4 Inter-Integrated Circuit (I2C)

□

Does ESP8266 support I2C slave mode?

No. If you want to use this function, it is recommended to choose ESP32 or ESP32-S2 chips instead. For ESP32 examples, please refer to `i2c_self_test`.

Is ESP8266 I2C realized via software programming?

Yes, ESP8266 I2C is realized via GPIO software programming.

When the I2C of the ESP32 series chip is operating (especially in fast mode), spikes often occur on the data lines, especially after the falling edge of the 8th/9th clock. Is this normal?

The spike on the data line at the 8th/9th clocks is caused by the I2C master-slave control handover. It is a normal phenomenon and is mentioned in the I2C protocol.

How can I realize data are received by ESP32 series chips, which are used as the I2C master, only after these data are processed by the slave? For example, when ESP32 chips read data through `i2c_master_read_to_device`, the slave should return data immediately after receiving the command. However, some slave devices wait for a while to return data after receiving the command.

This can be realized by dividing `i2c_master_read_device` into the following three steps:

1. Input commands and address: `i2c_cmd_link_create_static > i2c_master_start > i2c_master_write_byte > i2c_master_cmd_begin > i2c_cmd_link_delete_static`
2. Delay
3. Read data of the slave: `i2c_cmd_link_create_static > i2c_master_read > i2c_master_stop > i2c_master_cmd_begin > i2c_cmd_link_delete_static`

When using ESP32 series chip, can we configure GPIO32 and GPIO33 as I2C_SDA and I2C_SCL respectively?

Yes. I2C pins of the ESP32 chip can be remapped by any available GPIOs. Please refer to “4.2 Peripheral Pin Configurations” of [ESP32 Datasheet](#). If you do not need an external 32.768 KHz crystal, you can use GPIO32 and GPIO33 as I2C pins.

4.4.5 Inter-IC Sound (I2S)

□

Does ESP32 support using crystal oscillator as the clock source of I2S?

No. Please go to [ESP32 Technical Reference Manual](#) to read about clock source configurations of I2S.

When working as the I2S master, does ESP32 support connection to the I2S slave that only has the three signal lines, I2S_DATA, I2S_SCK, and I2S_WS?

Yes, but the connection to MCLK depends on the requirements of the codec chip on the other side.

Does the I2S interface of ESP32-C3 series chips support the PDM RX mode?

- In the software driver, the I2S interface of ESP32-C3 series chips does not support the PDM RX mode. Unlike ESP32-S3, the PDM RX of ESP32-C3 does not have a module supporting converting from PDM to PCM, which means the acquired data is in the RAW PDM format. The data in this format can't be used directly in most cases.

4.4.6 LCD

□

Where are the LCD drivers and reference examples for ESP32 series chips?

- You can find ESP's LCD driver in [components/esp_lcd](#) in **ESP-IDF**. However, this document is only available in ** release/v4.4 and newer** versions. **esp_lcd** can drive LCD screens with four interfaces (**I2C**, **SPI**, **8080**, and **RGB**) supported by ESP series chips. For the LCD interfaces supported by ESP32 series chips, see [ESP32 series chip screen interface](#).
- For the application examples of the LCD driver of each interface, please refer to [examples/peripherals/lcd](#) in ESP-IDF. Currently, these examples are only available in **release/v5.0** and newer versions. As the API names of **esp_lcd** in **release/v4.4** are basically same as those of higher versions, you can also refer to the above examples (please note the API implementations are a little different).
- **NOT RECOMMENDED** Use the LCD driver and examples in esp-iot-solution.

- ESP-IDF **release/v5.1** is recommended for RGB LCD applications as some features are not supported in release/v4.4.
-

Which adapted ICs can be used by the LCD screen of ESP32 series chips?

Currently, the adapted ICs for the *esp_lcd*-based LCD driver include:

- *esp_lcd* : st7789, nt35510, ssd1306
- **Package Manager** : gc9a01, ili9341, ili9488, ra8875, sh1107, st7796 (continuously updated)

Please note that even if driver ICs are same, different screens vary in register configuration parameters. In addition, screen manufacturers generally provide matched configuration parameters (code). Thus, it is recommended to use the above two methods to obtain code of similar driver ICs, and to update the code based on the parameters of your own screen.

Currently, the adapted ICs of *esp_lcd_touch* based touch driver include:

- **Package Manager**: FT5x06, GT1151, GT911, STMPE610, TT21100, XPT2046, CST816 (continuously updated).
-

How can I improve the display frame rate of LCD screens?

- The actual display frame rate of LCD screens is determined by the “interface frame rate” and “rendering frame rate”. Generally, the “interface frame rate” is much bigger than the “rendering frame rate”. So this question actually is “how can I improve the rendering frame rate of the LCD”.
 - The following ESP configuration items can improve the frame rate (ESP-IDF release/v5.0):
 - CONFIG_FREERTOS_HZ=1000
 - CONFIG_ESP_DEFAULT_CPU_FREQ_MHZ_240=y
 - CONFIG_ESPTOOLPY_FLASHMODE_QIO=y
 - CONFIG_ESPTOOLPY_FLASHFREQ_120M=y [needs to be consistent with PSRAM]
 - CONFIG_SPIRAM_MODE_OCT=y
 - CONFIG_SPIRAM_SPEED_120M=y [Need to be consistent with FLASH]
 - CONFIG_SPIRAM_FETCH_INSTRUCTIONS=y
 - CONFIG_SPIRAM_RODATA=y
 - CONFIG_ESP32S3_DATA_CACHE_LINE_64B=y
 - CONFIG_COMPILER_OPTIMIZATION_PERF=y
 - The following LVGL configuration items can improve the frame rate (LVGL v8.3):
 - #define LV_MEM_CUSTOM 1
 - #define LV_MEMCPY_MEMSET_STD 1
 - #define LV_ATTRIBUTE_FAST_MEM IRAM_ATTR
-

Is there any example code for I2S driving LCD with ESP32?

- The interface type of the screen driven by I2S in ESP32/ESP32-S2 is i80(8080)
 - For the application examples, please refer to [LCD examples](#).
-

What is the maximum resolution supported by ESP32 LCD? What is the corresponding frame rate?

- There is no “maximum” limit on how much resolution can be supported. Due to the limited data transmission bandwidth of the interface, the interface frame reduces as the LCD resolution increases. Thus, you need confirm the LCD resolution based on this.
 - Over the RGB interface, the maximum resolution of ESP32 LCD is 800×480 ; the maximum interface frame rate is 59 (PCLK 30 MHz); and the average frame rate of LVGL is about 23. The average upper limit of the frame rate of LVGL is 26, and the corresponding interface frame rate is 41 (PCLK 21 MHz).
-

How to enable PSRAM 120M Octal(DDR) on ESP32R8?

- ESP-IDF v5.1 or later versions are required.
 - For details, please refer to [SPI Flash and External SPI RAM Configuration](#).
 - Note: This feature is an experimental and has the following temperature-related risks:
 - The chip may not work properly even with ECC enabled when the temperature is above 65°C.
 - Temperature changes may also cause program crashes when accessing PSRAM/flash. For more details, please refer to [SPI Flash and External SPI RAM Configuration](#).
-

What models of display touch panels are supported for testing the LVGL example on ESP32-S3?

The driver and examples in esp-iot-solution are not recommended. For details, please refer to [Where are the LCD drivers and reference examples for ESP32 series chips?](#).

Does ESP32-S3 require an external PSRAM to use the RGB screen?

Yes, and it must be an Octal PSRAM at least and the clock must be set to 80 MHz or above. Otherwise, the PCLK of RGB LCD cannot be set to a higher PCLK frequency and the frame rate will be too low.

Which image decoding formats are supported by the ESP32-S3 series of chips?

- Currently, ESP-IDF only supports the JPEG decoding format. For an application example, please refer to [esp-idf/examples/peripherals/lcd/tjpgd](#).
- If you develop based on LVGL, PNG, BMP, SJPG and GIF decoding formats are supported. For details, please refer to [LVGL libs](#).

Why do I get drift (overall drift of the display) when driving an RGB LCD screen?

• Reasons

- PCLK is set to a too big number, and the PSRAM bandwidth is not applicable.
- PSRAM is disabled due to the write operation of flash.

• Solutions

- Improve bandwidths of PSRAM and flash. You can set flash to QIO 120 M and set PSRAM to Octal 120 M.
- Enable `CONFIG_COMPILER_OPTIMIZATION_PERF`.
- Reduce `data_cache_line_size` to 32 bytes.
- Enable `CONFIG_SPIRAM_FETCH_INSTRUCTIONS` and `CONFIG_SPIRAM_RODATA`.
- Enable `CONFIG_LCD_RGB_RESTART_IN_VSYNC`. But this operation may cause the screen to flash blurred and drop the frame rate, so we generally do not recommend this way. However, you can try it if you have interests.

• Applications

- If you need to use Wi-Fi and continuous write operation to flash, please use *XIP PSRAM + RGB Bounce buffer* method, and the settings are as follows:
 - * Make sure the ESP-IDF version is (> 2022.12.12) release/v5.0 and above (released after 2022.12.12), as older versions do not support the *XIP PSRAM* function.
 - * Verify that whether `SPIRAM_FETCH_INSTRUCTIONS` and `SPIRAM_RODATA` can be enabled in the PSRAM configuration (too large rodata segment will cause insufficient space in the PSRAM).
 - * Check if there is any memory (SRAM) left, and it takes about $[10 * \text{screen_width} * 4]$ bytes.
 - * Set *Data cache line size* to 64 Bytes (you can set *Data cache size* to 32 KB to save memory).
 - * If all the above conditions are met, then you can refer to [Documentation](#) to modify the RGB driver to *Bounce buffer* mode.
 - * If you still have the drift problem when dealing with Wi-Fi, you can try to turn off `CONFIG_SPIRAM_TRY_ALLOCATE_WIFI_LWIP` in PSRAM, which takes up much SRAM space.
 - * The effects of this setting include higher CPU usage, possible interrupt watchdog reset, and higher memory overhead.
- For the drift caused by short-term operations of flash, such as before and after Wi-Fi connection, you can call `esp_lcd_rgb_panel_set_pclk()` before the operation to reduce the PCLK (such as 6 MHz) and delay about 20 ms (the time for RGB to complete one frame), and then increase PCLK to the original level after the operation. This operation may cause the screen to flash blank in a short-term.

- Enable *flags.refresh_on_demand* in *esp_lcd_rgb_panel_config_t*, and manually refresh the screen by calling the *esp_lcd_rgb_panel_refresh()* interface. In addition, you need to reduce the refreshing frequency as much as possible while ensuring that the screen does not flash blank.
 - If unavoidable, you can call the *esp_lcd_rgb_panel_restart()* interface to reset the RGB timing to prevent permanent drift.
-

Why is there vertical dislocation when I drive SPI/8080 LCD screen to display LVGL?

If you use DMA interrupt to transfer data, *lv_disp_flush_ready* of LVGL should be called after DMA transfer instead of immediately after calling *draw_bitmap*.

When I use ESP32-C3 to drive the LCD display through the SPI interface, is it possible to use RTC_CLK as the SPI clock, so that the LCD display can normally display static pictures in Deep-sleep mode?

- Deep-sleep mode: CPU and most peripherals are powered down, and only the RTC memory is active. For details, please refer to “Low Power Management” in [ESP32-C3 Datasheet](#).
 - The SPI of ESP32-C3 only supports two clock sources, APB_CLK and XTAL_CLK, and does not support RTC_CLK. Therefore, the LCD screen cannot display static pictures in Deep-sleep mode. For details, please refer to *ESP32-C3 Technical Reference Manual > Reset and Clock* [PDF].
 - For the LCD screen driven by the SPI interface, the driver IC generally has built-in GRAM. Thus, the static pictures can be displayed normally without the ESP continuously outputting the SPI clock, but the pictures cannot be updated during this period.
-

Are 9-bit bus and 18-bit color depth supported if I use the ILI9488 LCD screen to test the screen example?

The ILI9488 driver chip can support 9-bit bus and 18-bit color depth. However, Espressif’s driver can only support 8-bit bus and 16-bit color depth for now. You can modify the driver according to the ILI9488 datasheet to support 9-bit bus and 18-bit color depth.

4.4.7 LED Control (LEDC)

□

What is the frequency range for ESP8266 PWM?

The PWM of ESP8266 is realized via software programming, so the maximum CLK value is 1 M limited by timer. It is recommended to set the frequency to 1 K. The PWM frequency can also be improved by decreasing the resolution of duty cycle.

Are there any limits on outputting PWM via ESP32 GPIO pins? Can I distribute it to any I/O?

- The ESP32 can output PWM using any GPIO via IO Matrix. Theoretically, the PWM can be distributed to any I/O except for those that only have input functions (e.g., GPIO34 ~ GPIO39).
 - In the actual use, this could also be affected by the limitations of chips and modules, the un-pinned I/Os, flash occupations and etc.
-

The PWM of ESP8266 NonOS SDK changes slow. What could be the reasons?

- If you are using the gradient APIs in SDK example/IOT_demo, e.g., `light_set_aim` or `light_set_aim_r`, it will need a gradual process for PWM changes.
 - If you need the PWM Duty to take effect immediately after configuration, please call API `pwm_set_duty`, and call `pwm_start` next to make this configuration take effect.
-

When the LEDC is in decremental fade mode, a duty overflow error can occur. How to solve the issue?

When using LEDC, avoid the concurrence of following three cases:

- The LEDC is in decremental fade mode;
 - The scale register is set to 1;
 - The duty is $2^{\text{LEDC_HSTIMERx_DUTY_RES}}$ or $2^{\text{LEDC_LSTIMERx_DUTY_RES}}$.
-

When using ESP8266 to generate PWM by directly writing to the register of the hardware timer FRC1, I found there are error PWM outputs after Wi-Fi is initialized since it may disturb the interrupt of FRC1. Is it possible to use FRC2 instead to generate PWM? Or is it possible to set FRC1 higher priority than Wi-Fi?

FRC2 cannot be used as it is occupied by the system. Wi-Fi uses NMI interrupt, which have a higher priority than other ordinary interrupts. It is recommended to use the PWM library of ESP8266_RTOS_SDK. Please refer to [ESP8266_RTOS_SDK/examples/peripherals/pwm](#) example.

I'm using v3.3.3 version of ESP-IDF to test the ledc example on ESP32. The LED PWM outputs when Auto Light Sleep mode is disabled, but does not output when this mode is enabled. According the description of LED PWM in ESP-IDF programming guide, LED PWM should work in sleep modes. What is the reason?

v3.3.3 does not support LED PWM working in sleep modes. Please use the LEDC example under the new versions of ESP-IDF (v4.0 and later versions) to test, e.g., ESP-IDF release/v4.2 version of the SDK. Plus, it is also necessary to change the LED PWM clock source to the internal RTC_8M clock source. Please see below:

```
ledc_timer_config_t ledc_timer = {  
    .duty_resolution = LEDC_TIMER_13_BIT,  
    .freq_hz = 5000,  
    .speed_mode = LEDC_LOW_SPEED_MODE,  
    .timer_num = LEDC_TIMER_0,  
    .clk_cfg = LEDC_USE_RTC8M_CLK,  
};
```

Does ESP32 PWM support complementary outputs with dead bands on two channels?

- This feature is not supported by LEDC but by the MCPWM peripheral.
- By measurement, ESP32-S3 can generate complementary output waveforms with the frequency of 10 k, the duty cycle accuracy of 1 us and the dead band accuracy of 100 ns by MCPWM.

4.4.8 Motor Control Pulse Width Modulator (MCPWM)

□

Does ESP32 support using the MCPWM Timer to trigger AD sampling?

No, it does not.

Can ESP32-S3 generate fully complementary PWM with accurate clock and duty cycle and adjustable dead band?

By measurement, ESP32-S3 can generate complementary output waveforms with the frequency of 10 k, the duty cycle accuracy of 1 us and the dead band accuracy of 100 ns by MCPWM.

4.4.9 Pulse Counter (PCNT)

□

Does ESP8266 support pulse counting?

- The ESP8266 does not include a hardware pulse counting module, thus only supports counting via the interrupt of GPIO rising edge or falling edge.
 - When Wi-Fi is turned on in ESP8266, it may cause a vacuum in the GPIO sampling due to its high priority, thus interrupting the collected counts and causing data loss.
 - In conclusion, it is recommended to use ESP32 and subsequent chips for scenarios with high counting demands.
-

Can ESP32-S3 realize low-level pulse counting with a frequency of 200 k?

Yes.

4.4.10 Remote Control Transceiver (RMT)

□

What can the RMT peripheral on ESP chips be used for in practical?

- For applications of RMT, please refer to [RMT application examples](#). It can be used for infrared remote control, LED strip lighting, D-shot motor control, and so on.
-

Which ESP chip is recommended for utilizing the RMT functionality?

- ESP32-S3 is recommended because it is currently the only chip with RMT DMA support. This ensures RMT is not interfered by interrupts caused by Wi-Fi, Bluetooth, and other peripherals.
-

How to change the clock to REF_TICK in RMT?

CHIP: ESP32 | ESP32-S2 | ESP32-C3

By using the `rmt_set_source_clk` interface.

When ESP32 RMT controls the WS2812 light strip with Wi-Fi or Bluetooth enabled, there are some data frame exceptions. How to solve the issue?

- This problem is difficult to solve on non-ESP32-S3 chips, because the RMT lighting up LED (especially when many LEDs) rely heavily on interrupts, and does not support DMA, thus requiring software to switch ping-pong buffer in the interrupt. If the interrupt does not respond in time, there will be a problem. By default (one memory block is set), after lighting up two LEDs, the RMT driver will go into the interrupt to switch the internal ping-pong buffer.
 - Workaround:
 - For esp-idf release/v4.4 and earlier versions, increase `mem_block_num`. There is a change in release/v5.0. Please see [Breaking Changes in Usage](#)
 - Install the RMT interrupts on a specific CPU core by calling the driver install function in a pin-to-core task, thus avoiding the core used by Wi-Fi or Bluetooth.
 - You can also use SPI DMA as an alternative to RMT to solve this issue. For more details, please refer to the [SPI DMA LED strip example](#).
 - If you are in the product selection stage, it is recommended to use ESP32-S3 RMT.
-

How can I quickly adapt other infrared protocols based on the IR NEC example in ESP-IDF

You can utilize [RMT Encoder](#) based on the [IR NEC example](#) to expedite the adaptation of other infrared protocols.

ESP32-S3 RMT supports configuring 4 RMT RX/TX channels. Why does it fail when creating more than 2 RMT TX channels in a row using `rmt_new_tx_channel`?

- This is because the `mem_block_symbols` parameter configured in the `tx_chan_config` structure is too large. On ESP32-S3, the size of each dedicated memory block for RMT is 48 bytes. If the `mem_block_symbols` parameter exceeds 48, creating a TX channel will actually occupy the memory block of the next adjacent channel as well. Therefore, if you want to create and use 4 RMT RX/TX channels simultaneously, the `mem_block_symbols` parameter should not exceed 48.
 - Please note that the size of each dedicated memory block for RMT on ESP32 is 64 bytes.
-

Can ESP32-S3 RMT achieve synchronized output for multiple TX channels?

- Yes, please refer to the following code snippet:

```
rmt_channel_handle_t tx_channels[TEST_RMT_CHANS];
rmt_sync_manager_handle_t synchro = NULL;
rmt_sync_manager_config_t synchro_config = {
    .tx_channel_array = tx_channels,
    .array_size = TEST_RMT_CHANS,
};
rmt_new_sync_manager(&synchro_config, &synchro);
```

(continues on next page)

(continued from previous page)

```

for (int i = 0; i < TEST_RMT_CHANS; i++) {
    rmt_transmit(tx_channels[i], led_strip_encoders[i], leds_grb, TEST_LED_NUM_
↪ * 3, &transmit_config);
}

```

How can I achieve cyclic data transmission using the RMT TX channel on ESP32-S3, such as an infinite loop?

- You can realize infinite loop transmission by setting the *rmt_transmit_config_t::loop_count* to -1. For more details, please refer to [Initiate TX Transaction](#)

Does the ESP32-S3 support One-Wire?

- ESP32-S3 supports [One-Wire bus protocol](#) with the RMT peripheral. For specific applications, please refer to “[esp-idf/examples/peripherals/rmt/ onewire_ds18b20](#)”.

4.4.11 Secure Digital Input Output (SDIO)

□

What is the maximum speed supported by the SDIO interface?

- The maximum clock speed supported by ESP32 SDIO is 50 MHz, and ESP32 SDIO supports the Quad mode at the maximum.
- The maximum clock speed supported by ESP32-S3 SDIO is 80 MHz, and ESP32-S3 SDIO supports the Octal mode at the maximum.
- The practical speed is influenced by the read and write speed of storage media at the same time.

Does the hardware SDIO interface support SD cards?

Please note that the SDIO hardware only supports the device or slave profile, i.e. it cannot act as a host to control SDIO devices such as SD cards.

What is the maximum capacity for ESP32 SD card?

- In the SD3.01 Specifications, the SDXC card supports a maximum capacity of 2 TB (2048 GB).
 - The ESP32 SDMMC Host also complies with the SD3.01 Specifications, which means up to 2 TB areas of it can be accessed by peripherals. When accessing the card via SPI bus using the SDSPI driver, there are also 2 TB of areas can be accessed in hardware level.
 - In software level, the usable area of the card is also affected by the file system.
-

Is it possible to use ESP32 SD card together with flash & PSRAM?

- Yes, they can be used simultaneously when they apply different pins.
 - Note that when ESP32 uses the SDMMC host driver, the SDMMC Slot0 pins in the ESP32-WROOM and ESP32-WROVER modules conflict with the flash.
-

Does ESP-WROOM-S2 module support using SDIO as a slave?

Yes, it does.

Since ESP32-S2 has removed the SDIO interface, does it still support external TF card?

You can use the interface of SPI2/SPI3 to connect an external TF card. When doing so, please use the SPI mode of the TF card.

Does ESP32-S2 support eMMC?

CHIP: ESP32-S2

No.

Does ESP32-S2 support SDIO as a slave?

ESP32-S2 has no SDIO interface and does not support SDIO as a slave.

How does ESP32 enable and disable the interrupt for the SDIO slave to receive data?

The data reception of the SDIO slave is related to the state of the mounted buffer. After the data is received, you need to call `sdio_slave_recv_load_buf` to release the buffer. Otherwise, the SDIO host will not be able to continue to send data to the SDIO slave.

4.4.12 Serial Peripheral Interface (SPI)

□

Is ESP-WROOM-02D module able to connect SPI flash?

The ESP-WROOM-02D module is a Wi-Fi module based on the ESP8266 chip, which supports communication with external SPI flash devices using SPI interfaces. Specifically, the ESP-WROOM-02D module provides 4 SPI interface pins, GPIO12, GPIO13, GPIO14, and GPIO15. Among these pins, GPIO12~GPIO14 can be used as the MISO, MOSI, and SCLK pins for the SPI master interface, and GPIO15 can be used as the CS pin for the SPI slave interface.

To connect an external SPI flash device to the ESP-WROOM-02D module, the MOSI, MISO, SCK, and CS pins of the SPI flash device should be connected to the GPIO12~GPIO14 and GPIO15 pins of the ESP-WROOM-02D module. Additionally, the SPI interface needs to be properly configured and initialized in the firmware to ensure correct communication of ESP8266 with the external SPI flash device.

It should be noted that the model and capacity of the external SPI flash device should be selected based on the specific application requirements. In addition, timing characteristics and reliability of the SPI flash device should be considered to ensure data can be transmitted correctly and stably. Moreover, factors such as environment noise and physical distance between the SPI flash device and the ESP-WROOM-02D module should be considered to improve the reliability and performance of the system as much as possible.

Taking ESP-WROOM-S2 as the slave device and STM32 as MCU, is it possible to download through SPI interface?

No, we use UART0 to download by default. You can also design OTA support yourself in firmware.

What is the difference among SPI0, SPI1, HSPI and VSPI in ESP32?

- ESP32 has 4 SPIs, SPI0 and SPI1 are two peripherals, also known as MSPI. SPI0 and SPI1 share the same SPI bus (same signals and IOs). The difference is, MSPI CS0 is connected to the main flash for firmware storage, and MSPI CS1 is connected to PSRAM. SPI2 and SPI3 are general-purpose SPIs that are available for customers to use.
 - HSPI represents the above-mentioned SPI2, and VSPI represents the SPI3. The two sets of SPIs are general-purpose SPIs and support QSPI.
-

The maximum data transmission of ESP32 SPI DMA is 4095 bytes. Is it because of hardware limitation?

- Yes, this is a hardware limitation.
 - A single node in the DMA table can only mount 4095 bytes of data, but it is possible to send more data through several nodes.
 - The maximum number of bytes that the SPI can send through the DMA table is also limited by the hardware register `SPI_LL_DATA_MAX_BIT_LEN` (the value varies by chip family and can be obtained in the ESP-IDF), i.e. `max_transfer_sz <= (SPI_LL_DATA_MAX_BIT_LEN >> 3)`.
-

The SPI of ESP32-S2 accesses three SPI Slave devices at the same time, do I need to synchronize the semaphore to access it?

- The same SPI peripheral, as the master, can only communicate with one slave at a time, and CS decides which slave to communicate with. If you connect 3 slave devices to the SPI driver and communicate with them separately, it is okay and recommended.
 - It is recommended to share one SPI device in one task. Otherwise, the threads are not safe, and they communicate through semaphore synchronization. For details, please refer to [SPI Master driver-feature](#).
-

When using an ESP32 board for development and testing based on ESP-IDF release/v4.3, I received the following error log during compilation. What is the reason?

```
spi_flash:Detected size(8192K) smaller than the size in the binary image_
↪header(16384K).Probe failed.
```

The reason is that the configured flash size is larger than the actual flash size. In order to avoid misuse of a larger address space, the actual flash size is checked.

What is the maximum transmission speed supported by SPI slave?

CHIP: ESP32

ESP32 can support up to 10 M of transmission speed when serves as an SPI slave.

When using ESP32 as an SPI Master device, how many bytes of data can be transfered at one time in non-DMA mode?

- Up to 64 Bytes of data can be transferred at one time in such condition.
 - When the transmitted data does not exceed 32 bits, you can use the 4-byte array in the SPI Master driver as the buffer for data transmission. For details, please refer to [Transactions with Data Not Exceeding 32 Bits](#).
 - But when the transmitted data exceeds 32 bits, you need to set the buffer for SPI data transmission by yourself. For details, please refer to [SPI Master Transactions](#).
-

- When using ESP32 as an SPI Master device to transmit more than 32 bits of SPI data in non-DMA mode, please refer to the example [lcd](#).

When using the ESP32-S3-WROOM-1 (ESP32-S3R2) module to enable its PSRAM configuration based on the “hello-world” example in ESP-IDF v4.4, the following error is printed. What is the reason?

```
E (232) spiram: Virtual address not enough for PSRAM!
```

- ESP32-S3R2 chip integrates a 4-wire 2 MB PSRAM, please set PSRAM Mode to **Quad** mode in menuconfig before your action as follows:

```
menuconfig > Component config > ESP32S3 Specific > Support for
external, SPI connected RAM > SPI RAM config > Mode (QUAD/OCT) of
SPI RAM chip in use (Quad Mode PSRAM)
```

When using the ESP32-S3-WROOM-2 (ESP32-S3R8V) module to enable the PSRAM configuration based on the “hello-world” example in ESP-IDF v4.4, the following error is printed. What is the reason?

```
E (453) psrm: psrm ID read error: 0x00ffff
E (454) cpu start: Failed to init external RAM!
```

ESP32-S3R8V chip integrates a 8-wire 8 MB PSRAM, please set PSRAM mode to **Octal** mode in menuconfig before your action as follows:

```
menuconfig > Component config > ESP32S3 Specific > Support for
external, SPI connected RAM > SPI RAM config > Mode (QUAD/OCT) of
SPI RAM chip in use (Octal Mode PSRAM)
```

Does ESP8266 RTOS SDK support full duplex for SPI?

CHIP: ESP8266

No, it doesn't. Because ESP8266 doesn't support DMA, in order to improve the transmission performance, the entire FIFO is used. So it can only be half duplex. Please refer to [spi readme](#) for more details.

Can ESP32 support 9-bit clock mode for 3-wire SPI (i.e. a mode where the first bit indicates whether the next 8 bits are command or data)?

No. Currently, all ESP32 series chips does not support non-byte-aligned data transfer, i.e., only support 8-bit-aligned data transfer. For details, please refer to [Github issue](#).

