



Qualcomm Technologies International, Ltd.

QCC711 Development Kit

Quick Start Guide

80-70852-1 Rev. AF

September 30, 2025

Qualcomm Technologies International, Ltd. is a company registered in England and Wales with a registered office at: Churchill House, Cambridge Business Park, Cowley Road, Cambridge, CB4 0WZ, United Kingdom.
Registered Number: 3665875 | VAT number: GB787433096

© Qualcomm Technologies, Inc. and/or its subsidiaries. All rights reserved.

Revision history

Revision	Date	Change reason
AA	June 2024	Initial release
AB	July 2024	Updated: <ul style="list-style-type: none">■ Development kit hardware■ QC711 jumper settings
AC	September 2024	Updated: <ul style="list-style-type: none">■ Development environment setup■ Build sample demo applications■ Program the Bluetooth MAC address
AD	March 2025	Editorial updates
AE	August 2025	Editorial updates
AF	September 2025	<ul style="list-style-type: none">■ Added Software dependencies deployment■ Updated QCC711 image

Contents

Revision history	2
1 QCC711 development kit introduction	6
1.1 Development kit contents	6
1.2 Development kit hardware.....	7
1.2.1 QC711 jumper settings	11
2 Development environment setup	12
2.1 Software dependencies deployment	12
2.1.1 Download software dependencies automatically	13
2.1.2 Configure environment variables and install python packages	13
2.2 Supported toolchains	15
2.2.1 GN package.....	16
2.2.2 xPack Arm-embedded GCC.....	16
2.3 Other requirements.....	17
2.3.1 Ninja	17
2.3.2 J-Link software.....	17
2.3.3 OpenOCD for JTAG adaptor tool	17
2.3.4 CH347 Microsoft Windows driver	17
2.3.5 Python	18
3 Build sample demo applications	19
3.1 Build images using qccsdk.py	19
4 QCC711 image	21
4.1 Flash command usage	21
4.2 Autoboot and run the application	22
5 Program the Bluetooth MAC address	24
Document references	25
Glossary	26

Tables

Table 1-1: QCC711 development kit jumper setting details.....	11
Table 2-1: Details to automatically download software dependencies.....	13
Table 2-2: Details to automatically configure environment variables	14
Table 3-1: qccsdk.py set parameters.....	19
Table 3-2: qccsdk.py build parameters	20
Table 4-1: Flash parameters for qccsdk.py.....	21

Figures

Figure 1-1: QCC711 development kit.....7

Figure 1-2: QCC711 development kit component layout8

Figure 1-3: QCC711 development kit block diagram9

Figure 1-4: QCC711 development kit pin map.....10

Figure 1-5: QCC711 development kit jumper settings.....11

Figure 2-1: Set-ExecutionPolicy command.....12

Figure 2-2: A script output example13

Figure 2-3: Output example.....14

Figure 2-4: System variables.15

Figure 2-5: Environment variable details16

Figure 4-1: UART settings.....22

Figure 4-2: appsGeneral console log.....23

1 QCC711 development kit introduction

This document provides the following information to configure the QCC711 development kit for connectivity testing:

- Hardware introduction
- Set up the development environment
- Build a sample demo
- Flash images
- Program a Bluetooth MAC address
- Use appsGeneral

1.1 Development kit contents

The QCC711 development kit contains the following parts:

- Carrier board and reference module
- Accessories:
 - USB Type-C cable for power supply
 - UART console

1.2 Development kit hardware

QCC711 is a micropower Bluetooth solution that supports the Bluetooth v5.4 specification.

Figure 1-1 shows the QCC711 development kit.

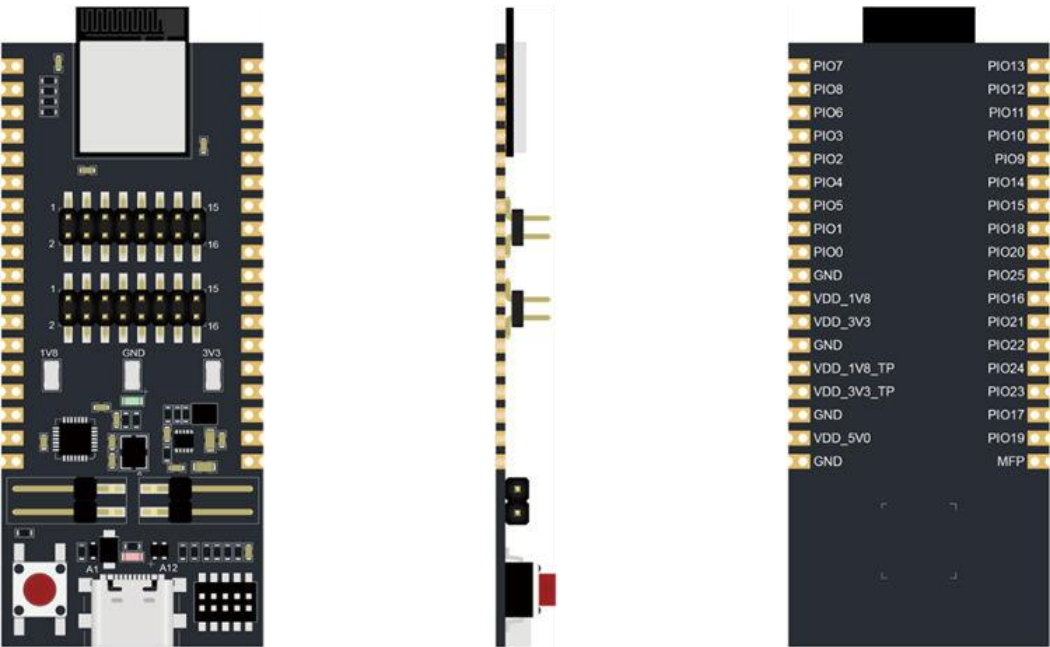


Figure 1-1 QCC711 development kit

Figure 1-2 shows the QCC711 development kit component layout.

