

# Espressi ESP8684 MINI1 Smart Wi-Fi And Bluetooth Module User Manual

## Contents

- [1 Product Information](#)
- [2 Features](#)
- [3 Product Usage Instructions](#)
- [4 Module Overview](#)
- [5 Pin Definitions](#)
- [6 Get Started](#)
- [7 Documents / Resources](#)
  - [7.1 References](#)

## Product Information

### Module Overview

The ESP8684-WROOM-02UC is a general-purpose Wi-Fi and Bluetooth LE module. It is equipped with a rich set of peripherals and offers high performance, making it an ideal choice for various applications such as smart homes, industrial automation, health care, and consumer electronics.

## Features

- CPU and On-Chip Memory
- **Wi-Fi modes:** Station mode, SoftAP mode, Station + SoftAP mode, and promiscuous mode
- Bluetooth integration with shared antenna
- Peripherals and integrated components on module
- Antenna Options
- Operating Conditions

## Pin Definitions

The module has 19 pins with various functions. Please refer to the table below for pin definitions:

Name	No.	Type	Function
3V3	1	Power supply	High: on, enables the chip.
EN	2	I	Low: off, the chip powers off. Note: Do not leave the CHIP_EN pin floating.
IO4	3	I/O/T	GPIO4, ADC1_CH4, FSPIHD, MTMS
IO5	4	I/O/T	GPIO5, FSPIWP, MTDI
IO6	5	I/O/T	GPIO6, FSPICLK, MTCK
IO7	6	I/O/T	GPIO7, FSPID, MTDO
IO8	7	I/O/T	GPIO8 This pin is internally pulled high.
IO9	8	I/O/T	GPIO9

## Product Usage Instructions

### Hardware Connection

Follow the hardware connection instructions provided in the user manual to connect the ESP8684-WROOM-02UC module with the required components.

### Set up Development Environment

Follow the steps below to set up the development environment:

1. Install the prerequisites as mentioned in the user manual.
2. Download and install the ESP-IDF (Espressif IoT Development Framework).
3. Set up the necessary tools for development.
4. Configure the environment variables as instructed.

### Create Your First Project

To create your first project, follow these steps:

1. Start a new project using the provided instructions.
2. Connect your device to the development environment.
3. Configure the project settings.
4. Build the project.
5. Flash the project onto the ESP8684-WROOM-02UC module.
6. Monitor the project execution.

### U.S. FCC Statement

The ESP8684-WROOM-02UC module complies with U.S. FCC regulations. Refer to the user manual for more details.

## Related Documentation and Resources

For additional information and resources related to the ESP8684-WROOM-02UC module, please refer to the documentation provided by Espressif Systems.

## Module Overview

### Features

#### CPU and On-Chip Memory

- ESP8684H2 or ESP8684H4 embedded, 32-bit
  - RISC-V single-core processor, up to 120 MHz
- 576 KB ROM
- 272 KB SRAM (16 KB for cache)
- SiP flash (see details in Table 1
  - ESP8684-WROOM-02UC Ordering Information)
- Access to flash accelerated by cache
- Supports flash in-Circuit Programming (ICP)

#### Wi-Fi

- IEEE 802.11 b/g/n-compliant
- Center frequency range of operating channel: 2412 ~ 2462 MHz
- Supports 20 MHz bandwidth in 2.4 GHz band
- 1T1R mode with data rate up to 72.2 Mbps
- Wi-Fi Multimedia (WMM)
- TX/RX A-MPDU, TX/RX A-MSDU
- Immediate Block ACK
- Fragmentation and defragmentation
- Transmit opportunity (TXOP)
- Automatic Beacon monitoring (hardware TSF)
- 3 × virtual Wi-Fi interfaces
- Simultaneous support for Infrastructure BSS in Station mode, SoftAP mode, Station + SoftAP mode, and promiscuous mode Note that when ESP8684 series scans in Station mode, the SoftAP channel will change along with the Station channel.