Newer versions of ESP32 series chips may support non-byte-aligned data transfer. However, there is no specific schedule for this functionality.

After routing the SDA signal of the SPI screen to GPIO35 of ESP32-S2, I expect that the SDA signal is low when idle and high when writing data. But why does this pin turn out to be high when idle and low when writing data on power-up? How to achieve my expected result?

Please modify the `mode` member variable in the `spi_device_interface_config_t` structure.

4.4.13 Timer

□

What should I pay attention to when using the HW timer interrupt with ESP8266?

- Please refer to [ESP8266 Technical Reference Manual](#) regarding the related APIs.
 - If you are using NonOS SDK, please refer to [ESP8266 Non-OS SDK API Reference](#).
 - Generally, when using hardware interrupts, you should finish executions as soon as possible and put the callback function into IRAM to avoid the potential impacts of Cache.
 - For RTOS SDK, `IRAM_ATTR` should be added to the function.
 - For NonOS SDK, `ICACHE_FLASH_ATTR` should not be added before the function.
-

How to set interrupt priority for timers?

- Only with the chip of ESP32, `esp_timer` can set the interrupt priority by modifying `CONFIG_ESP_TIMER_INTERRUPT_LEVEL`.
- General Purpose Timer can set the interrupt priority by modifying the last parameter of the interface `timer_isr_callback_add`.

4.4.14 Touch Sensor

□

When using ESP32 to develop Touch Sensor applications, where can I find references?

Please refer to [Software and Hardware Designs](#).

When there is water on ESP32-S2 Touch Sensor, does it block or recognize the Touch event with its waterproof function?

When there is a small amount of water droplets, the waterproof function of the ESP32-S2 Touch Sensor can work normally. In the event of a large area of standing water (i.e. the Touch contact area is completely covered), the Touch will temporarily lock and will not resume operation until the water is cleared.

While the waterproof feature of ESP32-S2 Touch Sensor shielding the Touchpad with water flow, does other pads with no water still usable?

Yes, the specific shielding channel can be selected via software.

Are there any recommendations for materials that can be used to test Touch Sensor, can trigger Touch Sensor stably and is close to the parameters of human touches?

For experiments with high consistency requirements, it is doable to replace human hands with cell phone pencils.

Can the pins of Touch Sensor be remapped?

No, because Touch Sensor is realized via software programming.

Do I need to reset a check threshold for Touch Sensor after covering it with a acrylic plate?

Yes.

Is it possible for Touch Sensor to detect whether there is a acrylic plate on the top, so that it can switch to the pre-defined threshold value automatically when there is a acrylic plate added or removed?

For now, it cannot adapt to the impacts brought by physical changes.

What reference drivers does the ESP32 touch screen have?

- Code: please refer to [touch_panel_code](#).
- Documentation: please refer to [touch_panel_doc](#).

4.4.15 Two-Wire Automotive Interface (TWAI)

□

What are the considerations when using the ESP32 TWAI® controller?

Please refer to the [ESP32 Series SoC Errata](#) > Section *ESP32 TWAI Errata*.

4.4.16 Universal Asynchronous Receiver/Transmitter (UART)

□

When using ESP8266 RTOS SDK v2.1 and previous versions, how to set log to UART1?

After initializing UART1, you can switch log to UART1 via API:

```
UART_SetPrintPort (UART1);
```

When using ESP8266 RTOS SDK v3.0 and later versions, how to set log to UART1?

Go to menuconfig -> Component config -> ESP8266-specific -> UART for console output -> custom -> UART peripheral to use for console output -> UART0 and change the option to “UART1”.

How to enable UART Flow Control in ESP32 IDF?

- Hardware enable: [uart-flow-control](#).
 - Software enable: [software-flow-control](#).
-

When using UART0 as a serial communication port for ESP32, what should I pay attention to?

- Generally, it is not recommended to use UART0 as a normal serial communication port, because it is the default LOG output port.
- If the UART number in ESP32 is not enough for you or it is not convenient to change your hardware designs anymore, and UART0 is therefore going to be used as a normal communication port, please pay attention to the following suggestions:

Software: You need to protect the serial communication port from being affected by printing. The UART0 mainly has three print settings in the default program:

- First, power-on ROM print. You can set the MTDO pin as low level when powered on to block the power-on ROM print.
- Second, bootloader log output. You can set `menuconfig -> Bootloader config -> Bootloader log verbosity` as Not output to block bootloader log output.
- Third, app log output. You can set `menuconfig -> Component config -> Log output -> Default log verbosity` as Not output to block app log output.

Hardware:

- Pay attention to other devices on UART0 when downloading programs since they could affect downloading. It is recommended to reserve a 0 resistance between ESP32 and other devices so that if there is something wrong while downloading, you can still disconnect this resistance.
-

Is it possible to use GPIO34 GPIO39 from ESP32-SOLO-1 as the RX signal pin for UART and TWAI®?

Yes, GPIO34 GPIO39 are for receive only and can be used as the RX signal pins for UART and TWAI®.

How to dynamically change the serial baud rate and make it take effect immediately with ESP32?

Please use the API `uart_set_baudrate()` to change the baud rate of UART. Please see [API Reference](#).

Does the ESP32 chip support USRAT (Universal Synchronous Asynchronous Receiver Transmitter)?

It's not support. ESP32 only supports UART and cannot provide the synchronous clock.

Does the serial port verification of the ESP32 chip support MARK and SPACE verification?

No.

What is the size of the hardware FIFO in ESP8266's serial port?

Both UART0 and UART1 of ESP8266 have a 128-byte hardware FIFO and a 128-byte RW FIFO, which operate at the same address. Please refer to Section 11.2. Hardware Resources in [ESP8266 Technical Reference Manual](#).

What is the serial port baud rate range of ESP8266?

300 ~ 115200*40 bps. Please refer to Section 11.3.1. Baud Rate in [ESP8266 Technical Reference Manual](#).

How to modify the output port of UART0?

CHIP: ESP32 | ESP32 | ESP32-C3

This can be set in menuconfig: `idf.py menuconfig` → Component config → Common ESP-related → Channel for console output (custom UART).

When using ESP8266, I want to use UART0 exclusively for downloading, and then use UART1 to communicate with other chips. Can GPIO4 and GPIO5 be configured as UART1 serial ports?

- Since the RXD of UART1 is occupied, UART1 cannot be used to communicate with other chips, but the TXD pin of UART1 can be used to output logs.
 - ESP8266 can only communicate with other chips by swapping CTS and RTS pins of UART0. It will be invalid to configure GPIO4 and GPIO5.
 - ESP8266 can communicate with other chips by calling “`uart_enable_swap()`” to swap the CTS and RTS pins of UART0 to MTCK (IO13) and MTDO (IO15). After this, ESP8266 can communicate with other chips via GPIO13 (TXD) and GPIO15 (RXD).
-

Can ESP32's UART0 be used for inputting from the computer console while it is being used for outputting logs?

- Yes. Outputting logs only requires using the TXD0 pin, while receiving input from the computer console only requires using the RXD0 pin. You can use the “[esp-idf/examples/system/console/basic](#)” example for testing.

4.4.17 USB

□

Does ESP32 support USB function?

- No, ESP32 does not support USB function.
 - However, ESP32-S2/S3 supports USB2.0 Full-speed mode.
-

Does the ESP-IDF SDK USB interface support HID and MSC modes?

- ESP32S2/S3 can be used as MSC Host to support reading from or writing to storage devices such as USB flash disks. For details, please refer to [esp-idf](#).
 - ESP32S2/S3 can be used as MSC Device to simulate storage of USB flash disks. For details, please refer to [esp-iot-solution](#).
 - ESP32S2/S3 can be used as HID Host. For details, please refer to [ESP-IDF Host HID](#).
 - ESP32S2/S3 can be used as HID Device. For details, please refer to [ESP-IDF Device HID](#).
-

What is the stable current output for ESP32-S2's USB interface?

The current output capability of the VBUS power line is determined by the power supply, not by the ESP32-S2 chip. If the chip is self-powered, please refer to [Self-Powered Device](#).

Does ESP32-S3's USB peripheral supports USB Host?

Yes, regarding this function, ESP32-S3 is the same as ESP32-S2.

Does ESP32-C3 USB support USB serial port function and USB JTAG function?

Yes, but you cannot define the descriptor by yourself.

What are the USB features of ESP32-S2 and ESP32-S3?

ESP32-S3 and ESP32-S2 support USB 2.0 OTG (supporting full-speed mode), and both support Host and Device functions. On top of that, ESP32-S3 also supports USB-Serial-JTAG peripheral, which can be used to download and debug firmware.

Are there any references to the library and demo of ESP32-S2 USB Host?

Please refer to [USB Host](#) in ESP-IDF.

The USB protocol supported by ESP32-S2 is OTG 1.1, with the maximum speed of 12 Mbps. Can it communicate with USB 2.0 devices?

In the full speed mode, USB 2.0 devices are compatible with USB 1.1 devices, so they can communicate with each other.

Does ESP32-S2 support USB camera?

Yes. For the demo code of ESP32-S2/ESP32-S3 USB Host UVC, please refer to [usb_stream](#).

Does ESP32-S3 support USB cameras with microphones and speakers?

Yes. For the demo code of ESP32-S2/ESP32-S3 USB Host UVC, please refer to [usb_stream](#).

Is there any reference for the example of using ESP32S2 as a USB flash drive (MSC DEVICE)?

Please refer to [usb_msc_wireless_disk](#) demo. The average read and write speed currently tested is: read 540 KB/s, write 350 KB/s.

As ESP32-C3 already has USB function, can I download firmware directly via USB without using the cp2102 chip?

Yes, ESP32-C3 can download firmware via USB, The USB serial port number should be displayed as COMx on Windows devices and ttyACMx on Linux devices.

Does ESP32-C3 support USB Host?

No, it only supports USB-Serial-JTAG function and can only be used as the USB device.

The ESP32-C3 chip can use USB to download firmware, but it is not supported under ESP-IDF v4.3. How to use USB to download firmware?

You need to compile under ESP-IDF v4.4 or later versions. After pulling the latest branch and [updating the IDF tool](#), you can compile normally and download it using USB. Please refer to [usb-serial-jtag-console](#) for the usage.

Does the ESP32-S2 support USB HID?

Yes. For the example of USB HID Device, please refer to [ESP-IDF Device HID](#). For the example of USB HID Host, please refer to [ESP-IDF Host HID example](#).

Why is this error log printed when I am testing the USB Camera + Wi-Fi Transfer example?

```
E (1437) UVC STREAM: Configuration descriptor larger than control transfer_↵  
↵max length
```

This error log is reported because the length of the descriptor sent by the USB Camera is larger than the default length (256). You can modify the following configuration to 2048 for testing:

```
Component config > UVC Stream > (2048) Max control transfer data size  
(Bytes)
```

Does ESP32-S3 support USB CDC for printing program log and downloading firmware?

Yes, ESP32-S3 supports printing program log and downloading firmware using USB CDC when the following configuration option is enabled:

```
Component config> ESP System Settings> Channel for console output > USB  
CDC
```

Does ESP32-S3 support devices with USB Device being Class 0?

- Yes, please refer to the example [esp-idf/components/tinyusb/additions/src/usb_descriptors.c](#). When class code == 00H, the class category is specified by the interface.
-

Can the ESP32-S3's USB OTG interface be used in both USB Host and USB Device modes?

- The ESP32-S3's USB OTG interface can not be used as USB Host and USB Device at the same time. However, it is possible to switch between the USB Host mode and the USB Device mode by software.
 - If you need the standard negotiation function of USB OTG, please note that currently ESP32-S3 only supports this function on the hardware, and does not support it in software protocol.
-

When testing the `esp-idf/examples/peripherals/usb/device/tusb_serial_device` example to send data using TinyUSB, do I have to use the `tinyusb_cdcacm_write_flush()` function?

To prevent sending FIFO overflows, you can use the ‘`tinyusb_cdcacm_write_flush()`’ function to flush. However, a large number of cycles of flushing may fail. So, it is recommended to set it according to the actual application.

Can ESP32-S3 use an external USB hub chip with two of its USB ports connecting to a USB 4G module and a dongle at the same time?

The ESP32-S3 USB does not support connection to an external USB hub chip currently because there is no driver support.

When ESP32-S2/ESP32-S3 serves as the UVC Host and connects some models of UVC cameras, why is there an error `HID_PIP1_EVENT_ERROR_OVERFLOW` in the log?

This is because MPS of the Alt interface endpoint of the selected camera is too large (ESP32-S2/ESP32-S3 supports up to 512 bytes). Please confirm whether the camera has an interface of less than or equal to 512 bytes under USB1.1.

Does ESP32-S2/ESP32-S3 have a USB 4G Internet access solution?

Yes, please refer to [USB CDC 4G Module Example](#).

Is there any USB CDC Host example for ESP32-S2/ESP32-S3?

Yes, please refer to [ESP-IDF USB CDC Host example](#) or [esp-iot-solution USB CDC Host example](#).

When burning firmware through the ESP32-C3/ESP32-S3 USB Serial/JTAG Controller function, I found that the PC sometimes cannot recognize the USB serial port, or automatically disconnects from the USB serial port repeatedly after recognizing it. What is the reason?

At present, the startup logic of the ESP32/ESP32-S2/ESP32-S3/ESP32-C3 chip is: if it cannot start normally (due to empty flash, no correct data/firmware in flash, the power-on sequence problem of flash, etc.), the internal timer will trigger the chip to restart every few seconds. The chip cannot connect stably until the program starts normally or enters the download mode. The ESP32-S3/ESP32-C3 USB-Serial-JTAG peripheral will be re-initialized when the chip restarts, so the corresponding result is that the chip tries to connect to and disconnects from the PC every few seconds. We provide the following two solutions:

- You need to boot the chip to enter the download mode manually before the first download or after flash is erased, so that the chip can be connected stably.

- You can also burn the firmware that can run stably through UART in advance to solve this issue. If there is firmware that can run stably in the chip, the USB serial port of the chip can be connected stably in subsequent programming.

If there is no strap pin test point reserved for booting manually, you may need to try several times in the initial USB download.

Why does ESP32-S2/ESP32-S3 not reach the maximum USB full speed, 12 Mbps?

We will explain this issue with the TinyUSB protocol stack as an example. As this USB mode does not use DMA, but directly uses CPU polling, some time slices are wasted in each transfer. As a result, the TinyUSB protocol stack is only expected to reach 6.4 Mbps (it can reach 9.628 Mbps theoretically if the batch transfer is adopted).

How can I confirm if ESP32-S2/ESP32-S3 USB supports a certain USB camera or not?

ESP32-S2/ESP32-S3 USB only supports USB cameras that correspond to `wMaxPacketSize` Video Streaming endpoints which include 512 bytes at the maximum. You can use *USB Stream Example* <https://github.com/espressif/esp-iot-solution/tree/master/examples/usb/host/usb_camera_mic_spk>__ to test. An error log will be printed if the camera is not supported.

What is the maximum resolution of USB cameras that ESP32-S2/ESP32-S3 support?

- If we do not consider local JPEG decoding, the bottleneck is the throughput rate of USB. The USB camera generally adopts synchronous transmission. ESP USB has a limitation of FIFO size, which can reach 500 KB/s at the maximum currently. Thus, if you want to achieve 15 frames, the size of each frame can only be 33 KB. The maximum resolution that can be achieved by 33 KB depends on the compression rate, and generally it can reach 480 * 320.
 - If you take local JPEG decoding into consideration, you also need to consider whether this resolution can reach 15 frames per second.
-

Can the ESP32-S2/ESP32-S3 USB recognize the USB plugging and unplugging action when it is used as a USB CDC Device?

- Yes, the USB device uses the tinyusb protocol stack, including mount and umount callback functions to response the USB plugging and unplugging events.
 - It should be noted that if this device is a self-powered USB device, please reserve a VBUS detection pin if you need to detect plugging and unplugging actions when it is powered on. For detail, please refer to *Solution of Self-Powered USB Device* <https://docs.espressif.com/projects/espressif-esp-iot-solution/en/latest/esp32/usb/usb_device_self_power.html>__.
-

After enabling the RNDIS and CDC functions on the ESP32-S3 USB, I found that the PC can recognize the COM port. However, the automatic programming function of the COM port is invalid. Is it expected?

- Yes. The USB auto-programming function is implemented through the USB-Seial-JTAG peripheral, and the USB RNDIS function is implemented through the USB-OTG peripheral. However, only one of the two peripherals can work at a moment.
 - If the USB-OTG peripheral is used in the application, the automatic programming function implemented by the USB-Seial-JTAG peripheral will not be available. But you can manually enter the download mode for USB burning.
-

Does the ESP32-S2/ESP32-S3 support the USB CDC NCM protocol?

- Currently, ESP32-S2/ESP32-S3 only supports the USB CDC ECM protocol, but does not support the USB CDC NCM protocol.

After I initialize the USB pins of ESP32-C3/ESP32-S3 to GPIO or other peripheral pins, why cannot I burn firmware through USB?

-The USB pin of the ESP32-C3/ESP32-S3 can be initialized to GPIO or other peripheral pins. However, please note that after the initialization, the original USB download function will be disconnected, and the download mode cannot be entered automatically through USB. But you can manually pull down the Boot pin (GPIO9 in ESP32-C3 and GPIO0 in ESP32-S3) to make ESP32-C3/ESP32-S3 enter the download mode. Then you can download firmware through USB.

What should I pay attention to if I want to use the USB interface of ESP32-C3/ESP32-S3 as the unique download interface of firmware?

- DO NOT use USB pins of ESP32-C3 (GPIO18 and GPIO19) / ESP32-S3 (GPIO19 and GPIO20) as other peripheral functions.
 - If the USB pins have to be reused as other functions in the application, the BOOT pin (GPIO9 in ESP32-C3, GPIO0 in ESP32-S3) must be wired to manually enter the chip into the download mode.
-

When I attempted to download and print log via the USB interface using the command `idf.py -p com35 flash monitor` on Windows, I encountered the following error. What's the reason for it?

- The error is as follows:

```
Connecting...
Failed to get PID of a device on com35, using standard reset sequence.
```

- On Windows, the COM port must be configured in the upper case, not the lower case `com`.
-

How can I apply for USB VID/PID for ESP32-S series products?

- You can use default TinyUSB PIDs if your software stack is based on TinyUSB. Otherwise, you need to apply for PID for each ESP32-S series product. For details, please refer to “[usb-pids](#)”.

Is it possible to fix the COM port when downloading firmware using the USB-Serial-JTAG interface on Windows?

- Please open the Windows CMD as the administrator and execute the following command to add a registry entry. In this way, you can prevent incremental numbering based on the Serial number. Then you need to restart the computer to enable the modification.

```
REG ADD HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\usbflags\
↪303A10010101 /V IgnoreHWSerNum /t REG_BINARY /d 01
```

- For more information, please refer to *Prevent Windows from Increasing COM Number According to Serial Numbers of USB Devices* <https://docs.espressif.com/projects/espressif-esp-iot-solution/en/latest/esp32/usb/usb_device_const_COM.html>__.

4.4.18 Other Peripherals

□

Can the REF_TICK clock frequency be modified?

CHIP: ESP32 | ESP32-S2 | ESP32-C3

No, the REF_TICK clock is fixed.

Does ESP32 support PCI-E protocol?

No, it doesn't.

4.5 Protocols

□

4.5.1 ESP-TLS

□

When testing RTOS SDK `mqtts/ssl_mutual_auth` with ESP8266, the server connection failed. Why?

- The failure of SSL connection may due to insufficient memory of ESP8266.
 - Please use the master version of ESP8266-RTOS-SDK to test this example, since it supports dynamic memory allocation in menuconfig so as to reduce the usage of memory peak. The specific action is: menuconfig -> Component config -> mbedTLS -> (type “Y” to enable) Using dynamic TX /RX buffer -> (type “Y” to enable) Free SSL peer certificate after its usage -> (type “Y” to enable) Free certificate, key and DHM data after its usage.
-

Can ESP HTTPS skip the server certificate check?

- Yes, if you enable the following options in menuconfig.
 - Menu path: (Top) -> Component config -> ESP-TLS -> Allow potentially insecure options
 - Menu path: (Top) -> Component config -> ESP-TLS -> Allow potentially insecure options-> Skip server certificate verification by default
 - Besides, make sure that the `cert_pem` member variable is not set in the `esp_http_client_config_t` structure. Otherwise, the server certificate will still be verified with this CA certificate.
 - If you want to test HTTP OTA at the same time, you need to enable the Menu path: (Top) -> Component config->ESP HTTPS OTA->Allow HTTP for OTA option in menuconfig.
-

How to set the `esp_tls_conn_read` API in ESP-TLS to non-blocking mode? Or is there any other way to implement non-blocking?

- You can set `non_block` to true in the `esp_tls_cfg_t` structure in `esp_tls.h` to achieve non-blocking.
 - Alternatively, you can call `esp_transport_connect_async` to achieve non-blocking.
-

What are the TLS versions supported by ESP-IDF?

- The recommended TLS protocol in ESP-IDF is the Mbed TLS protocol.
- ESP-IDF v5.0 and later no longer support SSL 3.0, TLS 1.0 and TLS 1.1, but only support TLS 1.2 and TLS 1.3.

4.5.2 HTTP



Does ESP8266 support HTTP hosting?

Yes, it does. ESP8266 can run as a server in both SoftAP and Station modes.

- When running as a server in SoftAP mode, clients can directly access the ESP8266 host or server at 192.168.4.1 (default) IP address.
- When the server is accessed via a router, the IP address should be the one allocated to the ESP8266 by the router.
- When using SDK to write native code, please refer to relevant examples.
- When using AT commands, start a server using AT+CIPSERVER command.

How to use esp_http_client to send chunked data?

- Please use [HTTP Stream](#) by setting the write_len parameter of esp_http_client_open() to -1. Then the “Transfer-Encoding” will be set to “chunked” automatically please see http_client_prepare_first_line() in esp_http_client.c.
- The code snippet is listed below for your reference

```
static void http_post_chunked_data()
{
    esp_http_client_config_t config = {
        .url = "http://httpbin.org/post",
        .method = HTTP_METHOD_POST, // This is NOT required. write_len < 0 will force_
        ↪ POST anyway
    };
    char buffer[MAX_HTTP_OUTPUT_BUFFER] = {0};
    esp_http_client_handle_t client = esp_http_client_init(&config);

    esp_err_t err = esp_http_client_open(client, -1); // write_len=-1 sets header
    ↪ "Transfer-Encoding: chunked" and method to POST
    if (err != ESP_OK) {
        ESP_LOGE(TAG, "Failed to open HTTP connection: %s", esp_err_to_name(err));
        return;
    }

    // Post some data
    esp_http_client_write(client, "5", 1); // length
    esp_http_client_write(client, "\r\n", 2);
    esp_http_client_write(client, "Hello", 5); // data
    esp_http_client_write(client, "\r\n", 2);
    esp_http_client_write(client, "7", 1); // length
    esp_http_client_write(client, "\r\n", 2);
    esp_http_client_write(client, " World!", 7); // data
    esp_http_client_write(client, "\r\n", 2);
    esp_http_client_write(client, "0", 1); // end
}
```

(continues on next page)

(continued from previous page)

```
esp_http_client_write(client, "\r\n", 2);
esp_http_client_write(client, "\r\n", 2);

// After the POST is complete, you can examine the response as required using:
int content_length = esp_http_client_fetch_headers(client);
ESP_LOGI(TAG, "content_length: %d, status_code: %d", content_length, esp_http_
↪client_get_status_code(client));

int read_len = esp_http_client_read(client, buffer, 1024);
ESP_LOGI(TAG, "receive %d data from server: %s", read_len, buffer);
esp_http_client_close(client);
esp_http_client_cleanup(client);
}
```

Is there a way to set cookies when ESP32 operates as an HTTP client?

ESP32 itself does not have an API for setting cookies directly, but you can use `esp_http_client_set_header` to add cookies to the HTTP header.

How do I set the maximum number of clients that are allowed for connection when ESP32 serves as an HTTP server? What will happen if the number exceeds the limit?

- The maximum number of client connections can be set by configuring `max_open_sockets` in the `httpd_config_t` structure.
 - If the number of clients exceeds the limit, you can set the `lru_purge_enable` parameter in the `httpd_config_t` structure to true. In doing so, if there is no socket available (which is determined by `max_open_sockets`), the least used socket will be cleared to accept the coming one.
-

Does ESP32 have an example of implementing a gRPC client over HTTP/2 and above versions?

Not yet.

How to download a specific segment of a file over HTTP in ESP-IDF (i.e., add `Range: bytes` information to the header)?

Please refer to the `http_partial_download` function in the `esp http client` example.

4.5.3 lwIP



How soon can the associated resources be released after the TCP connection is closed?

The associated resources can be released in 2 MSL, i.e. 120 seconds, or after the sent `linger/send_timeout` parameter is timeout.

After the SNTP calibration for ESP8266 RTOS SDK v3.2, errors gradually increase. How to resolve such issue?

This is because the ESP8266 uses software timer, which brings large errors itself. You can improve it with the following solutions:

- For branch v3.2, you can resynchronize time (300 s is recommended) from the server regularly by creating a task.
 - For branch release-v3.3, the code of system timer has been refactored and is tested with low errors. On the other hand, you can still synchronize time from the server regularly.
 - The master branch has inherited the refactored code from branch release-v3.3. In addition, you can configure the SNTP synchronization interval in menuconfig: `Component config > LWIP > SNTP > Request interval to update time (ms)`.
-

Does ESP8266 support loop-back for the device end?

- Yes, it does.
 - Please enable the LOOPBACK option from lwIP in menuconfig: `menuconfig > Component config > LWIP > Enable per-interface loopback (type “Y” to enable)`.
 - The device end sends data to the loopback address 127.0.0.1, and can read it from the address.
-

What is the default packet length for TCP/IP?

Please go to `menuconfig > Component config > LWIP > TCP > Maximum Segment Size (MSS)` for the length.

When using UTC and GMT methods in SNTP protocol, why can't I get the time of the target time zone

- The “TZ = UTC-8” refers to POSIX time, in which “UTC” is the abbreviation of any time zone and the number is the number of hours that the time zone is behind UTC.
 - “UTC-8” indicates a certain time zone, “UTC” for short, which is -8 hours later than the actual UTC. Therefore, “UTC+8” is 8 hours later than the actual UTC, and also 16 hours later than Beijing.
-

Is there any special firmware or SDK in ESP32 that can only provide AP/STA (TCP/IP bypass) without using its internal TCP/IP so as to give developers more permissions?

The software solution ESP-Dongle can fit your requirements. Please contact [Business Team](#) to sign NDA and then get related solutions.

When ESP32 & ESP8266 are used as TCP servers, how can the ports be used again immediately after they are released?

- On both ESP32 and ESP8266, TCP ports are not immediately released after being closed. They remain in a TIME_WAIT state for a certain period of time. During this period, binding a socket with the same port and source address as before will fail. This is to ensure that you can receive the FIN signal sent by the server and can close the connection successfully. In this state, the port cannot be immediately reused. To address this issue, the socket option “SO_REUSEADDR” should be used, which allows the device to bind a TCP socket with the same port and source address in the TIME-WAIT state.
- Therefore, a TCP server program can set the “SO_REUSEADDR” socket option before calling bind() to bind the same port.
- Alternatively, the setsockopt() function can be used to set the SO_REUSEADDR option. Here is an example:

```
int reuse = 1;
if (setsockopt(socket, SOL_SOCKET, SO_REUSEADDR, &reuse, sizeof(reuse)) < 0) {
    ESP_LOGE(TAG, "setsockopt(SO_REUSEADDR) failed");
    return ESP_FAIL;
}
```

In the above code, “socket” means a socket that has already been created, and “reuse” is an integer variable with a value of 1, indicating that the SO_REUSEADDR option is enabled. If the setsockopt() function returns a negative value, it means that the setting has failed. Enabling the SO_REUSEADDR option allows the port to be immediately reused after being closed. However, there are some potential risks. If another connection uses the same port while it is still in the TIME_WAIT state, it may cause packet confusion. Thus, it's better to make choice based on actual situations.

After downloading the `tcp_client` example for an ESP32 module, I connected the module to the router via Wi-Fi and performed a Ping test on the computer. Then the it shows high latency sometimes, what is the reason?

When Wi-Fi is connected, Power Save mode will be turned on by default, which may cause high Ping delay. To solve this issue, you can turn off Power Save mode to reduce the delay by calling `esp_wifi_set_ps(WIFI_PS_NONE)` after `esp_wifi_start()`.

