ESP32-DevKitC

Getting Started Guide

Version 1.4

About This Guide

This user guide introduces how to use the ESP32-DevKitC development board.

The document structure is as follows:

Chapter	Title	Content
Chapter 1	Overview	Introduction to the ESP32-DevKitC.
Chapter 2	Compilation and Download	Presentation of how to compile and download BIN files to the ESP32-DevKitC by using ***example/01_hello_world*** in ESP-IDF as an example.

Release Notes

Date	Version	Release notes
2016.09	V1.0	Initial release.
2016.10	V1.1	Updated Figure 1-2. ESP32-DevKitC Layout.
2016.11	V1.2	Added Figure 1-1. ESP32-DevKitC Dimensions. Updated Figure 1-2. ESP32-DevKitC Layout.
2016.12	V1.3	Major revision based on the release of ESP-IDF 1.0.
2017.03	V1.4	Updated Figure 1-2. ESP32-DevKitC Layout.

Related Documents

You may find the following documents helpful.

Document	Web link
ESP32 Datasheet	*http://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf*
ESP-WROOM-32 Datasheet	*http://www.espressif.com/sites/default/files/documentation/esp_wroom_32_datasheet_en.pdf*
ESP-IDF Getting Started Guide	*http://www.espressif.com/sites/default/files/documentation/esp-idf_getting_started_guide_en.pdf*

Table of Contents

1. Overview

1.1. Product Introduction

ESP32-DevKitC is a small-sized ESP32-based development board produced by Espressif. Most of the I/O pins are led out to the pin headers on both sides for easy interfacing. Developers can connect these pins to peripherals as needed. Standard headers also make development easy and convenient when using a breadboard.