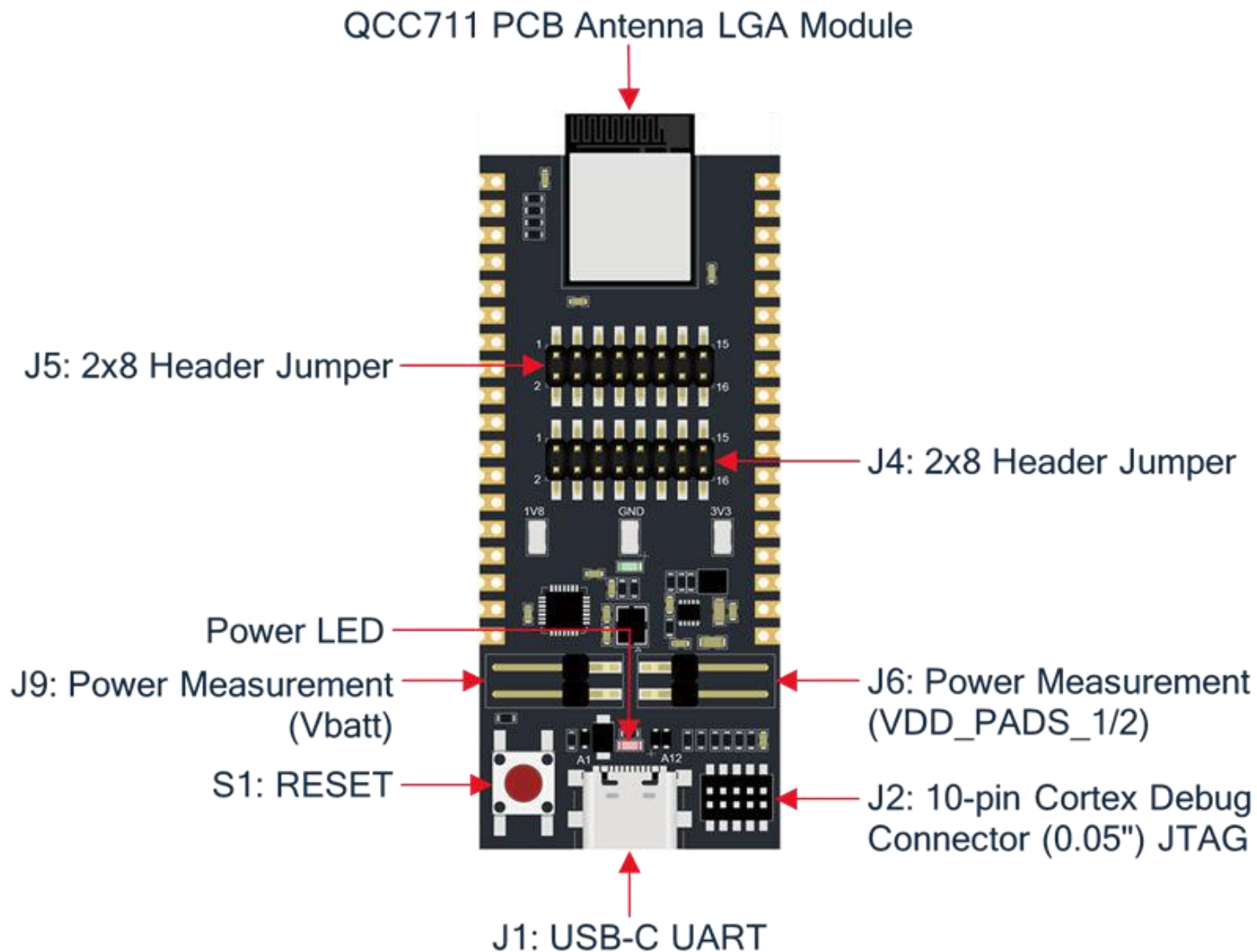


Figure 1-2 QCC711 development kit component layout

Figure 1-3 shows the QCC711 development kit block diagram.

