ESPRESSIF ESP32-C6-MINI-1 2.4 GHz Wi-Fi Module User Manual

ESP32-C6-MINI-1
User Manual

Contents

ESP32-C6-MINI-1 2.4 GHz Wi-Fi Module

Module that supports 2.4 GHz Wi-Fi 6 (802.11 ax), Bluetooth® 5 (LE), Zigbee and Thread (802.15.4)
Built around ESP32-C6 series of SoCs, 32-bit RISC-V single-core microprocessor
4 MB flash in chip package
22 GPIOs, rich set of peripherals
On-board PCB antenna


Pre-release v1.0
Espressif Systems

www.espressif.com

Module Overview

1.1 Features

CPU and On-Chip Memory
ESP32-C8FH4 embedded. 32-bit RISC-V single-core microprocessor, up to 160 MHz

- ROM: 320 KB
- HP SRAM: 512 KB
- LP SRAM: 16 KB
- 4.NB fash in Chip package

Wi-Fi

- 1T1Rin 2.4 GHz band
- Operating frequency: 2412 ~ 2462 MHz
- (EEE 802.1 1ax-compiant
  – 20 MHz-only non-AP mode
  – MCSO -MCS9
  — Uplink and downlink OFDMA. especialy Suitable for samultangous connections in
  — Downlink MU-MIMO (mutti-user, muftiple input, muftiple output) to increase network capacay
  — Baamformee that improves signal quality
  — Channel quality indication (CO

— DCM (dual carrier moduiation) to improve link robusiness
— Spatial reuse to maximize parallel transmissions
— Target wake time (TW) that optimizes Power saving mechanisms

- Fuly compatible with IEEE 802.11 b/g/n protocol
  — 20 MHz and 40 MHz bandvadth
  — Data rate up to 150 Mbps
  — Wi-Fi Muttimedia (WM)
  — TX/AX A-MPDU, TX/RX A-MSDU
  — Immedate Block ACK
  — Fragmentation and defragmentation
  — Transmit opportunity (TXOP)
  — Automatic Beacon monitoring (hardware TSF)
  — 4 virtual Wi-Fi interfaces
  — Simutaneous support for Infrastructure BSS in Station mode, SoftAP mode, Station + SORAP mode. and promiscuous mode Note that when ESP32-O6 scars itt Station mode, the SomAP channel wil change along with the Staton chance’
  — 802.11m¢ FTM

Bluetooth

- Bauetooth LE: Bauetooth 5.3 cartified
- Bauetooth mesh
- High power mode
- Speed: 1 Mbps, 2 Mops
- Advertising extensions
- Multiple advertisement sets
- Channel salection aigonthm #2
- LE power control
- Internal co-existence mechanism between Wi-Fi and Bauetooth to share the same antenna IEEE 802.15.4
- Compliant with IEEE 802.15.4-2015 protocol
- COPSK PHY in 2.4 GHz band
- Data rate: 250 Kbps
- Thread 1.3
- Zigbee 3.0

Peripherals

- GPIO, SPI, parallel |O interface, UART, I2C, I2S, RMT (TX/RX), pulse counter, LED PWM, USB Serial/JTAG controller, MCPWM, SDIO2.0 slave controller, GDMA, TWAI® controller, on-chip debug functionality via JTAG, event task matrix, ADC, temperature sensor, general-purpose timers, watchdog timers, etc.

Integrated Components on Module

- 40 MHz crystal oscillator

Antenna Options

- On-board PCB antenna

Operating Conditions

- Operating voltage/Power supply: 3.0 ~ 3.6 V
- Operating ambient temperature:
  – 85 °C version module: —40 ~ 85 °C
  – 105 °C version module: —40 ~ 105 °C

1.2 Description
ESP32-C6-MINI-1 is a general-purpose Wi-Fi, IEEE 802.15.4, and Bluetooth LE module. The rich set of peripherals and high performance make the module an ideal choice for smart homes, industrial automation, health care, consumer electronics, etc.
The ordering information for ESP32-C6-MINI-1 is as follows:
Table 1: ESP32-C6-MINI-1 Ordering Information

Ordering Code	Flash	Ambient Temp. CC)	Size (mm)
ESP32-C6-MINI-1-N4	4 MB (Quad SPI)	-40 -v 85	13.2 x 16.6 x 2.4
ESP32-C6-MINI-1-H4		-40^- 105	