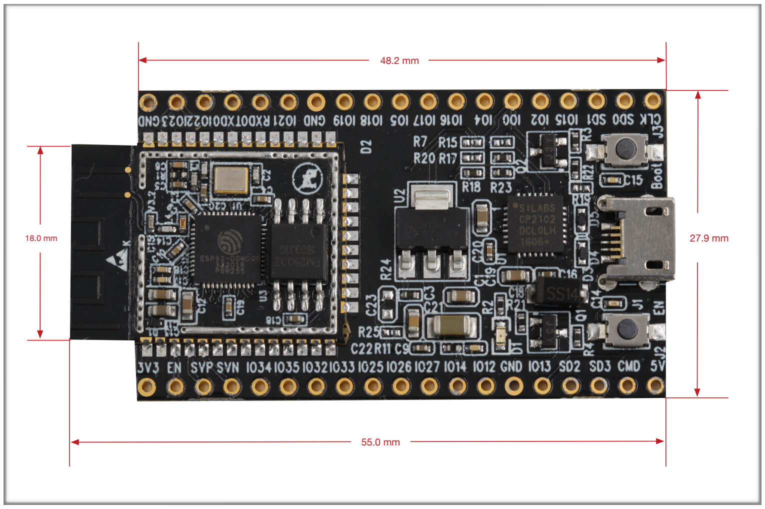


Figure 1-1. ESP32-DevKitC Dimensions

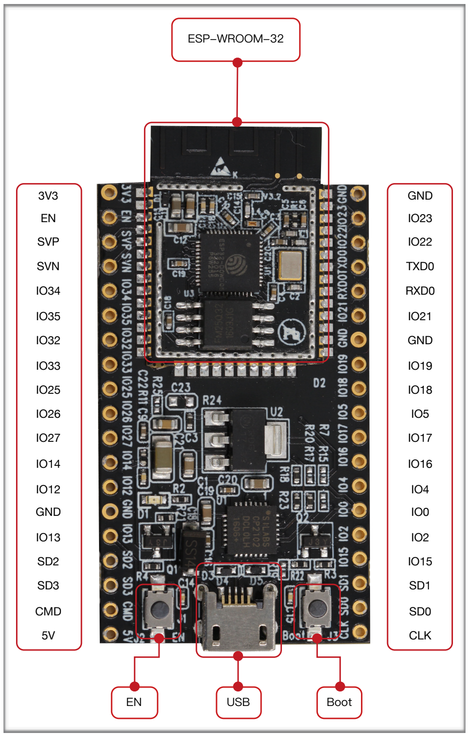


Figure 1-2. ESP32-DevKitC Layout

1.2. Functional Description

Table 1-1. ESP32-DevKitC Functional Description

Interface/Module	Description
ESP-WROOM-32	ESP-WROOM-32 module.
EN	Reset button: pressing this button resets the system.
Boot	Download button: holding down the Boot button and pressing the EN button initiates the firmware download mode. Then users can download firmware through the serial port.
USB	USB interface. It functions as the power supply for the board and the communication interface between PC and ESP-WROOM-32.
I/O	Most of the pins on the ESP-WROOM-32 are led out to the pin headers on the board. Users can program ESP32 to enable multiple functions such as PWM, ADC, DAC, I2C, I2S, SPI, etc.

2. Compilation and Download

We are using ESP-IDF as an example to show how to download firmware to the ESP32-DevKitC.

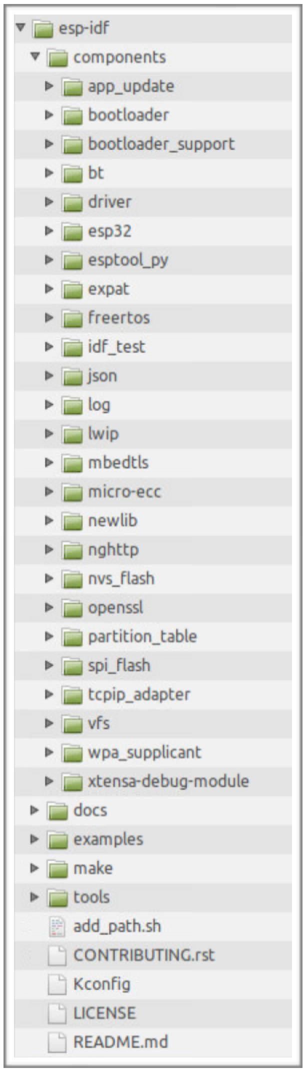
- The hardware listed below are required:
  - 1 × ESP32-DevKitC
  - 1 × PC (with Windows OS as an example in this document)
  - 1 × USB cable
- The tools listed below are required:
  - ESP32 Flash Download Tool: *http://espressif.com/en/support/download/other-tools?keys=&field_type_tid%5B%5D=13*
  - ESP-IDF：*https://github.com/espressif/esp-idf.git*

2.1. Create Serial Communication

Connect the ESP32-DevKitC to the PC using the USB cable. Check the list of identified external COM ports in the Windows Device Manager and confirm the COM port number of the ESP32-DevKitC.

2.2. ESP-IDF Directory Structure

The following figure shows the directory structure of ESP-IDF, including ***components***, ***examples***, ***make***, ***tools*** and ***docs***. The ***components*** folder contains the core components of ESP-IDF; the ***examples*** folder contains the program examples of ESP-IDF; the ***make*** folder contains makefiles for ESP-IDF; the ***tools*** folder is the toolkit; the ***docs*** folder contains ESP-IDF-relevant documentation.



- esp-idf
  - components
    - app_update
    - bootloader
    - bootloader_support
    - bt
    - driver
    - esp32
    - esptool_py
    - expat
    - freertos
    - idf_test
    - json
    - log
    - lwip
    - mbedtls
    - micro-ecc
    - newlib
    - nghttp
    - nvs_flash
    - openssl
    - partition_table
    - spi_flash
    - tcpip_adapter
    - vfs
    - wpa_supplicant
    - xtensa-debug-module
  - docs
  - examples
  - make
  - tools
  - add_path.sh
  - CONTRIBUTING.rst
  - Kconfig
  - LICENSE
  - README.md

2.3. The *hello_world* Example

The ***esp-idf/examples/01_hello_world*** directory contains a sample code that can be run on the ESP32.