How can I set static IP when using ESP-IDF?

For details, please refer to `static_ip` example.

Does ESP32 have an LTE connection demo?

- Yes. For ESP-IDF v4.2 and later versions, please refer to the `example/protocols/pppos_client` demo.
 - For ESP-IDF v5.0 and later versions, please refer to `examples` in the *esp-protocols* repo.
-

Will memory leak occur when ESP32 TCP repeatedly closes and rebuilds socket (IDF 3.3)?

In ESP-IDF v3.3, every time a socket is created, a lock will be assigned, given that this internal socket array has not been assigned any lock before. This lock will not be reclaimed after the socket is released. Thus, next time the same socket array is allocated, the previous lock will be used again. That is to say, every time a new socket array is allocated and released, there will be one lock memory used. After all socket arrays being allocated, there will be no memory leak any more.

Are there any limits on the maximum number of TCP client connection after the ESP32 additionally opens the TCP server?

Yes. The number of simultaneously connected socket fd number for ESP32 is limited by `LWIP_MAX_SOCKETS`, which is 10 by default.

What is the default MTU of lwIP for an ESP32?

The default MTU of lwIP is 1500. This is a fixed value and it is not recommended to change it.

How to increase the DNS request time for ESP32?

You can manually modify the `#define DNS_MAX_RETRIES 4` in `esp-idf/components/lwip/lwip/src/include/lwip/opt.h`. For example, you can change the value of `#define DNS_MAX_RETRIES` to 10. In this way, the maximum time that DNS waits for a response from the server is 46 s (1+1+2+3+4+5+6+7+8+9).

After creating and closing TCP SOCKET several times, an error is reported as “Unable to create TCP socket: errno 23”. How to resolve such issue?

:CHIP: ESP8266 | ESP32 | ESP32-S2 | ESP32-C3 | ESP32-S3 :

- Reason: “errno 23 ” means open many open files in system. Closing a socket takes 2 MSL of time, which means sockets will not be closed immediately after calling the close interface. Due to this reason, open sockets are accumulated and exceeds the maximum connection number (the default is 10 in menuconfig, the maximum connection is 16) thus triggering this error.
- Solution: Set `SO_LINGER` via the `setsockopt` interface to adjust the TCP close time.

```
linger link ;
link.on_off = 1 ;
link.linger = 0 ;
setsockopt(m_sockConnect, SOL_SOCKET, SO_LINGER, (const char*)&link, sizeof(linger));
```

What happens when ESP8266 receives a “tcp out of order” message?

- If `CONFIG_LWIP_TCP_QUEUE_OOSEQ`(Component config -> LWIP -> TCP -> Queue incoming out-of-order segments) is enabled, the out-of-order messages will be stored at the cost of memory consumption.
 - If this configuration is disabled, after receiving the “out of order” message, data will be discarded and a retransmission will be requested. For example, there are four data packets namely 1, 2, 3 and 4, ESP8266 receives 1 first, and then receives 4. If this configuration is enabled, ESP8266 will store the data of 4, wait until it receives 2, 3, and then report the four packets to the application layer; if this configuration is disabled, ESP8266 will discard the packet of 4 when it receives it, and let the other side send packet 2, and then the other side will send from 2. Under this condition, the retransmission is increased.
-

Does ES32 support PPP functionality?

Yes, please refer to `esp_modem` example.

Every time ESP32 attempts to read 4 KB of data with `read` and `recv` APIs in the socket, it can not always read the 4 KB of data. Why?

- Both `read` and `recv` APIs are used to read the data in the underlying buffer. For example, if there are 100 bytes of data in the underlying buffer, and the `len` size passed in by `read` and `recv` is only 50, then the API will return after it reads 50 bytes. If `len` exceeds the length of the data received in the underlying buffer, say 200, the API will return after it reads 100 bytes and will not wait until it receives 200 bytes. So, an attempt to read 4 KB of data will not necessarily return 4 KB of data, but only the data available in the underlying buffer at the time of reading.
 - If you need to read 4 KB of data every time, it is recommended to use application code on top of the socket layer to design the corresponding logic, which reads data recursively until it reaches 4 KB.
-

What is the version of lwIP currently used in ESP-IDF?

lwIP v2.1.3 is used currently.

In DHCP mode, will ESP32 renew the IP or apply for a new IP when the lease expires?

There are two lease periods, T1 (1/2 time of the lease) and T2 (7/8 time of the lease) in DHCP mode. When both lease expires, ESP32 usually renews the same IP. Only when both of them fail to renew will ESP32 apply for a new IP.

Why does ESP-IDF report an error when `SO_SNDBUF` option of `setsockopt` are used to get or set the size of the send buffer?

By default, lwIP does not support `SO_SNDBUF`. To set the send buffer size, go to `menuconfig -> Component config -> LWIP -> TCP -> Default send buffer size`. To get or set the receive buffer size, you need to enable the `CONFIG_LWIP_SO_RCVBUF` option in `menuconfig` before you can use the `SO_RCVBUF` option of `setsockopt` to get or set the receive buffer size.

I find that the network data latency of TCP & UDP is large when testing ESP-IDF. What is the buffering data mechanism of TCP & UDP protocols?

- For TCP, there is `TCP_NODELAY` option in socket option. You can enable this option to disable the Nagle algorithm which is enabled by default, so that the data will not be cached locally and then sent together.
 - For UDP, UDP data is sent directly. If there is any delay, it is because of the delay of the Wi-Fi network environment, not the UDP itself.
 - If TCP retransmission is caused by poor network environment, and the transmission interval is set too long, high latency will occur. You can try to shorten the RTO value (by modifying `component config -> lwip -> tcp -> Default TCP rto time` and `TCP timer interval` options in `menuconfig`).
-

How to choose the default route when ESP32 works as dual NICs (e.g. ETH+STA)?

The following summarizes how the default route is selected for dual NICs, using ETH and STA as examples.

- Supposing ETH and STA are in the same LAN:
 - When the device accesses the LAN address, the data will go to the last up netif.
 - When the device accesses the non-LAN address, the data will go to the netif with the larger `route_prio` value.
 - Supposing ETH and STA are not in a LAN, ETH belongs to 192.168.3.x segment, and STA belongs to 192.168.2.x segment:
 - When the device accesses 192.168.3.5, it will take the ETH netif.
 - When the device accesses 192.168.2.5, it will take the STA netif.
 - When the device accesses 10.10.10.10, it takes the default route (the netif with the larger `route_prio` value). When netif is up, it sets the default route based on the `route_prio` value size, and the default route is often the netif with the larger `route_prio` value. When the device accesses an address that is not inside the routing table, the data takes the default route.
-

How do I enable keepalive for TCP in ESP-IDF?

You can refer to the code for enabling TCP keepalive in `esp_tls.c`.

Is it possible to operate the same socket in multiple threads in ESP-IDF?

In ESP-IDF, it is possible for multiple threads to share one single socket for communication. Each thread can use the same socket to send and receive data, but it is important to ensure thread synchronization when accessing the socket to avoid race conditions and deadlocks. Typically, a mutex can be used to control access to the socket, ensuring that each thread's access to the socket is mutually exclusive to avoid data corruption caused by concurrent access to the socket. However, operating on the same socket from multiple threads is risky, and it is not recommended.

How much time do ESP devices allocate to other device's IPs in ESP DHCP server mode?

The default is 120 s. Please refer to the `DHCP_SERVER_LEASE_TIME_DEF` parameter, which is not recommended to be set to a small value.

What are the three lease related times in ESP-IDF DHCP? What parameters in the code do they correspond to?

They are Address Lease Time, Lease Renewal Time and Lease Rebinding Time, corresponding to the lwIP codes `offered_t0_lease`, `offered_t1_renew`, and `offered_t2_rebind` respectively.

What is the maximum length for each data transmission in the ESP-IDF lwIP?

If you are using the socket interface `send`, the maximum length supported is determined by the `SSIZE_MAX` parameter. If you use the `tcp_write` function, the maximum length is limited by `snd_buf` (send buffer length). `send` is a socket interface wrapped by lwIP based on the sequential API, which is a higher-level interface than `tcp_write` and is more suitable for user-level calls. There is basically no difference in resource usage between the two API calls.

If I need more debug logs for lwIP layer related issues with ESP-IDF, how can I enable the corresponding debug log to be printed (e.g. DHCP, IP)?

- To print lwIP-related debug log, open `menuconfig`, go to `Component config->LWIP`, and enable the option `Enable LWIP Debug`. There are sub-options, including `Enable IP debug messages` and `Enable DHCP debug messages`. You could enable them as needed.
 - If you don't find the desired debug log module in the above `menuconfig`, such as, UDP module, first check if there is `#define UDP_DEBUG` in `esp-idf/components/lwip/port/esp32/include/lwipopts.h`. If yes, change `#define UDP_DEBUG LWIP_DBG_OFF` to `#define UDP_DEBUG LWIP_DBG_ON` manually. If no, add `#define UDP_DEBUG LWIP_DBG_ON` to `esp-idf/components/lwip/port/esp32/include/lwipopts.h` referring to `#define UDP_DEBUG LWIP_DBG_OFF` in `esp-idf/components/lwip/lwip/src/include/lwip/opt.h` file.
-

What is the difference between socket blocking and non-blocking in ESP-IDF?

- For reads, the difference is whether the read interface returns immediately when no data arrives at the bottom. A blocking read will wait until data has arrived or until an exception occurs, while a non-blocking read will return immediately with or without data.
 - For writes, the difference is whether the write interface returns immediately when the underlying buffer is full. For blocking write, if the underlying buffer is not writable (the underlying buffer is full or the peer has not acknowledged the previously sent data), the write operation will keep blocking until it is writable or an exception occurs. For non-blocking write, it will write as much as it can without waiting for the underlying buffer to be writable or the length of the write to be returned.
 - The non-blocking interface call does not block the current process, while the blocking interface does.
-

Can ESP32 use the IP of the previous successful connection for communication after connecting to the router, and in case of failure, re-enter the authentication process and use DHCP to obtain a new IP?

- Yes, if you enable Component config -> LWIP -> DHCP: Restore last IP obtained from DHCP server option in menuconfig.
 - Note that you cannot use a static IP instead, because static IP settings do not have conflict detection. It may lead to IP conflict.
-

How do I achieve connect_timeout when programming with sockets?

- If you set the socket to the non-blocking mode, the connect() function will also be non-blocking. Then you can set the timeout by the select() function to determine whether the socket is connected successfully or not. For details, please refer to “[connect_timeout settings of sockets](#)”.
-

When ESP32 uses SNTP to synchronize the current time, I found that there is a random delay. After further analysis, I found that it is caused by SNTP_STARTUP_DELAY in the IDF lwip component, the default value of which is 1. Is there any way to avoid the random delay without modifying the IDF component?

- There is no way to avoid the random delay without modifying the IDF component. You need to manually add the code `#define SNTP_STARTUP_DELAY 0` to lwipopts.h in the lwip component. This code reduces the time that SNTP takes to send a request, so it can reduce the total time for ESP devices connecting to the cloud after they are powered up as a result.
 - The reason for enabling this random delay option by default is that it is mandated by the SNTP RFC protocol. A random delay can reduce the number of simultaneous accessing devices, so this can prevent the SNTP server from being overloaded.
-

Does IPv6 support setting static IP?

IPv6's link-local address is generated automatically according to certain protocol rules, so there is no need to manually set it. Therefore, it cannot set static addresses as IPv4.

4.5.4 Mbed TLS

□

Does ESP8266 OpenSSL support hostname validation?

Yes. ESP8266 OpenSSL is based on Mbed TLS encapsulation, which supports hostname validation. ESP-TLS can be used to switch between Mbed TLS and wolfSSL.

How to optimize memory when ESP32 uses Mbed TLS?

- You can enable dynamic buffer in menuconfig, the specific operation is `menuconfig > Component config > mbedTLS > Using dynamic TX/RX buffer` (key "Y" to enable).
- At the same time, you can enable the sub-options `Free SSL peer certificate after its usage` and `Free certificate, key and DHM data after its usage` in the `Using dynamic TX/RX buffer` in the previous step.
- However, ESP-IDF v5.0 and later no longer have sub-option `Free SSL peer certificate after its usage`, and Mbed TLS enables `MBEDTLS_SSL_KEEP_PEER_CERTIFICATE` by default. If you want to save memory, you can close it by `menuconfig > Component config > mbedTLS > mbedTLS v3.x related > Keep peer certificate after handshake completion` (key "N" to disable).

When I connected an ESP32 module with the HTTPS Server, I got the following log. What is the reason?

```
free heap size: 181784 bytes
I (4285) esp_https_server: Starting server
E (4285) esp_https_server: Could not allocate memory
I (4295) example: Error starting server!
I (4295) SSDP Server: SSDP server started
free heap size: 178636 bytes
```

- The error is caused by low memory. The log shows that you use the `esp_get_free_heap_size()` API to get the remaining memory. However, the remaining memory includes the chip's internal RAM as well as external PSRAM.
- By default, mbedTLS uses internal RAM memory, and you can use the `esp_get_free_internal_heap_size()` API to obtain the remaining internal memory.
- If the module has an external PSRAM, you can modify the configuration from `menuconfig > Component config > mbedTLS > Memory allocation strategy > Internal memory` to `menuconfig > Component config > mbedTLS > Memory allocation strategy > External SPIRAM` for testing.

When resolving a hostname on ESP32, I encountered the following error. What could be the reason?

- The error is caused by DNS request timeout.
- You can enable DNS log with the debug level or capture wireless packets for further analysis.
- To enable the debug level DNS log, you can add `#define DNS_DEBUG LWIP_DBG_ON` code to the `esp-idf/components/lwip/lwip/src/include/lwip/opt.h` file, and then enable the Component config > LWIP > Enable LWIP Debug configuration.

4.5.5 MQTT

[]

How to configure the server address so as to make it an autonomic cloud platform by using MQTT?

Please refer to [MQTT Examples](#).

I'm using ESP8266 release/v3.3 version of SDK to test the example/protocols/esp-mqtt/tcp example. Then during Wi-Fi configuration, the connection fails after configuring SSID, password and connecting to the default server. The log is as follows, what is the reason?

```
W (4211) MQTT_CLIENT: Connection refused, not authorized
I (4217) MQTT_CLIENT: Error MQTT Connected
I (4222) MQTT_CLIENT: Reconnect after 10000 ms
I (4228) MQTT_EXAMPLE: MQTT_EVENT_DISCONNECTED
I (19361) MQTT_CLIENT: Sending MQTT CONNECT message, type: 1, id: 0000
```

When such error occurs, it indicates that the server rejected the connection because the client's wrong MQTT username and password caused the server-side authentication to fail. Please check if you are using the correct MQTT username and password.

What is the default keepalive value of the MQTT component in ESP-IDF?

The default value is 120 s, which is defined by `MQTT_KEEPA_LIVE_TICK` in file `mqtt_config.h`.

Does MQTT support automatic reconnection?

- The automatic reconnection of MQTT is controlled by the `disable_auto_reconnect` variable of struct `esp_mqtt_client_config_t`. The default value of `disable_auto_reconnect` is false, which means that automatic reconnection is enabled.
 - The reconnection timeout value can be set using `reconnect_timeout_ms`.
-

What are the supported MQTT versions of ESP-IDF?

ESP-IDF currently supports MQTT 3.1 and MQTT 3.1.1, and MQTT 5.0.

When a Wi-Fi connection is disconnected in ESP-IDF, will the memory previously requested by MQTT upper layer protocol be automatically released?

- No, but you do not need to care about this memory. What you need to care about is the application layer that ESP encapsulates.
 - For MQTT application layer components, you get an MQTT handle when initializing MQTT. You only need to care about the memory in this handle. When not using MQTT, you can call `stop` or `destroy` to release the corresponding MQTT memory. When Wi-Fi is disconnected and connected, you do not need to release the MQTT memory or reapply for the handle, because there is an automatic reconnection mechanism in the MQTT component.
-

For ESP32-C3 MQTT, can I not set corresponding `client_id` but configure it as an empty string by default?

- Yes, you can achieve this by setting `set_null_client_id` to `true` in the application code.
-

When `MQTT_EVENT_PUBLISHED` is triggered after an ESP-IDF MQTT client has published data with QoS of 1 or 2, does it mean that a proper ack has been received from the other side to prove that the publish has completed? Or does it just mean that the data was successfully sent to the server once?

The `MQTT_EVENT_PUBLISHED` event triggered means that the broker has acknowledged receipt of the messages published by the client, proving that the publish has completed successfully.

How does an ESP MQTT client manually release MQTT resources after disconnection?

Calling the `esp_mqtt_client_destroy` API will do the trick.

How should I configure the MQTT keepalive time when ESP32 Wi-Fi and Bluetooth LE coexist? Is there any appropriate configuration time?

- When using Wi-Fi and Bluetooth LE concurrently in ESP32, it is recommended to configure the MQTT keepalive time properly. Since both Wi-Fi and Bluetooth LE require system resources, setting the keepalive time too short may cause high system load, affecting system stability and performance.

- Generally, it is advisable to set the MQTT keepalive time based on actual needs to ensure the device stays online while minimizing system resource consumption. In the case of Wi-Fi and Bluetooth LE coexistence, it is recommended to set the MQTT keepalive time to a longer duration, such as 30 seconds or 60 seconds, to reduce communication between the device and the MQTT broker, thereby reducing system load.
 - It is important to note that setting the keepalive time too long may cause a delay in detecting the device offline when it disconnects, which may affect real-time performance and reliability. Therefore, the MQTT keepalive time should be set based on actual needs and system performance.
-

When will the disconnect event message be triggered for ESP-MQTT clients?

The disconnect message only occurs in the follow cases:

- A TCP connection error occurs while the MQTT connection is being established.
 - An MQTT connection error occurs while the MQTT connection is being established.
 - You actively call the `disconnect` function.
 - An exception is received or sent.
 - The MQTT `PING RESPONSE` is not received within the specified time.
 - The MQTT `PING` request failed to be sent.
 - Reconnection.
-

Does the ESP32 MQTT client automatically try to reconnect after disconnecting from the server?

The `esp_mqtt_client_config_t` structure in the ESP-MQTT client has the `disable_auto_reconnect` parameter, which can be configured as `true` or `false` to determine to reconnect or not. By default, it will reconnect.

How to check if the ESP32 is disconnected from the MQTT server?

To detect if the ESP32 has been disconnected from the server, you can use MQTT's `PING` mechanism by configuring the keepalive parameters `disable_keepalive` and `keepalive` in the `esp_mqtt_client_config_t` structure in ESP-MQTT. For example, if you configure `disable_keepalive` to `false` (default setting) and `keepalive` to 120 s (default setting), the MQTT client will periodically send `PING` to check if the connection to the server is working.

4.5.6 Other Protocols

□

How to optimize communication latency for ESP32?

- It is recommended to turn off the sleep function for Wi-Fi by calling the API `esp_wifi_set_ps(WIFI_PS_NONE)`.
 - You can also disable the AMPDU function in `menuconfig`.
-

Does ESP8285 support CCS (Cisco Compatible eXtensions)?

No, it doesn't.

Does ESP32 support LoRa (Long Range Radio) communication?

No, the ESP32 itself does not have the LoRa protocol stack and the corresponding RF parts. However, to realize communication between Wi-Fi and LoRa devices, you can connect an external chip integrated with LoRa protocol to ESP32. In this way, ESP32 can be used as the master control MCU to connect the LoRa chip.

After calling `esp_netif_t* wifiAP = esp_netif_create_default_wifi_ap()` for ESP32-S2 chips, a following call of `esp_netif_destroy(wifiAP)` to deinit caused a 12-byte of memory leakage. What is the reason?

- It is necessary to call `esp_wifi_clear_default_wifi_driver_and_handlers(wifiAP)` before `esp_netif_destroy(wifiAP)`. This is the correct deinit process. Following this process, there will be no memory leakage.
 - Alternatively, call `esp_netif_destroy_default_wifi(wifiAP)`, which is supported by ESP-IDF v4.4 and later versions.
-

How to implement the certificate auto-download function?

CHIP: ESP32

Please refer to [aws certificate automatic download function](#).

How to get more debug information based on errno in ESP-IDF?

- The `errno` list in ESP-IDF v3.x exists directly in the IDF. Click [errno.h](#) to check it.
 - The `errno.h` for ESP-IDF v4.x is located under the compiler toolchain. For example, for `esp-2020r3`, the path of `errno.h` is `/root/.espressif/tools/xtensa-esp32-elf/esp-2020r3-8.4.0/xtensa-esp32-elf/xtensa-esp32-elf/include/sys/errno.h`.
-

Does the ESP8266_RTOS_SDK support the TR-069 protocol?

No. The ESP8266_RTOS_SDK itself does not provide native support for the TR-069 protocol, but you can implement TR-069 protocol stack according to your requirements and integrate it into the ESP8266_RTOS_SDK. You can also use third-party TR-069 protocol stacks for integration. In summary, ESP8266_RTOS_SDK can support the TR-069 protocol, but requires yourself to integrate and implement.

Does the ESP32 support SAVI?

No, SAVI (Source Address Validation Improvements) is to establish a binding relationship based on IPv6 source address, source MAC address and access device port on the access device (AP or switch) by listening to control packets (such as ND, DHCPv6), i.e. CPS (Control Packet Snooping), and then perform source address validation on IP packets passing through the specified port. Only when the source address of the message matches with the binding table entry can it be forwarded to ensure the authenticity of the source address of data messages on the network. This is generally a policy protocol for switches or enterprise-class AP routers. Currently ESP32 supports IPv6 link-local address and global address for communication.

When Ethernet and Wi-Fi coexist, does Ethernet take precedence over Wi-Fi in data transfer?

:CHIP: ESP32 :

- Call `esp_netif_get_route_prio` first to check the priority of Ethernet and Wi-Fi. If Wi-Fi takes priority over Ethernet, you can prioritize them by modifying `route_prio` in the structure `esp_netif_t`.

4.6 Provisioning

□

4.6.1 Can I add any broadcast data I want to Android ESP-Touch (e.g., add a device ID so that ESP32 can receive this ID)?

- ESP-Touch is a communication protocol used to establish a Wi-Fi connection between a mobile phone and ESP8266/ESP32. It uses a special broadcast method to transmit Wi-Fi SSID and password. Typically, the data broadcasted by ESP-Touch should be in a fixed format and cannot include customized data.
- If you want ESP32 to receive customized data such as device ID, you can consider using other communication protocols such as MQTT or HTTP. With these protocols, you can define data formats as you wish and communicate between an Android App and ESP32.
- If you still want to broadcast customized data, you can use BluFi, which is the networking protocol based on Bluetooth LE. Please refer to the following references for BluFi:
 - Android APP <https://github.com/EspressifApp/EspBluFiForAndroid>.
 - iOS APP <https://github.com/EspressifApp/EspBluFiForiOS>.

4.7 Security



4.7.1 Is the firmware in ESP8266 readable?

Yes, because the firmware in ESP8266 is located in the external flash, thus can be read externally. In addition, ESP8266 does not support flash encryption and all the data is written in plaintext.

4.7.2 Is it possible to encrypt firmware for ESP8285?

- No, the ESP8285 chip does not support firmware encryption function.
 - Both ESP32 and ESP32-S2 support firmware encryption, thus can be your substitution.
 - If you insist on using ESP8285, you can achieve data encryption by adding an encrypted chip externally.
-

4.7.3 What is the difference between secure boot v1 and v2?

Compared with secure boot v1, secure boot v2 has the following improvements: - The bootloader and app use the same signature format. - The bootloader and app use the same signing key.

Currently, [secure boot v1](#) is only recommended for earlier versions than ESP32 v3.0. For ESP32 v3.0 and later versions, ESP32-C3, ESP32-S2, and ESP32-S3, it is recommended to use [secure boot v2](#).

4.7.4 After enabling secure boot, there is a build error indicating missing files. What could be the reasons

Error log: /Makefile.projbuild:7/f/ESP32Root/secure_boot_signing_key.pem

Reason: security boot is a function for firmware signature verification, which requires generating key pairs. - For the method of generating a key pair when secure boot v1 is enabled, please refer to [secure boot v1 key generation](#). - For the method of generating a key pair when secure boot v2 is enabled, please refer to [secure boot v2 key generation](#).

4.7.5 After enabling secure boot, is it possible for modules to be flashed again?

- If the secure boot v1 is configured as one-time, then it can only be flashed once and the bootloader firmware cannot be reflashed.
 - If the secure boot v1 is configured as reflashable, then the bootloader firmware can be flashed again.
 - The secure boot v2 allows reflashing the bootloader and app firmware.
-

4.7.6 With flash encryption enabled, a module reports an error as flash read error after reflashed. How to resolve such issue?

With flash encryption enabled, the module will not support plaintext firmware flash. For common failures, please refer to [Possible Failures](#). You can use the `espefuse` script to disable the encryption and then reflash the plaintext firmware, or directly flash the encrypted firmware to devices referring to the [flash encryption example](#).

Note: Please note there is a time limit for the flash encrypted function.

4.7.7 After enabling flash encryption and secure boot for ESP32, how to disable them?

- If you are using the one-time flash (Release) mode, both flash encryption and secure boot cannot be disabled.
 - If you are using the reflashable (Development (NOT SECURE)) mode, the flash encryption can be disabled, please refer to [Disabling Flash Encryption](#); while the secure boot cannot be disabled.
-

4.7.8 Is there any security strategy for ESP32 to protect its firmware?

- ESP32 supports flash encryption and secure boot.
 - For flash encryption, please refer to [flash encryption](#).
 - For secure boot, please refer to [secure boot](#).
 - For secure boot V2, please refer to [secure boot V2 for chip revision v3.0](#).
-

4.7.9 When ESP32 debugging GDB after enabling flash encryption, why does it continuously reset and restart?

- After ESP32 enabling flash encryption or secure boot, it will restrict JTAG debugging by default, please refer to [Tips and Quirks](#).
 - You can read the current JTAG status of your chip using the `espefuse.py summary` command from esptool.
-

4.7.10 How to enable flash encryption for ESP32?

- It can be enabled via `menuconfig` or `idf.py menuconfig` by configuring `Security features -> Enable flash encryption on boot` (READ DOCS FIRST).
 - Please refer to [Flash encryption instructions](#).
-

4.7.11 After GPIO0 is pulled down, the ESP32 cannot enter download mode and prints “download mode is disable”. What could be the reason?

- The log means the chip’s UART Download mode has been disabled. You can check this via the `UART_DOWNLOAD_DIS` bit in [eFuse](#).
 - Please note that after the Production mode of flash encryption is enabled, the UART Download mode will be disabled by default. For more information, please refer to [UART ROM download mode](#).
-

4.7.12 Can the secure boot function be enabled for ESP32 in Arduino development environment?

- No. If you want to use Arduino for development, the only way to enable the secure boot function is to use Arduino as an IDF component.
-

4.7.13 What are the use scenarios for secure boot and flash encryption?

- When secure boot is enabled, the device will only load and run firmware that is signed by the specified key. Therefore, it can prevent the device from loading illegal firmware and prevent unauthorized firmware from being flashed to the device.
 - When flash encryption is enabled, the partitions on the flash where firmware is stored and the data in the partitions marked as “encrypted” will be encrypted. Therefore, it can prevent the data from being illegally viewed, and firmware data copied from flash cannot be applied to other devices.
-

4.7.14 What are the data stored in eFuse involved in secure boot and flash encryption?

- For the data stored in eFuse used in secure boot v1, please refer to [secure boot v1 efuses](#)
 - For the data stored in eFuse used in secure boot v2, please refer to [secure boot v2 efuses](#)
 - For the data stored in eFuse used in flash encryption, please refer to [flash encryption efuses](#)
-

4.7.15 Enabling secure boot failed with the log “Checksum failure”. How to fix it?

- After enabling secure boot, the size of bootloader.bin will increase, please check whether the size of the bootloader partition is enough to store the compiled bootloader.bin. For more information, please refer to [Bootloader Size](#)

4.7.16 NVS encryption failed to start and an error occurred as `nvs: Failed to read NVS security cfg: [0x1117] (ESP_ERR_NVS_CORRUPT_KEY_PART)`. How can I solve this issue?

- Please erase flash once using the flash tool before starting NVS encryption, and then flash the firmware which can enable the NVS encryption to the SoC.

4.7.17 After flash encryption was enabled, a warning occurred as `esp_image: image at 0x520000 has invalid magic byte (nothing flashed here)`. How can I solve this issue?

