



ESPRESSIF ESP8685-WROOM-04 WiFi and Bluetooth LE Module User Manual

[Home](#) » [ESPRESSIF](#) » ESPRESSIF ESP8685-WROOM-04 WiFi and Bluetooth LE Module User Manual 

Contents

- [1 ESPRESSIF ESP8685-WROOM-04 WiFi and Bluetooth LE Module](#)
- [2 Overview](#)
- [3 Get Started](#)
- [4 U.S. FCC Statement](#)
- [5 Learning Resources](#)
- [6 Documents / Resources](#)
 - [6.1 References](#)
- [7 Related Posts](#)



ESPRESSIF ESP8685-WROOM-04 WiFi and Bluetooth LE Module



Overview

Module Overview

ESP8685-WROOM-04 is a general-purpose Wi-Fi and Bluetooth LE module. The rich set of peripherals and a small size make this module an ideal choice for smart homes, industrial automation, health care, consumer electronics, etc.

ESP8685-WROOM-04 comes with a PCB antenna.

Table 1: ESP8685WROOM04 Specifications

Categories	Parameters	Specifications
Wi-Fi	Protocols	IEEE 802.11 b/g/n (1T1R mode with data rate up to 150 Mbps)
	Frequency range	2412 ~ 2462 MHz
Bluetooth®	Protocols	Bluetooth® LE: Bluetooth 5 and Bluetooth mesh
	Radio	Class-1, class-2 and class-3 transmitter
		AFH
	Audio	CVSD and SBC
Hardware	Module interfaces	GPIO, SPI, UART, I2C, I2S, remote control peripheral, LED PWM controller, general DMA controller, TWAI® c controller (compatible with ISO 11898-1), USB Serial/JTAG controller, temperature sensor, SAR ADC
	Integrated crystal	40 MHz crystal oscillator
	Operating voltage/Power supply	3.0 V ~ 3.6 V
	Operating current	Average: 80 mA
	Minimum current delivered by power supply	500 mA
	Ambient temperature	–40 °C ~ +105 °C
	Moisture sensitivity level (MSL)	Level 3

Pin Description

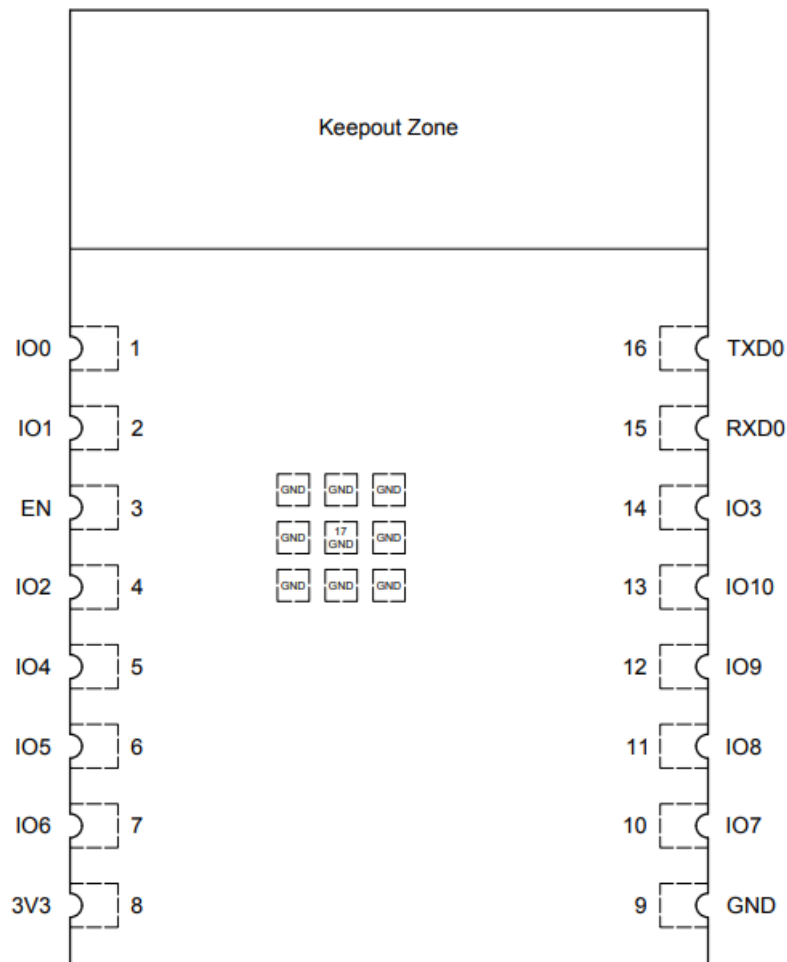


Figure 1: Pin Layout (Top View)

The module has 17 pins. See pin definitions in Table 2.