1. Using the command terminal, change the current directory to ***example/01_hello_world***:

```
cd examples/01_hello_world/
```

2. Configure ***IDF_PATH***:

```
export IDF_PATH=/home/share/esp-idf-driver/esp-idf
```

3. Check the ***IDF_PATH*** configuration to make sure it is properly set. Failing to set the path will cause failure to the linking of dependent files later.

```
echo $IDF_PATH
```

4. Compile the program to generate BIN files. These BIN files have to be downloaded to the ESP32-DevKitC. Please see Sections 2.3.1 and 2.3.2 for detailed instructions.

2.3.1. Using the ESP32 DOWNLOAD TOOL

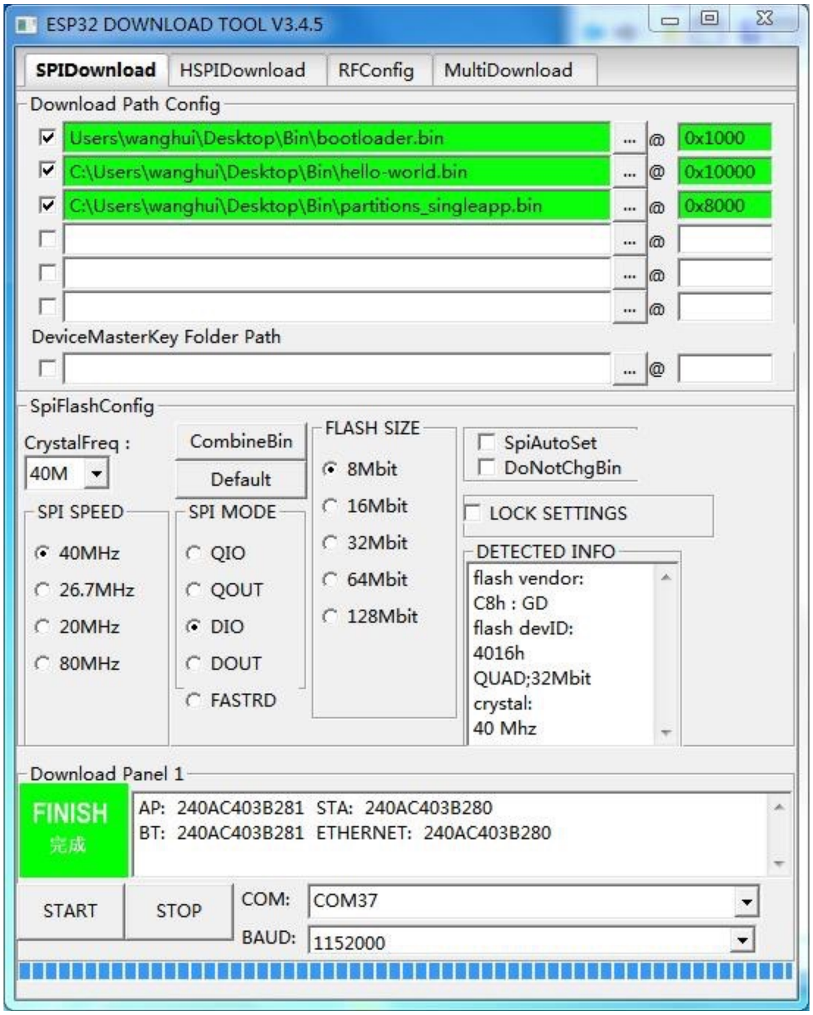
Execute the following command in the terminal to make the example project and generate executable BIN files:

```
make
```

Three BIN files need to be downloaded: ***example/01_hello_world/bootloader/bootloader.bin***, ***example/01_hello_world/partitions_singleapp.bin*** and ***example/01_hello_world/hello-world.bin***. Then, users can flash these BIN files by using the ESP32 DOWNLOAD TOOL. Please follow the steps below:

1. Open the ESP32 DOWNLOAD TOOL.
2. Configure the download tool and click on "START", as shown below:

> ⚠ ***Notice:***
>
> *Most computers will automatically reset the ESP32 into download mode when you start uploading. If this does not work on your computer, try holding down the Boot button (and possibly pressing and releasing the EN button) when starting the upload.*