- After SoC starts flash encryption, it will try to encrypt the data of all the partitions of the app type. If there is no corresponding app firmware stored in one app partition, the above log will appear. To avoid this warning, you can flash pre-compiled app firmware to the partitions of the app type when starting flash encryption.

4.7.18 Why is reltead data not encrypted after I enable `CONFIG_EFUSE_VIRTUAL` and flash encryption?

- Currently, Virtual eFuses is only used to test the update of eFuse data. Thus, flash encryption is not enabled completely even this function is enabled.

4.7.19 Can I update an app firmware which enables flash encryption in a device which does not enable flash encryption through OTA?

- Yes, please deselect Check Flash Encryption enabled on app startup when compiling.

4.7.20 How can I delete keys of secure boot?

- Keys of secure boot should be deleted in the firmware `new_app.bin`. First, please assure that `new_app.bin` is employed with two signatures. Then, flash `new_app.bin` to the device. At last, when the original signatures are verified, you can delete the original keys through `esp_ota_revoke_secure_boot_public_key()` in `new_app.bin`. Please note that if you use the OTA rollback scheme, please call `esp_ota_revoke_secure_boot_public_key()` after `esp_ota_mark_app_valid_cancel_rollback()` returns `ESP_OK`. For more details, please refer to [Key Revocation](#).

4.7.21 After I enabled secure boot or flash encryption (development mode), I cannot flash the new firmware, and an error occurred as `Failed to enter Flash download mode`. How can I solve this issue?

- Generally, the above log indicates that your flash command is incorrect. Please use script `idf.py` to execute `idf.py bootloader` and `idf.py app` to compile `bootloader.bin` and `app.bin`. Then execute the flash command through `idf.py` according to the tips after compiling. If you still cannot flash your firmware, please use `espefuse.py -p PORT summary` to check the eFuse of the current device and check whether the flash download mode is enabled or not.

4.7.22 After I input the command `espefuse.py read_protect_efuse BLOCK3` command in the terminal configured with ESP-IDF to enable the read-protection for Efuse BLOCK3, why is the data of the Efuse BLOCK3 all 0x00 when I input `esp_efuse_read_block()` to read the Efuse BLOCK3?

- After the Efuse BLOCK3 is read protected, it cannot be read anymore.
-

4.7.23 How can I enable secure boot or flash encryption by pre-burning eFuse?

By default, you can enable secure boot or flash encryption by burning firmware with secure boot or flash encryption enabled. In addition, you can also enable secure boot or flash encryption by pre-burning eFuse in the following two methods: - With [flash_download_tool](#), eFuse will be pre-burned automatically if secure boot or flash encryption is enabled. - You can generate the key and burn corresponding eFuse blocks with [espsecure.py](#) and [espefuse.py](#).

4.7.24 After enabling Secure Boot, why can't I flash the new bootloader.bin using the `idf.py build` command?

After enabling Secure Boot, please use the `idf.py bootloader` command to compile the new bootloader.bin. Then, flash the new bootloader.bin using the command `idf.py -p (PORT) bootloader-flash`.

4.7.25 After enabling Secure Boot or flash encryption, how can I view the security-related information in the device?

Please use the command `esptool.py --no-stub get_security_info` to view the security information of the device.

4.7.26 After enabling Secure Boot or flash encryption, what should I pay attention to during OTA (Over-The-Air) updates?

- After enabling Secure Boot, you must sign the new firmware to be used for OTA updates. Otherwise, the new firmware cannot be applied to the device.
- After enabling flash encryption, when generating a new firmware, please ensure that the flash encryption option is enabled.

4.8 Storage

[]

4.8.1 FAT Filesystem

[]

How to improve the damage to FatFs file system caused by accidental power loss?

Since FatFs is designed to not support write transactions, the accidental power loss may cause error to partitions, which cannot be restored by simply modifying FatFs. For now, it is recommended to resolve this problem in application level by creating two identical FatFs partitions to do backups, or you can also choose a more secure file system instead, such as [LittleFS](#) and [SafeFAT](#) (charged).

How to make and flash the image of a FatFs file system?

Here we will use a third-party tool, since there is no such tool provided in ESP-IDF now. The entire process shows as below:

- Step 1: use the [mkfatfs](#) tool to create image in a specified folder. Here we create a 1048576-byte image named `fat_img.bin` in the `file_image` folder.

```
./mkfatfs -c file_image -s 1048576 ./fat_img.bin
```

- Step 2: flash the image to address 0x110000:

```
esptool.py -p /dev/ttyUSB1 -b 460800 --before default_reset --after hard_
↪reset write_flash --flash_mode dio --flash_size detect --flash_freq 80m_
↪0x110000 ~/Desktop/fat_img.bin
```

- Step 3: mount the image in program:

```
static void initialize_filesystem() {
    static wl_handle_t
    wl_handle = WL_INVALID_HANDLE;
    const esp_vfs_fat_mount_config_t
    mount_config = { .max_files = 10, };
    ESP_LOGI(TAG, "Mounting FATfilesystem");
    esp_err_t err = esp_vfs_fat_spiflash_mount("/spiflash", "storage", &mount_
↪config, &wl_handle);
    if (err != ESP_OK) {
        ESP_LOGE(TAG, "Failed to mount FATFS (%s)", esp_err_to_name(err));
        return;
    }
}
```

Note: The address to be flashed in step 2 must be the corresponding partition address in the partition table where FatFs is mounted, and the image created must be the same size as the one set in the partition table. Please remember to

go to menuconfig and set Component config -> Wear Levelling -> Wear Levelling library sector size to 512, or the mounting would fail.

What are the differences between the two file systems, FatFs and SPIFFS, and how do we choose?

Please refer to *File System* <https://github.com/espressif/esp-iot-solution/blob/master/docs/en/storage/file_system.rst>.

What is the maximum size supported by FatFs?

Due to the limitations of the Windows system, FatFs is currently generally only available on storage devices up to 32 GB. Storage devices larger than 32 GB use other file systems, such as exFAT.

I cannot open the files with long names when I use the FAT file system. How can I fix this issue?

You can change the configuration in menuconfig > Component config > FAT Filesystem support > Long filename support by selecting the option Long filename buffer in heap or Long filename buffer on stack. Then you can update the maximum length in Component config > FAT Filesystem support > Max long filename length.

When I used the `ext_flash_fatfs` example to test, I encountered an error `vfs_fat_spiflash : f_mks failed(14), config:Failed to mount FATFS (ESP_FAIL)` if I set the fatfs partition to less than 512 KB. How can I solve it?

- A FAT partition has 128 sectors at the minimum, so the minimum size of the file system should be $128 \times 4 + 4 \times 4 = 528$ KB. The extra four sectors are used for wear leveling information. As a result, the size of the fatfs partition must not be less than 528 KB.

4.8.2 Non-Volatile Storage (NVS)

□

If data needs to be stored or updated to flash every minute, can ESP32 NVS meet this requirement?

According to [NVS Specifications](#), NVS uses two main entities in its operation: pages and entries. Logical page corresponds to one physical sector of flash memory. For now, we assume that flash sector is 4096 bytes and each page can contain 126 entries (32 bytes for one entry), with the left spaces for page header (32 bytes) and entry state bitmap (32 bytes). Typical flash lifetime is 100 k erase cycles. Assuming that the device is expected to run for 10 years, and the data size written to flash is 4 bytes per minute with flash encryption disabled, then the number of flash write operation can be calculated as: $60 \times 24 \times 365 \times 10 = 5256000$. In this way, no more than 42 k of erase cycles ($5256000/126$) will be caused in NVS, which does not exceed 100 k. Therefore, such operation is supported even without the effects of multiple sectors. In actual use, usually there will be multiple sectors given to NVS, and the NVS can distribute erase cycles to different sectors, making the number of erase cycles in each sector necessarily less than 42 k.

Therefore, the NVS of ESP32 can meet such requirement.

Does NVS have wear levelling function?

Yes, NVS (Non-Volatile Storage) has wear leveling functionality. When storing data in flash memory, the limited number of writing and erasing operations to the flash will cause some storage blocks to have shorter service life than others, affecting the overall working life of the memory. To address this issue, NVS uses an erase-write balancing mechanism implemented internally, rather than the wear_levelling component in ESP-IDF, to evenly distribute data across the flash memory blocks, ensuring that each block is used as much as possible to extend the overall working life of the flash memory.

Can NVS sectors be corrupted by accidental power loss during writing?

No, NVS is designed to resist accidental power loss, so it will not be damaged.

Will the configured Wi-Fi SSID and PASSWORD disappear after the ESP series development board is powered on again and need to be reconfigured?

- It will be stored in NVS by default and will not disappear due to power failure. You can also set it through `esp_wifi_set_storage()`, which can be divided into two situations:
 - If you want to save the Wi-Fi SSID and PASSWORD when powered off, you can store the Wi-Fi information in flash by calling `esp_wifi_set_storage(WIFI_STORAGE_FLASH)`.
 - If you want to achieve the operation of not saving the Wi-Fi SSID and PASSWORD when powered off, you can call `esp_wifi_set_storage(WIFI_STORAGE_RAM)` to store the Wi-Fi information in RAM.
-

How to realize that stored user data can be saved after power off, not erased by OTA, and can be re-written or modified?

- Non-volatile storage can meet the above requirements, and only eFuse and flash can be used to store the data. Since the data should be modifiable, only flash is suitable. It is recommended to use NVS or MFG mechanism. For details, please refer to
 - [Manufacturing Utility](#)
 - [NVS Partition Generator Utility](#)

4.8.3 PSRAM

□

When using ESP32 modules, how to check the size of their PSRAM?

For ESP32 modules, the `esp_spiram_get_size()` function in ESP-IDF can be used to obtain the size of a module's PSRAM. This function returns the total size of the PSRAM in bytes and can be used for memory allocation and management.

The following is an example for obtaining the size of PSRAM:

```
size_t psram_size = esp_spiram_get_size();
printf("PSRAM size: %d bytes\n", psram_size);
```

Note that the `esp_spiram_get_size()` function should be called before using the PSRAM to ensure the correct PSRAM size can be obtained. Additionally, PSRAM functionality should be enabled in `make menuconfig`, so that PSRAM can be used and configured. Furthermore, the PSRAM size can also be obtained in the bootloader log.

When ESP32 connected to a PSRAM externally, how to change its clock source?

In menuconfig: menuconfig -> Component config -> ESP32-specific -> SPI RAM config.

When a 8 MB PSRAM mounted on ESP32, why only 4 MB of it is actually mapped?

- Up to 4 MB (0x3F80_0000 ~ 0x3FBF_FFFF) of external RAM can be mapped into data address space, please refer to the specifications of Section 3.1.4 Memory Map in [ESP32 Datasheet](#).
 - For a 8 MB PSRAM, you can access the other 4 MB following example [himem](#).
-

I'm using an ESP32 development board with the official PSRAM chip PSRAM64H embedded. But after replacing another type of PSRAM chip to PSRAM64H, it failed to recognize when I ran an ESP-IDF example and enabled the PSRAM configuration. What is the reason?

- If you need to change the PSRAM chip, please update configuration options in “menuconfig -> Component config -> ESP32-specific -> Support for external, SPI-connected RAM -> SPI RAM config -> Type of SPI RAM chip in use”.
 - If you cannot find the corresponding type options of the new PSRAM chip you are about to use, please add the chip driver manually.
 - It is recommended to use Espressif's official ESP-PSRAM chip for ESP32 series.
-

Why is the following error printed when I download the hello-world example into the ESP32-WROOM-32E module?

```
E (225) psram: PSRAM ID read error: 0xffffffff
E (225) spiram: SPI RAM enabled but initialization failed. Bailing out.
```

The reason for the error is that the PSRAM (Component config>ESP32-specific>Support for external, SPI-connected RAM) setting is enabled in the software, but there is no PSRAM support in the hardware.

Does ESP32 support coexistence between 16 MB External Flash and 8 MB External PSRAM?

Yes, ESP32 supports coexistence between 16 MB External Flash and 8 MB External PSRAM.

4.8.4 SD/SDIO/MMC Driver

□

Is it possible to use ESP8266 together with TF card?

It is not recommended to use ESP8266 together with TF card.

- Although ESP8266 can be connected to TF card in hardware level (communicate through SPI), the chip may run out of memory in different application scenarios due to its limited resources. Thus, it is not recommended to use ESP8266 with TF card.
 - If all you need is a Wi-Fi-only module that can be connected to a TF card, it is recommended to use the [ESP32-S2](#) chip instead.
-

What is the maximum capacity of eMMC supported by ESP32-S3?

2 TB is the limit for SD and eMMC protocols. There is no specific maximum capacity limit for the ESP32-S3. But if your application is built upon a filesystem, the maximum capacity may also be restricted by the filesystem.

Does ESP32 support DDR52, HS200, HS400, and SDR52 modes when it is connected to an eMMC card?

- At the frequency of 40 MHz, ESP32 supports the DDR52 mode and does not support other modes. For details, please refer to [Supported Speed Modes](#).

4.8.5 SPI Flash



What is the requirement for the storage and usage of ESP32 flash?

The external flash can be mapped into CPU instruction space and RO data space simultaneously. ESP32 can support up to 16 MB of external flash.

- When it is mapped into CPU instruction space, up to 11 MB + 248 KB of data can be mapped at a time. If more than 3 MB + 248 KB is mapped at a time, the cache performance may be degraded due to speculative CPU reads.
 - When it is mapped into RO data space, up to 4 MB of data can be mapped at a time, and 8-bit, 16-bit and 32-bit reads are supported.
 - You should specify flash partition table when programming. Specifically, you should divide flash into different partitions, such as app partition, data partition, and OTA partition, and you should specify the size and offset address of each partition.
 - When using flash, you need to pay attention to its service life. As flash can sustain only a limited number of erase operations, you need to plan and manage the usage of flash. For example, you can use wear leveling to extend the service life of flash.
 - It should be noted that writing operations to flash occupy CPU resources, which may influence the system response time. As a result, the writing operation to flash should be avoided as much as possible.
-

What kind of sectors are reserved for customized use in ESP8266 modules?

- For previous versions of SDK rel3.0, besides for bootloader and app bin, the following sectors are reserved at the end of the configured flash: 1 for system information, 1 for OTA information and 1 for RF calibration information.
 - For SDK rel3.0 and later versions, we use partition_table to manage flash. Except for partition_table and bootloader, other bin files are all marked in partition_table.
-

How to read flash data for ESP8266?

- You can use the script tool under ESP8266-RTOS-SDK to read flash data. The whole process is shown as follows:
 - Install python environment and the required packages;
 - Go to ESP8266_RTOS_SDK/components/esptool_py/esptool;
 - Run `python esptool.py --chip esp8266 --port /dev/ttyUSB0 --baud 115200 read_flash 0x0 0x400000 esp8266.bin`. In this command, “esp8266.bin” is a self-defined file, where all flash data read will be stored; “/dev/ttyUSB0” is the serial port number in linux environment, which can be different in other environments and systems.
-

Why do different ESP32 modules have inconsistent flash erase times?

- Different ESP32 modules may have different flash chips or flash controllers, which may affect the erase time. Some models of flash do not have a mechanism to skip empty blocks during erasing, so it takes longer time. Specifically, different flash chips may have different erase time, for example, the erase time of SPI flash and QSPI flash is different. Even the same type of flash chip may have different erase time as they are produced and packaged in different batches. In addition, the design and performance of the flash controller may also affect the erase time. Therefore, erase time varies in ESP32 modules with different flash chips and flash controllers.
-

When the flash SPI mode is set to QIO mode on the ESP32-S3-WROOM-2-32R8V module, the running firmware prints the following error. Why?

```
E (47) qio_mode: Failed to set QIE bit, not enabling QIO mode
```

The ESP32-S3-WROOM-2-32R8V module uses the 32 MB Octal flash and the 8 MB Octal PSRAM. Please enable the settings in the configuration options:

- (Top) > Serial flasher config > [*] Enable Octal Flash > Flash SPI mode (OPI)
 - (Top) > Component config > ESP PSRAM > Support for external, SPI-connected RAM > SPI RAM config > Mode (QUAD/OCT) of SPI RAM chip in use
-

How can I confirm whether ESP-IDF supports a certain flash?

- You can refer to [Optional features for flash](#) to learn more about the flash information supported by ESP-IDF. It should be noted that this document only indicates that the ESP-IDF code has supported these flash, but this flash list is not fully certified by Espressif.
 - For further support on flash selection, please contact [Espressif](#).
-

For the SPI flash connected to ESP32-S3, what is the maximum amount of data that can be written in a single operation?

- Due to hardware limitations, ESP32-S3 allows a maximum of 64 bytes of data per operation.

4.8.6 SPIFFS Filesystem

□

Can SPIFFS partition be encrypted?

CHIP: ESP32, ESP32S2, ESP32S3, ESP32C3

- SPIFFS does not provide native disk encryption. However, as SPIFFS is realized based on flash, the data can be encrypted using flash encryption. Standard encryption libraries such as mbedtls or OpenSSL can be used to encrypt and decrypt files in SPIFFS. When writing files, the data is encrypted first, and then written to SPIFFS. When reading files, data is first read from SPIFFS, and then decrypted using corresponding decryption algorithms.
-

How do I store the keys and certs of ESP32 devices in SPIFFS?

You can generate an SPIFFS image from files and flash it to the corresponding partition. See [SPIFFS Filesystem](#) for details.

Can ESP32 mount a SPIFFS file system partition in the external SPI flash

Yes, this function has been added in ESP-IDF v4.0 and later versions. Please note that when two partitions are mounted to ESP32, it is not permitted for multiple tasks to write files into the same partition at the same time.

4.8.7 Other Storages

□

Can ESP32 use LittleFS file system?

Currently, LittleFS is not included in ESP-IDF, but there is a third-party porting component [esp_littlefs](#) that can be used directly in ESP-IDF. You can use the [mklittlefs](#) tool for the image of LittleFS file system.

How to check the memory usage (e.g., DRAM, IRAM, rodata) of ESP32 chips?

You can check the estimated occupied static storage of ESP32 chips by inputting the instruction `idf.py size-components` under corresponding directories in terminal. You can use [heap_caps_get_info](#) to obtain dynamic applied memory during operation.

What is the available size of RTC RAM in ESP8266 for users?

- The available RTC RAM in ESP8266 for users is '0x200'. Please see descriptions in [esp8266.ld](#).
-

How to enable exFAT?

CHIP: ESP32

- please modify `#define FF_FS_EXFAT 0` as `#define FF_FS_EXFAT 1`, please refer to [ffconf.h](#) for details.
-

Is there a limit to the number of partitions in the partition table of ESP32?

- The length of partition table is 0xC00 bytes (can store up to 95 partition table entries). Please refer to the description in [partition table](#).
-

How to read the remaining memory of the ESP32 chip?

- The remaining memory of the chip RAM can be read through [esp_get_free_heap_size\(\)](#).
-

What should I pay attention to when using `xTaskCreateStatic()` in ESP-IDF?

- Please refer to [xTaskCreateStatic\(\)](#) introduction.

4.9 System



4.9.1 My application does not really need the watchdog timer, can I disable it?

There are two types of watchdog in ESP-IDF: task watchdog and interrupt watchdog. You can disable these two types of watchdog in menuconfig. For more details, please refer to [Watchdogs](#).

4.9.2 What are the differences between RTOS SDK and Non-OS SDK?

The main differences are as follows:

Non-OS SDK

- Non-OS SDK uses timers and callbacks as the main way to perform various functions - nested events and functions triggered by certain conditions. Non-OS SDK uses the espconn network interface; users need to develop their software according to the usage rules of the espconn interface.

RTOS SDK

- FreeRTOS SDK is based on FreeRTOS, a multi-tasking OS. You can use the standard FreeRTOS interfaces to realize resource management, recycling operations, execution delay, inter-task messaging and synchronization, and other task-oriented process design approaches. For the specifics of interface methods, please refer to the official website of FreeRTOS or the book USING THE FreeRTOS REAL TIME KERNEL-A Practical Guide.
 - The network operation interface in RTOS SDK is the standard lwIP API. RTOS SDK provides a package which enables BSD Socket API interface. Users can directly use the socket API to develop software applications; and port other applications from other platforms using socket API to ESP8266, effectively reducing the learning and development cost arising from platform switch.
 - RTOS SDK introduces cJSON library whose functions make it easier to parse JSON packets.
 - RTOS is compatible with Non-OS SDK in Wi-Fi interfaces, SmartConfig interfaces, Sniffer related interfaces, system interfaces, timer interface, FOTA interfaces and peripheral driver interfaces, but does not support the AT implementation.
-

4.9.3 Why does the log output `ets_main.c` when ESP8266 starts?

If ESP8266 prints `ets_main.c` when it starts, it indicates there are no programs that can be operated. Please check the binary file and burn address if you encounter this issue.

4.9.4 Why do I get compile errors when using `IRAM_ATTR` in Non-OS SDK?

The default function attribute is `IRAM_ATTR` in Non-OS SDK. Therefore, if you want the function to reside in IRAM, please leave out the `ICACHE_FLASH_ATTR` attribution in the function definition/declaration.

4.9.5 Where is main function in ESP8266?

- ESP8266 SDK does not provide main function. Main function is stored in first-stage bootloader in ROM, which is used to load second-stage bootloader. The entry function of the second-stage bootloader is `ets_main`. After startup, the `user_init` in the user application will be loaded to lead the user to the program.
-

4.9.6 Which part of ESP8266 partition-tables should be paid special attention to?

Compared to those of ESP32, partition-tables of ESP8266 have some special requirements on OTA partitions due to cache characteristics of ESP8266. For details, please refer to [Offset & Size of ESP8266 Partition Tables](#)

4.9.7 Is it possible to compile the binaries in application layer and bottom layer separately?

No, they cannot be compiled separately.

4.9.8 How can I to reduce the IRAM occupied by the ESP32 system?

- Please disable `menuconfig>Component config>LWIP>Enable LWIP IRAM optimization` by typing N.
 - Please change the configurations in `menuconfig>Compiler option>Optimization Level>Optimize for size (-Os)`.
 - Please disable `WiFi IRAM speed optimization (N)` and `WiFi RX IRAM speed optimization (N)` in `menuconfig>Component config>wifi`.
 - For more details, please refer to [Minimizing RAM Usage](#)
-

4.9.9 How can I optimize the size of binary files compiled by ESP32?

- Please optimize GCC compilation by `idf.py menuconfig > Compiler options > Optimization level (Optimize for size(-Os))`
 - You can also optimize your code to improve the code reusability, and you can also adjust the log level to only print necessary logs.
 - For more details, please refer to [Minimizing Binary Size](#)
-

4.9.10 How can I turn off log output in ESP32

- You can turn off the bootloader log by setting `menuconfig > bootloader config > bootloader log verbosity` to `No output`.
 - You can turn off the program log by setting `menuconfig > Component config > log output > Default log verbosity` to `No output`.
 - For ESP-IDF release/v4.3 and earlier versions, you can turn off UART0 output log by `menuconfig > Component Config > Common ESP-related > Channel for console output > None`.
 - For ESP-IDF release/v4.4 and later versions, you can turn off UART0 output log by `Component config > ESP System Settings > Channel for console output > None`.
-

4.9.11 How to modify the GPIO used for log output on ESP32?

- Go to `menuconfig > Component Config > ESP System Settings > Channel for console output > Custom UART` and select the UART port.
 - Go back to the previous level of menu, find the options *UART TX on GPIO#* and *UART RX on GPIO#*, and use them to modify the log output GPIO.
-

4.9.12 When ESP8266 is in Deep sleep mode, can the data stored in RTC Memory work?

- When ESP8266 is in Deep sleep mode, only the RTC timer continues to work. The data saved in the RTC Memory will not run, but can still be saved here. However, the data saved in RTC memory will lose after ESP8266 is powered off.
-

4.9.13 What is the maximum length of the NVS Key for ESP32?

- The maximum length of the NVS key for ESP32 is 15 characters, which cannot be changed. Please see the description of [key-value pair](#).
 - But you can use the value of `nvs_set_str()` to store data.
-

4.9.14 Does the cJSON in ESP-IDF release/v4.2 support uint64_t data analysis?

- No. The cJSON library has restrictions on parsing long integers, and the longest type is Double.
-

4.9.15 Given that the GDB debugging function is working before the flash encryption is disabled, then why does the device keeps restarting during the GDB debugging after the flash encryption is enabled?

- The JTAG debugging function will be disabled by default when flash encryption or secure boot is enabled. For more information, please refer to [JTAG with Flash Encryption or Secure Boot](#).
-

4.9.16 While using mobile's hotspot for an ESP32 to download the OTA firmware, after a few seconds when turning of the hotspot and restarts ESP32, the program sticks in the OTA operation (the same situation for plugging and unplugging the wan port network cable when using a router), why?

- This is a normal situation based on the protocol. When using the `esp_https_ota` component to run OTA, you can set the network timeout value (via `http_config->timeout_ms`) to 10 ~ 30 s (not too low) and enable `http_config->keep_alive_enable` to see if there are any errors in the link layer.
 - If you are using an self-implemented OTA module, please set a timeout value via the `select` configuration or enable the TCP keep-alive mechanism to detect the link layer.
-

4.9.17 Which GPIOs can be used to wake up ESP32-C3 from Deep-Sleep mode?

- Only GPIO0 ~ GPIO5 in VDD3P3_RTC domain can be used to wake up ESP32-C3 from Deep-sleep mode. Please read Chapter 5.9.1 Power Supplies of GPIO Pins in [ESP32-C3 Technical Reference Manual](#).
-

4.9.18 When using the ESP-WROOM-02D module with a battery for power supply, are there any risks in frequently formatted reading and writing flash as the battery is low (the module barely starts up)?

Frequent formatting and read/write operations on flash storage in low power situations may have some risks. It may not work properly or be susceptible to cause errors under low power conditions. In addition, frequent formatting and read/write operations in this situation may lead to the following risks:

- Data loss or corruption: Flash storage may not be able to write data properly under low power conditions. Frequent formatting and read/write operations may result in data loss or corruption.
- Module crash or damage: Frequent formatting and read/write operations on flash storage in low power conditions will consume the module's power, which may cause the module to crash or damage.

Therefore, it is recommended to minimize access and operations on flash storage in low power conditions and avoid frequent formatting and read/write operations. If formatting and read/write operations are necessary, ensure that the module has sufficient power, backup data before the operation to prevent data loss, use low power mode and optimize code to minimize power consumption.

4.9.19 How to check the maximum stack size used by a thread for ESP32?

- You can call the `UBaseType_t uxTaskGetStackHighWaterMark(TaskHandle_t xTask)` function. This function will return the minimum remaining stack space after the task is started.

4.9.20 What is the meaning of the "SW_CPU_RESET" log when using ESP32?

On ESP32, the "SW_CPU_RESET" log is usually caused by abnormal termination of the program. ESP32 has two cores, the main core and the assistant core. In some cases, if the program is executed on the main core and some abnormal situations occur, such as accessing illegal addresses or unhandled interrupts, it may cause the main core to enter into an exception state and restart. When this happens, ESP32 will print the "SW_CPU_RESET" log on the serial terminal (UART). In addition, when developing applications using ESP-IDF, it is also possible to call the `esp_restart()` function in the application to restart ESP32. In this case, ESP32 will also print the "SW_CPU_RESET" log on the serial terminal. It should be noted that the appearance of the "SW_CPU_RESET" log does not necessarily mean that there is a problem with the program or ESP32 hardware. It may be a normal phenomenon caused by some abnormal situations. However, if the program frequently encounters exceptions and restarts, it is necessary to debug and troubleshoot the problem. You can determine the reason of the problem by checking the program log and hardware device status.

4.9.21 For ESP32 products, when testing NVS separately, I found it occupies a lot of memory. What is the reason?

- Please check the partition table settings. It is recommended to set a smaller NVS data partition in the partition table to test. The larger the NVS data partition setting, the more memory it will occupy.
-

4.9.22 How do I change the system time of a module ?

CHIP: ESP32 | ESP32 | ESP32-C3

- You can use the c language `time()` interface to set the system time.
-

4.9.23 During the OTA upgrade process, an `ESP_ERR_OTA_VALIDATE_FAILED` error occurred after calling `esp_ota_end`, how to troubleshoot such issue?