At the core of this module is ESP32-C6FH4, a 32-bit RISC-V single-core processor.
ESP32-C6FH4 integrates a rich set of peripherals including SPI, parallel IO interface, UART, I2C, 12S, RMT (TX/RX), LED PWM, USB Serial/UTAG controller, MCPWM, SDIO2.0 slave controller, GDMA, TWAI® controller, on-chip debug functionality via JTAG, event task matrix, as well as up to 22 GPIOs, etc.
Note:
* For more information on ESP32-C6FH4, please refer to **ESP32-C6 Senes Datasheet**,

Pin Definitions

2.1 Pin Layout
The pin diagram below shows the approximate location of pins on the module.

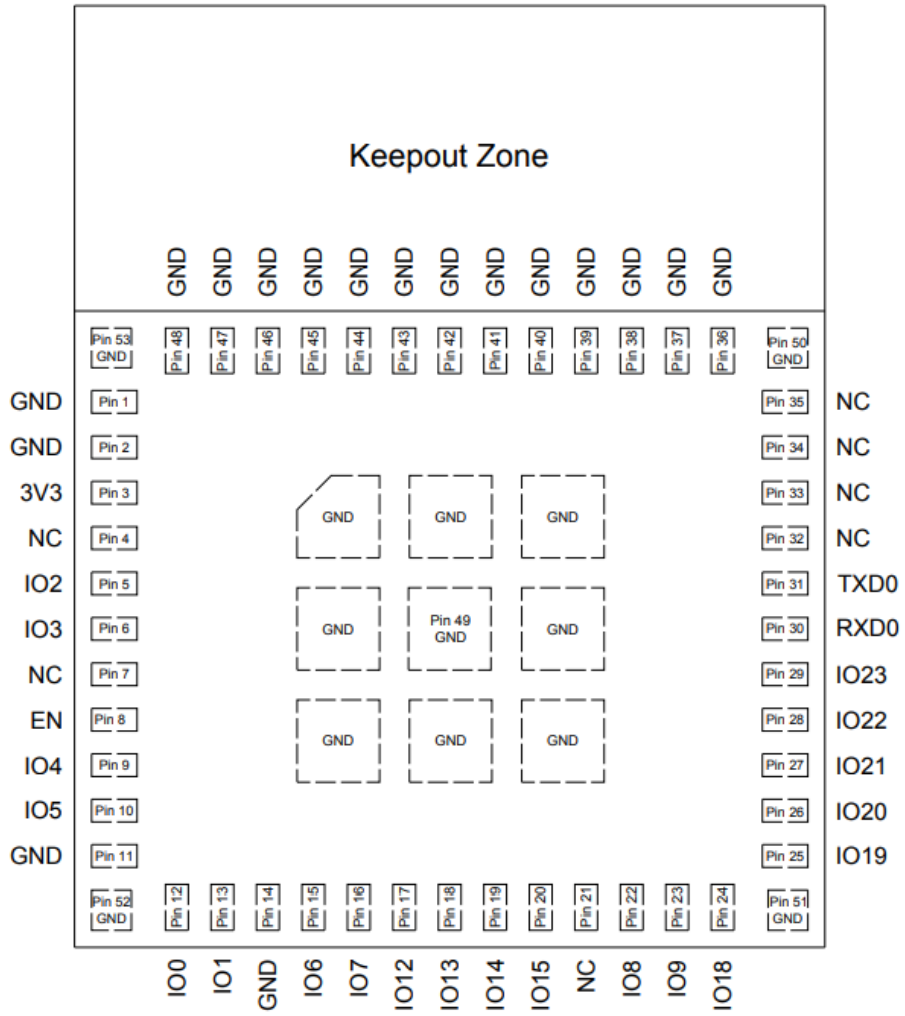


Figure 1: Pin Layout (Top View)

2.2 Pin Description

The module has 53 pins. See pin definitions in Table 2 Pin Definitions.

For peripheral pin configurations, please refer to **ESP32-C6 Series Datasheet**.

