




# ESPRESSIF ESP32-S3-WROOM-1 Bluetooth Module User Manual

[Home](#) » [ESPRESSIF](#) » ESPRESSIF ESP32-S3-WROOM-1 Bluetooth Module User Manual 



**ESP32-S3-WROOM-1**



**ESP32-S3-WROOM-1U**

2.4 GHz WiFi (802.11 b/g/n) and Bluetooth5 (LE) module  
Built around ESP32S3 series of SoCs, Xtensa® dualcore 32bit LX7 microprocessor  
Flash up to 16 MB, PSRAM up to 8 MB  
36 GPIOs, a rich set of peripherals  
Onboard PCB antenna or external antenna connector

## Contents

- [1 Module Overview](#)
- [2 Pin Definitions](#)
- [3 Get Started](#)
- [4 U.S. FCC Statement](#)
- [5 IC Statement](#)
- [6 Related Documentation and Resources](#)
- [7 Documents / Resources](#)
  - [7.1 References](#)
- [8 Related Posts](#)

## Module Overview

### 1.1 Features

#### CPU and OnChip Memory

- ESP32-S3 series of SoCs embedded, Xtensa ® dual-core 32-bit LX7 microprocessor, up to 240 MHz
- 384 KB ROM
- 512 KB SRAM
- 16 KB SRAM in RTC
- Up to 8 MB PSRAM

#### WiFi

- 802.11 b/g/n
- Bit rate: 802.11n up to 150 Mbps
- A-MPDU and A-MSDU aggregation
- 0.4 µs guard interval support
- Center frequency range of operating channel: 2412 ~ 2462 MHz

#### Bluetooth

- Bluetooth LE: Bluetooth 5, Bluetooth mesh
- 2 Mbps PHY
- Long-range mode
- Advertising extensions
- Multiple advertisement sets
- Channel selection algorithm #2

#### Peripherals

- GPIO, SPI, LCD interface, Camera interface, UART, I2C, I2S, remote control, pulse counter, LED PWM, USB 1.1 OTG, USB Serial/JTAG controller, MCPWM, SDIO host, GDMA, TWAI ® controller (compatible with ISO 11898-1), ADC, touch sensor, temperature sensor, timers and watchdogs

## Integrated Components on Module

- 40 MHz crystal oscillator
- Up to 16 MB SPI flash

## Antenna Options

- On-board PCB antenna (ESP32-S3-WROOM-1)
- External antenna via a connector (ESP32-S3-WROOM-1U)

## Operating Conditions

- Operating voltage/Power supply: 3.0 ~ 3.6 V
- Operating ambient temperature:
  - 65 °C version: –40 ~ 65 °C
  - 85 °C version: –40 ~ 85 °C
  - 105 °C version: –40 ~ 105 °C
- Dimensions: See Table 1

## 1.2 Description

ESP32-S3-WROOM-1 and ESP32-S3-WROOM-1U are two powerful, generic Wi-Fi + Bluetooth LE MCU modules that are built around the ESP32-S3 series of SoCs. On top of a rich set of peripherals, the acceleration for neural network computing and signal processing workloads provided by the SoC makes the modules an ideal choice for a wide variety of application scenarios related to AI and Artificial Intelligence of Things (IoT), such as wake word detection, speech commands recognition, face detection, and recognition, smart home, smart appliances, smart control panel, smart speaker, etc.

ESP32-S3-WROOM-1 comes with a PCB antenna. ESP32-S3-WROOM-1U comes with an external antenna connector. A wide selection of module variants is available for customers as shown in Table 1. Among the module variants, those embedded ESP32-S3R8 operate at –40 ~ 65 °C ambient temperature, ESP32-S3-WROOM-1-H4 and ESP32-S3-WROOM-1U-H4 operate at –40 ~ 105 °C ambient temperature, and other module variants operate at –40 ~ 85 °C ambient temperature.

### Table 1: Ordering Information

Ordering Code	Chip Embedded	Flash (MB)	PSRAM (MB)	Dimensions (mm)
ESP32-S3-WROOM-1-N4	ESP32-S3	4	0	18 × 25.5 × 3.1
ESP32-S3-WROOM-1-N8	ESP32-S3	8	0	
ESP32-S3-WROOM-1-N16	ESP32-S3	16	0	
ESP32-S3-WROOM-1-H4 (105 °C)	ESP32-S3	4	0	
ESP32-S3-WROOM-1-N4R2	ESP32-S3R2	4	2 (Quad SPI)	
ESP32-S3-WROOM-1-N8R2	ESP32-S3R2	8	2 (Quad SPI)	
ESP32-S3-WROOM-1-N16R2	ESP32-S3R2	16	2 (Quad SPI)	
ESP32-S3-WROOM-1-N4R8 (65 °C)	ESP32-S3R8	4	8 (Octal SPI)	
ESP32-S3-WROOM-1-N8R8 (65 °C)	ESP32-S3R8	8	8 (Octal SPI)	
ESP32-S3-WROOM-1-N16R8 (65 °C)	ESP32-S3R8	16	8 (Octal SPI)	
ESP32-S3-WROOM-1U-N4	ESP32-S3	4	0	18 × 19.2 × 3.2
ESP32-S3-WROOM-1U-N8	ESP32-S3	8	0	
ESP32-S3-WROOM-1U-N16	ESP32-S3	16	0	
ESP32-S3-WROOM-1U-H4 (105 °C)	ESP32-S3	4	0	
ESP32-S3-WROOM-1U-N4R2	ESP32-S3R2	4	2 (Quad SPI)	
ESP32-S3-WROOM-1U-N8R2	ESP32-S3R2	8	2 (Quad SPI)	
ESP32-S3-WROOM-1U-N16R2	ESP32-S3R2	16	2 (Quad SPI)	
ESP32-S3-WROOM-1U-N4R8 (65 °C)	ESP32-S3R8	4	8 (Octal SPI)	
ESP32-S3-WROOM-1U-N8R8 (65 °C)	ESP32-S3R8	8	8 (Octal SPI)	
ESP32-S3-WROOM-1U-N16R8 (65 °C) )	ESP32-S3R8	16	8 (Octal SPI)	