CHIP: ESP32

- Generally it is caused by the error content in the downloaded firmware. You can dump out such content via `read_flash` in `esptool` from your module. Then use the Beyond Compare tool to compare the two bin files in hexadecimal to see which part of the bin file is downloaded incorrectly.
-

4.9.24 How does ESP8266-RTOS-SDK store data to RTC memory?

- The definition method of storing data in RTC memory is as follows:
 - Please refer to the description in `esp_attr.h`.
-

4.9.25 After waking up from Deep-sleep mode, where does ESP8266 start boot?

- After ESP8266 wakes up from Deep-sleep mode, the device will boots up from `user_init`. Please refer to the description in `esp_deep_sleep()`.
-

4.9.26 When will the RTC clock be reset?

- Any reset (except the power-up reset) or sleep mode settings will not reset the RTC clock.
-

4.9.27 After using the `AT+GSLP` command to enter Deep-sleep mode for ESP32, can it be awakened by pulling down the EN pin?

- Yes, but it is not recommended.
 - Deep-sleep wakeup can be awakened by `RTC_GPIO`. Please refer to [ESP32 Technical Reference Manual](#).
-

4.9.28 When multiple threads want to use the watchdog of ESP32, should each thread enable the watchdog individually?

- Yes, please see [Task watchdog instructions](#).
-

4.9.29 When using the release/v3.3 version of ESP8266-RTOS-SDK, how to enter Light-sleep mode?

- First set the wake-up mode of Light-sleep mode, please refer to [ESP8266_RTOS_SDK/components/esp8266/include/esp_sleep.h](#).
 - Then use the `esp_light_sleep_start()` API to enter Light-sleep mode.
 - You can refer to the [esp-idf/examples/system/light_sleep/main/light_sleep_example_main.c](#) example for implementation logic.
 - Please read [API Reference](#) for API descriptions about sleep modes in ESP8266-RTOS-SDK.
-

4.9.30 How to wake up ESP8266 in Deep sleep mode?

- The ESP8266 can only be awakened from Deep sleep mode via RTC Timer, the timing duration is set by user via `esp_deep_sleep`, and `GPIO16(XPD_DCDC)` should be connected to `EXT_RSTB` through a 0 resistor to support such function. Please refer to [related API descriptions](#).
-

4.9.31 When using the ESP32-WROVER module, there is a problem of battery jitter or abnormal power-off and power-on, causing the system to crash and fail to wake up. What is the reason?

- Application scenario: The current is about 12 uA during sleep. When the battery is unplugged or the product is shaken, it will cause power failure, but there is still electricity in the capacitor. During the process of discharging ESP32 from 3.3 V to 0 V, ESP32 will fail to wake up when powered on again with 3.3 V.
 - Please check whether the chip VCC and EN meet the power-on sequence requirements.
 - Consider adding a reset chip to ensure normal timing.
 - For ESP32 power-on and reset timing description, please refer to [ESP32 Datasheet](#).
-

4.9.32 How to flash a customized mac address?

- You can start by understanding the MAC mechanics of ESP modules, please refer to [Introduction to Mac Addresses](#). There are currently 2 options for burning customized MAC addresses:
 - Option 1: directly flash it into efuse blk3.
 - Option 2: Store in flash. It is not recommended to store the MAC address in the default NVS partition. It is recommended to create a customized NVS partition for storing customized Mac addresses. For more information on the use of customized MAC addresses, please refer to [base_mac_address](#).
-

4.9.33 When ESP32 uses esp_timer, network communication or Bluetooth communication is abnormal. What is the reason?

- esp_timer is a high-precision hardware timer component, and some background components also use it to complete some system tasks. When using esp_timer, please do not call delay and blocking APIs in the callback function of the timer, and try to ensure that the function is executed as quickly as possible, so as not to affect the performance of other system components.
 - If you do not need very high time precision, please use the timer component [xTimer](#) in FreeRTOS.
-

4.9.34 With ESP32, are there any return instructions if I skip to a function using the jump instruction in ULP

Please see [here](#) for ULP CPU instructions list and corresponding specifications. Normally, a general register is used for return instructions to store backup PC addresses for later jumping backs. Since there are only four general registers in ULP for now, please make proper use of them.

4.9.35 How to adjust the warning level for project build?

When building the project, it is found that some warnings being treated as errors, causing build failure, as follows:

```
error: format '%d' expects argument of type 'int *', but argument 3 has type
↳ 'uint32_t *' {aka 'long unsigned int *'} [-Werror=format=]
```

For the error above, you can modify compilation flags at a component level (in component CMakeLists.txt) or at a project level (in project CMakeLists.txt). These two ways have roughly the same effect.

- To modify compilation flags for a specific component, use the standard CMake function `target_compile_options`. Please refer to [Controlling Component Compilation](#). For an example of `target_compile_options` at the component level, please see [CMakeLists.txt#L3](#).
 - To modify compilation flags for the whole project, either use standard CMake function `add_compile_options` or IDF-specific function `idf_build_set_property` to set `COMPILE_OPTIONS` property. Please refer to [overriding-default-build-specifications](#).
-

4.9.36 The firmware compiled based on the ESP-IDF SDK varies as it contains the information about `IDF_PATH` and compilation time. How to remove that information?

- For SDK v5.0 and the above versions, you can enable the `CONFIG_APP_REPRODUCIBLE_BUILD` configuration option. In doing so, the application built upon ESP-IDF does not depend on the build environment and both the .elf file and .bin file of the application remain unchanged even if the following variables change:
 - Directory where the project is located
 - Directory where ESP-IDF is located (`IDF_PATH`)
 - Build time

Please refer to the [Reproducible Builds](#) description.

- For SDK versions below v5.0, you can disable `CONFIG_APP_COMPILE_TIME_DATE=n` to remove the built timestamp information and enable `COMPILER_HIDE_PATHS_MACROS=y` to hide `IDF_PATH`.

4.9.37 When I downloaded the official application `hello_world` using ESP32-S3-DevKitM-1, the following error occurred. What is the reason for that?

```
ESP-ROM:esp32s3-20210327
Build:Mar 27 2021
rst:0x7 (TG0WDT_SYS_RST),boot:0x8 (SPI_FAST_FLASH_BOOT)
Saved PC:0x40043ac8
Invalid chip id. expected 9 read 4. bootloader for wrong chip?
ets_main.c 329
```

- The current error may be related to the chip version on the development board or to the fact that the software version of the esp-idf SDK is not the official production version. The chip (ROM) bootloader expects the chip ID is 9, which is the production version of the chip (not a test version). However, in the secondary bootloader header, it sees the chip ID is 4, which is the beta version of the chip. Please refer to the description in [esp-idf/issues/7960](#).
- The actual version of the chip can be obtained by the command `esptool.py chip_id`. If the chip version is the production version, this error is related to the version of the used esp-idf SDK. For ESP32-S3 series products, esp-idf release/v4.4 and later are necessary.

4.9.38 What is the accuracy of the internal 150 kHz RTC of ESP32 series chips?

- The accuracy of the internal 150 kHz RTC of ESP32 series chips is $\pm 5\%$.

4.9.39 What versions of esp-idf SDK are supported by ESP32-D0WDR2-V3 chip?

- Supported IDF versions are: v4.4.1, v4.3.3, v4.2.3, and v4.1.3.
-

4.9.40 When I test OTA applications based on the ESP32 chip, can I delete the default factory partition in the partition table and set the address of the OTA_0 partition to 0x10000?

- Yes. Please note that the offsets of partitions of any app type have to be aligned to 0x10000 (64K).
-

4.9.41 Why can I not burn data to BLOCK3 of ESP32-C3 eFuse with `espefuse.py burn_key`?

- `espefuse.py burn_key` can only burn data to eFuse blocks of the KEY_DATA type. However, BLOCK3 of ESP32-C3 is of the USR_DATA type by default.
 - You can burn data to eFuse blocks of the USR_DATA type with `espefuse.py burn_block_data`.
-

4.9.42 Why do different ESP32 modules have different flash erase time?

- This is caused by different type of flash models. Some module of flash don't have a mechanism for passing empty blocks when erasing, so it takes longer time.
-

4.9.43 Why I encountered the following error after I run the firmware based on the esp-idf SDK?

```
***ERROR*** A stack overflow in task sys_evt has been detected.
```

- The error is caused by insufficient system_event task stack. You can try to resolve it by increasing Component config>ESP System Setting>Event loop task stack size. However, the overflow occurs because too much logic is being processed within system_event. It is not recommended as it might lead to delayed handling of subsequent events. We suggest forwarding this event to other tasks for processing, either through a queue or other operations.

4.10 Wi-Fi

□

4.10.1 Do ESP32 and ESP8266 support Chinese SSID for Wi-Fi?

Both ESP32 and ESP8266 support Chinese SSID, but you need to use corresponding libraries and implement some settings. It should be noted you need to make special configurations when using Chinese SSID as Chinese characters occupy different numbers of bytes.

For ESP32, you can use the Wi-Fi related API provided by ESP-IDF. When connecting AP, you can use the function `esp_wifi_set_config()` to set Wi-Fi. The SSID parameter can be set to Chinese characters. For example,

```
wifi_config_t wifi_config = {
    .sta = {
        .ssid = "",
        .password = "password123",
    },
};
ESP_ERROR_CHECK(esp_wifi_set_config(ESP_IF_WIFI_STA, &wifi_config));
```

4.10.2 How much time does an ESP32 scan take?

The total time for scanning depends on:

- Active scan (by default) or passive scan.
- The time spent on each channel is 120 ms for active scanning and 360 ms for passive scanning.
- The country code and configured channel range from 1~13 channels (by default).
- Fast scan (by default) or full-channel scan.
- Station mode or Station-AP mode, and if any active connections are currently maintained.

By default, channels 1 to 11 use active scans, and channels 12 to 13 use passive scans.

- In the absence of connection in Station mode, the total time for a full-channel scan is: $11 \times 120 + 2 \times 360 = 2040$ ms.
- With active connections in Station mode or Station-AP mode, the total time for a full-channel scan is: $11 \times 120 + 2 \times 360 + 13 \times 30 = 2430$ ms.

4.10.3 [Scan] Do Espressif's products support boundary scans?

No, they don't.

4.10.4 What is the definition for Wi-Fi channel? Can I select any channel of my choice?

- A Wi-Fi channel is a frequency range used for wireless communication. Different countries and regions have regulations on the available Wi-Fi channels. For example, in North America, the Wi-Fi channel ranges from 1 to 11, while in Europe, the Wi-Fi channel ranges from 1 to 13. For more details, please refer to [ESP8266 Wi-Fi Channel Selection Guidelines](#).
-

4.10.5 [LWIP] With ESP-IDF v4.1, how to configure ESP32's IP address when it is in SoftAP mode?

Since ESP-IDF v4.1 and later versions do not have TCP/IP interfaces anymore, it is recommended to use the [ESP-NETIF](#) interface instead.

Code example

```
{
    ...
    esp_netif_t *ap_netif = esp_netif_create_default_wifi_ap();
    char* ip= "192.168.5.241";
    char* gateway = "192.168.5.1";
    char* netmask = "255.255.255.0";
    esp_netif_ip_info_t info_t;
    memset(&info_t, 0, sizeof(esp_netif_ip_info_t));

    if (ap_netif)
    {
        ESP_ERROR_CHECK(esp_netif_dhcps_stop(ap_netif));
        info_t.ip.addr = esp_ip4addr_aton((const char *)ip);
        info_t.netmask.addr = esp_ip4addr_aton((const char *)netmask);
        info_t.gw.addr = esp_ip4addr_aton((const char *)gateway);
        esp_netif_set_ip_info(ap_netif, &info_t);
        ESP_ERROR_CHECK(esp_netif_dhcps_start(ap_netif));
    }
    ...
}
```

4.10.6 [LWIP] How to configure ESP32's static IP when it is in Station mode

Since ESP-IDF v4.2 and later versions do not have tcp/ip interfaces anymore, it is recommended to use the [ESP-NETIF](#) interface instead. The code example is as follows

```
esp_netif_t *sta_netif = esp_netif_create_default_wifi_sta();
if (sta_netif)
{
    esp_netif_ip_info_t info_t = {0};
    esp_netif_dhcpc_stop(sta_netif);

    info_t.ip.addr = ESP_IP4TOADDR(192, 168, 3, 23);
    info_t.gw.addr = ESP_IP4TOADDR(192, 168, 3, 1);
    info_t.netmask.addr = ESP_IP4TOADDR(255, 255, 255, 0);
}
```

(continues on next page)

(continued from previous page)

```

    esp_netif_set_ip_info(sta_netif, &info_t);
}
esp_netif_dns_info_t dns_info = {0};

```

4.10.7 [LWIP] How to configure the Option contents of DHCP Server in ESP-IDF?

Since ESP-IDF v4.1 and later versions do not have TCP/IP interfaces anymore, it is recommended to use the [ESP-NETIF](#) interface instead. You can also refer to this example when dealing with DHCP Client configuration. The code example is as follows:

```

// Set up the handle for softap netif
esp_netif_t *ap_netif = esp_netif_create_default_wifi_ap();

// ESP_NETIF_IP_ADDRESS_LEASE_TIME, DHCP Option 51, Set the lease time for
↳distributed IP address
uint32_t dhcp_lease_time = 60; // The unit is min
ESP_ERROR_CHECK(esp_netif_dhcp_option(ap_netif, ESP_NETIF_OP_SET, ESP_NETIF_
↳IP_ADDRESS_LEASE_TIME, &dhcp_lease_time, sizeof(dhcp_lease_time)));

// ESP_NETIF_DOMAIN_NAME_SERVER, DHCP Option 6, Set DNS SERVER
// Set the local domain DNS first
esp_netif_dns_info_t dns_info = {0};
dns_info.ip.u_addr.ip4.addr = ESP_IP4TOADDR(8,8,8,8);
ESP_ERROR_CHECK(esp_netif_set_dns_info(ap_netif, ESP_NETIF_DNS_MAIN, &dns_
↳info));

uint8_t dns_offer = 1; // Pass 1 to make the modified DNS take effect, if it
↳is 0, then it means the gw ip of softap is used as the DNS server (0 by
↳default)
ESP_ERROR_CHECK(esp_netif_dhcp_option(ap_netif, ESP_NETIF_OP_SET, ESP_NETIF_
↳DOMAIN_NAME_SERVER, &dns_offer, sizeof(dns_offer)));

// ESP_NETIF_ROUTER_SOLICITATION_ADDRESS, DHCP Option 3 Router, Pass 0 to
↳make the DHCP Option 3(Router) un-shown (1 by default)
uint8_t router_enable = 0;
ESP_ERROR_CHECK(esp_netif_dhcp_option(ap_netif, ESP_NETIF_OP_SET, ESP_NETIF_
↳ROUTER_SOLICITATION_ADDRESS, &router_enable, sizeof(router_enable)));

// ESP_NETIF_SUBNET_MASK, DHCP Option 1, Configure the subnet mask
// If it fails to configure the subnet mask via ESP_NETIF_SUBNET_MASK,
↳please make modifications using esp_netif_set_ip_info

```

4.10.8 [Performance] How to test the bit rate of Wi-Fi modules?

Please use the example `iperf` in ESP-IDF for testing.

4.10.9 [LWIP] What is the default IP address of ESP8266 SoftAP?

Why do I have problem connecting to router with IP 192.168.4.X in SoftAP + Station mode?

- The default network segment used by ESP8266 SoftAP is 192.168.4.*, and its IP address is 192.168.4.1. When connecting ESP8266 to the router of 192.168.4.X, it cannot distinguish whether this address indicates its own SoftAP or the external router.
-

4.10.10 [Connect] How many devices is ESP8266 able to connect in SoftAP mode?

The ESP8266 chip in SoftAP mode supports connecting eight devices at most. This is because the NAT (Network Address Translation) mechanism used by the ESP8266 chip in SoftAP mode only supports eight devices at most. However, it should be noted that each connected device will occupy a certain amount of bandwidth and resources. Therefore, we recommend connecting four devices as too many devices may affect the performance and stability of the Wi-Fi module.

4.10.11 Do ESP8266/ESP32/ESP32-S2/S3/C2/C3 support web/SoftAP provisioning?

Yes.

- For ESP8266, please refer to example `ESP8266 softap_prov`.
 - For ESP32/ESP32-S2/S3/C2/C3, please refer to example `ESP32/ESP32-S2/S3/C2/C3 wifi_prov_mgr`.
-

4.10.12 [Connect] How do ESP8266 and ESP32 hide SSID in SoftAP mode?

To hide ESP8266 or ESP32 as SSID in SoftAP mode, you can use the following methods:

- Use the `esp_wifi_set_config()` function to configure the SSID and password in SoftAP mode and whether to hiding them. For example, the following code sets the SSID to “MySoftAP”, the password to “MyPassword”, and set `.ssid_hidden = 1` to hide the SSID:

```
wifi_config_t config = {
    .ap = {
        .ssid = "MySoftAP",
        .ssid_len = strlen("MySoftAP"),
        .password = "MyPassword",
        .max_connection = 4,
        .authmode = WIFI_AUTH_WPA_WPA2_PSK
        .ssid_hidden = 1
    },
};
esp_wifi_set_config(WIFI_IF_AP, &config);
```

Then use `esp_wifi_start()` function to starts Wi-Fi.

4.10.13 Does the buffer parameter in `esp_wifi_802.11_tx` interface include FCS?

No, the FCS frame is generated automatically by hardware.

4.10.14 What is the supported Wi-Fi frequency band and power meter for ESP-WROOM-32D?

The Wi-Fi frequency band is 2412 ~ 2484 MHz, and the available channels and corresponding operating frequencies can be configured in software. There are default values in power meter, and it can also be configured by software. For detailed guidance, please refer to [ESP32 Phy Init Bin Parameter Configuration Guide](#).

4.10.15 What is the maximum value of ESP32 Wi-Fi RF power

The output RF power of ESP32 can be set to 20 dBm at maximum. Please note that the maximum output power may vary in different countries and regions. Please ensure that you comply with local rules and regulations when using ESP32. In addition, high power output also influence battery life and Wi-Fi signal stability. As a result, you should confirm the output power depending on applications and requirements.

4.10.16 How does ESP32 adjust Wi-Fi TX power?

- Configure Component `config > PHY > Max Wi-Fi TX power (dBm)` via `menuconfig`, and the max value is 20 dBm.
 - Use API `esp_err_t esp_wifi_set_max_tx_power(int8_t power);`.
-

4.10.17 [Connect] How many devices is ESP32 able to connect in AP mode?

Up to 10 devices in AP mode. It is configured to support four devices by default.

4.10.18 [Connect] How do Wi-Fi modules rank signal strength levels based on RSSI values

We do not have a rating for RSSI signal strength. You can take the calculation method from Android system for reference if you need a standard for classification.

```
@UnsupportedAppUsage
private static final int MIN_RSSI = -100;

/** Anything better than or equal to this will show the max bars. */
@UnsupportedAppUsage
private static final int MAX_RSSI = -55;

public static int calculateSignalLevel(int rssi, int numLevels) {
    if(rssi <= MIN_RSSI) {
        return 0;
    } else if (rssi >= MAX_RSSI) {
        return numLevels - 1;
    } else {
        float inputRange = (MAX_RSSI - MIN_RSSI);
        float outputRange = (numLevels - 1);
        return (int)((float)(rssi - MIN_RSSI) * outputRange / inputRange);
    }
}
```

4.10.19 [Connect] Why does ESP32 disconnect from STA when it is in Soft-AP mode?

- By default, the ESP32 will disconnect from the connected STA if it doesn't receive any data from this STA for continuous 5 minutes. This time can be modified via API [esp_wifi_set_inactive_time](#).
 - Note: `esp_wifi_set_inactive_time` is a newly added API.
 - master commit: 63b566eb27da187c13f9b6ef707ab3315da24c9d
 - 4.2 commit: d0dae5426380f771b0e192d8ccb051ce5308485e
 - 4.1 commit: 445635fe45b7205497ad81289c5a808156a43539
 - 4.0 commit: 0a8abf6ffececa37538f7293063dc0b50c72082a
 - 3.3 commit: 908938bc3cd917edec2ed37a709a153182d511da
-

4.10.20 [Connect] While ESP32 connecting Wi-Fi, how can I determine the reason of failure by error codes?

For ESP-IDF v4.0 and later versions, please refer to the following codes to get the reason

```
if (event_base == WIFI_EVENT && event_id == WIFI_EVENT_STA_DISCONNECTED) {
    wifi_event_sta_disconnected_t *sta_disconnect_evt = (wifi_event_sta_
    ↪ disconnected_t*)event_data;
    ESP_LOGI(TAG, "wifi disconnect reason:%d", sta_disconnect_evt->reason);
    esp_wifi_connect();
    xEventGroupClearBits(s_wifi_event_group, CONNECTED_BIT);
}
```

When the callback function received `WIFI_EVENT_STA_DISCONNECTED` event, you can get the reason through the `reason` variable from `wifi_event_sta_disconnected_t`.

- `WIFI_REASON_AUTH_EXPIRE`: This code is returned during the auth phase when the STA sends an auth but do not received any auth reply from the AP within the specified time. The possibility of this code occurrence is low.
- `WIFI_REASON_AUTH_LEAVE`: This code is sent by AP, normally because the AP is disconnected from the STA for some reason.
- `WIFI_REASON_4WAY_HANDSHAKE_TIMEOUT` or `WIFI_REASON_HANDSHAKE_TIMEOUT`: Wrong password.

`WIFI_REASON_4WAY_HANDSHAKE_TIMEOUT` is the standard generalized error code, while `WIFI_REASON_HANDSHAKE_TIMEOUT` is a customized error code. The main difference is: `WIFI_REASON_4WAY_HANDSHAKE_TIMEOUT` occurs when the router tells the device the password is wrong; `WIFI_REASON_HANDSHAKE_TIMEOUT` occurs when the device itself performs a timeout mechanism without being informed about the wrong password by the router.

- `WIFI_REASON_CONNECTION_FAIL`: This code is returned during the scan phase when the STA scanned a matched AP while the AP is in the blacklist. This is probably because that the AP has actively disconnected from the STA last time or something wrong happened when the STA connecting the AP.

4.10.21 Does ESP32 perform domain name resolution each time it connects to the server?

The domain name is resolved via DNS within the stack, and the resolved data will be cached within the specified time. The cache time is based on the TTL data obtained from the DNS server, which is a parameter filled when configuring the domain name, usually 10 minutes.

4.10.22 [Connect] What does the number after the state machine switch in Wi-Fi log mean?

eg: run -> init (fc0), fc0 means the STA has received the deauth frame and reason is password error.

- c0 indicates the received frame type (00 indicates a timeout)
- f indicates reason

Frame type: [a0 disassoc], [b0 auth], [c0 deauth].

4.10.23 [Connect] What does `bcn_timeout`, `ap_probe_send_start` mean

The STA does not receive the Beacon frame within the specified time (6 s by default for ESP32, equals to 60 Beacon Intervals). - The reason could be:

- Insufficient memory. “ESP32_WIFI_MGMT_SBUF_NUM” is not enough (there will be errors like “esf_buf: t=8, l=beacon_len, ...” in the log). You can check this by typing the heap size when received a Disconnect event.
- The AP did not send a beacon. This can be checked by capturing beacons from AP.

- Rssi too low. When the Rssi value is too low in complex environments, the STA may not receive the beacon. This can be checked by retrieving Rssi values via `esp_wifi_sta_get_ap_info`.
- Hardware related issues. Bad package capturing performance.

When there is a `bcn_timeout`, the STA will try to send Probe Request for five times. If a Probe Response is received from the AP, the connection will be kept, otherwise, the STA will send a Disconnect event and the connection will fail.

4.10.24 [Connect] How to reconnect Wi-Fi after it disconnected?

Call `esp_wifi_connect` after received the `WIFI_EVENT_STA_DISCONNECTED` event.

4.10.25 [Connect] When does ESP32 disconnect from SoftAP in station mode

By default, the ESP32 will disconnect from the AP if it does not receive any beacon for 6 s. This time can be modified via `esp_wifi_set_inactive_time`.

4.10.26 [Scan] Why does the STA cannot find any AP sometimes during the scanning?

There are many possible reasons why ESP32 and ESP8266 cannot scan any AP, and some common reasons and solutions are listed below.

- The AP is too far away or the signal is too weak, while Wi-Fi of ESP32 and ESP8266 can only work within a certain range. If the AP is too far away or the Wi-Fi signal is too weak, ESP32 and ESP8266 may not be able to scan the AP. You can move ESP32 or ESP8266 closer to the AP or use a signal amplifier to enhance the signal strength.
 - SSID of the AP is hidden. SSID of some APs may be hidden, so they will not be broadcasted to nearby devices. In this case, ESP32 and ESP8266 cannot scan these APs. You can connect these APs by inputting their SSID and passwords manually.
 - The AP is overloaded or malfunctioning. If the AP is overloaded or malfunctioning, it may not be able to handle new connection requests, and thus ESP32 and ESP8266 cannot connect to the AP. You can try to wait for a while and then scan the AP again.
 - ESP32 or ESP8266 has some software issues. Sometimes, software issues of ESP32 or ESP8266 may cause problems with scanning for APs. In this case, you can try to reset ESP32 or ESP8266 and restart Wi-Fi. If this method does not work, you may need to update the firmware of ESP32 or ESP8266.
 - Other reasons include wireless interference, security settings, and network configuration. These reasons may also affect Wi-Fi of ESP32 or ESP8266. In this case, you need to carefully check the Wi-Fi environment and implement corresponding settings.
-

4.10.27 [Scan] What is the maximum number of APs that can be scanned

There is no limit to the maximum number of APs that can be scanned. The number depends on how many APs are around and configurations of the scanning parameters, such as the time spent on each channel, the longer time spent on each channel the more likely it is to find all the APs.

4.10.28 [Scan] Can I choose to connect the best AP when there are multiple APs with identical ssid/password during the scan

By default, the scan type is WIFI_FAST_SCAN, which makes the STA always connects the first AP during the scan. If you expect to connect the best AP, please set `scan_method` to WIFI_ALL_CHANNEL_SCAN and configure `sort_method` to determine whether to choose the AP with the strongest RSSI or connect to the most secure AP.

4.10.29 [Scan] How to configure `scan_method` in the `wifi_sta_config_t` structure? What is the difference between `all_channel_scan` and `fast_scan`?

`all_channel_scan` and `fast_scan` are used to find the appropriate AP before connecting. The `scan_method` is set to `fast_scan` by default, which is mainly used together with `threshold` to filter APs with weak signal or encryption.

- When `fast_scan` is set, the STA will stop scanning once it finds the first matched AP and then connect to it, so as to save time for connection.
 - When `all_channel_scan` is set, the STA will scan all channels and store four APs with the best signal or the most secure encryption according to the sorting method configured in `sort_method`. After the scan is completed, the STA will connect one of the four APs with the best signal or the most secure encryption.
-

4.10.30 [LWIP] How to get error code of the socket?

- For ESP-IDF v4.0 and later versions: use the value of `errno` directly to get the error code after the socket API returns failure.
 - For previous versions of ESP-IDF v4.0: call `getsockopt(sockfd, SOL_SOCKET, SO_ERROR, ...)` to get the error code after the socket API returns failure, otherwise you may get wrong error code when multiple sockets operate simultaneously.
-

4.10.31 [LWIP] What is the default keep-alive time of TCP?

By default, a TCP keep-alive message will be sent every 75 seconds for 9 times if no TCP message is received for two consecutive hours. Then, if there is still no message received, the LWIP will disconnect from the TCP.

The keep-alive time can be configured via socket option.

4.10.32 [LWIP] What is the retransmission interval of TCP

When ESP32 acts as the sender, the initial retransmission interval of the TCP protocol is usually set to 3 seconds. If the receiver does not send an ACK message, the next retransmission interval will be determined based on the Jacobson algorithm. The retransmission interval will be exponentially increased. Specially, it will be increased by 2, 4, 8, 16, 32 seconds gradually. This retransmission interval is not fixed and you can adjust it by changing some parameters such as timeout time and the size of sliding window.

4.10.33 [LWIP] What is the maximum number of sockets that can be created?

32 for most, and the default number is 10.

4.10.34 [Sleep] What kinds of sleeping mode does ESP32 have? What are the differences?

- There are mainly three sleeping modes: Modem sleep, Light sleep and Deep sleep.
 - Modem sleep: the station WMM sleeping mode specified in the Wi-Fi protocol (the station sends NULL data frame to tell the AP to sleep or wake up). The Modem sleep mode is enabled automatically after the station connected to AP. After entering this mode, the RF block is disabled and the station stays connected with the AP. The Modem sleep mode will be disabled after the station disconnected from the AP. The ESP32 can also be configured to decrease the CPU's clock frequency after entering Modem sleep mode to further reduce its current.
 - Light sleep: this is a station sleep mode based on Modem sleep mode. The difference between is that, besides for the RF block being disabled, the CPU will also be suspended in this mode. After exiting from Light sleep mode, the CPU continues to operate from where it stopped.
 - Deep sleep: a sleeping mode un-specified in the Wi-Fi protocol. During Deep sleep mode, all the blocks except for RTC is disabled, and the station cannot be connected to AP. After exiting from this mode, the whole system will restart to operate (similar to system restart).
-

4.10.35 [Sleep] How can I enable the dynamic frequency scaling function for ESP32 in Modem-sleep mode?