Table 2: Pin Definitions

Name	No.	Types 1	Function
GND	1, 2, 11, 14, 36 ~ 53	P	Ground
3V3	3	P	Power supply
NC	4	–	NC
102	5	VO/T	GPIO2, LP_GPIO2, LP_UART_RTSN, ADC1_CH2, FSPIQ
103	6	VO/T	GPI03, LP_GPI03, LP_UART_CTSN, ADC1_CH3
NC	7	–	NC
EN	8	I	High: on, enables the chip. Low: off, the chip powers off. Note: Do not leave the EN pin floating.
104	9	VO/T	MTMS, GPIO4, LP_GPIO4, LP_UART_FIXD, ADC1_CH4, FSPIHD
105	10	VO/T	MTDI, GPIO5, LP_GPIO5, LP_UART_TXD, ADC1_CH5, FSPIWP
100	12	VO/T	GPI00, XTAL_32K_P, LP_GPI00, LP_UART_DTRN, ADC1_CHO

101	13	VO/T	GPI01, XTAL_32K_N, LP_GPI01, LP_UART_DSRN, ADC1_CH1
106	15	VO/T	MICK, GPIO6, LP_GPI06, LP_12C_SDA, ADC1_CH6, FSPICLK
107	16	VO/T	MTDO, GPIO7, LP_GPIO7, LP_12C_SCL, FSPID
1012	17	VO/T	GPIO12, USB_D-
1013	18	VO/T	GPI013, USB_D+
1014	19	VO/T	GPIO14
1015	20	VO/T	GPIO15
NC	21	–	NC
108	22	VO/T	GP108
109	23	VO/T	GP109
1018	24	VO/T	GPIO18, SDIO_CMD, FSPICS2
1019	25	VO/T	GPIO19, SDIO_CLK, FSPICS3
1020	26	VO/T	GPIO20, SDIO_DATAO, FSPICS4
1021	27	VO/T	GPIO21, SDIO_DATA1, FSPICS5
1022	28	VO/T	GPIO22, SDIO_DATA2
1023	29	VO/T	GPIO23, SDIO_DATA3
RXDO	30	VO/T	UORXD, GPI017, FSPICS1
TXDO	31	VO/T	UOTXD, GPIO16, FSPICSO
NC	32	–	NC
NC	33	–	NC
NC	34	–	NC
NC	35	–	NC

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

Get Started

3.1 What You Need

To develop applications for module you need:

- 1 x ESP32-C6-MINI-1
- 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-C6-MINI-1 module to the RF testing board as shown in Figure 2.

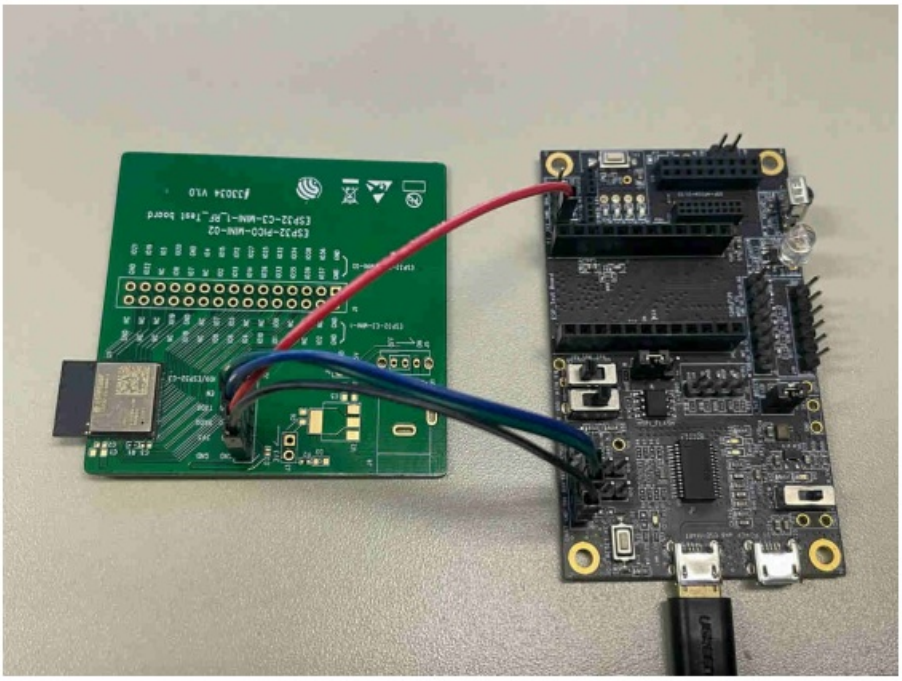


Figure 2: Hardware Connection

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 IO9 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. The module 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 ESP32-C6-MINI-1, please refer to **ESP32-C6 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-C6 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 cmake ninja-build ccache dfu-util libusbx

Ubuntu and Debian:

1 sudo apt-get install git wget flex bison gperf python3 python3-venv 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 cmake ninja ccache 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.