At the core of the modules is an ESP32-S3 series of SoC \*, an Xtensa® 32-bit LX7 CPU that operates at up to 240 MHz. You can power off the CPU and make use of the low-power co-processor to constantly monitor the peripherals for changes or crossing of thresholds.

ESP32-S3 integrates a rich set of peripherals including SPI, LCD, Camera interface, UART, I2C, I2S, remote control, pulse counter, LED PWM, USB Serial/JTAG controller, MCPWM, SDIO host, GDMA, TWAI® controller (compatible with ISO 11898-1), ADC, touch sensor, temperature sensor, timers, and watchdogs, as well as up to 45 GPIOs. It also includes a full-speed USB 1.1 On-The-Go (OTG) interface to enable USB communication.

#### Note:

\* For more information on the ESP32-S3 series of SoCs, please refer to ESP32-S3 Series Datasheet.

## Pin Definitions

### 2.1 Pin Layout

The pin diagram is applicable for ESP32-S3-WROOM-1 and ESP32-S3-WROOM-1U, but the latter has no keep-out zone.

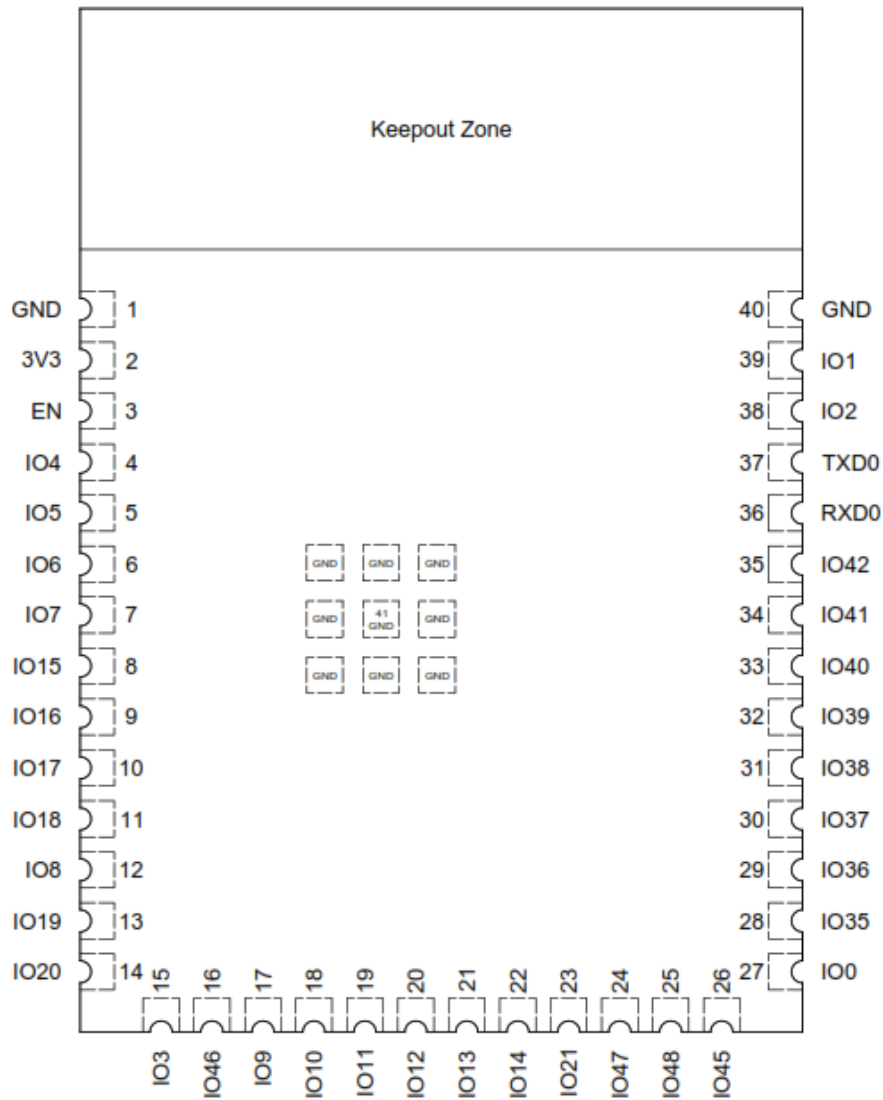


Figure 1: Pin Layout (Top View)

## 2.2 Pin Description

The module has 41 pins. See pin definitions in Table 2.