Table 2: Pin Definitions

Name	No.	Type ¹	Function
IO0	1	I/O/T	GPIO0 ADC1_CH0, XTAL_32K_P
IO1	2	I/O/T	GPIO1, ADC1_CH1, XTAL_32K_N
EN	3	I	High: on, enables the chip. Low: off, the chip powers off. Default: internally pulled-up
IO2	4	I/O/T	GPIO2, ADC1_CH2, FSPIQ
IO4	5	I/O/T	GPIO4, ADC1_CH4, FSPIHD, MTMS, LED PWM
IO5	6	I/O/T	GPIO5, ADC2_CH0, FSPIWP, MTDI, LED PWM
IO6	7	I/O/T	GPIO6, FSPICLK, MTCK, LED PWM
3V3	8	P	Power supply

Table 2 – cont'd from previous page

Name	No.	Type ¹	Function
GND	9,17	P	Ground
IO7	10	I/O/T	GPIO7, FSPID, MTDO, LED PWM
IO8	11	I/O/T	GPIO8
IO9	12	I/O/T	GPIO9
IO10	13	I/O/T	GPIO10, FSPICS0, LED PWM
IO3	14	I/O/T	GPIO3, ADC1_CH3, LED PWM
RXD0	15	I/O/T	GPIO20, U0RXD
TXD0	16	I/O/T	GPIO21, U0TXD

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

Get Started

What You Need

To develop applications for ESP8685-WROOM-04 module you need:

- 1 x ESP8685-WROOM-04 module
- 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

1. Solder the ESP8685-WROOM-04 module to the RF testing board as shown in Figure 2

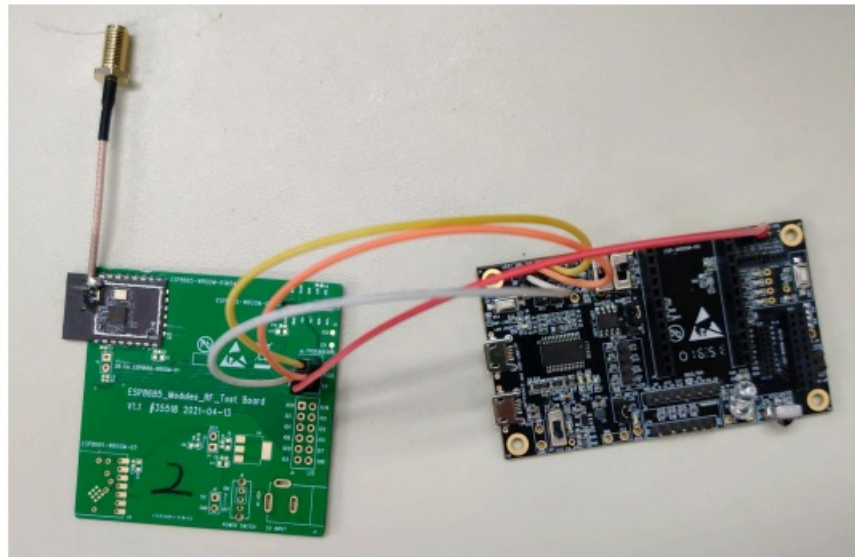


Figure 2: Hardware Connection

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 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 IO9 and GND.
8. Power up the RF testing board again. ESP8685-WROOM-04 will switch to working mode. The chip will read programs from flash upon initialization.

Note:

IO9 is internally logic high. If IO9 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 ESP8685-WROOM-04, please refer to ESP8685-WROOM-04 Datasheet.

Set up Development Environment

The Espressif IoT Development Framework (ESP-IDF for short) is a framework for developing applications based on the Espressif chips. Users can develop applications with ESP chips 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`
- Ubuntu and Debian:
`sudo apt-get install git wget flex bison gperf python3 python3-pip python3-setuptools cmake ninja-build ccache dfu-util libusb-1.0`
- Arch:
`sudo pacman -S --needed gcc git make flex bison gperf python-pip cmake ninja ccache dfu-util libusb`
- 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 ESPIDF

To build applications for ESP8685-WROOM-04 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': `mkdir -p ~/.esp cd ~/.esp 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.
`cd ~/.esp/esp-idf./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: `$HOME/esp/esp-idf/export.sh` Now everything is ready, you can build your first project on ESP8685-WROOM-04 module.

Create Your First Project

Start a Project

Now you are ready to prepare your application for ESP8685-WROOM-04 module. You can start with get-started/hello_world project from examples directory in ESP-IDF.

Copy get-started/hello_world to ~/.esp directory: `cd ~/.esp 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 ESP8685-WROOM-04 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: `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

Start a Project, set ESP8685 as the target and run the project configuration utility 'menuconfig'. `cd ~/.esp/hello_world idf.py set-target esp8685 idf.py menuconfig`

Setting the target with 'idf.py set-target esp8685' should be done once, after opening a new project. If the project contains some existing builds and configuration, they will be cleared and initialized. The target may be saved in 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)
```

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.

Build the Project

Build the project by running:

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.