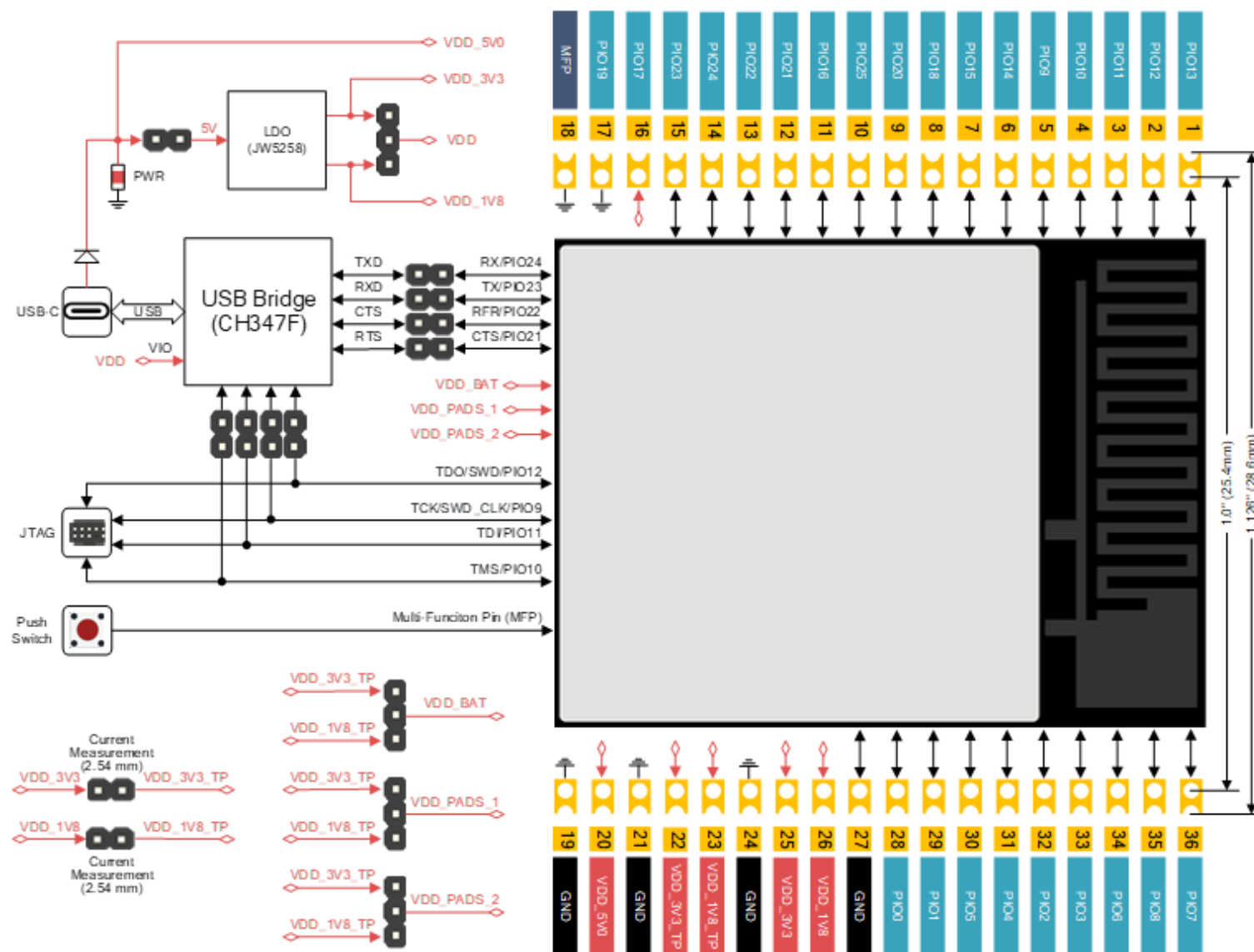


Figure 1-3 QCC711 development kit block diagram

Figure 1-4 shows the QCC711 development kit pin map.

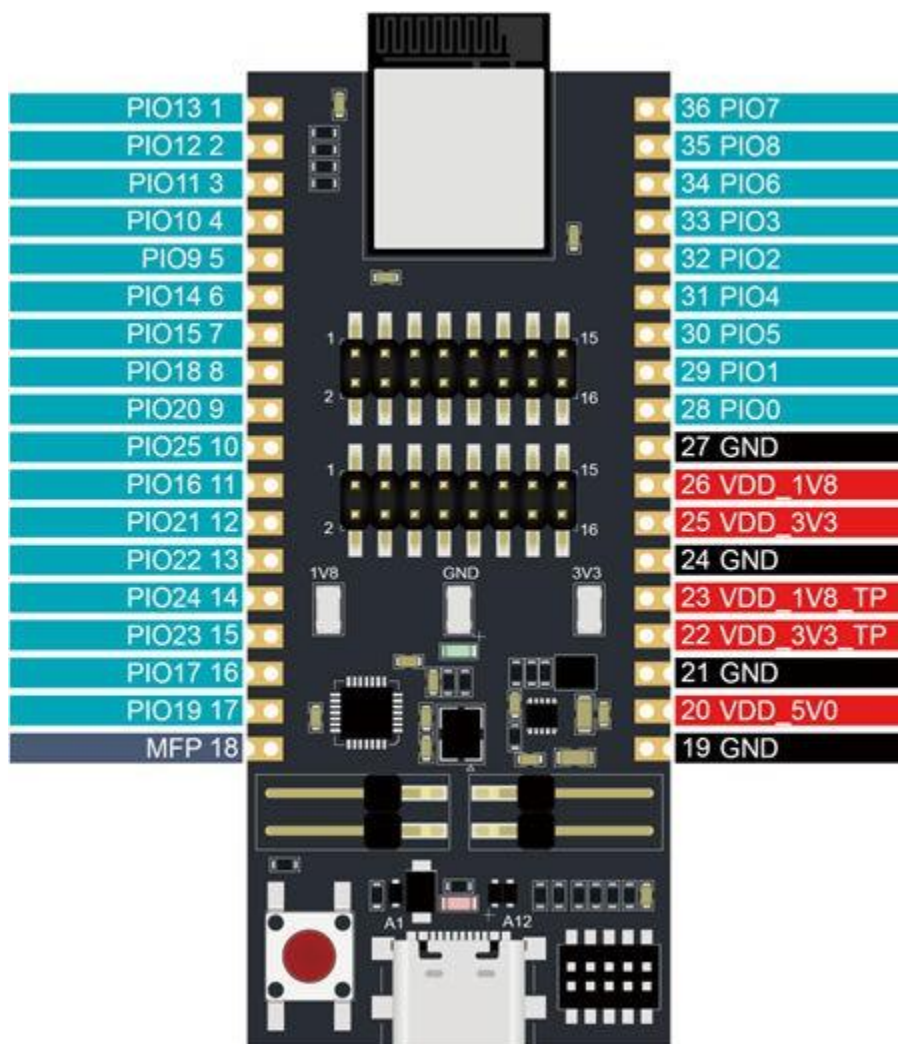


Figure 1-4 QCC711 development kit pin map

1.2.1 QCC711 jumper settings

Figure 1-5 shows the QCC711 development kit jumper settings.

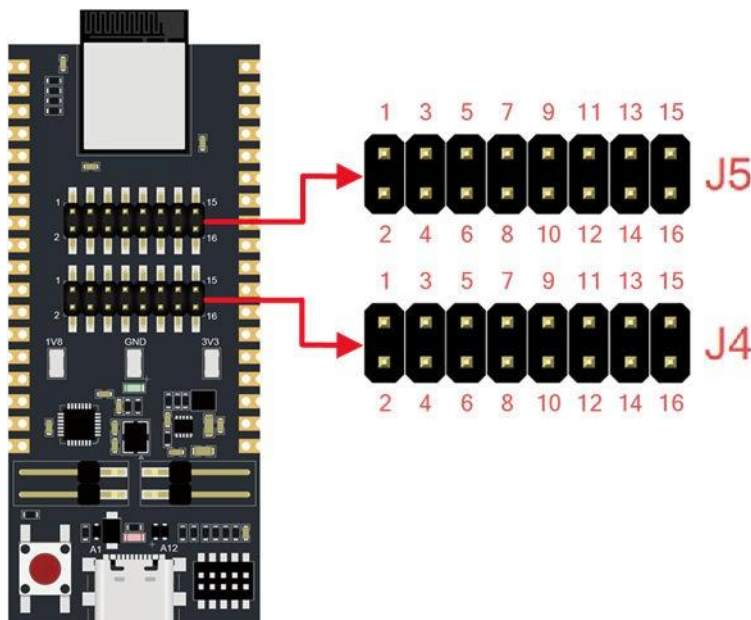


Figure 1-5 QCC711 development kit jumper settings

Table 1-1 lists the QCC711 development kit jumper settings.