For explanations of pin names and function names, as well as configurations of peripheral pins, please refer to **ESP32-S3 Series Datasheet**.

Table 2: Pin Definitions

Name	No.	Type <sup>a</sup>	Function
GND	1	P	GND
3V3	2	P	Power supply
EN	3	I	High: on, enables the chip. Low: off, the chip powers off. Note: Do not leave the EN pin floating.
IO4	4	I/O/T	RTC_GPIO4, GPIO4, TOUCH4, ADC1_CH3
IO5	5	I/O/T	RTC_GPIO5, GPIO5, TOUCH5, ADC1_CH4
IO6	6	I/O/T	RTC_GPIO6, GPIO6, TOUCH6, ADC1_CH5

IO7	7	I/O/T	RTC_GPIO7, GPIO7, TOUCH7, ADC1_CH6
IO15	8	I/O/T	RTC_GPIO15, GPIO15, U0RTS, ADC2_CH4, XTAL_32K_P
IO16	9	I/O/T	RTC_GPIO16, GPIO16, U0CTS, ADC2_CH5, XTAL_32K_N
IO17	10	I/O/T	RTC_GPIO17, GPIO17, U1TXD, ADC2_CH6
IO18	11	I/O/T	RTC_GPIO18, GPIO18, U1RXD, ADC2_CH7, CLK_OUT3
IO8	12	I/O/T	RTC_GPIO8, GPIO8, TOUCH8, ADC1_CH7, SUBSPICS1
IO19	13	I/O/T	RTC_GPIO19, GPIO19, U1RTS, ADC2_CH8, CLK_OUT2, USB_D-
IO20	14	I/O/T	RTC_GPIO20, GPIO20, U1CTS, ADC2_CH9, CLK_OUT1, USB_D+
IO3	15	I/O/T	RTC_GPIO3, GPIO3, TOUCH3, ADC1_CH2
IO46	16	I/O/T	GPIO46
IO9	17	I/O/T	RTC_GPIO9, GPIO9, TOUCH9, ADC1_CH8, FSPIHD, SUSPEND
IO10	18	I/O/T	RTC_GPIO10, GPIO10, TOUCH10, ADC1_CH9, FSPICS0, FSPIIO4, SUBSPICS0
IO11	19	I/O/T	RTC_GPIO11, GPIO11, TOUCH11, ADC2_CH0, FSPID, FSPIIO5, SUSPEND
IO12	20	I/O/T	RTC_GPIO12, GPIO12, TOUCH12, ADC2_CH1, FSPICLK, FSPIIO6, SUBSPICLK
IO13	21	I/O/T	RTC_GPIO13, GPIO13, TOUCH13, ADC2_CH2, FSPIQ, FSPIIO7, SUBSPIQ
IO14	22	I/O/T	RTC_GPIO14, GPIO14, TOUCH14, ADC2_CH3, FSPIWP, FSPIDQS, SUBSPIWP
IO21	23	I/O/T	RTC_GPIO21, GPIO21
IO47	24	I/O/T	SPICLK_P_DIFF,GPIO47, SUBSPICLK_P_DIFF
IO48	25	I/O/T	SPICLK_N_DIFF,GPIO48, SUBSPICLK_N_DIFF
IO45	26	I/O/T	GPIO45
IO0	27	I/O/T	RTC_GPIO0, GPIO0
IO35 <sup>b</sup>	28	I/O/T	SPIIO6, GPIO35, FSPID, SUBSPID
IO36 <sup>b</sup>	29	I/O/T	SPIIO7, GPIO36, FSPICLK, SUBSPICLK
IO37 <sup>b</sup>	30	I/O/T	SPIDQS, GPIO37, FSPIQ, SUBSPIQ
IO38	31	I/O/T	GPIO38, FSPIWP, SUBSPIWP
IO39	32	I/O/T	MTCK, GPIO39, CLK_OUT3, SUBSPICS1
IO40	33	I/O/T	MTDO, GPIO40, CLK_OUT2
IO41	34	I/O/T	MTDI, GPIO41, CLK_OUT1

**Table 2 – contd from the previous page**

Name	No.	Type <sup>a</sup>	Function
IO42	35	I/O/T	MTMS, GPIO42
RXD0	36	I/O/T	U0RXD, GPIO44, CLK_OUT2
TXD0	37	I/O/T	U0TXD, GPIO43, CLK_OUT1
IO2	38	I/O/T	RTC_GPIO2, GPIO2, TOUCH2, ADC1_CH1
IO1	39	I/O/T	RTC_GPIO1, GPIO1, TOUCH1, ADC1_CH0
GND	40	P	GND
READ	41	P	GND

<sup>a</sup> P: power supply; I: input; O: output; T: high impedance. Pin functions in bold font are the default pin functions.