#### Bluetooth®

- **Bluetooth LE:** Bluetooth 5
- **Speed:** 125 kbps, 500 kbps, 1 Mbps, 2 Mbps
- Advertising extensions
- Multiple advertisement sets
- Channel selection algorithm #2
- Internal co-existence mechanism between Wi-Fi and Bluetooth to share the same antenna

## Peripherals

- GPIO, SPI, UART, I2C, LED PWM controller, general DMA controller, temperature sensor, SAR ADC, timers and watchdogs

## Integrated Components on Module

- 26 MHz crystal oscillator

## Antenna Options

- External antenna via a connector

## Operating Conditions

- Operating voltage/Power supply: 3.0 ~ 3.6 V
- Operating ambient temperature: -40 ~ 85 °C

## Description

ESP8684-WROOM-02UC is a general-purpose Wi-Fi and Bluetooth LE module. The rich set of peripherals and high performance make this module an ideal choice for smart homes, industrial automation, health care, consumer electronics, etc. ESP8684-WROOM-02UC comes with a connector for an external antenna. The ordering information for ESP8684-WROOM-02UC is as follows:

**Table 1:** ESP8684-WROOM-02UC Ordering Information

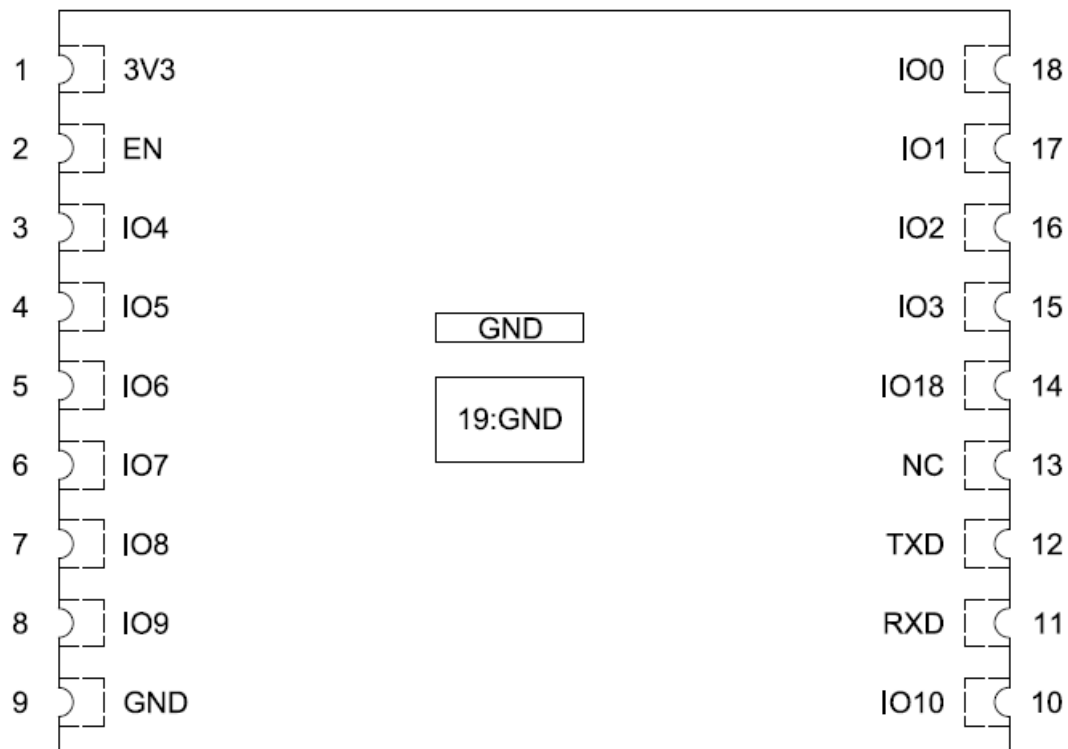
Module	Ordering Code	Chip Embedded	SiP Flash	Module Dimensions (mm)
ESP8684-WROOM-02UC	ESP8684-WROOM-02UC-N2	ESP8684H2	2 MB	14.3 × 18.0 × 3.2
	ESP8684-WROOM-02UC-N4	ESP8684H4	4 MB	

The ESP8684H2 chip and the ESP8684H4 chip fall into the same category, namely ESP8684 chip series. ESP8684 integrates a rich set of peripherals including UART, I2C, LED PWM controller, general DMA controller, temperature sensor, and SAR ADC.