3.3.2 Get ESP-IDF

To build applications for ESP32-C6-MINI-1 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.

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 esp32c6
```

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 ESP32-C6-MINI-1 module.

3.4 Create Your First Project

3.4.1 Start a Project

Now you are ready to prepare your application for ESP32-C6-MINI-1 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.

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-C6 chip as the target and run the project configuration utility `menuconfig'.

```
1 cd ~/esp/hello_world
2 idf.py set-target esp32c6
3 idf.py menuconfig
```

Setting the target with `idf.py set-target ESP32-C6′ 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:

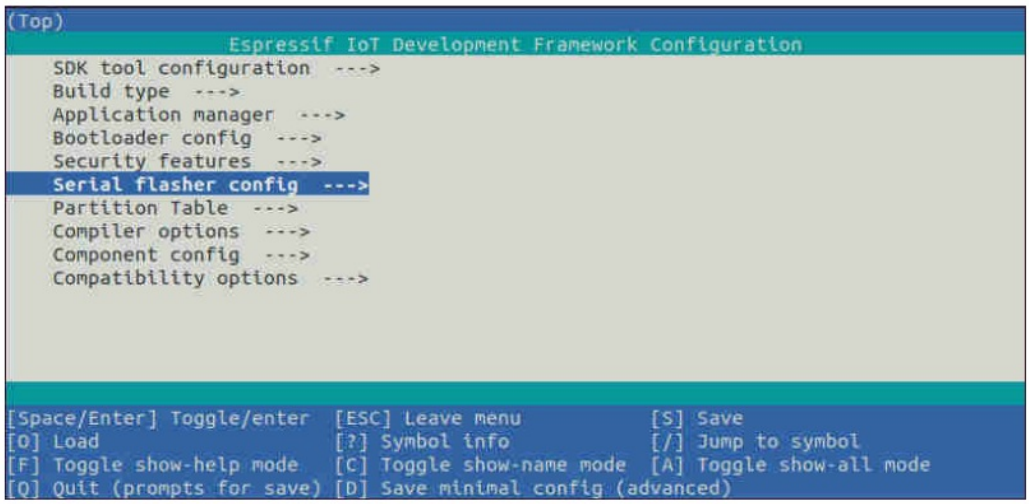


Figure 3: Project Configuration - Home Window

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 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: …
10 … (more lines of build system output)
12 [527/527] Generating hello_world.bin
13 esptool.py v2.3.1
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-C6 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 esp32c6 -p /dev/ttyUSB0 -b 460800 –before=default_reset –after=hard_reset –no- stub write_flash –flash_mode dio –flash_freq 80m –flash_size 2MB 0x0 bootloader/ bootloader.bin 0x10000 hello_world.bin 0x8000 partition_table/partition-table.bin

3 esptool.py v4.3
4 Serial port /dev/ttyUSB0
5 Connecting….
6 Chip is ESP32-C6 (revision v0.0)
7 Features: WiFi 6, BT 5
8 Crystal is 40MHz
9 MAC: 60:55:f9:f6:01:38
10 Changing baud rate to 460800
11 Changed.
12 Enabling default SPI flash mode…
13 Configuring flash size…
14 Flash will be erased from 0x00000000 to 0x00004fff…
15 Flash will be erased from 0x00010000 to 0x00028fff…
16 Flash will be erased from 0x00008000 to 0x00008fff…
17 Erasing flash… 18 Took 0.17s to erase flash block
19 Writing at 0x00000000… (5 %)
20 Writing at 0x00000c00… (23 %)
21 Writing at 0x00001c00… (47 %)
22 Writing at 0x00003000… (76 %)
23 Writing at 0x00004000… (100 %)
24 Wrote 17408 bytes at 0x00000000 in 0.5 seconds (254.6 kbit/s)…
25 Hash of data verified.
26 Erasing flash…
27 Took 0.85s to erase flash block
28 Writing at 0x00010000… (1 %)
29 Writing at 0x00014c00… (20 %)
30 Writing at 0x00019c00… (40 %)
31 Writing at 0x0001ec00… (60 %)
32 Writing at 0x00023c00… (80 %)
33 Writing at 0x00028c00… (100 %)
34 Wrote 102400 bytes at 0x00010000 in 3.2 seconds (253.5 kbit/s)…
35 Hash of data verified.
36 Erasing flash…
37 Took 0.04s to erase flash block
38 Writing at 0x00008000… (33 %)
39 Writing at 0x00008400… (66 %)
40 Writing at 0x00008800… (100 %)
41 Wrote 3072 bytes at 0x00008000 in 0.1 seconds (269.0 kbit/s)…
42 Hash of data verified.
43
44 Leaving…
45 Hard resetting via RTS pin…
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 <PORT> monitor
2 Running idf_monitor in directory […]/esp/hello_world/build
3 Executing “python […]/esp-idf/tools/idf_monitor.py -b 115200 […]/esp/hello_world/build /hello_world.elf”…
4 — idf_monitor on <PORT> 115200 —
5 — Quit: Ctrl+] | Menu: Ctrl+T | Help: Ctrl+T followed by Ctrl+H —
6 ets Jun 8 2016 00:22:57
7
8 rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
9 ets Jun 8 2016 00:22:57
10 …
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 esp32c6 chip with 1 CPU core(s), WiFi/BLE, 802.15.4 (Zigbee/Thread), silicon revision v0.0, 2 MB external flash
5 Minimum free heap size: 337332 bytes
6 Restarting in 9 seconds…
7 Restarting in 8 seconds…
8 Restarting in 7 seconds…
To exit IDF monitor use the shortcut Ctrl+]. That's all what you need to get started with ESP32-C6-MINI-1 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

The device complies with KDB 996369 D03 OEM Manual v01. Below are integration instructions for host product manufacturers according to the KDB 996369 D03 OEM Manual v01.
List of Applicable FCC Rules
FCC Part 15 Subpart C 15.247
Specific Operational Use Conditions
The module has WiFi and BLE functions.

- Operation Frequency:

  WiFi: 2412 ~ 2462 MHz

  Bluetooth: 2402 ~ 2480 MHz Zigbee/Thread: 2405 ~ 2480 MHz
- Number of Channel:

  WiFi: 11

  Bluetooth: 40

  Zigbee/Thread: 26
- Modulation:

  WiFi: DSSS; OFDM

  Bluetooth: GFSK

  Zigbee/Thread:O-QPSK
- Type: On-board PCB Antenna
- Gain: 3.96 dBi Max

The module can be used for IoT applications with a maximum 3.96 dBi antenna. The host manufacturer installing this module into their product must ensure that the final composit product complies with the FCC requirements by a technical assessment or evaluation to the FCC rules, including the transmitter operation. The host manufacturer 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 show in this manual.
Limited Module Procedures
Not applicable. The module is a single module and complies with the requirement of FCC Part 15.212.
Trace Antenna Designs
Not applicable. The module has its own antenna, and does not need a host's printed board microstrip trace antenna, etc.
RF Exposure Considerations
The module must be installed in the host equipment such that at least 20cm is maintained between the antenna and users' body; and if RF exposure statement or module layout is changed, then the host product manufacturer required to take responsibility of the module through a change in FCC ID or new application. The FCC ID of the module cannot be used on the final product. In these circumstances, the host manufacturer will be responsible for re-evaluating the end product (including the transmitter) and obtaining a separate FCC authorization.
Antennas

Antenna specification are as follows:

Type: PCB Antenna
Gain: 3.96 dBi

This device is intended only for host manufacturers under the following conditions:

- The transmitter module may not be co-located with any other transmitter or antenna.
- The module shall be only used with the external antenna(s) that has been originally tested and certified with this module.
- The antenna must be either permanently attached or employ a `unique' antenna coupler.

As long as the conditions above are met, further transmitter test will not be required. However, the host manufacturer is still responsible for testing their end-product for any additional compliance requirements required with this module installed (for example, digital device emissions, PC peripheral requirements, etc.).

Label and Compliance Information

Host product manufacturers need to provide a physical or e-label stating “Contains FCC ID: 2AC7Z-ESPC6MINI1” with their finished product.

Information on test modes and additional testing requirements

- Operation Frequency:
  WiFi: 2412 ~ 2462 MHz
  Bluetooth: 2402 ~ 2480 MHz
  Zigbee/Thread: 2405 ~ 2480 MHz
- Number of Channel:
  WiFi: 11
  Bluetooth: 40
  Zigbee/Thread: 26
- Modulation:
  WiFi: DSSS; OFDM
  Bluetooth: GFSK
  Zigbee/Thread:O-QPSK

Host manufacturer must perform test of radiated and conducted emission and spurious emission, etc., according to the actual test modes for a stand-alone modular transmitter in a host, as well as for multiple simultaneously transmitting modules or other transmitters in a host product. Only when all the test results of test modes comply with FCC requirements, then the end product can be sold legally.

Additional testing, Part 15 Subpart B compliant

The modular transmitter is only FCC authorized for FCC Part 15 Subpart C 15.247 and that the host product manufacturer is responsible for compliance to any other FCC rules that apply to the host not covered by the modular transmitter grant of certification. If the grantee markets their product as being Part 15 Subpart B compliant (when it also contains unintentional-radiator digital circuity), then the grantee shall provide a notice stating that the final host product still requires Part 15 Subpart B compliance testing with the modular transmitter installed. 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 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.

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.