<sup>b</sup> In module variants that have embedded OSPI PSRAM, i.e., that embed ESP32-S3R8, pins IO35, IO36, and IO37 connect to the OSPI PSRAM and are not available for other uses.

## Get Started

### 3.1 What You Need

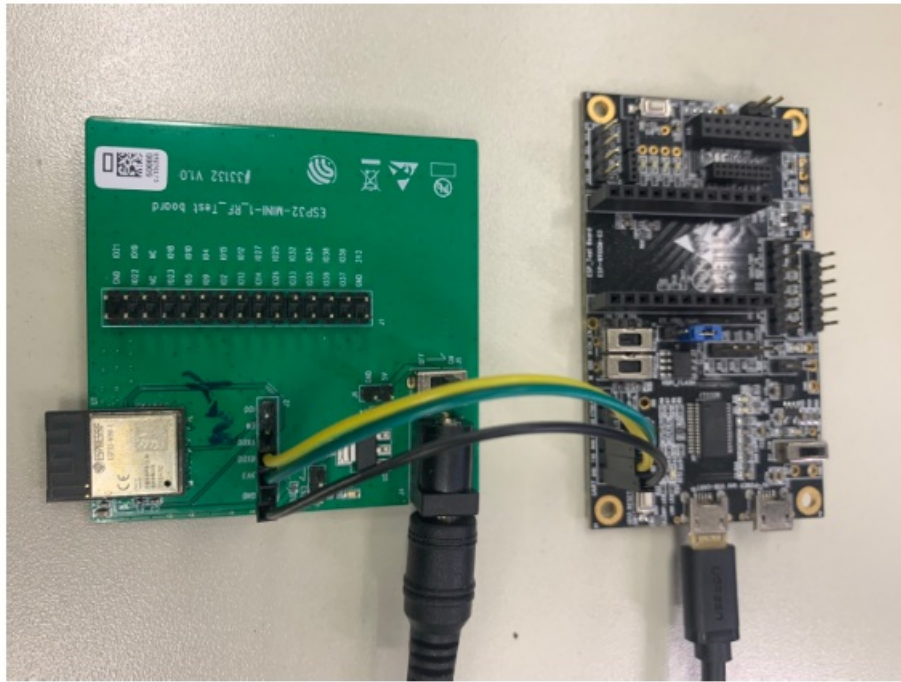
To develop applications for the module you need:

- 1 x ESP32-S3-WROOM-1 or ESP32-S3-WROOM-1U
- 1 x Espressif RF testing board
- 1 x USB-to-Serial board
- 1 x Micro-USB cable
- 1 x PC running Linux

In this user guide, we take Linux operating system as an example. For more information about the configuration on Windows and macOS, please refer to ESP-IDF Programming Guide.

### 3.2 Hardware Connection

1. Solder the ESP32-S3-WROOM-1 or ESP32-S3-WROOM-1U module to the RF testing board as shown in Figure 2.



**Figure 2: Hardware Connection**

2. Connect the RF testing board to the USB-to-Serial board via TXD, RXD, and GND.
3. Connect the USB-to-Serial board to the PC.
4. Connect the RF testing board to the PC or a power adapter to enable a 5 V power supply, via the Micro-USB cable.
5. During download, connect IO0 to GND via a jumper. Then, turn "ON" the testing board.
6. Download firmware into flash. For details, see the sections below.
7. After download, remove the jumper on IO0 and GND.
8. Power up the RF testing board again. The module will switch to working mode. The chip will read programs from flash upon initialization.

**Note:**

IO0 is internally logic high. If IO0 is set to pull-up, the Boot mode is selected. If this pin is pull-down or left floating, the Download mode is selected. For more information on ESP32-S3-WROOM-1 or ESP32-S3-WROOM-1U, please refer to ESP32-S3 Series Datasheet.

### 3.3 Set up Development Environment

The Espressif IoT Development Framework (ESP-IDF for short) is a framework for developing applications based on the Espressif ESP32. Users can develop applications with ESP32-S3 in Windows/Linux/macOS based on ESP-IDF. Here we take Linux operating system as an example.

#### 3.3.1 Install Prerequisites

To compile with ESP-IDF you need to get the following packages:

- **CentOS 7 & 8:**

- 1 `sudo yum -y update && Sudo yum install git wget flex bison gperf python3 python3pip`
- 2 `python3-setuptools CMake ninja-build ccache dfu-util busby`

- **Ubuntu and Debian:**

- 1 `Sudo apt-get install git wget flex bison gperf python3 python3-pip python3setuptools`
- 2 `cmake ninja-build ccache life-dev libssl-dev dfu-util libusb-1.0-0`

- **Arch:**

- 1 `sudo Pacman -S --needed GCC git make flex bison gperf python-pip CMake ninja ccache 2 dfu-util`



### Note:

- This guide uses the directory ~/esp on Linux as an installation folder for ESP-IDF.
- Keep in mind that ESP-IDF does not support spaces in paths.