You can enable this function by selecting `menuconfig > Component Config > Power Management > Support for power management > Enable dynamic frequency scaling (DFS) at startup (NEW)`.

4.10.36 [Sleep] How low can the speedstep function go for ESP32 in modem sleep mode

For now, the CPU clock can go down to as low as 40 MHz.

4.10.37 [Sleep] What affects the average current of ESP32 in modem sleep mode?

The modem sleep mode of ESP32 is achieved by setting wakeup cycles. ESP32 opens RF functions to communicate during every cycle and closes these functions in the rest of the time.

The average current of this mode is influenced by several factors, including:

- Wakeup cycle: A shorter wakeup cycle means the chip will be waked up more frequently, which increases the average current.
 - Signal quality: If the Wi-Fi signal is weak, the chip will keep trying to reconnect or send data, which will increase the average current. Using communication protocols with bigger transmission power will also increase the average current.
 - Hardware configuration: The hardware configuration of the chip also affects power consumption, such as the number of CPU cores, CPU clock frequency, CPU idle time ratio, power supply voltage, external crystal oscillator, etc. All of these factors influence the size of average current.
 - Other factors include whether the testing router accurately sends beacon timestamps, whether too many broadcast packets have been sent, whether peripheral modules are working, etc.
-

4.10.38 [Sleep] Why the average current measured in modem sleep mode is a bit high?

- A lot of Wi-Fi data sent and received during the test. The more data there is, the less chance there will be for entering sleeping mode and the higher average current will be.
 - The time when the router sends out beacon is not accurate. The station needs to wake up and monitor the beacon regularly, thus it will wait longer if the beacon time is not accurate. In this way, the station has less time in sleeping mode and the average current will be high.
 - There are peripheral modules working during the test. Please close them before the test.
 - The station+SoftAP mode is enabled. During modem sleep state, the current will only be lower in station-only mode.
-

4.10.39 [Sleep] Why the average current measured in light sleep mode is a bit high?

Besides for the reasons listed in the last question, the possible reasons also could be:

- The application layer code is running continuously, thus the CPU does not get chance to suspend.
 - The application layer has enabled ets timer or esp timer and the timeout interval is short, thus the CPU does not get chance to suspend.
-

4.10.40 [Sleep] What kinds of power-saving modes does ESP32 have? What are the differences?

There are mainly three modes: minimum modem power-saving, maximum modem power-saving, and no power save modes.

- Minimum modem: default type. In this mode, the station wakes up to receive beacon every DTIM period, which is equal to $(DTIM * 102.4)$ ms. For example, if the DTIM of the router is 1, the station will wake up every 100 ms.
 - Maximum modem: in this mode, the interval to receive beacons is determined by the `listen_interval` parameter in `wifi_sta_config_t`. The interval is equal to $(listen_interval * 102.4)$ ms. For example, if the DTIM of the router is 1, and the listen interval is 10, the station will wake up every 1 s.
 - No power save: no power save.
-

4.10.41 Does ESP8266 support 802.11k/v/r protocol?

For now, the ESP8266 only supports 802.11k and 802.11v, please refer to example [roaming](#).

4.10.42 Does ESP32 Wi-Fi support roaming between different APs with the same SSID?

Yes, currently it supports 802.11k and 802.11v protocols. Please refer to the example [roaming](#).

4.10.43 [Connect] After the NONOS_SDK updated from version 2.1.0 to 2.2.2, why does the connecting time become longer

Please update to version *master*, which has solved the incompatibility issue between the CCMP encryption and some APs.

4.10.44 How does ESP32 receive and transmit Wi-Fi 802.11 packets?

- By using the following APIs:

```
esp_err_t esp_wifi_80211_tx(wifi_interface_t ifx, const void *buffer, int_
↪len, bool en_sys_seq);
esp_wifi_set_promiscuous_rx_cb(wifi_sniffer_cb);
```

- The abovementioned APIs are also used in the ESP-MDF project, please refer to [mconfig_chain](#).

4.10.45 [Connect] The ESP32 and ESP8266 failed to connect to router, what could be the reasons

- Please check if the SSID or password is wrong.
- There could be errors in different Chinese codes, so it is not recommended to use an SSID written in Chinese.
- The settings of bssid_set. If the MAC address of the router does not need to be identified, the station-Conf.bssid_set should be configured to 0.
- It is recommended to define the wifi_config field in wifi_config_t using the static variable *static*.

4.10.46 [Connect] What kind of networking methods does ESP8266 have

- SmartConfig mode: using SmartConfig. The device scans feature pack in sniffer mode.
- SoftAP mode: the device enables SoftAP and sends SSID and password after the phone connects to SoftAP and set up a stable TCP/UDP connection.
- WPS mode: an additional button should be added on the device; or using the phone to enable WPS after it connected to SoftAP.

4.10.47 [Connect] What are the specifications of Wi-Fi parameters when using SmartConfig?

SmartConfig is a method of configuring Wi-Fi parameters via local network broadcasting. Users can send Wi-Fi account and password to the device through a matching APP. SmartConfig network configuration has some requirements on Wi-Fi parameters:

- SSID name: Supports Chinese characters and English letters, digits, with a maximum length of 32 bytes.
- Wi-Fi password: 8-64 digits, case-sensitive.
- Wi-Fi security encryption: Currently, SmartConfig supports WPA, WPA2, and WEP encryption methods, and does not support open methods without encryption.

4.10.48 [Connect] Does ESP8266 Wi-Fi support WPA2 enterprise-level encryption

- Yes. Please refer to example [wpa2_enterprise](#).
 - To build RADIUS server, you can use [FreeRADIUS](#).
-

4.10.49 [Connect] What are the low-power modes for ESP32 to maintain its connection to Wi-Fi?

- In such scenarios, the chip switches between Active mode and Modem-sleep mode automatically, making the power consumption also varies in these two modes.
 - The ESP32 supports Wi-Fi keep-alive in Light-sleep mode, and the auto wakeup interval is determined by the DTIM parameter.
 - Please find examples in ESP-IDF - > examples - > wifi - > power_save.
-

4.10.50 Do Espressif's chips support WPA3?

- ESP32 series: WPA3 is supported from esp-idf release/v4.1 and enabled by default. Go to menuconfig > Component config > Wi-Fi for configuration.
 - ESP8266: WPA3 is supported from the release/v3.4 branch of ESP8266_RTOS_SDK and enabled by default. Go to menuconfig > Component config > Wi-Fi for configuration.
-

4.10.51 [Connect] How does the device choose AP when there are multiple identical SSIDs in the current environment?

- The device connects to the first scanned AP.
 - If you expect to sort APs by signal quality and etc., use the scan function to filter manually.
 - If you expect to connect to a specified AP, add BSSID information in connection parameters.
-

4.10.52 [Connect] Does ESP8266 have repeater solutions?

- We have not officially released such application solutions yet.
 - For relay related applications, please find on github. The relay rates should be set basing on real tests.
-

4.10.53 What is the retransmission time for ESP32's data frame and management frameCan this be configured

The retransmission time is 31 and it can not be configured.

4.10.54 How to customize the hostname for ESP32

- Taking ESP-IDF V4.2 as an example, you can go to menuconfig > Component Config > LWIP > Local netif hostname, and type in the customized hostname.
 - There may be a slight difference on naming in different versions.
-

4.10.55 How to obtain 802.11 Wi-Fi packets

- Please refer to [Wireshark User Guide](#) in ESP-IDF Programming Guide.
 - Please note that the wireless network interface controller (WNIC) that you use should support the Monitor mode.
-

4.10.56 Does ESP32 Wi-Fi support PMF (Protected Management Frames) and PFS (Perfect Forward Secrecy)

The PMF is supported both in WPA2 and WPA3, and PFS is supported in WPA3.

4.10.57 Why does ESP8266 print out an AES PN error log when using esptouch v2?

- This occurs when ESP8266 has received retransmitted packets from the router for multiple times. However, this will not affect your usage.
-

4.10.58 Does ESP32 WFA certification support multicast?

- No. It is recommended to refer to the ASD-1148 method of testing.
-

4.10.59 When using ESP32 to establish a hotspot, can I scan all APs and the occupied channels first, and then select the smallest and cleanest channel to establish my own AP?

- You can scan all APs and occupied channels before establishing a hotspot. Refer to the API `esp_wifi_scan_get_ap_records`.
 - It cannot be performed automatically. You need to customize the channel selection algorithm to implement such operation.
-

4.10.60 I'm scanning Wi-Fi on an ESP32 device using release/v3.3 version of ESP-IDF. When there are some identical SSIDs, same SSID names will show in the Wi-Fi list repeatedly. Is there an API to filter such repeated names?

- No, same SSID names cannot be filtered out since identical SSID names may not mean identical servers. Their BSSID may not be the same.
-

4.10.61 Does ESP8266 support EDCF (AC) scheme?

The master version of ESP8266-RTOS-SDK supports EDCF (AC) applications, but no application examples are provided for now. You can enable Wi-Fi QoS configuration in `menuconfig -> Component config -> Wi-Fi` to get support.

4.10.62 I'm using the master version of ESP8266-RTOS-SDK to open the WiFi Qos application to get EDCF support. How does ESP8266 decide which data packet should be allocated to the EDCF AC category?

- It can be determined by setting `IPH_TOS_SET(iphdr, tos)`.
-

4.10.63 Using ESP32, how to configure the maximum Wi-Fi transmission speed and stability without considering memory and power consumption?

- To configure the maximum Wi-Fi transmission speed and stability, please refer to [How to improve Wi-Fi performance](#) in ESP-IDF programming guide and set the relevant configuration parameters in `menuconfig`. The option path can be found by searching “/” in the `menuconfig` interface. The optimal configuration parameters need to be tested according to the actual environment.
-

4.10.64 In Wi-Fi SoftAP mode, how many Station devices can ESP8266 be connected at most?

- ESP8266 supports up to 8 Station device connections.
-

4.10.65 How to get CSI data when using ESP32 device in Station mode?

- CSI data can be obtained by calling `'esp_wifi_set_csi_rx_cb()'`. See description in [API](#).
 - For specific steps, please refer to [Espressif CSI examples](#).
-

4.10.66 In AP + STA mode, after an ESP32 is connected to Wi-Fi, will the Wi-Fi connection be affected if I enable or disable its AP mode?

- After an ESP32 is connected to Wi-Fi in AP + STA dual mode, AP mode can be enabled or disabled at will without affecting Wi-Fi connection.

4.10.67 I'm using ESP-IDF release/v3.3 for ESP32 development, but only bluetooth function is needed, how to disable Wi-Fi function through software?

- Please call `esp_wifi_stop()` to disable the Wi-Fi function. For API description, please see `esp_err_t esp_wifi_stop(void)`.
- If you need to reclaim the resources occupied by Wi-Fi, call `esp_wifi_deinit()`. For API description, please see `esp_err_t esp_wifi_deinit(void)`.
- Here is a simple example:

```
#include "esp_wifi.h"
#include "esp_bt.h"

void app_main()
{
    // Turn off Wi-Fi functionality
    esp_wifi_stop();

    // Initializing Bluetooth functionality
    esp_bt_controller_config_t bt_cfg = BT_CONTROLLER_INIT_CONFIG_DEFAULT();
    esp_bt_controller_init(&bt_cfg);
    esp_bt_controller_enable(ESP_BT_MODE_BTDM);

    // ...
}
```

In this example, the `esp_wifi_stop()` function is called to turn off Wi-Fi and then to initialize Bluetooth. It should be noted that once Wi-Fi is stopped, Wi-Fi related APIs cannot be used.

4.10.68 In ESP-IDF, the `esp_wifi_80211_tx()` interface can only be used to send data packets, is there a corresponding function to receive packets?

- Please use callback function to received data packets as follows:

```
esp_wifi_set_promiscuous_rx_cb(wifi_sniffer_cb);
esp_wifi_set_promiscuous(true);
```

- The above data receive method is also used in another open-sourced project, please see [esp-mdf](#).

4.10.69 What are the reasons for the high failure rate of esptouch networking?

CHIP: ESP32, ESP32S2, ESP32S3, ESP32C3, ESP8266

- The same hotspot is connected too many people.
 - The signal quality of the hotspot connected by cell phone is poor.
 - The router does not forward multicast data.
 - The router has enabled dual-band integration, and the phone is connected to the 5G frequency band.
-

4.10.70 How to optimize the IRAM when ESP32 uses Wi-Fi?

- You can disable `WIFI_IRAM_OPT`, `WIFI_RX_IRAM_OPT` and `LWIP_IRAM_OPTIMIZATION` in menu-config to optimize IRAM space, but this will degrade Wi-Fi performance.
-

4.10.71 How to test ESP32's Wi-Fi transmission distance?

- You can use the [iperf example](#) and configure the ESP32 device to iperf UDP mode. Then, you can distance the device continuously to see at which point the Wi-Fi data transmission rate will drop to 0.
-

4.10.72 What is the maximum length of Wi-Fi MTU for an ESP32 and how to change it?

- The maximum Wi-Fi MTU length for ESP32 is 1500. You can change this value in the LwIP component by `netif>mtu`. However, it is not recommended to change this value.
-

4.10.73 During the on-hook test for an ESP32 device, the following log shows. What does it mean?

log

```
[21-01-27_14:53:56]I (81447377) wifi:new:<7,0>, old:<7,2>, ap:<255,255>, sta:
↪<7,0>, prof:1
[21-01-27_14:53:57]I (81448397) wifi:new:<7,2>, old:<7,0>, ap:<255,255>, sta:
↪<7,2>, prof:1
[21-01-27_14:53:58]I (81449417) wifi:new:<7,0>, old:<7,2>, ap:<255,255>, sta:
↪<7,0>, prof:1
[21-01-27_14:53:59]I (81450337) wifi:new:<7,2>, old:<7,0>, ap:<255,255>, sta:
↪<7,2>, prof:1
```

- The value after new represents the current primary and secondary channel; the value after old represents the last primary and secondary channel; and the value after ap represents the primary and secondary channel of the current ESP32 AP, which will be 255 if softAP is not enabled; the value after sta represents primary and secondary channel of the current ESP32 sta; and prof is the channel of ESP32's softAP stored in NVS.

- For the meaning of secondary channel values, please refer to [wifi_second_chan_t](#).
- The above log indicates that router is switching between HT20 and HT40 minus. You can check the Wi-Fi bandwidth setting of the router.

4.10.74 How to disable AP mode when ESP32 is in AP + STA mode?

- This can be done through the configuration of `esp_wifi_set_mode(wifi_mode_t mode);` function.
- Just call `esp_wifi_set_mode(WIFI_MODE_STA);`.

4.10.75 After ESP32 used the Wi-Fi function, are all ADC2 channels unavailable?

- When an ESP32 device is using Wi-Fi function, the ADC2 pins that are not occupied by Wi-Fi can be used as normal GPIOs. You can refer to the official [ADC Description](#).

4.10.76 How do I set the country code for a Wi-Fi module ?

CHIP: ESP8266 | ESP32 | ESP32 | ESP32-C3

- Please call `esp_wifi_set_country` to set the country code.

4.10.77 When using ESP32 as a SoftAP and have it connected to an Iphone, a warning prompts as “low security WPA/WPA2(TKIP) is not secure. If this is your wireless LAN, please configure the router to use WPA2(AES) or WPA3 security type”, how to solve it?

IDF: release/v4.0 and above

- You can refer to the following code snippet:

```
wifi_config_t wifi_config = {
    .ap = {
        .ssid = EXAMPLE_ESP_WIFI_SSID,
        .ssid_len = strlen(EXAMPLE_ESP_WIFI_SSID),
        .channel = EXAMPLE_ESP_WIFI_CHANNEL,
        .password = EXAMPLE_ESP_WIFI_PASS,
        .max_connection = EXAMPLE_MAX_STA_CONN,
        .authmode = WIFI_AUTH_WPA2_PSK,
        .pairwise_cipher = WIFI_CIPHER_TYPE_CCMP
    },
};
```

- WIFI_AUTH_WPA2_PSK is AES, also called CCMP. WIFI_AUTH_WPA_PSK is TKIP. WIFI_AUTH_WPA_WPA2_PSK is TKIP+CCMP.

4.10.78 Since ESP32's Wi-Fi module only supports 2.4 GHz of bandwidth, can Wi-Fi networking succeed when using a multi-frequency router with both 2.4 GHz and 5 GHz of bandwidth

- Please set your router to multi-frequency mode (can support 2.4 GHz and 5 GHz for one Wi-Fi account), and the ESP32 device can connect to Wi-Fi normally.
-

4.10.79 How to obtain the RSSI of the station connected when ESP32 is used in AP mode?

- You can call API `esp_wifi_ap_get_sta_list`, please refer to the following code snippet:

```
{
    wifi_sta_list_t wifi_sta_list;
    esp_wifi_ap_get_sta_list(&wifi_sta_list);
    for (int i = 0; i < wifi_sta_list.num; i++) {
        printf("mac address: %02x:%02x:%02x:%02x:%02x:%02x\t rssi:%d\n",wifi_sta_
→list.sta[i].mac[0], wifi_sta_list.sta[i].mac[1],wifi_sta_list.sta[i].mac[2],
        wifi_sta_list.sta[i].mac[3],wifi_sta_list.sta[i].mac[4],wifi_
→sta_list.sta[i].mac[5],wifi_sta_list.sta[i].rssi);
    }
}
```

- The RSSI obtained by `esp_wifi_ap_get_sta_list` is the average value over a period of time, not real-time RSSI. The previous RSSI has a weight of 13, and the new RSSI has a weight of 3. The RSSI is updated when it is or larger than 100ms, the old `rssi_arg` is used when updating as: `rssi_avg = rssi_avg*13/16 + new_rssi * 3/16`.
-

4.10.80 Does ESP32 support FTM(Fine Timing Measurement)?

- No, it doesn't. FTM needs hardware support, but ESP32 doesn't have it.
 - ESP32-S2 and ESP32-C3 can support FTM in hardware.
 - ESP-IDF can support FTM from v4.3-beta1.
 - For more information and examples of FTM, please refer to [FTM](#).
-

4.10.81 When ESP32 is in STA+AP mode, how to specify whether using STA or AP interface to send data?

Background:

The default network segment of ESP32 as AP is 192.168.4.x, and the network segment of the router to which ESP32 as STA is connected is also 192.168.4.x. The PC connects to the same router and creates a tcp server. In this case, the tcp connection between ESP32 as tcp client and PC as tcp server cannot be established successfully.

Solutions:

- It is possible for ESP32 to specify whether to use STA or AP interface for data transmission. Please see example [tcp_client_multi_net](#), in which both ethernet and station interface are used and each can be specified for data transmission.
- There are two ways to bind socket to an interface:
 - use netif name (use socket option SO_BINDTODEVICE)
 - use netif local IP address (get IP address of an interface via `esp_netif_get_ip_info()`, then call `bind()`)

Note:

- The tcp connection between ESP32 and PC can be established when an ESP32 is bound to the STA interface, while the connection cannot be established when it is bound to the AP interface.
- By default, the tcp connection between ESP32 and mobile phone can be established (the mobile phone as a station is connected to ESP32).

4.10.82 ESP8266 wpa2_enterprise How to enable Wi-Fi debugging function?

- Open menuconfig via `idf.py menuconfig` and configure the following parameters:

```
menuconfig==>Component config ==>Wi-Fi ==>
[*]Enable WiFi debug log ==>The DEBUG level is enabled (Verbose)
[*]WiFi debug log submodule
[*] scan
[*] NET80211
[*] wpa
[*] wpa2_enterprise

menuconfig==>Component config ==>Supplicant ==>
[*] Print debug messages from WPA Supplicant
```

4.10.83 Is there a standard for the number of Wi-Fi signal frames?

CHIP: ESP8266 | ESP32 | ESP32 | ESP32-C3

- There is no such standard for now. You can do the calculation by yourself based on the received RSSI. For example, if the received RSSI range is [0,-96], and the required signal strength is 5, then [0~-20] is the full signal, and so on.

4.10.84 What is the current progress of WFA bugs fixing?

CHIP: ESP32 | ESP32-S2 | ESP32-C3 | ESP8266

- Please refer to `Security Advisory for WFA vulnerability` <https://www.espressif.com/sites/default/files/advisory_downloads/AR2021-003%20Security%20Advisory%20for%20WFA%20vulnerability.pdf>`_ for more details.
-

4.10.85 When Wi-Fi connection failed, what does the error code mean?

CHIP: ESP32

- Any error occurred during the Wi-Fi connection will cause it coming to init status, and there will be a hexadecimal number in the log, e.g., `wifi:state, auth-> init(200)`. The first two digits indicate error reasons while the last two digits indicate the type code of the received or transmitted management frame. Common frame type codes are 00 (received nothing, timeout), A0 (disassoc), B0 (auth) and C0 (deauth).
 - Error reasons indicated by the first two digits can be found in [Wi-Fi Reason Code](#). The last two digits can be checked in frame management code directly.
-

4.10.86 When using ESP32's Release/v3.3 of SDK to download the Station example, the device cannot be connected to an unencrypted Wi-Fi. What is the reason?

- In the example, it is by default to connect to an encrypted AP as:

```
.threshold.authmode = WIFI_AUTH_WPA2_PSK,
```

- If you need connect to an unencrypted AP, please set the following parameter to 0:

```
.threshold.authmode = 0,
```

- For AP mode selection instructions, please refer to [esp_wifi_types](#).
-

4.10.87 What is the maximum PHY rate of Wi-Fi communication of ESP32-S2 chip?

- The theoretical maximum PHY rate of ESP32-S2 Wi-Fi communication is 150 Mbps.
-

4.10.88 Does ESP modules support EAP-FAST?

:CHIP: ESP32 | ESP32-S2 | ESP32-C3 :

- Yes, please refer to [wifi_eap_fast](#) demo.

4.10.89 Does ESP modules support the WiFi NAN (Neighbor Awareness Networking) protocol?

CHIP: ESP8266 | ESP32 | ESP32-C3 | ESP32-S2 | ESP32-S3

- No.

4.10.90 When using ESP32 with release/v3.3 version of ESP-IDF. When configuring the router, is there an API to directly tell that the entered password is wrong?

- There is no such API. According to the Wi-Fi protocol standard, when the password is wrong, the router will not clearly tell the Station that the 4-way handshake is due to the password error. Under normal circumstances, the password is obtained in 4 packets (1/4 frame, 2/4 frame, 3/4 frame, 4/4 frame). When the password is correct, the AP will send 3/4 frames, but when the password is wrong, the AP will not send 3/4 frame but send 1/4 frame instead. However, when the AP sends 3/4 frame which is lost in the air for some reason, the AP will also re-send 1/4 frame. Therefore, for Station, it is impossible to accurately distinguish between these two situations. In the end, it will report a 204 error or a 14 error.
- Please refer to [Wi-Fi reason code](#).

4.10.91 When testing the Station example of ESP32 base on v4.4 version of ESP-IDF, how to support WPA3 encryption mode?

- Open `menuconfig` → `Component config` → `Wi-Fi` → `Enable WPA3-Personal configuration`;
- Set `capable = true` in `pmf_cfg` in the application code;
- Please refer to [Wi-Fi Security](#) for more descriptions.

4.10.92 How does ESP32 speed up the Wi-Fi connection?

You can try the following approaches:

- Set the CPU frequency to the maximum to speed up the key calculation speed. In addition, you can also set the flash parameters to `QIO, 80MHz`, which will increase power consumption.
- Disable `CONFIG_LWIP_DHCP_DOES_ARP_CHECK` to greatly reduce the time of getting IP. But there will be no checking on whether there is an IP address conflict in the LAN.

- Open `CONFIG_LWIP_DHCP_RESTORE_LAST_IP`, and save the IP address obtained last time. When DHCP starts, send DHCP requests directly without performing DHCP discover.
 - Use fixed scanning channel.
-

4.10.93 Does ESP32 WPA2 Enterprise Authentication support Cisco CCKM mode?

- This mode is currently not supported, even though the enumeration in `esp_wifi_driver.h` has `WPA2_AUTH_CCKM`.
-

4.10.94 Using `wpa2_enterprise` (EAP-TLS method), what is the maximum length supported for client certificates?

- Up to 4 KB.
-

4.10.95 Does ESP8089 support Wi-Fi Direct mode?

- Yes, but ESP8089 can only use the default fixed firmware and cannot be used for secondary development.
-

4.10.96 How does ESP32 connect to an AP whose RSSI does not fall below the configured threshold when there are multiple APs in the environment?

- In ESP32 station mode, there is a `wifi_sta_config_t` structure with 2 variables underneath, i.e., `sort_method` and `threshold`. The RSSI threshold is configured by assigning values to these two variables.
-

4.10.97 ESP32 Wi-Fi has a beacon lost and sends 5 probe requests to the AP after 6 seconds. If the AP does not respond, disconnection will be caused. Can this 6 seconds be configured?

Use API `esp_wifi_set_inactive_time` to configure the time.

4.10.98 Does ESP32 Wi-Fi work with PSRAM?

- For information on using Wi-Fi with PSRAM, please refer to [Using PSRAM](#).
-

4.10.99 [Connect] How to troubleshoot the issue that ESP32 series of products cannot connect to the router over Wi-Fi from the hardware and software aspects?

Please follow the steps below to troubleshoot the issue:

- Firstly, use the [Wi-Fi error code](#) to determine the possible cause for the failure.
 - Then, try connecting another device, such as a phone, to the router to determine whether this is a problem with the router or ESP32.
 - If the phone cannot connect to the router either, please check if there is any problem with the router.
 - If it can, please check whether there is any issue with ESP32.
 - Steps to troubleshoot router issues:
 - Check whether the router is in the stage of power off and rebooting. In this stage, the router cannot be connected. Please do not connect to it until it is initialized.
 - Check whether the configured SSID and PASSWORD are consistent with those of the router.
 - Check whether the router can be connected after being configured in OPEN mode.
 - Check whether the router can connect to other routers.
 - Steps to troubleshoot ESP32 issues:
 - Troubleshoot the ESP32 hardware:
 - * Check whether the issue occurs only in a specific ESP32. If it occurs in a small number of specific ESP32 devices, identify how likely the issue is to occur and compare the hardware differences between them and regular ESP32 devices.
 - Troubleshoot the ESP32 software:
 - * Check whether the Wi-Fi connection works using the [station example](#) in ESP-IDF. The example has a reconnecting mechanism by default, so please watch if ESP32 can connect to Wi-Fi as it is trying reconnecting.
 - * Check whether the configured SSID and PASSWORD are consistent with those of the router.
 - * Check whether ESP32 can connect to the router when the router is configured in OPEN mode.
 - * Check whether ESP32 can connect to Wi-Fi after calling the API `esp_wifi_set_ps(WIFI_PS_NONE)` additionally before executing the code for connecting to Wi-Fi.
 - If all the above steps still fail to locate the issue, please capture Wi-Fi packets for further analysis by referring to [Espressif Wireshark User Guide](#).
-

4.10.100 After being connected to the router, ESP32 prints `w (798209) wifi:<ba-add>idx:0 (ifx:0, f0:2f:74:9b:20:78), tid:0, ssn:154, winSize:64` and `w (798216) wifi:<ba-del>idx` several times every 5 minutes and consumes much more power. Why?

- This log does not indicate any issue. It is related to the Wi-Fi block acknowledgment mechanism. `ba-add` means the ESP32 received an add block acknowledgment request frame from the router. `ba-del` means the ESP32 received a delete block acknowledgment request frame from the router. Frequent printing of this log suggests that the router has been sending packets.
 - If this log is printed periodically every five minutes, it may indicate that the router is updating the group secret key. You could double-check it according to the following steps:
 - Print log in `wpa_supplicant_process_1_of_2()` to check if this function is called every 5 minutes when the group key is updated every 5 minutes.
 - In the router's Wi-Fi configuration interface, check if there is the `Group Key Update Time` option and it is set to 5 minutes.
-

4.10.101 Why can't ESP32 keep the Wi-Fi sending rate at a fixed value with the function `esp_wifi_config_80211_tx_rate()` to maintain stable transmission?