```
$ idf.py build
Running cmake in directory /path/to/hello_world/build
Executing "cmake -G Ninja --warn-uninitialized /path/to/hello_world"...
Warn about uninitialized values.
-- Found Git: /usr/bin/git (found version "2.17.0")
-- Building empty aws_iot component due to configuration
-- Component names: ...
-- Component paths: ...

... (more lines of build system output)

[527/527] Generating hello-world.bin
esptool.py v2.3.1

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'
```

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

Flash onto the Device

Flash the binaries that you just built onto your ESP8685-WROOM-04 module by running:

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

Replace PORT with your module'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

```
...
esptool.py --chip esp8685 -p /dev/ttyUSB0 -b 460800 --before=default_reset --after=hard_reset writ
esptool.py v3.0
Serial port /dev/ttyUSB0
Connecting....
Chip is ESP8685
Features: Wi-Fi
Crystal is 40MHz
MAC: 7c:df:a1:40:02:a4
Uploading stub...
Running stub...
Stub running...
Changing baud rate to 460800
Changed.
Configuring flash size...
Compressed 3072 bytes to 103...
Writing at 0x00008000... (100 %)
Wrote 3072 bytes (103 compressed) at 0x00008000 in 0.0 seconds (effective 4238.1 kbit/s)...
Hash of data verified.
Compressed 18960 bytes to 11311...
Writing at 0x00000000... (100 %)
Wrote 18960 bytes (11311 compressed) at 0x00000000 in 0.3 seconds (effective 584.9 kbit/s)...
Hash of data verified.
Compressed 145520 bytes to 71984...
Writing at 0x00010000... (20 %)
Writing at 0x00014000... (40 %)
Writing at 0x00018000... (60 %)
Writing at 0x0001c000... (80 %)
Writing at 0x00020000... (100 %)
Wrote 145520 bytes (71984 compressed) at 0x00010000 in 2.3 seconds (effective 504.4 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
Done
```

If everything goes well, the "hello_world" application starts running after you remove the jumper on IO0 and GND, and re-power up the testing board.

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:

```

$ idf.py -p /dev/ttyUSB0 monitor
Running idf_monitor in directory [...]/esp/hello_world/build
Executing "python [...]/esp-idf/tools/idf_monitor.py -b 115200 [...]/esp/hello_world/build/hello-world.elf
--- idf_monitor on /dev/ttyUSB0 115200 ---
--- Quit: Ctrl+] | Menu: Ctrl+T | Help: Ctrl+T followed by Ctrl+H ---
ets Jun  8 2016 00:22:57

rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
ets Jun  8 2016 00:22:57
...

```

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

```

...
Hello world!
Restarting in 10 seconds...
This is esp8685 chip with 1 CPU core, WiFi/BLE
Restarting in 9 seconds...
Restarting in 8 seconds...
Restarting in 7 seconds...

```

To exit IDF monitor use the shortcut Ctrl+].

That’s all what you need to get started with ESP8685-WROOM-04 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

FCC ID: 2AC7ZESP868504

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 Part 15 of the FCC Rules. These limits are designed to provide reasonable protection 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.

Caution:

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 installation 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 transmit 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 3 conditions above are met, further transmitter test 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 and circumstance, the OEM integrator will be responsible for re-evaluating. The end product (including the transmitter) and obtaining a separate FCC authorization.

The final end product must be labeled in a visible area with the following: "Contains Transmitter Module FCC ID: 2AC7ZESP868504"

Learning Resources

MustRead Documents

Please familiarize yourself with the following documents:

- ESP-IDF Programming Guide

Extensive documentation for the ESP-IDF development framework, ranging from hardware guides to API reference.

- Espressif Products Ordering Information

Important Resources

Here are the important ESP8685-related resources.

- ESP32 BBS

Engineer-to-Engineer (E2E) Community for Espressif products where you can post questions, share knowledge, explore ideas, and help solve problems with fellow engineers.

Revision History

Date	Version	Release notes
2021-05-10	V0.1	Preliminary release

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, NONINFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, NOR DOES ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION OR SAMPLE.

All liability, including liability for infringement of any proprietary rights, relating to 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 property of their respective owners, and are hereby acknowledged.

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

Documents / Resources

	<u>ESPRESSIF ESP8685-WROOM-04 WiFi and Bluetooth LE Module</u> [pdf] User Manual ESP868504, 2AC7Z-ESP868504, 2AC7ZESP868504, ESP8685 -WROOM- 04 Module, ESP8685 -WROOM- 04, Module, ESP8685 -WROOM- 04 WiFi and Bluetooth LE Module, WiFi and Bluetooth LE Module, Bluetooth LE Module, LE Module
---	--

References

-  [Certificates | Espressif Systems](#)
-  [Subscribe | Espressif Systems](#)
-  [Wi-Fi & Bluetooth MCUs and AIoT Solutions | Espressif Systems](#)
-  [ESP32 Forum - Index page](#)
-  [Wi-Fi & Bluetooth MCUs and AIoT Solutions | Espressif Systems](#)
-  [Technical Documents | Espressif Systems](#)