### 3.3.2 Get ESPIDF

To build applications for ESP32-S3-WROOM-1 or ESP32-S3-WROOM-1U module, you need the software libraries provided by Espressif in the ESP-IDF repository.

To get ESP-IDF, create an installation directory ( ~/esp) to download ESP-IDF to and clone the repository with 'git clone':

1. `mkdir -p ~/esp`
2. `cd ~/esp`
3. `git clone --recursive https://github.com/espressif/esp-idf.git`

ESP-IDF will be downloaded into ~/esp/esp-idf. Consult ESP-IDF Versions for information about which ESP-IDF version to use in a given situation.

### 3.3.3 Set up Tools

Aside from the ESP-IDF, you also need to install the tools used by ESP-IDF, such as the compiler, debugger, Python packages, etc. ESP-IDF provides a script named 'install.sh' to help set up the tools in one go.

- 1 `cd ~/esp/esp-idf`
- 2 `./install.sh`

### 3.3.4 Set up Environment Variables

The installed tools are not yet added to the PATH environment variable. To make the tools usable from the command line, some environment variables must be set. ESP-IDF provides another script `export.sh` which does that. In the terminal where you are going to use ESP-IDF, run:

- 1 `. $HOME/esp/esp-IDF/export.sh`

Now everything is ready, you can build your first project on the ESP32-S3-WROOM-1 or ESP32-S3-WROOM-1U module.

## 3.4 Create Your First Project

### 3.4.1 Start a Project

Now you are ready to prepare your application for the ESP32-S3-WROOM-1 or ESP32-S3-WROOM-1U module. You can start with the `get-started/hello_world` project from the `examples` directory in ESP-IDF.

Copy `get-started/hello_world` to ~/esp directory:

- 1 `cd ~/esp`
- 2 `cp -r $IDF_PATH/examples/get-started/hello_world .`

There is a range of example projects in the `examples` directory in ESP-IDF. You can copy any project in the same way as presented above and run it. It is also possible to build examples in place, without copying them first.

### 3.4.2 Connect Your Device

Now connect your module to the computer and check under what serial port the module is visible. Serial ports in Linux start with `/dev/TTY` in their names. Run the command below two times, first with the board unplugged, then with plugged in. The port which appears the second time is the one you need:

- 1 `ls /dev/tty*`

### Note:

Keep the port name handy as you will need it in the next steps.

### 3.4.3 Configure

Navigate to your 'hello\_world' directory from Step 3.4.1. Start a Project, set ESP32-S3 chip as the target, and run the project configuration utility 'menuconfig'.

```
1 cd ~/esp/hello_world
2 idf.py set-target esp32s3
3 idf.py menuconfig
```

Setting the target with 'idf.py set-target esp32s3' should be done once, after opening a new project. If the project contains some existing builds and configurations, they will be cleared and initialized. The target may be saved in the environment variable to skip this step. See [Selecting the Target](#) for additional information.

If the previous steps have been done correctly, the following menu appears:

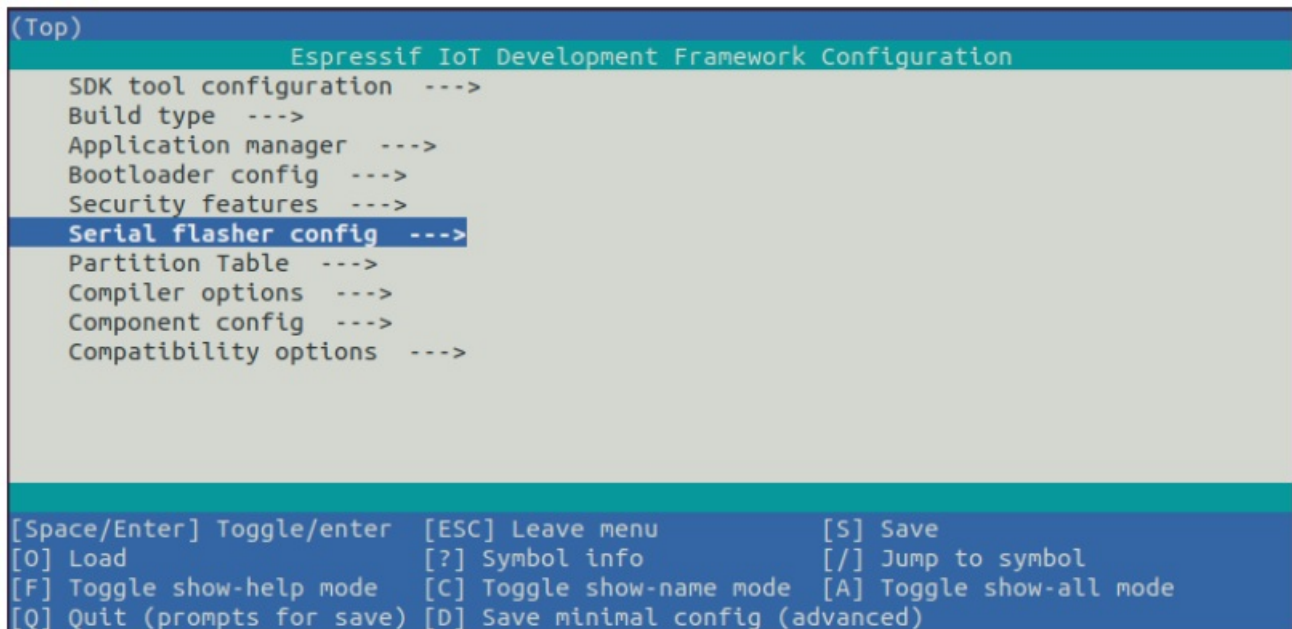


Figure 3: Project Configuration - Home Window

You are using this menu to set up project-specific variables, e.g. Wi-Fi network name and password, the processor speed, etc. Setting up the project with menuconfig may be skipped for "hello\_word". This example will run with the default configuration. The colors of the menu could be different in your terminal. You can change the appearance with the option '-style'. Please run 'idf.py menuconfig --help' for further information.

### 3.4.4 Build the Project

Build the project by running:

```
1 idf.py build
```

This command will compile the application and all ESP-IDF components, then it will generate the bootloader, partition table, and application binaries.

```
1 $ idf.py build
2 Running CMake in directory /path/to/hello_world/build
3 Executing "CMake -G Ninja --warn-uninitialized /path/to/hello_world"...
4 Warn about uninitialized values.
5 — Found Git: /usr/bin/git (found version "2.17.0")
6 — Building empty aws_iot component due to configuration
7 — Component names: ...
8 — Component paths: ...
9
10 ... (more lines of build system output)
11
12 [527/527] Generating hello_world.bin
13 esptool.py v2.3.1
14
15 Project build complete. To flash, run this command:
16 ../../components/esptool_py/esptool/esptool.py -p (PORT) -b 921600
```

```
17 write_flash -flash_mode dio -flash_size detect -flash_freq 40m
18 0x10000 build/hello_world.bin build 0x1000 build/bootloader/bootloader.bin 0x8000
19 build/partition_table/partition-table.bin
20 or run 'idf.py -p PORT flash'
```

If there are no errors, the build will finish by generating the firmware binary .bin file.

### 3.4.5 Flash onto the Device

Flash the binaries that you just built onto your module by running:

```
1 idf.py -p PORT [-b BAUD] flash
```

Replace PORT with your ESP32-S3 board's serial port name from Step: Connect Your Device.

You can also change the flasher baud rate by replacing BAUD with the baud rate you need. The default baud rate is 460800.

For more information on idf.py arguments, see idf.py.

#### Note:

The option 'flash' automatically builds and flashes the project, so running 'idf.py build' is not necessary.

When flashing, you will see the output log similar to the following:

```
1 ...
2 esptool.py esp32s3 -p /dev/ttyUSB0 -b 460800 --before=default_reset --after=hard_reset
3 write_flash -flash_mode dio -flash_freq 80m -flash_size 2MB 0x0 bootloader/bootloader.
bin
4 0x10000 hello_world.bin 0x8000 partition_table/partition-table.bin
5 esptool.py v3.2-dev
6 Serial port /dev/ttyUSB0
7 Connecting....
8 Chip is ESP32-S3
9 Features: WiFi, BLE
10 Crystal is 40MHz
11 MAC: 7c:df:a1:e0:00:64
12 Uploading stub...
13 Running stub...
14 Stub running...
15 Changing baud rate to 460800
16 Changed.
17 Configuring flash size...
18 Flash will be erased from 0x00000000 to 0x00004fff...
19 Flash will be erased from 0x00010000 to 0x00039fff...
20 Flash will be erased from 0x00008000 to 0x00008fff...
21 Compressed 18896 bytes to 11758...
22 Writing at 0x00000000... (100 %)
23 Wrote 18896 bytes (11758 compressed) at 0x00000000 in 0.5 seconds (effective 279.9 kbit/s)
...
24 Hash of data verified.
25 Compressed 168208 bytes to 88178...
26 Writing at 0x00010000... (16 %)
27 Writing at 0x0001a80f... (33 %)
28 Writing at 0x000201f1... (50 %)
29 Writing at 0x00025dcf... (66 %)
30 Writing at 0x0002d0be... (83 %)
31 Writing at 0x00036c07... (100 %)
32 Wrote 168208 bytes (88178 compressed) at 0x00010000 in 2.4 seconds (effective 569.2 kbit/s)
)...
33 Hash of data verified.
34 Compressed 3072 bytes to 103...
```

```
35 Writing at 0x00008000... (100 %)
36 Wrote 3072 bytes (103 compressed) at 0x00008000 in 0.1 seconds (effective 478.9 kbit/s)...
37 Hash of data verified.
38
39 Leaving...
40 Hard resetting via RTS pin...
41 Done
```

If there are no issues by the end of the flash process, the board will reboot and start up the “hello\_world” application.

### 3.4.6 Monitor

To check if “hello\_world” is indeed running, type ‘idf.py -p PORT monitor’ (Do not forget to replace PORT with your serial port name).