**Note:** For more information on ESP8684, please refer to ESP8684 Series Datasheet.

## Pin Definitions

### Pin Layout



**Figure 1: Pin Layout (Top View)**

- The pin diagram below shows the approximate location of pins on the module.

### Pin Description

- The module has 19 pins. See pin definitions in Table 2.
- For peripheral pin configurations, please refer to ESP8684 Series Datasheet.

**Table 2: Pin Definitions**

Name	No.	Type <sup>1</sup>	Function
3V3	1	P	Power supply
EN	2	I	High: on, enables the chip. Low: off, the chip powers off. Note: Do not leave the CHIP_EN pin floating.
IO4	3	I/O/T	GPIO4, ADC1_CH4, FSPIHD, MTMS
IO5	4	I/O/T	GPIO5, FSPIWP, MTDI
IO6	5	I/O/T	GPIO6, FSPICLK, MTCK
IO7	6	I/O/T	GPIO7, FSPID, MTDO
IO8	7	I/O/T	GPIO8 This pin is internally pulled high.
IO9	8	I/O/T	GPIO9

GND	9	P	Ground
IO10	10	I/O/T	GPIO10, FSPICS0
RXD	11	I/O/T	GPIO19, U0RXD
TXD	12	I/O/T	GPIO20, U0TXD
NC	13	—	NC
IO18	14	I/O/T	GPIO18
IO3	15	I/O/T	GPIO3, ADC1_CH3
IO2	16	I/O/T	GPIO2, ADC1_CH2, FSPIQ
IO1	17	I/O/T	GPIO1, ADC1_CH1
IO0	18	I/O/T	GPIO0, ADC1_CH0
GND	19	P	Ground

1. **P**: power supply; **I**: input; **O**: output; **T**: high impedance.

## Get Started

### What You Need

To develop applications for module you need:

- 1 x ESP8684-WROOM-02UC

- 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.

## Hardware Connection

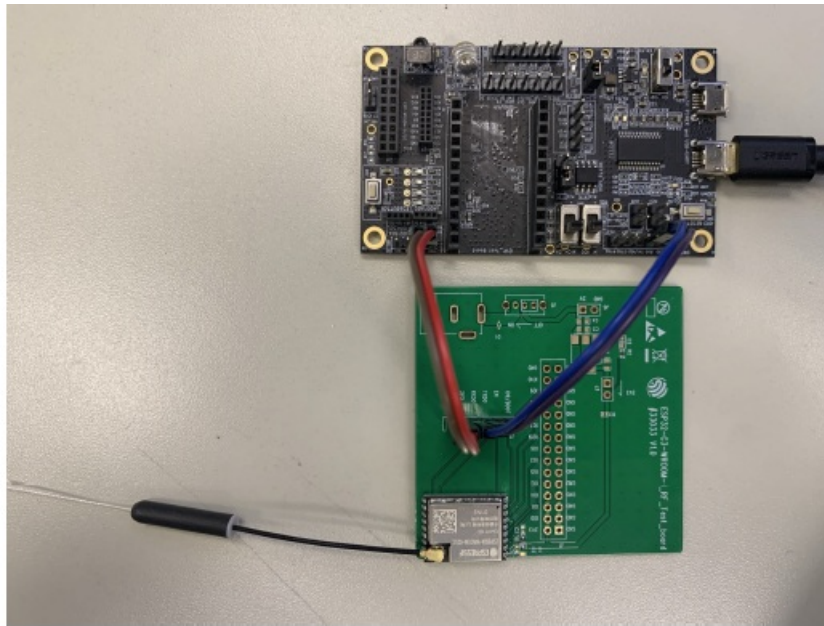


Figure 2: Hardware Connection

1. Solder the ESP8684-WROOM-02UC module to the RF testing board as shown in Figure 2.
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 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 ESP8684-WROOM-02UC, please refer to ESP8684 Series Datasheet.