ESP32 DOWNLOAD TOOL V3.4.5

Download Path Config

☑	Users\wanghui\Desktop\Bin\bootloader.bin	@	0x1000
☑	C:\Users\wanghui\Desktop\Bin\hello-world.bin	@	0x10000
☑	C:\Users\wanghui\Desktop\Bin\partitions_singleapp.bin	@	0x8000
□		@	
□		@	
□		@	

DeviceMasterKey Folder Path

□ @

SpiFlashConfig

CrystalFreq : 40M

SPI SPEED: ● 40MHz ○ 26.7MHz ○ 20MHz ○ 80MHz

SPI MODE: ○ QIO ○ QOUT ● DIO ○ DOUT ○ FASTRD

FLASH SIZE: ● 8Mbit ○ 16Mbit ○ 32Mbit ○ 64Mbit ○ 128Mbit

□ SpiAutoSet
□ DoNotChgBin
□ LOCK SETTINGS

DETECTED INFO

```
flash vendor:
C8h : GD
flash devID:
4016h
QUAD;32Mbit
crystal:
40 Mhz
```

Download Panel 1

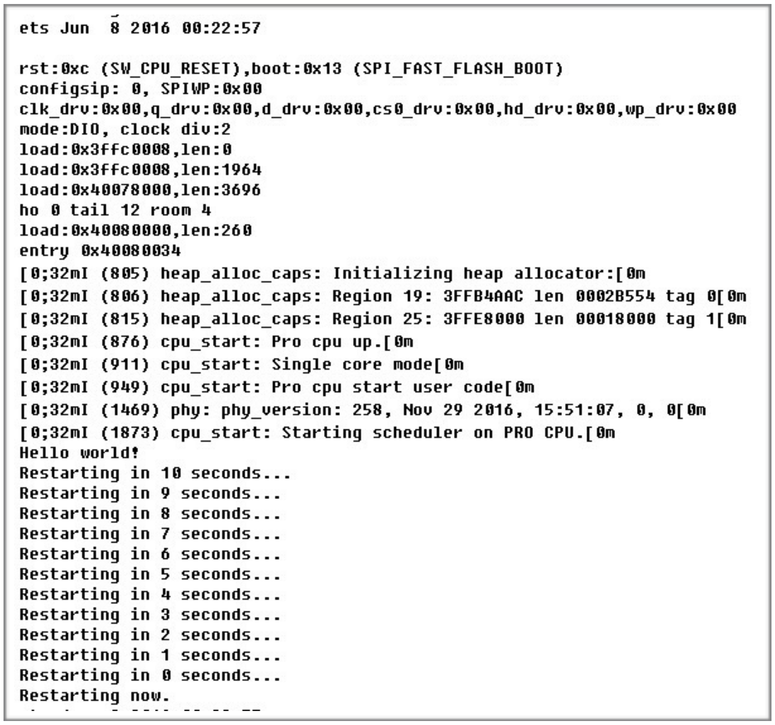
FINISH
完成

```
AP:  240AC403B281  STA:  240AC403B280
BT:  240AC403B281  ETHERNET:  240AC403B280
```

COM: COM37

BAUD: 1152000

3. Open the serial port. Set the Port, Baud rate = 115200, Data bits = 8, and Stop bits = 1. If the log below is printed, then it shows that the firmware has been downloaded to the ESP32-DevKitC successfully.



```
ets Jun  8 2016 00:22:57

rst:0xc (SW_CPU_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0x00
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:2
load:0x3ffc0008,len:0
load:0x3ffc0008,len:1964
load:0x40078000,len:3696
ho 0 tail 12 room 4
load:0x40080000,len:260
entry 0x40080034
[0;32mI (805) heap_alloc_caps: Initializing heap allocator:[0m
[0;32mI (806) heap_alloc_caps: Region 19: 3FFB4AAC len 0002B554 tag 0[0m
[0;32mI (815) heap_alloc_caps: Region 25: 3FFE8000 len 00018000 tag 1[0m
[0;32mI (876) cpu_start: Pro cpu up.[0m
[0;32mI (911) cpu_start: Single core mode[0m
[0;32mI (949) cpu_start: Pro cpu start user code[0m
[0;32mI (1469) phy: phy_version: 258, Nov 29 2016, 15:51:07, 0, 0[0m
[0;32mI (1873) cpu_start: Starting scheduler on PRO CPU.[0m
Hello world!
Restarting in 10 seconds...
Restarting in 9 seconds...
Restarting in 8 seconds...
Restarting in 7 seconds...
Restarting in 6 seconds...
Restarting in 5 seconds...
Restarting in 4 seconds...
Restarting in 3 seconds...
Restarting in 2 seconds...
Restarting in 1 seconds...
Restarting in 0 seconds...
Restarting now.
```