This command launches the IDF Monitor application:

```
1 $ idf.py -p /dev/ttyUSB0 monitor
2 Running idf_monitor in directory [...]esp/hello_world/build
3 Executing "python [...]esp-idf/tools/idf_monitor.py -b 115200
4 [...]esp/hello_world/build/hello-world.elf"...
5 — idf_monitor on /dev/ttyUSB0 115200 —
6 — Quit: Ctrl+] | Menu: Ctrl+T | Help: Ctrl+T followed by Ctrl+H —
7 ets Jun 8 2016 00:22:57
8
9 rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
10 ets Jun 8 2016 00:22:57
11 ...
```

After startup and diagnostic logs scroll up, you should see “Hello world!” printed out by the application.

```
1 ...
2 Hello world!
3 Restarting in 10 seconds...
4 This is esp32s3 chip with 2 CPU core(s), This is esp32s3 chip with 2 CPU core(s), WiFi/BLE
5 ,
6 silicon revision 0, 2MB external flash
7 Minimum free heap size: 390684 bytes
7 Restarting in 9 seconds...
8 Restarting in 8 seconds...
9 Restarting in 7 seconds...
```

To exit the IDF monitor use the shortcut Ctrl+].

That’s all that you need to get started with the ESP32-S3-WROOM-1 or ESP32-S3-WROOM-1U module! Now you are ready to try some other examples in ESP-IDF, or go right to developing your own applications.

## U.S. FCC Statement

This device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions:

- This device may not cause harmful interference.
- This device must accept any interference received, including interference that may cause undesired operation.

This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to Part15 of the FCC Rules.

These limits are designed to protect reasonably against harmful interference in a residential installation. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee

that interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one of the following measures:

- Reorient or relocate the receiving antenna.
- Increase the separation between the equipment and receiver.
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected.
- Consult the dealer or an experienced radio/TV technician for help.

Any changes or modifications not expressly approved by the party responsible for compliance could void the user's authority to operate the equipment.

This equipment complies with FCC RF radiation exposure limits set forth for an uncontrolled environment. This device and its antenna must not be co-located or operating in conjunction with any other antenna or transmitter.

The antennas used for this transmitter must be installed to provide a separation distance of at least 20 cm from all persons and must not be co-located or operating in conjunction with any other antenna or transmitter.

#### **OEM Integration Instructions**

This device is intended only for OEM integrators under the following conditions. The module can be used to install in another host. The antenna must be installed such that 20 cm is maintained between the antenna and users, and the transmitter module may not be co-located with any other transmitter or antenna. The module shall be only used with the integral antenna(s) that has been originally tested and certified with this module. As long as the 3 conditions above are met, further transmitter tests will not be required. However, the OEM integrator is still responsible for testing their end-product for any additional compliance requirement with this module installed (for example, digital device emission, PC peripheral requirements, etc.)

#### **Notice:**

In the event that these conditions cannot be met (for example certain laptop configuration or co-location with another transmitter), then the FCC authorization for this module in combination with the host equipment is no longer considered valid and the FCC ID of the module cannot be used on the final product. In these circumstances, the OEM integrator will be responsible for re-evaluating the end product (including the transmitter) and obtaining a separate FCC authorization.

#### **End Product Labeling**

This transmitter module is authorized only for use in devices where the antenna may be installed such that 20 cm may be maintained between the antenna and users. The final end product must be labeled in a visible area with the following: "Contains FCC ID: 2AC7Z-ESPS3WROOM1".

#### **IC Statement**

This device complies with Industry Canada's license-exempt RSS. Operation is subject to the following two conditions:

- This device may not cause interference; and
- This device must accept any interference, including interference that may cause undesired operation of the device.

#### **Radiation Exposure Statement**

This equipment complies with IC radiation exposure limits set forth for an uncontrolled environment. This equipment should be installed and operated with a minimum distance of 20 cm between the radiator & your body.

#### **RSS247 Section 6.4 (5)**

The device could automatically discontinue transmission in case of the absence of information to transmit or operational failure. Note that this is not intended to prohibit transmission of control or signaling information or the use of repetitive codes where required by the technology.

**This device is intended only for OEM integrators under the following conditions: (For module device use)**

- The antenna must be installed such that 20 cm is maintained between the antenna and users, and
- The transmitter module may not be co-located with any other transmitter or antenna.

As long as the 2 conditions above are met, further transmitter tests will not be required. However, the OEM integrator is still responsible for testing their end-product for any additional compliance requirements required with this module installed.

#### **IMPORTANT NOTE:**

In the event that these conditions can not be met (for example certain laptop configurations or colocation with another transmitter), then the Canada authorization is no longer considered valid and the IC ID can not be used on the final product. In these circumstances, the OEM integrator will be responsible for re-evaluating the end product (including the transmitter) and obtaining a separate Canada authorization.

#### **End Product Labeling**

This transmitter module is authorized only for use in devices where the antenna may be installed such that 20 cm may be maintained between the antenna and users. The final end product must be labeled in a visible area with the following: "Contains IC: 21098-ESPS3WROOM1".

#### **Manual Information To the End User**

The OEM integrator has to be aware not to provide information to the end user regarding how to install or remove this RF module in the user's manual of the end product which integrates this module. The end user manual shall include all required regulatory information/warning as shown in this manual.