- `esp_wifi_config_80211_tx_rate()` is used to configure the sending rate of `esp_wifi_80211_tx()`.
 - To set and fix the Wi-Fi sending rate, use the function `esp_wifi_internal_set_fix_rate`.
-

4.10.102 How do I debug the ESP32 station that is connected to a router but does not get an IP properly?

- Open the debug log of DHCP in lwIP, go to ESP-IDF menuconfig, and configure `Component config > LWIP > Enable LWIP Debug (Y)` and `Component config -> LWIP > Enable DHCP debug messages (Y)`.
- Earlier IDF versions do not have the above options, so please refer to `lwipopts.h` line 806 and 807, to change `LWIP_DBG_OFF` to `LWIP_DBG_ON` in both lines of code as follows.

```
#define DHCP_DEBUG LWIP_DBG_ON
#define LWIP_DEBUG LWIP_DBG_ON
```

4.10.103 When ESP32 works as a softAP, the station connected to it does not get the IP. How to debug?

To open the debug log of DHCP in lwIP for debugging, please go to `dhcpserver.c` and change `#define DHCP_DEBUG 0` to `#define DHCP_DEBUG 1`.

4.10.104 In ESP-IDF menuconfig, after Component config > PHY > Max Wi-Fi TX power (dBm) is configured to adjust the Wi-Fi transmit power, what is the actual power? For example, what is the actual maximum transmit power when the option is configured to 17 dBm?

- For ESP32, the actual maximum transmit power in the example is 16 dBm. For the mapping rules, please refer to the function `esp_wifi_set_max_tx_power()`.
 - For ESP32-C3, the maximum transmit power value configured in menuconfig is the actual maximum power value.
-

4.10.105 ESP-IDF currently supports connecting to Chinese SSID routers with UTF-8 encoding. Is there a way to connect to Chinese SSID routers with GB2312 encoding?

Yes, please keep the encoding method of the ESP device consistent with that of the router. In this case, make the ESP device also use the GB2312-based Chinese SSID.

4.10.106 After connecting to the router, ESP32 consumes much power in an idle state, with an average current of about 60 mA. How to troubleshoot the issue?

- Please capture Wi-Fi packets for further analysis. See [Espressif Wireshark User Guide](#). After the packets are captured, check whether the NULL data packet sent by the device contains `NULL(1)`. If `NULL(1)` is sent every 10 seconds, it means that ESP32 is interacting with the router in keepalive mode.
 - You can also check the TIM (Traffic Indication Map) field of the beacon packet in the captured packets. If Traffic Indication is equal to 1, it means Group Frames Buffered. In this case, ESP32 will turn on RF, resulting in higher power consumption.
-

4.10.107 How to configure the Wi-Fi country code when the ESP end product needs to be sold worldwide?

- Different Wi-Fi country codes need to be set for different countries.
- The default country code configuration can be used for most countries, but it is not compatible with some special cases. The default country code is CHINA `{.cc="CN", .schan=1, .nchan=13, policy=WIFI_COUNTRY_POLICY_AUTO}`. While in the ESP-IDF v5.0 or later version, the default one has been changed to "01" (world safe mode) `{.cc="01", .schan=1, .nchan=11, .policy=WIFI_COUNTRY_POLICY_AUTO}`. Since channels 12 and 13 are passively scanned by default, they do not violate the regulations of most countries. Besides, the country code of the ESP product is automatically adapted to the router that it is connected to. When disconnected from the router, it automatically goes back to the default country code.

Note:

- There is a potential issue. If the router hides the SSID and is on channel 12 or 13, the ESP end product can not scan the router. In this case, you need to set `policy=WIFI_COUNTRY_POLICY_MANUAL` to enable ESP end products to actively scan on channels 12 and 13.
 - Some countries, such as Japan, support channels 1-14, and channel 14 only supports 802.11b. ESP end products cannot connect to routers on channel 14 by default.
-

4.10.108 Sometimes the rate drops or even a disconnection occurs after a period of iperf testing. What is the reason and how to solve it?

- Possible reasons:
 - Bad network environment.
 - Incompatibility between the computer or mobile phone and the ESP32-S2 or ESP32-S3 softAP.
- Solutions:
 - In the case of a bad network environment, change the network environment or test in a shielded box.
 - In the case of incompatibility, disable `menuconfig > Component config > Wi-Fi > WiFi AMPDU RX`. If disconnections occur again, disable `menuconfig > Component config > Wi-Fi > WiFi AMPDU TX`.

Note:

- AMPDU stands for Aggregated MAC Protocol Data Unit and is a technique used in the IEEE 802.11n standard to increase network throughput.
 - When `WiFi AMPDU RX` is disabled, the device will not receive AMPDU packets, which will affect the RX performance of the device.
 - When `WiFi AMPDU TX` is disabled, the device will not send AMPDU packets, which will affect the TX performance of the device.
-

4.10.109 Why is this log frequently printed when the phone connects to the ESP32-S3 that works as the Wi-Fi AP based on the ESP-IDF v5.0 SDK?

```
(13964) wifi:<ba-del>idx
(13964) wifi:<ba-add>idx:2 (ifx:1, 48:2c:a0:7b:4e:ba), tid:0, ssn:5,
↪winSize:64
```

This is because A-MPDU is created and deleted all the time. The printing is only auxiliary and does not affect communication. If you need to hide this log, add the following code before the Wi-Fi initialization code.

```
esp_log_level_set("wifi", ESP_LOG_ERROR);
```

4.10.110 Does ESP32 support the coexistence of the network port (LAN8720) and Wi-Fi (Wifi-AP)?

Yes, this can be achieved by writing the detection events of both connections as one.

4.10.111 How can I optimize ESP32's slow IP address acquisition after Wi-Fi is connected in a weak network environment or interference environment?

- You can disable Modem-sleep using `esp_wifi_set_ps(WIFI_PS_NONE)`; after Wi-Fi start, and enable Modem-sleep after getting the event `IP_EVENT_STA_GOT_IP`.
- For the situation of reconnection after disconnection, you can manually disable Modem-sleep before connection, and enable it after getting the event `IP_EVENT_STA_GOT_IP`.
- Note: This optimization is not applicable for Wi-Fi/BT coexistence scenarios.

4.10.112 When ESP32/ESP32-S2/ESP32-S3 series chips work in SoftAP mode, they are susceptible to disconnect from mobile phones and PCs of other manufacturers when they are communicating with each other. How can I optimize this situation?

It is recommended to turn off `WiFi AMPDU RX` and `WiFi AMPDU TX` options in `menuconfig`.

4.10.113 What is the value range of ESP32 Wi-Fi TX power?

- It ranges from 2 to 20 dBm. In ESP-IDF, you can use `esp_wifi_set_max_tx_power()` to set the maximum of TX power, and use `esp_wifi_get_max_tx_power()` to get the maximal TX power supported by the system.
- It should be noted that setting TX power too high may affect system stability and battery life, and may also violate wireless regulations in some countries and regions, so it should be used with caution. For more details, please refer to `esp_wifi_set_max_tx_power()` API.

4.10.114 How do I get Wi-Fi RSSI when using ESP32?

When using ESP32 as a station in ESP-IDF release/v4.1, you can use the following code example to obtain the RSSI of the connected AP:

```
wifi_ap_record_t ap_info;
if (esp_wifi_sta_get_ap_info(&ap_info) == ESP_OK) {
    int rssi = ap_info.rssi;
    // handle rssi
}
```

The `wifi_ap_record_t` structure contains information about the connected AP, including SSID, BSSID, channel, and encryption type. The RSSI field represents the RSSI value of the AP. Call `esp_wifi_sta_get_ap_info()` to get the information of this structure. For details of the API, please refer to `esp_err_t esp_wifi_sta_get_ap_info(wifi_ap_record_t *ap_info)`.

4.10.115 Does ESP32 support WPA3-Enterprise?

- ESP32 supports WPA/WPA2/WPA3/WPA2-Enterprise/WPA3-Enterprise/WAPI/WPS and DPP. For more details, please refer to [ESP32 Wi-Fi Feature List](#).
 - esp-idf release/v5.0 has provided [wifi_enterprise example](#), which supports setting WPA3-Enterprise mode for testing. You can configure it as following: `idf.py menuconfig > Example Configuration > Enterprise configuration to be used > WPA3_ENT`.
-

4.10.116 Does ESP modules support WAPI (Wireless LAN Authentication and Privacy Infrastructure)?

- Yes. Please refer to [WIFI_AUTH_WAPI_PSK](#)
-

4.10.117 How can I increase the time for scanning routers when using ESP32 as the Wi-Fi Station to connect routers?

- In ESP32, by default, channels 1 ~ 11 are active scanning, while channels 12 ~ 13 are passive scanning. Active and passive scanning require different amounts of time. For more details, please refer to [Wi-Fi scan configuration](#). The default duration for active scanning is 120 ms per channel, while for passive scanning, it is 360 ms per channel. You can call the following function before calling `esp_wifi_start()` to increase the time for scanning routers:

```
extern void scan_set_act_duration(uint32_t min, uint32_t max);
extern void scan_set_pas_duration(uint32_t time);
scan_set_act_duration(50, 500);
scan_set_pas_duration(500);
```

4.10.118 Does ESP32 support LDPC

- Yes. No additional configuration or calling is required as it is already implemented in the driver.

HARDWARE RELATED



5.1 Chip Comparison



5.1.1 What is the difference between ESP32 with a single core and ESP32 with dual cores regarding programming method, feature performance, power consumption, and so on?

Compared with the single-core chip, the dual-core chip has one more independent core, on which some highly real-time operations can be supported in a better way.

- The two chips employ the same programming method except the following step only. For single-core chip, you have to configure FreeRTOS to make it run on the first core. The configuration path is `make menuconfig > Component config > FreeRTOS > [*] Run FreeRTOS only on first core`. This step is not needed for the dual-core chip.
- The two chips have similar performance in most cases, except under high-load conditions, such as AI algorithm and high real-time interrupts.
- The dual-core chip consumes a little more power than the single-core chip in Modem-sleep mode. For more details, please refer to [ESP32 Series Datasheet](#).

5.1.2 What's the difference between ESP32 v3.0 and previous chip revisions in software and hardware?

- For software usage, there is no difference. ESP32 v3.0 is compatible with old firmwares. For hardware, some bugs have been fixed in ESP32 v3.0.
 - For more information on design changes, please refer to [ESP32 Chip Revision v3.0 User Guide](#).
-

5.1.3 Can the GPIO34 ~ GPIO39 of ESP32 only be set to input mode?

- Yes, the GPIO34 ~ GPIO39 of ESP32 can only be set to input mode. They cannot be set to the output mode as they cannot be pulled up or down by software.
 - For more details, please refer to [GPIO & RTC GPIO](#).
-

5.1.4 Does ESP32 support a driver for Linux?

Yes, please refer to the example [esp-hosted](#).

Note: This example is adapted to the 802.3 protocol rather than the 802.11 protocol.

5.1.5 What is the meaning of the acquired data when you scan the data matrix on the module's shielding case?

- Taking 0920118CAAB5D2B7B4 as an example. '09' indicates the factory code; '20' indicates the last two numbers of the year of manufacturing, which is 2020 in this example; '11' indicates the week number of manufacturing; and the last 12 numbers '8CAAB5D2B7B4' is the MAC address of this module. For the latest information, please refer to [Module Packaging Information](#).
-

5.1.6 Can VDD3P3_RTC of ESP32 support independent battery power supply?

- The internal RTC of ESP32 cannot work independently as it requires the main CPU to participate in the configuration. Even if the RTC domain is powered by an independent battery, it may still be powered down suddenly.
 - If you want to save the clock information when power-off occurs, please use an external RTC clock chip.
-

5.1.7 What is the difference between ESP32-PICO-D4, ESP32-PICO-V3, and ESP32-PICO-V3-02?

- Built-in chip version: The core chip of ESP32-PICO-V3 and ESP32-PICO-V3-02 is ESP32 (ECO V3), while that of ESP32-PICO-D4 is ESP32 (ECO V1).
 - Package size: The size of ESP32-PICO-D4 and ESP32-PICO-V3 is $7 \times 7 \times 0.94$ (mm), while the size of ESP32-PICO-V3-02 is $7 \times 7 \times 1.11$ (mm).
 - Built-in flash: ESP32-PICO-D4 and ESP32-PICO-V3 integrate 4 MB SPI Flash, while ESP32-PICO-V3-02 integrates 8 MB flash and 2 MB PSRAM.
 - Pin differences: Please refer to Section Compatibility with ESP32-PICO-V3 in the [ESP32-PICO-V3-02 Datasheet](#).
-

5.1.8 Does ESP8266 support 32 MHz crystal?

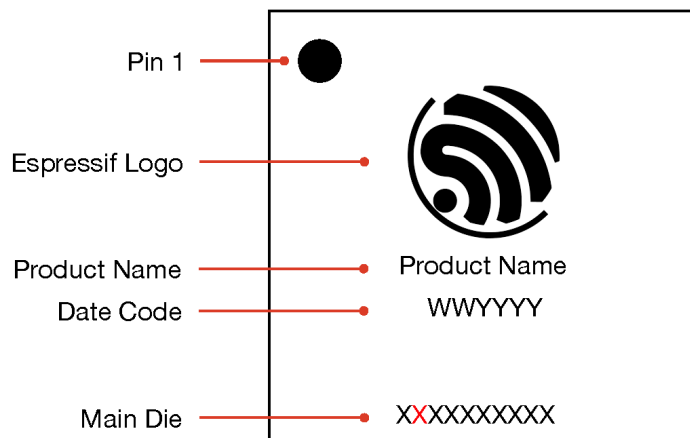
- No. Currently, ESP8266 supports 26 MHz and 40 MHz crystals, and the 26 MHz crystal is recommended.

5.1.9 Do ESP products support Zephyr?

- Please refer to [Espressif's Support for Zephyr](#) for detailed information about the support for Zephyr in ESP products. While there are only a few functional modules have been adapted so far, other modules will be updated further later. For feature requests, you may check or ask on the [Zephyr GitHub issue](#) first.
- You can also find information about ESP products in sections like [XTENSA Boards](#) and [RISC-V Boards](#) in *Zephyr Docs* <<https://docs.zephyrproject.org/latest/introduction/index.html>>.

5.1.10 How to identify the chip revision from the chip silk marking?

You can do it by checking the second character of main die line on the chip silk marking.



The mapping between the chip revision of all our chips and the second character of main die lie can be found in the table below:

Chip Series	Chip Revision	Marking Indicator
ESP32	v0.0	A
	v1.0	B
	v1.1	F
	v3.0	E
	v3.1	G
ESP32-S2	v0.0	A
	v1.0	B
ESP32-C3	v0.0	A
	v0.1	B
	v0.2	C
	v0.3	D
	v0.4	E
ESP32-S3	v0.0	A
	v0.1	B
	v0.2	C
ESP32-C2/ESP8684	v0.0	A
	v1.0	AA
	v1.1	B
	v1.2	C
ESP32-C6	v0.0	A
ESP32-H2	v0.0	A
	v0.1	B

- For detailed differences between chip revisions, please refer to respective chip errata documents from Espressif's [documentation page](#).
- For a complete explanation of the chip silk marking, please refer to [Espressif Chip Packaging Information](#).

5.2 Development board

□

5.2.1 How long does it take for the ESP-WROOM-02D module to restart after the reset signal?

- It will restart when the input level is lower than 0.6 V for more than 200 s.

5.2.2 According to the schematic of ESP32-LyraT-Mini, the analog output of the ES8311 codec chip is connected to the input of the ES7243 ADC chip. What is the purpose of this?

- The hardware acquisition circuit of the AEC reference signal simultaneously transmits the DAC output of the Codec (ES8311) to the speaker PA and the ADC (ES7243) AINLP/N, of which the signal collected would be send back to the ESP32 as the reference signal for the AEC algorithm.

5.2.3 When using the ESP32-MINI-1 module, the serial port printed the follows log when powered on. What could be the reason?

```
rst:0x10 (RTCWDT_RTC_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
invalid header: 0xffffffff
invalid header: 0xffffffff
invalid header: 0xffffffff
invalid header: 0xffffffff
invalid header: 0xffffffff
invalid header: 0xffffffff
invalid header: 0xffffffff
invalid header: 0xffffffff
ets Jul 29 2019 12:21:46
```

- The reason could be flash is not programmed.

Which GPIO is connected to the RGB LED of the [ESP32-S3-DevKitC-1](#) development board?

- The RGB LED on the [ESP32-S3-DevKitC-1 v1.0](#) development board is connected to GPIO48.
- The RGB LED on the [ESP32-S3-DevKitC-1 v1.1](#) development board is connected to GPIO38.
- The reason why the [ESP32-S3-DevKitC-1 v1.1](#) development board changed the RGB LED pin to GPIO38 is that the [ESP32-S3R8V](#) chip's VDD_SPI voltage has been set to 1.8 V. Therefore, unlike other GPIOs, GPIO47 and GPIO48 in the VDD_SPI power domain of this chip also operate at 1.8 V.

5.3 Hardware design

□

5.3.1 The I2S pins of ESP32 are scattered. Can I route I2S signals to adjacent pins? For example, to GPIO5, GPIO18, GPIO23, GPIO19, and GPIO22, or to GPIO25, GPIO26, GPIO32, and GPIO33.

- All I2S signals can be routed to different I/Os freely. Please note that some I/Os can only be set as input. For details, please refer to Section *Peripheral Pin Configurations* and Appendix *IO_MUX* in the [ESP32 Datasheet](#).

5.3.2 How can I stop the power loss of VDD3P3_RTC after ESP32 enters Light-sleep mode?

After entering Light-sleep mode, if RTC power loss occurs, the level of the GPIO corresponding to the VDD3P3_RTC pin will be pulled low, causing external RTC or other devices to malfunction. There are two possible ways to solve this problem:

- Set the RTC hardware voltage control register (RTC_CNTL_REG) to control the voltage. Specifically, set the FORCE_PU and FORCE_PD bits in the RTC_CNTL_REG register to 1, namely, `RTC_CNTL_REG |= RTC_CNTL_FORCE_PU_M | RTC_CNTL_FORCE_PD_M;`

- Set GPIO hold pins. The ESP32's Light-sleep mode supports GPIO hold function, which can set some GPIO pins as hold pins to maintain their voltage level when the system enters Low-power mode. Specifically, the VDD3P3_RTC pin can be set as a hold pin to maintain its voltage. The relevant code snippet is as follows:

```
esp_sleep_pd_config(ESP_PD_DOMAIN_RTC_PERIPH, ESP_PD_OPTION_ON);  
esp_sleep_enable_gpio_wakeup();  
gpio_hold_en(GPIO_NUM_X);
```

Among them, ESP_PD_DOMAIN_RTC_PERIPH represents the power domain of the RTC subsystem. ESP_PD_OPTION_ON enables the power domain. And the gpio_hold_en() function can set the specified GPIO pin as a hold pin. After setting the VDD3P3_RTC pin as a hold pin, even if the system enters Light-sleep mode, the voltage of the pin will be held.

Please note that using the GPIO hold function will increase the power consumption of the system, so the appropriate solution should be selected according to the specific application scenario. If only the power supply of the RTC hardware needs to be held, the first method can be adopted. If the power supply of other external devices needs to be held as well, the second method can be adopted.

5.3.3 What should be noted when I configure the pins of ESP32?

- You may assign most of the digital peripherals to any pins through GPIO Matrix. However, functions such as SDIO, high speed SPI, and analog can only be realized via IO MUX.
- For details, please refer to [GPIO & RTC GPIO](#).

Note:

- Strapping pins have default levels. Please refer to [ESP32 Datasheet](#).
 - GPIO34 ~ GPIO39 can only be set as input without software-enabled pull-up or pull-down functions.
 - GPIO6 ~ GPIO11 are saved for flash.
 - GPIO1 and GPIO3 are the TX and RX pins for UART0, which cannot be configured.
 - GPIO16 and GPIO17 are saved for PSRAM if there is any.
-

5.3.4 What is the voltage tolerance of GPIOs of ESP chips?

- The voltage tolerance of GPIO is 3.6 V. If the voltage exceeds 3.6 V, please add a voltage divider to protect GPIO pins from damage.
-

5.3.5 What are the power supply specifications for ESP8266?

- Digital working voltage range: 1.8 V ~ 3.3 V
- Analog working voltage range: 3.0 V ~ 3.6 V (the lowest possible value is 2.7 V)
- Peak analog circuit current: 350 mA
- Peak digital circuit current: 200 mA

Note: The operating voltage of SPI flash should be compatible with that of GPIO pins. The operating voltage of CHIP_EN ranges from 3.0 V to 3.6 V, so please use a level converter when GPIO pins operates at 1.8 V.

5.3.6 Do Espressif Wi-Fi modules support single-layer PCBs?

- The ESP32 module is a wireless device, which needs rather high-quality PCB materials to fulfill the RF performance requirements. We have tested four-layer and two-layer PCBs, but not single-layer ones.
 - Single-layer PCBs are not recommended as RF performance cannot be guaranteed. You may use single-layer PCBs in your end products and then mount Espressif modules.
 - Four-layer PCBs are recommended for desired RF performance.
-

5.3.7 What should be noted when I power ESP8266 with batteries?

- The operating voltage of ESP8266 ranges from 3.0 V to 3.6 V, so two AA batteries can be used to power ESP8266. Please ensure the battery voltage stays within the operating range of ESP8266 when it drops.
 - If the lithium battery voltage surpasses module operating voltage, and the voltage drops heavily during discharge, then such batteries should not be used to power ESP8266.
 - We recommend you to use DC/DC converters or LDO regulators to convert voltage before powering ESP8266. Please pay attention to the difference between the input and output voltages of converters or regulators.
-

5.3.8 Where can I get the footprint of ESP32 Series?

You may get the footprint in the PCB layout of different modules. Please refer to [reference designs](#).

5.3.9 For ESP32-S2 chips, can I have audio connection when the DVP camera interface is in use?

The LCD, DVP camera, and I2S interfaces of ESP32-S2 share one set of hardware, so they cannot be used at the same time.

5.3.10 What should be noted when I assign I2C signals to GPIO0 and GPIO4 of ESP32 modules?

Please pull GPIO0 up when assigning I2C signals to the pin. Please also ensure GPIO0 can be pulled down when powered on during flashing, which can be released afterwards. Only pull GPIO0 down when flashing firmware on ESP32 modules.

5.3.11 When the external flash is connected to GPIO6 ~ GPIO11, can they be set as SPI pins?

When the external flash is connected to GPIO6 ~ GPIO11, they cannot be set as SPI pins.

5.3.12 Do I need to connect an external crystal when using the ESP8285 chip?

You need to connect an external crystal, as the chip has no internal crystal.

5.3.13 Where can I find the reference design for connecting an external PSRAM to ESP32-D2WD?

You may refer to the design for the external PSRAM of ESP32-PICO-D4. Please refer to Chapter *Peripheral Schematics* in the [ESP32-PICO-D4 Datasheet](#).

Note: ESP32-D2WD has an 1.8 V flash, so please add a resistor and a capacitor to VDD_SDIO and connect an 1.8 V PSRAM.

5.3.14 Can I use ESP32 to play music with PWM or DAC?

You may use ESP32 to play music with PWM or DAC, and we recommend you play voice prompts. To run a test, please refer to [esp-adf/examples/player/pipeline_play_mp3_with_dac_or_pwm](#).

5.3.15 Why is the suggested voltage range of ESP32 modules different from that of ESP32 chips?

- This is because of the different working environments and usage scenarios. - The ESP32 chip is a bare die and requires external circuitry on a circuit board to function properly. The recommended operating voltage range for the ESP32 chip is 2.3 V to 3.6 V, which is determined by the chip's electrical parameters. Within this voltage range, the ESP32 chip can function properly and provide optimal performance and power consumption. - The ESP32 module, on the other hand, is a packaged electronic module that typically includes voltage regulators, external crystals, external antennas, and other peripheral chips, such as flash memory and RAM, and can be used directly. As the module's circuitry has already been optimized and tested, its recommended operating voltage range may be narrower. For example, the ESP32-WROOM-32 module has a recommended operating voltage range of 3.0 V to 3.6 V. Apart from that, as the module has to take flash voltage into account, the recommended operating voltage for the ESP32 module would thus be higher.
 - When using these chips and modules, appropriate power supplies and peripheral circuits should be chosen based on the specific situation to ensure that they function properly.
 - For more information, please check [module and chip datasheets](#).
-

5.3.16 Why does it take a longer time to erase the flash of self-developed modules than that of Espressif modules?

- It is common that the erasing time varies, as it depends on factors such as the manufacturer of your flash and the size of the block you erase.
 - If you want to shorten the erasing time, you may test flash memories from different manufacturers.
-

5.3.17 Why does the current surge when ESP8266 is powered on?

- The RF and digital circuits of ESP8266 are highly integrated. When ESP8266 is powered on, the RF automatic calibration starts to work, which requires high current.
 - The maximal current of the analog circuit can reach 500 mA, while that of the digital circuit is 200 mA.
 - Usually the average current is 100 mA.
 - To wrap up, ESP8266 needs a 500 mA power supply.
-

5.3.18 What choices do I have when configuring the RMI clock for the Ethernet of ESP32?

- We recommend use GPIO0 as the RMI clock input pin. Please note that the GPIO0 cannot be low level when the chip powered on.
 - For details, please refer to the [Configure MAC and PHY](#) guide.
-

5.3.19 What kind of socket is used on ESP32-LyraT development boards to connect a speaker?

Please use a PH-2A socket.

5.3.20 For modules housing ESP32, which pins cannot be set by users?

- For ESP32-WROOM Series of modules, GPIO6 ~ GPIO11 are pins for flash and cannot be set for other uses.
 - For ESP32-WROVER Series of modules, GPIO16 and GPIO17 are pins for PSRAM and cannot be set for other uses.
 - Besides, please note that ESP32 has five strapping pins. For details, please refer to [ESP32 Datasheet](#).
-

5.3.21 Which is the reset pin of ESP32?

- CHIP_PU serves as the reset pin of ESP32. The input level (VIL_nRST) for resetting the chip should be low enough and remain so for a period of time. Please refer to Section *Reset* in the [ESP32 Hardware Design Guidelines](#).
-

5.3.22 What should be noted when I design the power supply for ESP8266?

- If you use LDO regulators, please ensure the input voltage ranges from 2.7 V to 3.6 V and the output current is greater than 500 mA.
 - The decoupling capacitor must be as close to the chip as possible. The equivalent resistance should be low enough.
 - ESP8266 is not 5 V tolerant. It operates at 3.3 V, with the operating voltage ranging from 2.7 V to 3.6 V.
 - If you use DC/DC converters, please add LC filters when necessary.
 - Please refer to Section *Power Supply* in the [ESP8266 Hardware Design Guidelines](#).
-

5.3.23 When I use the TOUT pin of ESP8266 to collect ADC sample signals, will the pins be damaged if the voltage is greater than 1.0 V?