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 transmitter module may not be co-located with any other transmitter or antenna.
- The module shall be only used with the external antenna(s) that has been originally tested and certified with this module.

As long as the 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 requirements required with this module installed (for example, digital device emissions, PC peripheral requirements, etc.).
Validity of Using the Module Certification
In the event that these conditions cannot be met (for example certain laptop configurations 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
The final end product must be labeled in a visible area with the following: "Contains Transmitter Module FCC ID: 2AC7Z-ESPC6MINI1".

Industry Canada Statement

This device complies with Industry Canada's licence-exempt RSSs. 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 minimum distance 20 cm between the radiator and your body.
RSS-247 Section 6.4 (5)
The device could automatically discontinue transmission in case of 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 2 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 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.
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 show in this manual.

Related Documentation and Resources

Related Documentation

- **ESP32-C6 Series Datasheet** - Specifications of the ESP32-C6 hardware.
- **ESP32-C6 Technical Reference Manual** - Detailed information on how to use the ESP32-C6 memory and peripherals.
- **ESP32-C6 Hardware Design Guidelines** - Guidelines on how to integrate the ESP32-C6 into your hardware product.
- Certificates
  https://espressif.com/en/support/documents/certificates
- Documentation Updates and Update Notification Subscription
  https://espressif.com/en/support/download/documents

Developer Zone

- **ESP-IDF Programming Guide for ESP32-C6** Extensive documentation for the ESP-IDF development framework.
- ESP-IDF and other development frameworks on GitHub.
  https://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.
  https://esp32.com/
- The ESP Journal Best Practices, Articles, and Notes from Espressif folks.
  https://blog.espressif.com/
- See the tabs SDKs and Demos, Apps, Tools, AT Firmware.
  https://espressif.com/en/support/download/sdks-demos

Products

- ESP32-C6 Series SoCs Browse through all ESP32-C6 SoCs.
  https://espressif.com/en/products/socs?id=ESP32-C6
- ESP32-C6 Series Modules Browse through all ESP32-C6-based modules.
  https://espressif.com/en/products/modules?id=ESP32-C6
- ESP32-C6 Series DevKits Browse through all ESP32-C6-based devkits.
  https://espressif.com/en/products/devkits?id=ESP32-C6
- ESP Product Selector Find an Espressif hardware product suitable for your needs by comparing or applying filters.
  https://products.espressif.com/#/product-selector?language=en

Contact Us

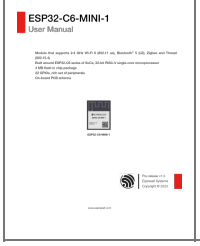
- See the tabs Sales Questions, Technical Enquiries, Circuit Schematic & PCB Design Review, Get Samples (Online stores), Become Our Supplier, Comments & Suggestions.
  https://espressif.com/en/contact-us/sales-questions

Revision History

Date	Version	Release r
2023-04-27	v1.0	Official release

www.espressif.com

Documents / Resources

	ESPRESSIF ESP32-C6-MINI-1 2.4 GHz Wi-Fi Module [pdf] User Manual ESP32-C6-MINI-1, ESP32-C6-MINI-1 2.4 GHz Wi-Fi Module, 2.4 GHz Wi-Fi Module, Wi-Fi Module, Module

References

- Wireless SoCs, Software, Cloud 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
- ESP-IDF Versions - ESP32 - — ESP-IDF Programming Guide latest documentation
- Get Started - ESP32-C6 - — ESP-IDF Programming Guide latest documentation
- ESP32 Forum - Index page
- Sales Questions | Espressif Systems
- ESP DevKits | Espressif Systems
- ESP Modules | Espressif Systems
- ESP SoCs | 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
- Wireless SoCs, Software, Cloud and AIoT Solutions | Espressif Systems
- Documentation Feedback | Espressif Systems
- User Manual