### **Related Documentation and Resources**

#### **Related Documentation**

- ESP32-S3 Series Datasheet – Specifications of the ESP32-S3 hardware.
- ESP32-S3 Technical Reference Manual – Detailed information on how to use the ESP32-S3 memory and peripherals.
- ESP32-S3 Hardware Design Guidelines – Guidelines on how to integrate the ESP32-S3 into your hardware product.
- Certificates  
<http://espressif.com/en/support/documents/certificates>
- Documentation Updates and Update Notification Subscription  
<http://espressif.com/en/support/download/documents>

#### **Developer Zone**

- ESP-IDF Programming Guide for ESP32-S3 – Extensive documentation for the ESP-IDF development framework.
- ESP-IDF and other development frameworks on GitHub.  
<http://github.com/espressif>
- ESP32 BBS Forum – Engineer-to-Engineer (E2E) Community for Espressif products where you can post questions, share knowledge, explore ideas, and help solve problems with fellow engineers.  
<http://esp32.com/>
- The ESP Journal – Best Practices, Articles, and Notes from Espressif folks.  
<http://blog.espressif.com/>

- See the tabs SDKs and Demos, Apps, Tools, AT Firmware.

<http://espressif.com/en/support/download/sdks-demos>

## Products

- ESP32-S3 Series SoCs – Browse through all ESP32-S3 SoCs.

<http://espressif.com/en/products/socs?id=ESP32-S3>

- ESP32-S3 Series Modules – Browse through all ESP32-S3-based modules.

<http://espressif.com/en/products/modules?id=ESP32-S3>

- ESP32-S3 Series DevKits – Browse through all ESP32-S3-based devkits.

<http://espressif.com/en/products/devkits?id=ESP32-S3>

- ESP Product Selector – Find an Espressif hardware product suitable for your needs by comparing or applying filters.

<http://products.espressif.com/#/product-selector?language=en>

## Contact Us

- See the tabs Sales Questions, Technical Enquiries, Circuit Schematic & PCB Design Review, Get Samples (Online stores), Become Our Supplier, Comments & Suggestions.

<http://espressif.com/en/contact-us/sales-questions>

## Revision History

Date	Version	Release notes
10/29/2021	v0.6	Overall update for chip revision 1
7/19/2021	v0.5.1	Preliminary release, for chip revision 0



[www.espressif.com](http://www.espressif.com)

## Disclaimer and Copyright Notice

Information in this document, including URL references, is subject to change without notice.

ALL THIRD-PARTY'S INFORMATION IN THIS DOCUMENT IS PROVIDED AS IS WITH NO WARRANTIES TO ITS AUTHENTICITY AND ACCURACY.

NO WARRANTY IS PROVIDED TO THIS DOCUMENT FOR ITS MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE, NOR DOES ANY WARRANTY OTHERWISE ARISE OUT OF ANY PROPOSAL, SPECIFICATION, OR SAMPLE.

All liability, including liability for infringement of any proprietary rights, relating to the use of information in this document is disclaimed. No licenses express or implied, by estoppel or otherwise, to any intellectual property rights are granted herein.

The Wi-Fi Alliance Member logo is a trademark of the Wi-Fi Alliance. The Bluetooth logo is a registered trademark of Bluetooth SIG.

All trade names, trademarks, and registered trademarks mentioned in this document are the property of their

respective owners and are hereby acknowledged.

**Pre-release v0.6 Copyright**

**© 2022 Espressif Systems (Shanghai) Co., Ltd. All rights reserved.**

## Documents / Resources

	<p><a href="#">ESPRESSIF ESP32-S3-WROOM-1 Bluetooth Module</a> [pdf] User Manual</p> <p>ESP32- S3- WROOM -1, ESP32 -S3 -WROOM -1U, Bluetooth Module, ESP32- S3- WROOM -1 Bluetooth Module</p>
---	---

## References

- [The ESP Journal](#)
- [ESP32 Forum - Index page](#)
- [Sales Questions | Espressif Systems](#)
- [Development Boards | Espressif Systems](#)
- [Modules | Espressif Systems](#)
- [Chipsets | Espressif Systems](#)
- [Certificates | Espressif Systems](#)
- [Technical Documents | Espressif Systems](#)
- [SDKs & Demos | Espressif Systems](#)
- [Espressif Systems · GitHub](#)
- [ESP Product Selector](#)
- [Wi-Fi & Bluetooth MCUs and AIoT Solutions | Espressif Systems](#)
- [Build System - ESP32-S3 - — ESP-IDF Programming Guide latest documentation](#)
- [Build System - ESP32-S3 - — ESP-IDF Programming Guide latest documentation](#)
- [Get Started - ESP32-S3 - — ESP-IDF Programming Guide latest documentation](#)
- [ESP-IDF Versions - ESP32-S3 - — ESP-IDF Programming Guide latest documentation](#)
- [Wi-Fi & Bluetooth MCUs and AIoT Solutions | Espressif Systems](#)
- [Documentation Feedback | Espressif Systems](#)