- If the input voltage is within the operating range of pins (0 V ~ 3.6 V), the pins will not be damaged.
 - If the voltage is greater than 1.0 V, it may lead to abnormal results.
-

5.3.24 For modules with PCB antennas, what should be noted when I design the PCB and the housing of the antenna?

- When adopting on-board design, you should pay attention to the layout of the module on the base board. The interference of the base board on the module's antenna performance should be reduced as much as possible.
 - It is recommended that the PCB antenna area of the module be placed outside the base board, while the module be put as close as possible to the edge of the base board so that the feed point of the antenna is closest to the board.
 - Please make sure that the module is not covered by any metal shell. The antenna area of the module and the area 15 mm outside the antenna should be kept clean (namely no copper, routing, components on it).
 - For details, please refer to [Hardware Design Guidelines](#).
-

5.3.25 Can GPIO 34 ~ GPIO39 of ESP32 be used as UART RX pins?

- GPIO 34 ~ GPIO39 can be used as UART RX pins.
-

5.3.26 Where can I find the design reference for the external 32 kHz crystal of ESP32 modules?

- Please refer to Section *RTC (optional)* in the [ESP32 Hardware Design Guidelines](#).
-

5.3.27 Does the flash of ESP32 modules support 80 MHz QIO access mode?

- ESP32 modules support 80 MHz QIO access mode.
 - You are recommended to load the second-stage bootloader in QIO mode, as QE is not set by default in some flash status registers.
-

5.3.28 How to configure the RMII synchronous clock for the Ethernet of ESP32?

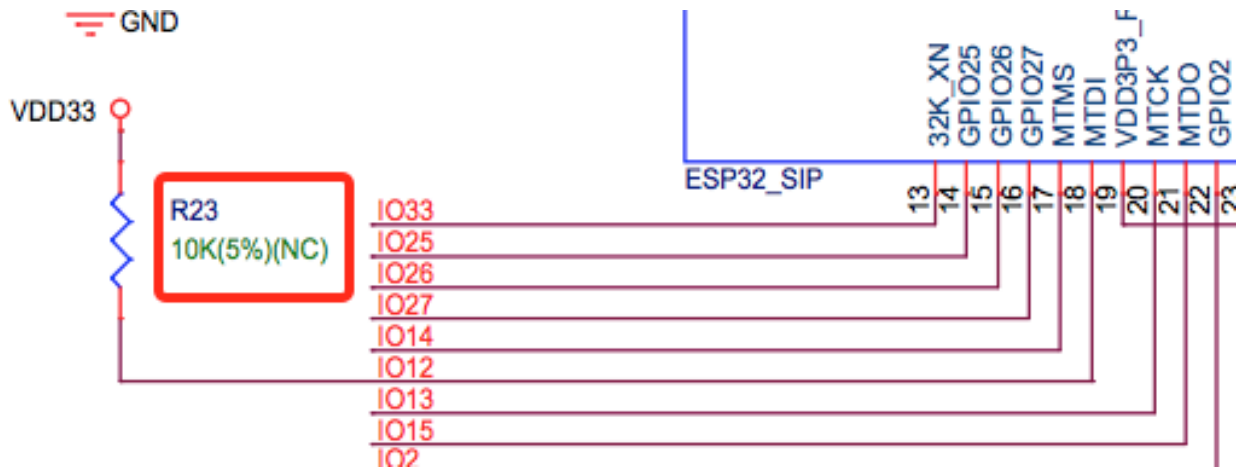
- To run a test, please refer to [esp-idf/examples/ethernet/basic](#).
 - When GPIO0 provides clock output for PHY, the Ethernet connection of the IP101 PHY chip can be unstable. Therefore, you are recommended to connect a 50 MHz crystal to PHY with GPIO0 as input.
 - Because of the characteristics of GPIO0, the IO should be set to control the enable pin of PHY.
 - Please read [Ethernet document](#).
 - You may also refer to [1SCH_ESP32-ETHERNET-KIT Schematics](#).
-

5.3.29 How can I hard reset ESP8266? Is hard reset active low or active high? What are the requirements for reset?

- The Pin32 EXT_RSTB of ESP8266 is the reset pin. This active low pin has an internal pull-up resistor. To prevent external factors triggering a restart, it is recommended that the EXT_RSTB cabling be as short as possible and an RC circuit be added to the EXT_RSTB pin.
- The CHIP_EN pin of ESP8266 can also be used as a hard reset pin. When you use the CHIP_EN pin as a reset pin, the reset is active low. To reset and restart ESP8266, the input level should be lower than 0.6 V and last for more than 200 s. It is recommended to use the CHIP_EN pin for chip reset. For more information, please refer to Section *Reset* in the [ESP8266 Hardware Design Guidelines](#).

5.3.30 What does the term NC mean in Espressif schematics?

- NC is the acronym of “No Component”. If you see a pull-up resistor is marked NC as shown in the figure below, it indicates that the component is not installed.



5.3.31 How can I use multiple antennas with ESP32-S2?

- Using multiple antennas with ESP32-S2 is similar to that with ESP32. You may refer to [ESP32-WROOM-DA Datasheet](#).
- For detailed instructions, please refer to [ESP-IDF Programming Guide](#).
- You can add an RF switch to select antennas.

5.3.32 Does ESP32-C3F SPI CS0 pin need an external 10 k pull-up resistor?

CHIP: ESP32-C3F

- The SPI controller of ESP32-C3F supports software-programmable CS (Chip Select) pin without external 10 k pull-up resistor. - In ESP32-C3F, the CS pin can be set to any GPIO pin via SPI controller configuration. The GPIO state can be set in the code to control the level of the CS pin. When the SPI bus is idle, the CS pin is automatically pulled up to the default state of the GPIO pin without an external pull-up resistor.
 - Please note that when using a software-programmable CS pin, to select the target device, the pin should be manually pulled down before the SPI bus transmission. After the transmission is completed, pull the CS pin high to release the device. Additionally, the level and status of the CS pin should be adjusted according to the actual situation to ensure the stability and reliability of the SPI bus.
-

5.3.33 Is there any hardware design reference for ESP-Skainet Speech Recognition?

- Please refer to [ESP32-Korvo V1.1 User Guide](#).
-

5.3.34 Is it necessary to connect a 32 kHz RTC crystal?

CHIP: ESP32 | ESP32-C3 | ESP32-S3

- The external 32 kHz crystal is often used for Bluetooth Light-sleep timing. Therefore, when Bluetooth LE Light-sleep mode is not necessary, there is no need to do so.
-

5.3.35 For the ESP32-MINI-1 module, is there a component library for Altium Designer?

- Our hardware schematics are developed with PADS. To find the .asc file that can be converted and opened in Altium Designer, please go to [ESP32-MINI-1 Reference Design](#).
 - For hardware reference designs of other modules, please refer to [technical documents](#).
-

5.3.36 Can I change the input voltage of UART0 of ESP8266 from 3.3 V to 1.8 V?

- Yes. VDDPST is the power domain for UART0, the input voltage of which can be 1.8 V theoretically.
-

5.3.37 Is the level of UART0 of ESP8266 determined by VDD (VCC_WIFI) or VDDPST (VCC_CODEC_IO)?

- The digital power voltage is determined by VDDPST, so the level of UART0 of ESP8266 is determined by VDDPST (hardware power domain).
-

5.3.38 What should be noted when I connect an external PSRAM to ESP32-D2WD?

- Please enable CPU frequency 240 Mhz and RTC clock 80 Mhz as follows:
 - `menuconfig>Serial flasher config>Flash SPI Speed (80 Mhz)`
 - `Component config>CPU frequency (240 Mhz)`
 - `Component config > ESP32 specific > [*]Support for external, SPI-connected RAM`
 - `Component config > ESP32 specific > SPI RAM config > Set RAM clock speed (80 Mhz clock speed)`
-

5.3.39 When the VDD power supply of ESP32 slowly rises from 0 V to 3.3 V, why does the chip not start as usual?

- This problem occurs because the power-on sequence requirements are not met. To start the chip, when VDD reaches 2.3 V, the EN voltage should not exceed 0.6 V.
 - If the VDD rise time is too long, the RC circuit on the EN side of the chip will not be able to delay EN.
 - You may modify the RC circuit, for example, increase the capacitance, adjust the resistance, or use the Reset chip to control EN state.
 - When the voltage provided to ESP32 is detected to be less than 2.3 V, you are recommended to pull down the EN pin of ESP32.
 - For ESP32 power-on sequence description, please refer to [ESP32 Datasheet](#).
-

5.3.40 When using the ESP32-WROOM-32D module, can I set GPIO12 for other uses?

- GPIO12 is a strapping pin that controls the startup voltage of SPI flash. The SPI flash startup voltage of the ESP32-WROOM-32D module is 3.3 V, so GPIO12 needs to be pulled down during powering on.
 - If you need to set GPIO12 for other uses, please use the command `espefuse.py set_flash_voltage 3.3v` in the esptool to set the voltage through VDD_SDIO as 3.3 V.
 - It is possible to connect VDD_SDIO to 3.3 V in hardware directly without burning eFuse again.
 - In the mass production stage, you can also download the firmware directly by modifying the default configuration of ESP32_EFUSE_CONFIG to `config_voltage = 3.3 V` in `config/esp32/utility.config` in the flash download tool.
-

5.3.41 When connecting an external flash to ESP32-WROOM-32D module, is it possible if I do not use GPIO6 ~ GPIO11 pins?

- ESP32 has 3 sets of SPIs (SPI, HSPI and VSPI), which can access the external flash through the SPI0/1(HSPI/VSPI) bus. The external flash connected to other pins (pins other than GPIO6 ~ GPIO11) can only receive data for storage, but not run code. If you need to run code from flash, please connect the flash to GPIO6 ~ GPIO11 pins only.
-

5.3.42 Do I need to add a shielding case to the PCB of ESP32 modules?

- Whether a shield needs to be added depends on the specific application scenarios and requirements.
 - In some high-demand application scenarios, such as environments with severe wireless communication interference or high electromagnetic compatibility (EMC) testing requirements, adding a shield can effectively reduce external interference and mutual interference on the PCB board, improving system stability and reliability. At this time, the shield should be made of conductive material and grounded to ensure its effectiveness.
 - On the other hand, if the application scenario is relatively simple, such as low wireless communication interference and low EMC requirements, the effect of adding a shield may not be very obvious and may increase system cost and complexity.
 - If the board has other signal interference, such as 2G, 3G, 4G, Wi-Fi, Bluetooth, Zigbee, etc., it is recommended to add a shielding case.
-

5.3.43 Do I must use GPIO0, GPIO1, or GPIO3 of ESP32 as the I2S CLK pin?

- The MCLK pin must use GPIO0, GPIO1, or GPIO3. The other clock pins can use any GPIOs. Note that GPIO0 is generally not recommended for other functions because it is a strapping pin.
-

5.3.44 Does the ESP32-U4WDH chip support external PSRAM chips?

- The ESP32-U4WDH chip supports external PSRAM chips. However, only the [ESP-PSRAMXXH](#) chip released by Espressif is supported. Third-party PSRAM chips are not supported.
 - For hardware design, all the PSRAM pins except for the CS pin can be multiplexed with Flash. For more information, please refer to the [ESP32 Hardware Design Guidelines](#).
 - Also, when designing the PCB, please make sure that the GND of the PSRAM to the GND of the ESP32-U4WDH is as short as possible; Otherwise, the signal quality may be affected.
-

5.3.45 Does ESP32 support connection to an external SD NAND flash chip (instead of the default NOR flash chip) via the SPI0/SPI1 interface for storing application firmware?

- The ESP32 chip does not support external SD NAND Flash chips using the SPI0/SPI1 (connect the core Flash) interface.
 - If you want to store external data, it is recommended to use the SPI2, SPI3, or SDIO interface of ESP32 to connect to an external NAND SD chip.
 - SPI2 and SPI3 can be used via any GPIOs, while the SDIO interface can only be used via the specified interface. For more information, please refer to Section *Peripheral Pin Configurations* in the [ESP32 Datasheet](#).
-

5.3.46 Does it support to connect a second PSRAM chip externally based on the ESP32-S3R8 chip?

- No, it is not supported. The reasons are as follows:
 - The PSRAM chip is connected to the MSPI bus. There are only two CS signals from the MSPI peripheral, one is connected to the flash, another is connected to the PSRAM.
 - CPU accesses external memory via cache and MSPI. A GPSPi peripheral is not accessible cache.
-

5.3.47 Could you please provide the 3D model and Footprint files of the ESP32-S3-WROOM-1 module?

- The 3D models and Footprint files for the modules are available under the [espressif/kicad-libraries](#) library.
-

5.3.48 Does ESP32/ESP32-S2/ESP32-C3/ESP32-S3 support powering the RTC power domain only to keep the chip working with low power consumption?

No, it is not supported. Take ESP32 as an example, detailed information will be updated to the RTC chapter in [ESP32 Hardware Design Guidelines](#).

5.3.49 How can I improve the EMC performance?

- At the hardware level, the following measures can be taken to improve the EMC performance of the PCB board.
 - The EMC performance with a four-layer board design will be better than a two-layer board hardware design.
 - Add filtering circuits to the power supply circuit.
 - Add ESD or magnetic beads to the antenna circuit.
 - Add a zero-ohm series resistor to the SPI Flash communication lines to lower the driving current, reduce interference to RF, and adjust timing for better interference shielding.
-

- Keep GND intact as much as possible.
 - For more hardware design suggestions, please refer to [ESP Hardware Design Guidelines](#).
-

5.3.50 Why do I need to connect a 499 resistor to U0TXD for ESP32-S3?

- The 499 resistor is reserved for the U0TXD to suppress 80 MHz harmonics. For more information, please refer to [ESP32S3 Series Hardware Design Guidelines](#).
-

5.3.51 How to calibrate the ESP32-S3 ADC in hardware?

- The ESP32-S3 already has the ADC calibrated in hardware on the chip. ESP32-S3 ADCs can be sensitive to noise, resulting in large differences in ADC readings. Depending on the usage scenario, you may need to connect a bypass capacitor (e.g. 100 nF ceramic capacitor) to the ADC input pads for minimising noise. In addition, multi-sampling can be used to further mitigate the effects of noise.
-

5.3.52 How to design an automatic download circuit based on the ESP32 series chip?

- You can refer to the hardware design of the automatic download circuit in the [ESP32-DevKitC V4 schematic](#).
-

5.3.53 Which crystal oscillator should be used on the ESP8266 chip?

- The ESP8266 chip requires the 26 MHz crystal oscillators to start the chip. The crystal precision should be ± 10 PPM. For details, please refer to [ESP8266 Hardware Design Guidelines](#).
-

5.3.54 Do the ESP32-C2, ESP32-C3 and ESP32-C6 chips support external PSRAM chips?

- ESP32-C2, ESP32-C3, and ESP32-C6 do not support external PSRAM chips.
-

5.3.55 When the ESP32-C3 is powered by a battery, it may fail to start if the supply voltage gradually decreases, for example, when the battery is fully discharged and then recharged. In such cases, the solution could only be disconnecting the battery from the ESP32-C3 and reconnect a fully charged battery, or to connect a voltage regulator diode between the 3.3 V pin and the EN pin to ensure the chip starts properly. What is the root cause of this situation? Is there an optimal solution?

- Root cause: When powering up and resetting the ESP32-C3 chip, the CHIP_EN pin needs to meet the power-up timing specifications outlined in the [ESP32-C3 datasheet](#) or [ESP32-C3 hardware design guidelines](#). If the battery discharge and power-up process are relatively slow, ESP32-C3 may not be fully reset, resulting in certain units of the chip being in an uncertain state.
- Solution: Currently, if using battery power or energy storage systems, this issue can be addressed by adjusting RC component values, using voltage divider circuits with two resistors, or using a reset IC, which is a more commonly used approach. For detailed information regarding RC component values and related resistors, please refer to [ESP32-C3 Family Hardware Design Guidelines](#).

5.4 RF Related

□

5.4.1 Does the RF performance of an ESP32 module degrade if it runs with a 2.8 V supply?

Yes, its RF performance may become unstable. It is recommended to follow the suggested operating voltage range specified in the [module's datasheet](#).

5.4.2 What are the modulation methods supported by ESP chips?

- ESP8266 supports BPSK/QPSK/16QAM/64QAM/DBPSK/DQPSK/CCK.
 - ESP32 supports BPSK/QPSK/16QAM/64QAM/DBPSK/DQPSK/CCK/GFSK /4-DQPSK 8-DPSK.
 - ESP32-S2 supports BPSK/QPSK/16QAM/64QAM/DBPSK/DQPSK/CCK.
 - ESP32-C3 supports BPSK/QPSK/16QAM/64QAM/DBPSK/DQPSK/CCK/GFSK.
 - ESP32-S3 supports BPSK/QPSK/16QAM/64QAM/DBPSK/DQPSK/CCK/GFSK.
-

5.4.3 How can I get the RF related information (e.g., antenna specification, antenna pattern, etc.) for certification?

For such information, please contact [Sales](#) for help.

5.4.4 Why does ESP32 automatically reduce its transmit power when it uses the RF Test Tool at 80 °C?

- Temperature compensation is disabled by default when ESP32 runs the fixed frequency firmware. Therefore, the power reduces at a high temperature. To enable temperature compensation, please send `txpwr_track_en 1` to ESP32 through the default log serial port.
-

5.4.5 How to improve the receiving distance and strength of Wi-Fi signals for ESP32-WROVER-E? (Application scenario: Wi-Fi probe)

- In terms of software, you can either set the maximum transmit power by API `esp_wifi_set_max_tx_power()`, or set that via menuconfig: `Component config > PHY > Max Wi-Fi TX power (dBm)`. The default maximum transmit power is 20 dBm.
 - If the transmit power has been set to the maximum value, you may improve the efficiency of the antenna and receiving devices as follows:
 - Adjust the module direction so that the stronger radiation direction of the antenna points to the receiving device, realizing the farthest radiation distance.
 - Make sure there is no metal or blocking object near the antenna of the module, no PCB on the back of the antenna, and the Wi-Fi signal is not interfered by other signals of the end device.
 - Use the IE series module with an antenna connector if the performance of the PCB antenna cannot meet requirements, so that an external antenna with higher directional gain can be connected.
 - Increase the radiation efficiency of the antenna in the receiving device.
-

5.4.6 How to write phy_init data to flash ?

:CHIP: ESP32 :

- You can write it via the power limit tool. Please download the [ESP_RF_TEST Tool](#), unzip the package, open the `EspRFTestTool_vx.x_Manual.exe` file, and then go to `help > Tool help > PowerLimitTool help` for detailed operations.
-

5.4.7 How can I optimize the second harmonic and other spurious signals created by my own products?

The second harmonic mainly comes from the radiation generated by the RF link and PA (power amplifier) power supply. The backplane (board size) and the product also make impact on the second harmonic. Therefore, it is recommended to:

- Add a ground capacitor of approximately 2.4 pF to the RF matching circuit to reduce the spurious radiation coming from the RF link.
 - Add a series inductor to the PA power supply (Pins 3 and 4 of ESP32) to reduce the spurious radiation coming from it.
-

5.4.8 How can I suppress the harmonics of 80 MHz?

If the harmonics of 80 MHz (160 MHz, 240 MHz, 320 MHz, etc.) exceed the allowable levels of spurious emissions, you can add a resistor of approximately 470 Ω to the data transmission circuit (TXD) to suppress those harmonics.

5.4.9 Is manual power calibration required for the Espressif modules that connect to external antennas?

No, it is not required. When an external antenna is used, please make sure it is connected properly before the module is powered up. Then, the module performs a self-calibration, including power calibration.

5.4.10 What is the default duty cycle for the “default” level when using the “RF Test Tool” to set the Wi-Fi “TX continues” mode?

- The default duty cycle for the “default” level is 98% or higher, and cannot be modified.

5.5 Process and ESD Protection

□

5.5.1 What should be paid attention to during the ESP32 ESD test?

- The Electrostatic Discharge (ESD) test for ESP32 is conducted to ensure that the device has sufficient tolerance to withstand electrostatic discharge. The precautions are as follows: - ESD testing should be conducted in an ESD laboratory or ESD protection area, which should have good grounding protection and electrostatic discharge protection facilities. - When conducting ESD testing, please use ESD testing equipment that meets international standards, including ESD generators and ESD grounding mats, to ensure the accuracy of the test results. - When conducting ESD testing, please make sure you are using a stable 3.3 V voltage. If the EN trace is too long, it may cause a reboot. - ESP32 devices should be tested multiple times to verify the reliability of their tolerance, and the test results should be recorded and analyzed. - If the module does not respond, please check the voltage of the air discharge or contact discharge used in the test.

5.6 Production Test



5.6.1 Why can some modules download firmware normally when using DIO/DOUT, but encounter program abnormality when using QOUT/QIO?

- Firstly, please check the modes supported by flash in the module and whether the module routing meets the requirements of modes.
 - Secondly, please check the QE bit of the status register of flash, which determines whether the flash supports the QUAD mode or not.
 - Different ESP chips/modules use flashes from different manufacturers. Some flashes have QE disabled by default. Thus, it is necessary to check whether the flash supports Quad mode through actual testing.
 - When ROM boots a second stage bootloader, the secondary read will fail if the configuration parameters are read in the QIO mode because the QE bit is disabled.
 - It is recommended to program firmware in the DIO mode and to configure the QIO mode in `menuconfig`. The configuration enables the QE bit in the second stage bootloader and then boots the app bin to use the QUAD mode.
-

5.6.2 How to get the production test tool?

CHIP: ESP32 | ESP8266

- Please click [production test tool](#) to download.
-

5.6.3 When I use the `esptool.py burn_custom_mac` command to write the user-defined MAC address, why is the MAC address read by the `esptool.py read_mac` command still factory default?

- The `esptool.py read_mac` command can only read the MAC address written in eFuse BLOCK0 by default, but the user-defined MAC address written with the `esptool.py burn_custom_mac` command is in eFuse BLOCK3. You may use the `espefuse.py get_custom_mac` command to check the MAC address written to eFuse BLOCK3.
 - For more information, please refer to [esptool documentation](#).
-

5.6.4 When downloading bin files to ESP32-WROVER-E (16 MB flash) using Flash Download Tool, multiple separate bin files can be downloaded successfully, but downloading the combined firmware (12 MB) failed. Why?

Since the combined firmware is mostly “0xFF” with relatively high compression rate, the amount of data after decompression would be relatively large for the same length of compressed data, resulting in a timeout error (default 7 seconds) after a long download time. To solve this issue, in Flash Download Tool, go to `configure > esp32 > spi_download`, and disable the compression configuration option as follows:

```
compress = False  
no_compress = True
```

TEST VERIFICATION

□

6.1 Power Consumption Verification

□

6.1.1 Why does ESP32 reboot when it is woken up from Deep-sleep mode?

When ESP32 is in Deep-sleep mode, the digital core is powered off and the information stored in CPU will be lost. After ESP32 is woken up from Deep-sleep mode, it re-boots firmwares and re-loads them to the internal memory. The application information that requires to be reserved can be saved in RTC, as RTC is still powered on in Deep-sleep mode. The reserved information can be loaded after wake-up.

6.1.2 What sleep modes does ESP32 support? What is the difference between them?

ESP32 supports three sleep modes: Modem-sleep, Light-sleep, and Deep-sleep.

- **Modem-sleep:** CPU works normally and the clock is configurable. The station (ESP32) automatically turns on after it is connected to the AP. After ESP32 enters Modem-sleep mode, the RF module is shut down, and the station remains connected to the AP. If ESP32 disconnects to the AP, it will not work in Wi-Fi Modem-sleep mode. In Modem-sleep mode, the CPU clock frequency can be lowered to further reduce the current consumption.
- **Light-sleep:** CPU is suspended and the digital core clock is limited. When ESP32 is in Light-sleep mode, not only the RF module is closed, CPU and partial system clocks are also suspended. After ESP32 exits Light-sleep mode, the CPU resumes working.
- **Deep-sleep:** The digital core is powered off and the information stored in CPU is lost. After ESP32 enters Deep-sleep mode, all modules are closed except for RTC. After it exits Deep-sleep mode, the entire system restarts, which is similar to the system reboot. ESP32 does not remain connected to the AP in Deep-sleep mode.

Please refer to *Table 8: Power Consumption by Power Modes* in [ESP32 datasheet](#) for the corresponding sleep power consumption.

6.1.3 Can ESP32 in Deep-sleep mode be woken up by any RTC_GPIO?

Yes. For the configuration of RTC_GPIO, please refer to [ESP32 datasheet](#) > Chapter *Pin Definitions* > Section *Pin Description*.

6.1.4 What is the power consumption of ESP8266 when the CHIP_PU pin is at the low level?

- CHIP_PU pin is the module EN pin. When the pin is set to the low level, the power consumption of the chip is about 0.5 A.
 - In Table Power Consumption by Power Modes of [ESP8266 Datasheet](#) > *Functional Description* > *Power Management* > Table 3-4. *Power Consumption by Power Modes*, shut down power mode means CHIP_PU is pulled down and the chip is disabled.
-

6.1.5 Why does the minimum current of ESP32 in Light-sleep increase when the timer is not used as a wakeup source?

- By default, to avoid potential issues, `esp_light_sleep_start` functions will not power down flash. This is to prevent errors that may be caused if the flash is not fully powered off and back on when the device has just gone to sleep and is immediately woken up.
 - For the issue details and on how to optimize power consumption in this scenario please refer to [Power-down of Flash](#) in the ESP-IDF Programming Guide.
-

6.1.6 In ESP32's Deep-sleep mode, using an internal 150 KHz RTC clock or using an external 32 KHz, which consumes more power?

- If the RTC clock source is external 32 kHz crystal, there is no difference in power consumption.
 - If an external 32 kHz oscillator is connected to the hardware, the power consumption will increase by 50 to 100 A regardless of which RTC clock source is selected.
-

6.1.7 What are the requirements for CPU frequency to ensure normal operation of the RF module when reducing power consumption by reducing the CPU frequency?

CPU frequency should be 80 Mhz at least.

COMMERCIAL FAQ



7.1 Which certificates have your products passed?

Please check our [Certificates](#) , where you can get all the relevant information about our products.

7.2 Does your company have the ISO Quality Management System Certification?

Yes, our company has passed the ISO9001:2015 Quality Management System Certification.

7.3 Do your chips and modules have environmental certificates such as REACH, ROHS, etc?

Our chips and modules comply with REACH, ROHS, Prop65 and many other environmental certification standards. To find out more about them, please contact our business support team by submitting a [Sales Questions](#) electronic form, where you should mention the specific environmental certificate you need to check.

7.4 Do you have distributors in China, Europe, the United States and Canada?

To get specific information about our worldwide distributors, you should contact us by filling in the required information on our [Sales Questions](#) webpage. Then, our business support team will contact you as soon as possible, giving you all the information you need.

7.5 How can I start a distribution business with Espressif?

If you are interested in becoming one of our distributors, please send your company information to: sales@espressif.com.

7.6 Where can I find your product information? Which of your products are in mass production?

You can get the basic information on our products by clicking [here](#) . If you are looking for the technical documents of our products, please click [here](#) .

7.7 Do your products have a longevity commitment?

Yes, Espressif provides a minimum longevity commitment of at least 12 years for all the products listed [here](#) .

7.8 Where can I find the SPQ (Standard Pack Quantity) and MOQ (Minimum Order Quantity) for your products?

Please refer to our [Product Ordering Information](#) , where you can find our products' SPQ and MOQ.

7.9 What is your recommended purchasing method?

If you need to make a bulk purchase, please go to our [Sales Questions](#) and fill in the required information. Then, our business support team will contact you as soon as possible. If you just want to buy samples, please click [here](#) to check the corresponding purchasing method.

7.10 What's the price for bulk purchasing? How can I purchase in bulk?

Please go to our [Sales Questions](#) and fill in the electronic form you will find there. Then, our business support team will contact you as soon as possible.

7.11 Where can I find all the differences between your products (e.g. in terms of series and types)?

Please click [here](#) to find some introductory information on our products. For detailed information, please contact our sales team by clicking [here](#) .

7.12 Do your products have firmware? Can I customize my module/chip flash before the product leaves the factory? How much does this process cost? How long does it take? How can you help me do this?

Espressif Systems has developed a set of AT commands that can be used for Espressif products to easily interface with other products. Most of our modules have a standard AT firmware by default. For more information, please go to our [Sales Questions](#) and fill in the required details. Our business support team will contact you as soon as possible. Additionally, in order to simplify and shorten our customers' manufacturing process, we also provide customized manufacturing services. You can go to our [Manufacturing Services](#) and check all the available flash projects. For more information, please go to our [Sales Questions](#) and fill in the required details. Then, our business support team will get in touch with you as soon as possible.

7.13 Which of your products support HomeKit? Where can I get the Espressif HomeKit SDK?

You can refer to the [Espressif HomeKit SDK](#) . Please note that the Espressif HomeKit SDK is available to MFi licensees only, and you need to provide your Account Number (6 digits) for verification purposes, when [requesting the SDK](#) .

7.14 What is your company's address?

Espressif Systems (688018.SH) is a public multinational, fabless semiconductor company established in 2008, with offices in China, the Czech Republic, India, Singapore and Brazil. Please click [here](#) to check the details of Espressif's global offices.

7.15 How can I contact your technical team?

Please click [here](#) and tell us your problems or questions. We will try to help you as soon as possible.

7.16 How can I get in touch with your company?

In order to better understand your questions and needs, please click [here](#) and fill in the required information. Then, we will get in touch with you as soon as possible.

7.17 How can I tell if an Espressif module is in mass production or an NPI product?

On each Espressif module, you can find a specification identifier (4 digits to 9 digits) in the lower left corner of the module marking. All Espressif mass production modules have a specification identifier starting with XX or Mn (n can be 0, 1, 2, . . . for example, M0). The rest of them are NPI products. Find more information about [Espressif Module Packing Information](#) here.

Also please note that NPI product can be different from the final mass production product and may experience hardware or software issues. Thank you for your interests in our new products and helping us providing better products. Please contact our [technical support team](#) if you have any questions using our NPI products.