Table 1-1 QCC711 development kit jumper setting details

Jumper	Pin 1	Pin 2	Function
J4 (2 x 8 Jumper)	1	2	CH347 TDO to PIO12 (TDO/SWO) connect enable.
	3	4	CH347 TCK to PIO9 (TCK/SWD_CLK) connect enable.
	5	6	CH347F TDI to PIO11 (TDI) connect enable.
	7	8	CH347F (TMS) to PIO10 (TMS/SWD_DIO) connect enable.
	9	10	CH347F TXD to PIO24 (RX) connect enable.
	11	12	CH347F RXD to PIO23 (TX) connect enable.
	13	14	CH347F CTS to PIO22 (RFR) connect enable.
	15	16	CH347F RTS to PIO21 (CTS) connect enable.
J5 (2 x 8 Jumper)	1	3	VDD_PADS_2 set to 3.3 V.
	3	5	VDD_PADS_2 set to 1.8 V.
	2	4	VDD_PADS_1 set to 3.3 V.
	4	6	VDD_PADS_1 set to 1.8 V.
	7	9	VDD_BAT set to 1.8 V.
	9	11	VDD_BAT set to 3.3 V.
	8	10	CH347F VDD set to 1.8 V.
	10	12	CH347F VDD set to 3.3 V.
	13	14	Not used (Floating).
	15	16	USB 5 V power supply enable.

2 Development environment setup

QCC711 software development kit (SDK) contains sample applications that demonstrate the use of the Qualcomm API (QAPI) and provide a command-line interface (CLI) based interface to test chip features.

To set up the development environment for the SDK, it is recommended to use the QCC IDE to test the necessary dependencies and configure the system environment automatically. For detailed instructions, see *QCC711 v2.1 Bluetooth Low Energy Software Programming Guide* (80-70850-1).

2.1 Software dependencies deployment

QCC711 SDK provides powershell scripts to download software dependencies, configure environment variables, and install python package dependencies.

Before running any powershell script, user must configure the powershell execution policy.

1. Open powershell with administrator rights.
2. Change the execution policy. By default, the execution policy of powershell is set to Restricted, which prevents all scripts from running and provides the highest level of security. Change it to RemoteSigned using the command `Set-ExecutionPolicy RemoteSigned`, as shown in the following figure. With this policy, only scripts originating from the internet or a network location must be signed, while local scripts stored on the user computer can be executed without a signature.

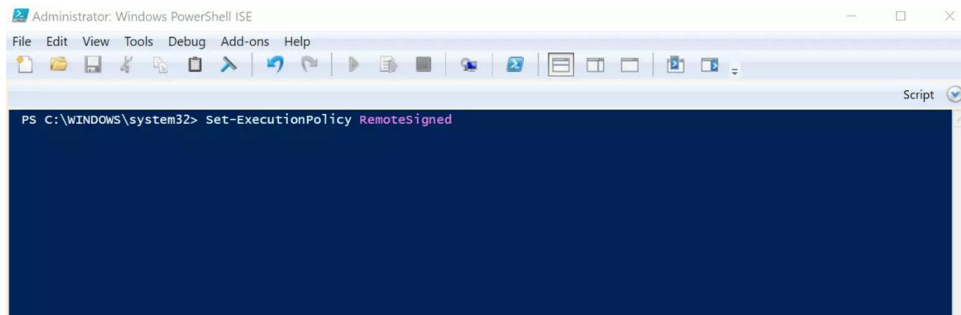


Figure 2-1 Set-ExecutionPolicy command

3. Confirm execution. Depending on the security settings of powershell environment, a prompt may appear asking whether to run the script. Enter `Y` or `A` to agree, or `N` if you do not want to run it.

After switching to RemoteSigned policy, the user can run any powershell without signature or restriction.

2.1.1 Download software dependencies automatically

QCC711 SDK provides a powershell script to automatically check and download the required software dependencies. The power shell script named `QCC711SDK_dependency_download_install.ps1` is located under `SRC_ROOT\qcc711_sdk\tools\scripts`.

Table 2-1 Details to automatically download software dependencies

Script description	Detail	Comments
Name	<code>QCC711SDK_dependency_download_install.ps1</code>	User has to manually install all downloaded packages after executing it.
Parameters	<code>download_dest_path</code>	Optional to specify. If not specified, the default download path is <code>./QCC711SDK_download_path</code> .
Usage	<code>.\QCC711SDK_dependency_download.ps1</code> <code>-download_dest_path ./my_download_folder</code>	-

Since not all dependencies are installed into the system, only a few programs can be detected as installed.

Example of a script output:

```
Summary download status ...
=====
[PKG Name]                                [Status]
Microsoft Windows 10 Enterprise or above----->[Already installed]
J-LINK(7.86f or above)----->[Download complete]
QCC711 Patch(latest or above)----->[Download complete]
Python3(3.11.0.0 or above)----->[Download complete]
gn-windows-amd64(latest or above)----->[Download complete]
ninja(v1.11.1 or above)----->[Download complete]
arm(v11.2.1-1.2 or above)----->[Download complete]
ch347(latest or above)----->[Download complete]
Java(TM) SE Development Kit(15.0.2 or above)----->[Download complete]
IAR Embedded Workbench for ARM(8.12.0 or above)----->[Pls manually download from https://www.iar.com/products/architectures/arm/iar-em
git(2.40.2 or above)----->[Can be installed automatically, refer to QCCIDE release doc]
vscode(1.85.1 or above)----->[Can be installed automatically, refer to QCCIDE release doc]
snapdragon LLVM(12.0.3 or above)----->[No need support]
openssl(3.0.0 or above)----->[Key generation can be done from web - https://www.cryptool.org/en/cto/openssl/]
```

Figure 2-2 A script output example

2.1.2 Configure environment variables and install python packages

QCC711 SDK provides a powershell script to automatically configure system environment variables and install required python package dependencies. The power shell script named `QCC711SDK_set_env_and_install_python_package.ps1` is located under `SRC_ROOT\qcc711_sdk\tools\scripts`.