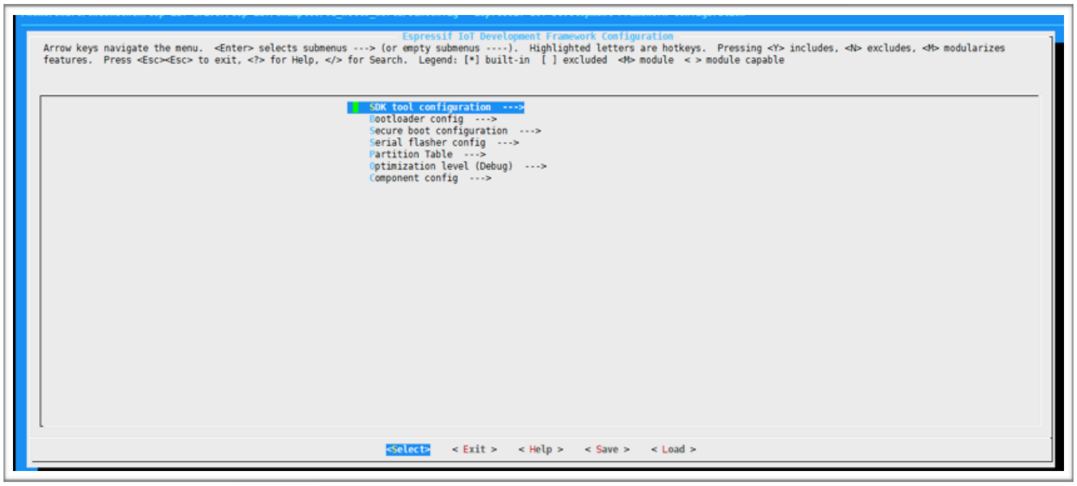
2.3.2. Using esptool

Users need to configure the serial port before compiling and downloading BIN files. Serial port configuration is not required if the Flash Download Tool is used to flash the BIN files on to the ESP32-DevKitC. However, other important system parameters may be set via `menuconfig`. Please complete this step before generating BIN files.

1. Enter `make menuconfig`:

```
make menuconfig
```

Then, the following interface is displayed:

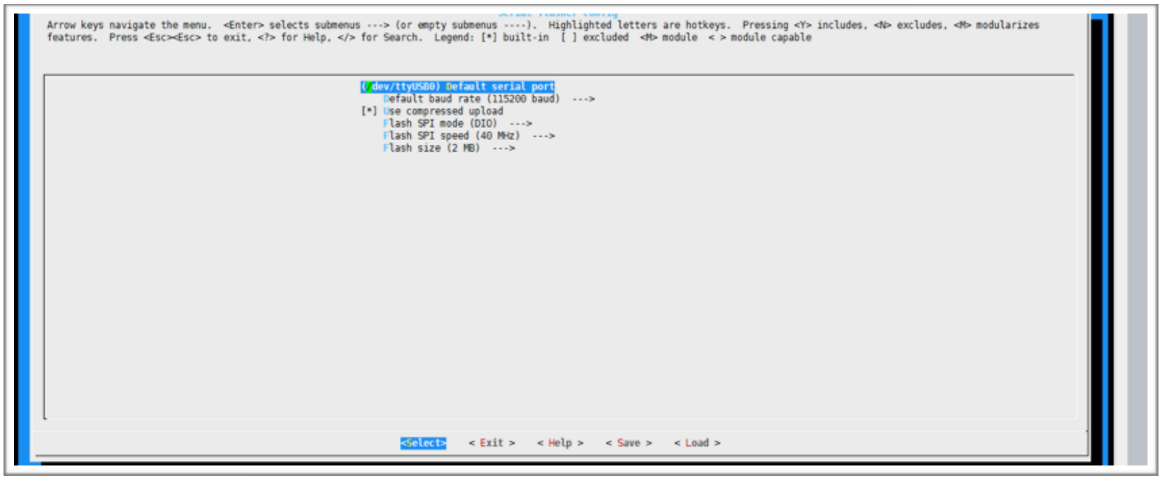


```
Espressif IoT Development Framework Configuration
Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty submenus ----).  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes, <M> modularizes
features.  Press <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in  [ ] excluded  <M> module  < > module capable

    SDK tool configuration  --->
    Bootloader config  --->
    Secure boot configuration  --->
    Serial flasher config  --->
    Partition Table  --->
    Optimization level (Debug)  --->
    Component config  --->

<Select>    < Exit >    < Help >    < Save >    < Load >
```