## 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 ESP8684 in Windows/Linux/macOS based on ESP-IDF. Here we take Linux operating system as an example.

## Install Prerequisites

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

- CentOS 7 & 8:
  - `sudo yum -y update && sudo yum install git wget flex bison gperf python3 python3- pip`
  - `python3-setuptools cmake ninja-build ccache dfu-util libusbx`
- Ubuntu and Debian:
  - `sudo apt-get install git wget flex bison gperf python3 python3-pip python3- setuptools`
  - `cmake ninja-build ccache libffi-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 libusb`

### 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.

## Get ESP-IDF

To build applications for ESP8684-WROOM-02UC module, you need the software libraries provided by Espressif in 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.

## 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`

## 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 ESP8684-WROOM-02UC module.

## Create Your First Project

### *Start a Project*

Now you are ready to prepare your application for ESP8684-WROOM-02UC module. You can start with get-started/hello\_world project from 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.

## 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.

## Configure

Navigate to your 'hello\_world' directory from Step 3.4.1. Start a Project, set ESP8684 chip as the target and run the project configuration utility 'menuconfig'.

1. `cd ~/esp/hello_world`
2. `idf.py set-target esp8684`
3. `idf.py menuconfig`

Setting the target with 'idf.py set-target ESP8684' 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 at all. See Selecting the Target for additional information. If the previous steps have been done correctly, the following menu appears:

```
(Top)
Espressif IoT Development Framework Configuration
SDK tool configuration --->
Build type --->
Application manager --->
Bootloader config --->
Security features --->
Serial flasher config --->
Partition Table --->
Compiler options --->
Component config --->
Compatibility options --->

[Space/Enter] Toggle/enter  [ESC] Leave menu          [S] Save
[O] Load                   [?] Symbol info          [/] Jump to symbol
[F] Toggle show-help mode  [C] Toggle show-name mode [A] Toggle show-all mode
[Q] Quit (prompts for save) [D] Save minimal config (advanced)
```

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 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 menu config -help’ for further information.

## 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 ESP8684 board's serial port name from Step: Connect Your Device.

You can also change the flash 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.

## Documents / Resources

	<p><a href="#">Espressi ESP8684 MINI1 Smart Wi-Fi And Bluetooth Module</a> [pdf] User Manual          ESP8684 MINI1 Smart Wi-Fi And Bluetooth Module, ESP8684, MINI1 Smart Wi-Fi And Bluetooth Module, Smart Wi-Fi And Bluetooth Module, Wi-Fi And Bluetooth Module, Bluetooth Module</p>
---	---

## References

- [Wi-Fi & Bluetooth MCUs and AIoT Solutions | Espressif Systems](#)
- [The ESP Journal](#)
- [Build System - ESP32 - — ESP-IDF Programming Guide latest documentation](#)
- [Build System - ESP32 - — ESP-IDF Programming Guide latest documentation](#)
- [Get Started - ESP32 - — ESP-IDF Programming Guide latest documentation](#)

-  [ESP-IDF Versions - ESP32 - — ESP-IDF Programming Guide latest documentation](#)
-  [Get Started - ESP32-C2 - — ESP-IDF Programming Guide latest documentation](#)
-  [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](#)
-  [GitHub - espressif/esp-idf: Espressif IoT Development Framework. Official development framework for Espressif SoCs.](#)
-  [esp-idf/examples/get-started/hello\\_world at c77c4ccf6c43ab09fd89e7c907bf5cf2a3499e3b · espressif/esp-idf · GitHub](#)
-  [esp-idf/examples at master · espressif/esp-idf · GitHub](#)
-  [ESP Product Selector](#)
-  [Wi-Fi & Bluetooth MCUs and AIoT Solutions | Espressif Systems](#)
-  [Documentation Feedback | Espressif Systems](#)