The script for setting environment should come after executing the download script, and then the user needs to install all dependency non-python packages manually.

Table 2-2 Details to automatically configure environment variables

Script description	Detail	Comments
Name	QCC711SDK_set_env_and_install_python_pkg.ps1	Requires administrator privileges to run.
Parameters	env_config	Mandatory to specify. A JSON file includes all the environment variables that need to be set.
Usage	.\QCC711SDK_set_env_and_install_python_pkg.ps1 ./my_config.json	-

The location for the template of environment variable configuration can be found at SRC_ROOT\qcc711_sdk\tools\scripts\env_config_example.json.

The variable to be set can be divided into two categories:

- Appended to system path
- Set as a new variable

These variables can be configured by the user in JSON format and later loaded by script to configure accordingly.

NOTE User should make sure the path to be added is correct.

Since installing the required python packages depends on the python environment variable being correctly set, the installation will be performed after all environment variables are correctly configured.

```
Summary Env configuration status ...
#####
[ENV Variable]                                     [Status]
sdk_root_path_from_download_path----->[Success]
demoChipBoard----->[Success]
gn----->[Success]
openocd----->[Success]
OPENOCD_PATH----->[Success]
python311----->[Success]
GDB_CLIENT_PATH----->[Success]
ninja----->[Success]
pth_file_name----->[Success]
arm_gcc----->[Success]
compiler_path----->[Success]
```

Figure 2-3 Output example


```

Summary python package install status ...
#####
[Python Package]                                [Status]
pyyaml(6.0.1 or above)-----> [Installed]
pywin32(306 or above)-----> [Installed]
pyserial(3.5 or above)-----> [Installed]
argparse(1.4.0 or above)-----> [Installed]
pycryptodomex(3.18.0 or above)-----> [Installed]
pycryptodome(3.2.0 or above)-----> [Installed]
beautifultable(1.0.1 or above)-----> [Installed]
termcolor(1.1.0 or above)-----> [Installed]
colorama(0.4.4 or above)-----> [Installed]
cryptography(3.4.7 or above)-----> [Installed]

```

2.2 Supported toolchains

The SDK contains build and flash scripts for the toolchains that are listed in the following sections. The following sections use the `set` command to configure environment variables. However, the `set` command only takes effect in the current command window. If you want to make the variables permanently effective, use one of the following methods in Microsoft Windows:

1. Configure environment variables in **Advanced system settings** as shown in [Figure 2-4](#) and [Figure 2-5](#).
2. Use the `setx` command.

NOTE The `setx` command has a limitation of 1024 characters.

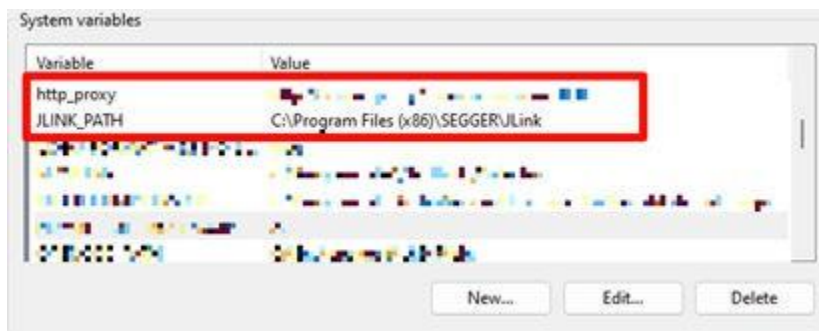


Figure 2-4 System variables

NOTE If you need a proxy to access any of the following resources, set the `http_proxy` variable in Microsoft Windows system to:

`http://<proxy_server>:<proxy_port>`

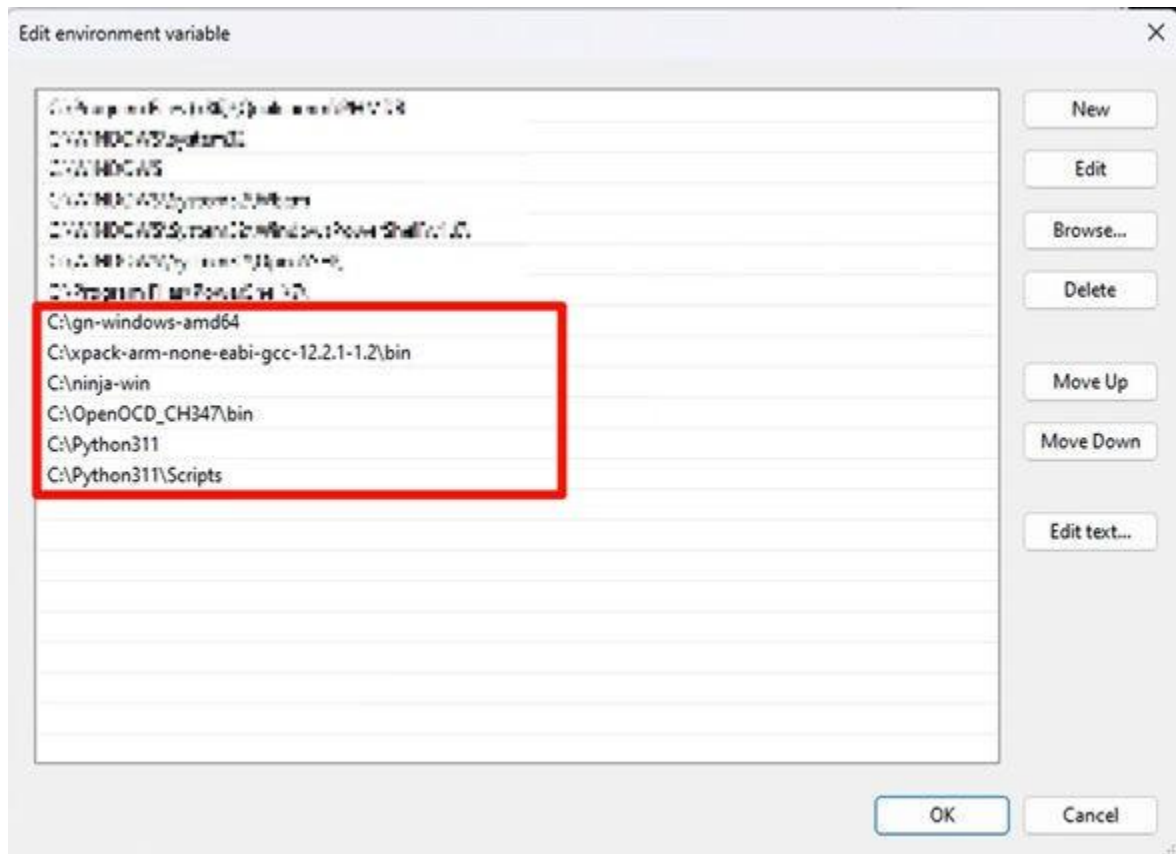


Figure 2-5 Environment variable details

2.2.1 GN package

The supported version is v21.2.1 or higher.

1. Download GN from the following path:
<https://chrome-infra-packages.appspot.com/dl/gn/gn/windows-amd64/+/latest>
2. Add the path of the toolchain binaries to the `PATH` environment variable.

Example:

```
set PATH=C:\gn-windows-amd64;%PATH%
```

2.2.2 xPack Arm-embedded GCC

The supported versions are those between v12.2.1 and 1.2.

1. Download the xPack Arm-embedded GNU C compiler (GCC) toolchain from the following path:
<https://github.com/xpack-dev-tools/arm-none-eabi-gcc-xpack/releases/tag/v12.2.1-1.2/xpack-arm-none-eabi-gcc-12.2.1-1.2-win32-x64.zip>
2. Add the path of the toolchain binaries to the `PATH` environment variable.

Example:

```
set PATH=C:\xpack-arm-none-eabi-gcc-11.2.1-1.2\bin;%PATH%
```


2.3 Other requirements

The following sections describe the components required by the development environment:

2.3.1 Ninja

The supported version is v1.10.2 or higher.

1. Download ninja from the following path:
<https://github.com/ninja-build/ninja/releases/download/v1.11.1/ninja-win.zip>
2. Add the path of the ninja binaries to the `PATH` environment variable.

Example:

```
set PATH=C:\ninja;%PATH%
```

2.3.2 J-Link software

The supported version is v7.64 or higher.

1. Download J-Link software from the following path:
<https://www.segger.com/downloads/jlink/>
2. Add the `JLINK_PATH` environment variable to the system environment and set the value to the directory that contains `JLink.exe`.

Example:

```
set JLINK_PATH=C:\ProgramFiles (x86)\SEGGER\Jlink
```

2.3.3 OpenOCD for JTAG adaptor tool

1. Download the driver (`openocd_ch347.zip`) for high-speed USB adapter chip CH347 from the following path:
https://github.com/seasteh/QCC711_SDK_Patches/raw/master/openocd_ch347.zip.
2. Add the `OPENOCD_PATH` environment variable to the system environment and set the value to the directory that contains `openocd.exe`.

Example:

```
set OPENOCD_PATH=C:\openocd_ch347\bin
```

2.3.4 CH347 Microsoft Windows driver

Download `CH341PAR.EXE` from the following path and run the Microsoft Windows one-click driver installation:

<https://www.wch.cn/downloads/file/64.html>

2.3.5 Python

The supported version is Python 3.11.5.

1. Some support scripts are Python-based. It is recommended to use Python v3.11 downloaded from the following path:

<https://www.python.org/downloads/release/python-3115>

2. Add the path of python binaries to the `PATH` environment variable.

Example:

```
set PATH=C:\Python311;C:\Python311\Scripts;%PATH%
```

3. Install the dependency library `pip3 install pyyaml` from the `C:\Python311\Scripts` folder.

3 Build sample demo applications

To build demo applications, use one of the following methods:

1. QCC IDE application: build a demo with one click. For detailed instructions, see *QCC711 v2.1 Bluetooth Low Energy Software Programming Guide* (80-70850-1).
2. `qccsdk.py` script file: this is provided in the QCC711 SDK and can build sample demo applications.

NOTE To prevent errors, ensure the path to the QCC711 SDK only contains English letters and does not contain spaces.

3.1 Build images using `qccsdk.py`

The `qccsdk.py` file is in the `qcc711_sdk\tools\scripts\qccsdk` folder.

As an example, the following commands build the `appsGeneral` image for CQM711:

```
python qccsdk.py set -I aGen
python qccsdk.py set -B CQM711
python qccsdk.py build --rebuild
```

By default, the generated images are saved to the following directory:

`qcc711_sdk\build\appsGeneral\output_CQM711\bin`

[Table 3-1](#) lists the `qccsdk.py` set parameters.

Table 3-1 `qccsdk.py` set parameters

Parameter	Description
<code>--help</code>	Print help messages.
<code>-I (--image)</code>	Image to build / flash (case insensitive): <ul style="list-style-type: none">■ <code>aGen / appsGeneral</code>■ <code>aGen_AT / appsGeneral_AT</code>■ <code>aPer / appsPeripheral</code>■ For more images, see <code>python qccsdk.py set -help</code>
<code>-B (--board)</code>	Board type: <ul style="list-style-type: none">■ <code>CQM711</code>■ <code>MQM711</code>
<code>-o (--out_dir)</code>	<ul style="list-style-type: none">■ <code>OUTPUT_DIRECTORY</code> (relative path)■ For more options, see <i>QCC711 v2.1 Bluetooth Low Energy Software Programming Guide</i> (80-70850-1).

Table 3-2 lists the qccsdk.py build parameters.

Table 3-2 qccsdk.py build parameters

Parameter	Description
-h (--help)	Print help messages.
-b (--build)	Build to generate images.
-c (--clean)	Clean build.
-r (--rebuild)	Clean and then build.