2. Select ***Serial flasher config*** to configure the serial port, as shown below:

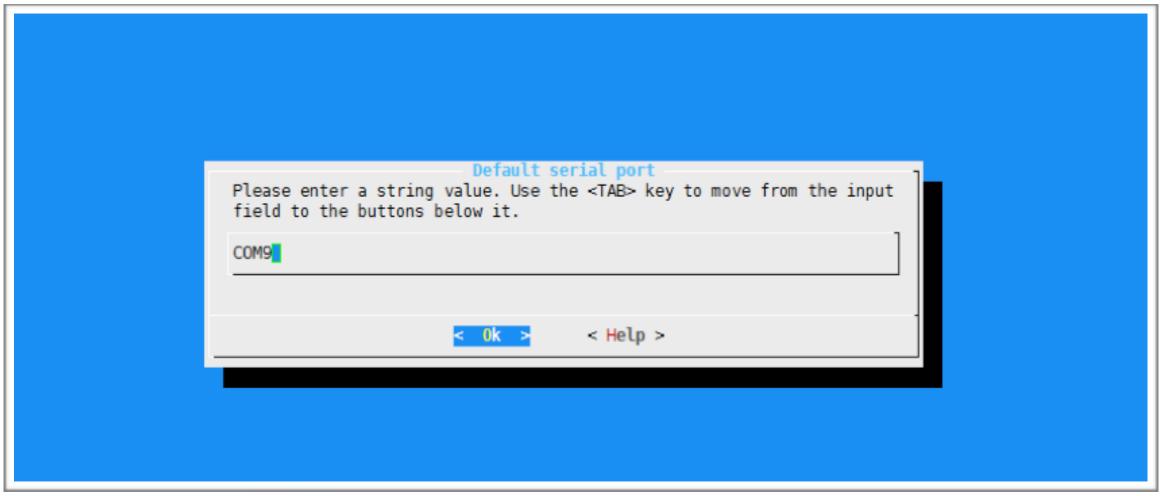


```
Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty submenus ----).  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes, <M> modularizes
features.  Press <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in  [ ] excluded  <M> module  < > module capable

    (/dev/ttyUSB0) Default serial port
        Default baud rate (115200 baud)  --->
    [*] Use compressed upload
        Flash SPI mode (DIO)  --->
        Flash SPI speed (40 MHz)  --->
        Flash size (2 MB)  --->

        <Select>    < Exit >    < Help >    < Save >    < Load >
```

3. Configure the serial port, as shown below:



```
Default serial port
Please enter a string value. Use the <TAB> key to move from the input
field to the buttons below it.

COM9

        <  Ok  >      < Help >
```

4. Click "OK" and exit `make menuconfig`.

5. Flash BIN files directly via the command line below:

```
make flash
```

📖 ***Notes:***

- *Please configure the serial port according to actual situation.*
- *Only DIO mode is supported currently. QIO mode would be supported afterwards.*
- *For more information on ESP-IDF, please see ESP-IDF Getting Started Guide.*

Espressif IOT Team

www.espressif.com

Disclaimer and Copyright Notice

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

THIS DOCUMENT IS PROVIDED AS IS WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NON-INFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY WARRANTY OTHERWISE ARISING 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 expressed 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.