4 QCC711 image

To flash images, use one of the following methods:

1. QCC IDE application: flash images with one click. For detailed instructions, see *QCC711 v2.1 Bluetooth Low Energy Software Programming Guide* (80-70850-1).
2. `qccsdk.py` or `nvm_programmer.py` script file: these are provided in the QCC711 SDK. For details about their use, see the following sub-sections:
 - [Flash command usage](#)
 - [Autoboot and run the application](#)

Prerequisites

Ensure that one of the demos (`appsGeneral`, `appsGeneral_AT`, `appsPeripheral`, `appsBeacon`, `appsBleCentral`, `appsBlePeripheral`, `appsBlePairing`, `appsBleBas`, `appsBleKpiTest`, `appsBlePawr`, `appsBleOtaUpdate`) have been compiled successfully before flashing.

4.1 Flash command usage

Using `qccsdk.py`

The `qccsdk.py` file is in the `qcc711_sdk\tools\scripts\qccsdk` folder. Use the following command to flash images to a QCC711 board:

```
python qccsdk.py flash --flash --reset
```

OpenOCD is the default mode for flashing images. To use the J-Link interface, run the following command:

```
qccsdk.py set --gdb_server segger
```

Table 4-1 Flash parameters for `qccsdk.py`

Parameter	Description
<code>-h</code> (<code>--help</code>)	Print help messages.
<code>-f</code> (<code>--flash</code>)	Flash images.
<code>-R</code> (<code>--reset</code>)	Reset the board after flash.
<code>-s</code> (<code>--sign</code>)	Flashed sign image.

Using nvm_programmer.py

The `nvm_programmer.py` file is in the `qcc711_sdk\tools\scripts\` folder. The syntax for programming an image to the flash is as follows:

```
nvm_programmer.py [-b ADDRESS] [-U] [-v] [-V] [-t TYPE] [--lock-debug]
filepath
```

Example:

```
python .\nvm_programmer.py --server ch347 --server-port 3333 --client-exe
arm-none-eabi-gdb-py3 --reset -b 0x10210000 -t 2 ..... \build\appsGeneral
\output_CQM711\bin\appsGeneral.bin
```

For more information, see *QCC711 v2.1 Bluetooth Low Energy Software Programming Guide* (80-70850-1)

4.2 Autoboot and run the application

The autoboot mode refers to the mode in which the system boots up upon a power cycle.

When using the demos, if the UART is connected and a **serial console** is active with a port configuration of **115200,8,n,1**, as shown in [Figure 4-1](#), a CLI menu becomes visible. You can then type commands to test the system features.

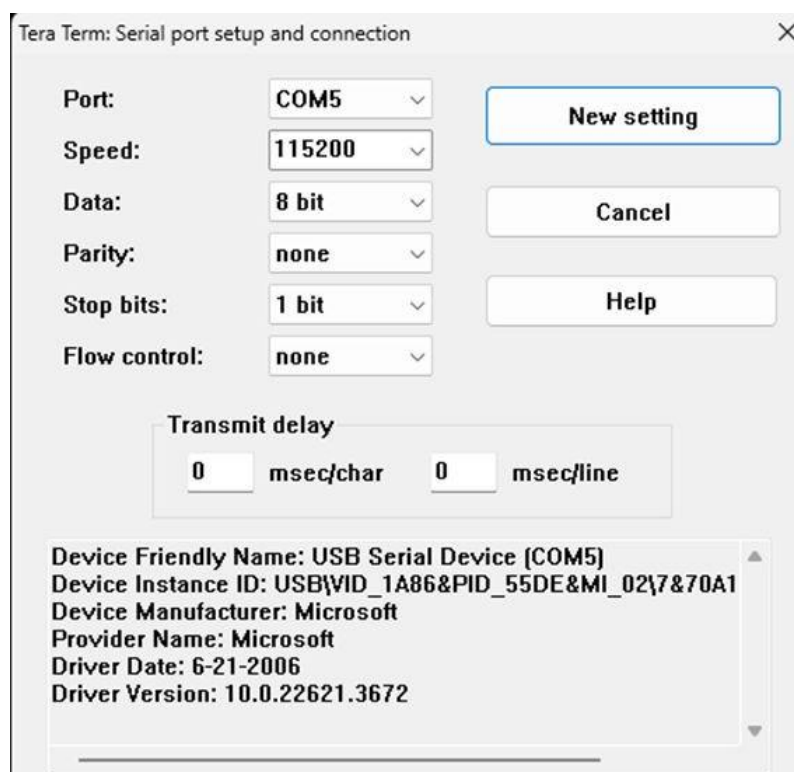
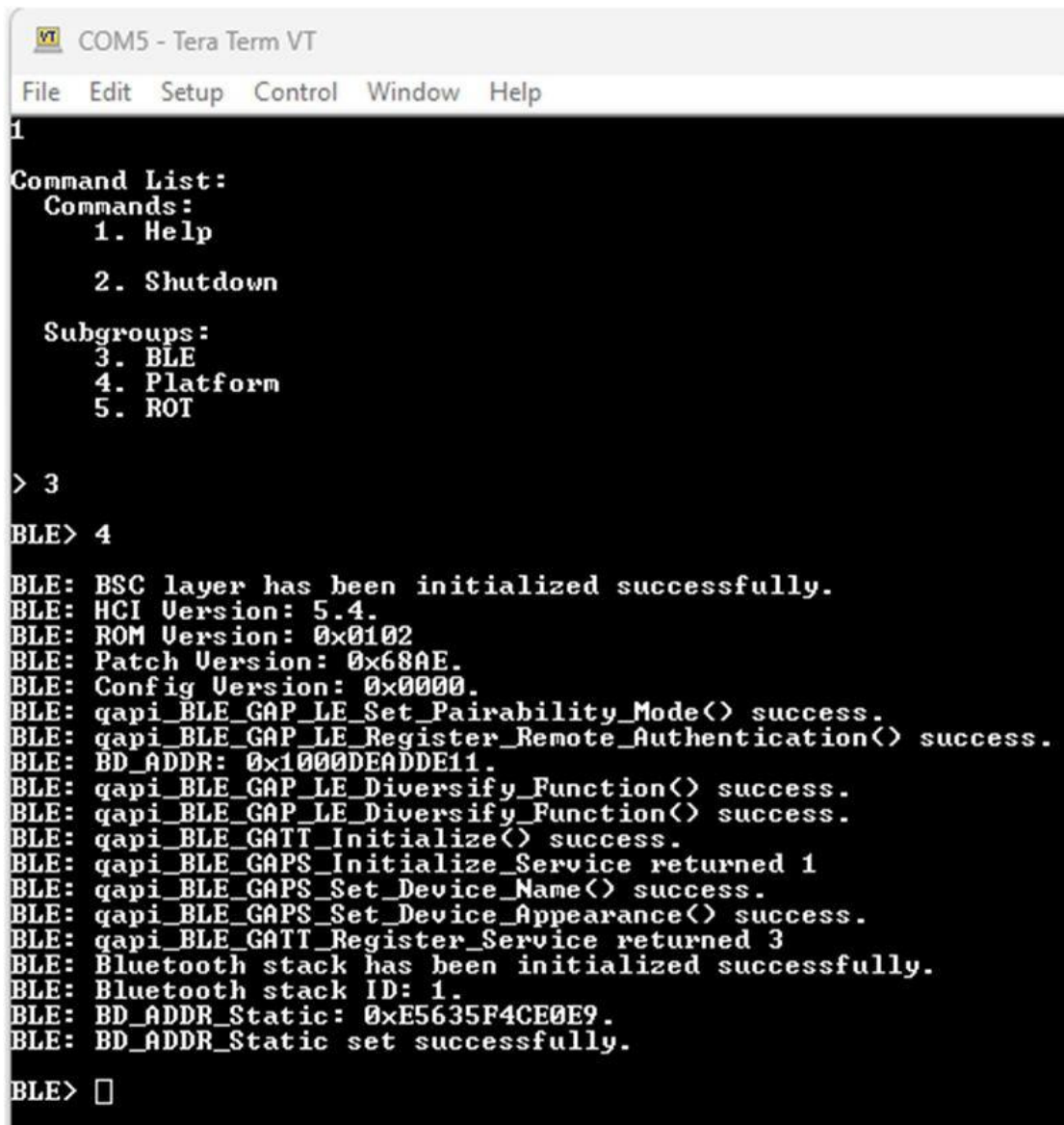


Figure 4-1 UART settings

Figure 4-2 shows the console log.



```
COM5 - Tera Term VT
File Edit Setup Control Window Help

1
Command List:
  Commands:
    1. Help
    2. Shutdown

  Subgroups:
    3. BLE
    4. Platform
    5. ROT

> 3
BLE> 4

BLE: BSC layer has been initialized successfully.
BLE: HCI Version: 5.4.
BLE: ROM Version: 0x0102
BLE: Patch Version: 0x68AE.
BLE: Config Version: 0x0000.
BLE: qapi_BLE_GAP_LE_Set_Pairability_Mode() success.
BLE: qapi_BLE_GAP_LE_Register_Remote_Authentication() success.
BLE: BD_ADDR: 0x1000DEADDE11.
BLE: qapi_BLE_GAP_LE_Diversify_Function() success.
BLE: qapi_BLE_GAP_LE_Diversify_Function() success.
BLE: qapi_BLE_GATT_Initialize() success.
BLE: qapi_BLE_GAPS_Initialize_Service returned 1
BLE: qapi_BLE_GAPS_Set_Device_Name() success.
BLE: qapi_BLE_GAPS_Set_Device_Appearance() success.
BLE: qapi_BLE_GATT_Register_Service returned 3
BLE: Bluetooth stack has been initialized successfully.
BLE: Bluetooth stack ID: 1.
BLE: BD_ADDR_Static: 0xE5635F4CE0E9.
BLE: BD_ADDR_Static set successfully.

BLE> □
```

Figure 4-2 appsGeneral console log

QCC IDE also provides a console that can communicate with the QCC711 development board. For more details, see *QCC711 v2.1 Bluetooth Low Energy Software Programming Guide* (80-70850-1).

5 Program the Bluetooth MAC address

The Bluetooth devices are identified using a Bluetooth device address (BD_ADDR). The following commands can be used to read the BD_ADDR from the MTP, and write it to the MTP, which can be programmed using the `otp_programmer.py` tool.

The `otp_programmer.py` file is in the `qcc711_sdk\tools\scripts\` folder. For more information, see *QCC711 v2.1 Bluetooth Low Energy Software Programming Guide* (80-70850-1).

- To read the BD_ADDR, use: `python .\otp_programmer.py --read BD_ADDR -s ch347 -p 3333`
- To write the BD_ADDR, use: `python .\otp_programmer.py --write BD_ADDR=0x0202DEAD0001 -s ch347 -p 3333`

Document references

Document	Reference
<i>QCC711 v2.1 Bluetooth Low Energy Software Programming Guide</i>	80-70850-1

Glossary

Term	Definition
Bluetooth	Set of technologies providing audio and data transfer over short-range radio connections
GCC	GNU C compiler
MAC	Medium access control
SDK	Software development kit

LEGAL INFORMATION

Your access to and use of this material, along with any documents, software, specifications, reference board files, drawings, diagnostics and other information contained herein (collectively this “Material”), is subject to your (including the corporation or other legal entity you represent, collectively “You” or “Your”) acceptance of the terms and conditions (“Terms of Use”) set forth below. If You do not agree to these Terms of Use, you may not use this Material and shall immediately destroy any copy thereof.

1) Legal Notice.

This Material is being made available to You solely for Your internal use with those products and service offerings of Qualcomm Technologies, Inc. (“Qualcomm Technologies”), its affiliates and/or licensors described in this Material, and shall not be used for any other purposes. If this Material is marked as “Qualcomm Internal Use Only”, no license is granted to You herein, and You must immediately (a) destroy or return this Material to Qualcomm Technologies, and (b) report Your receipt of this Material to qualcomm.support@qti.qualcomm.com. This Material may not be altered, edited, or modified in any way without Qualcomm Technologies’ prior written approval, nor may it be used for any machine learning or artificial intelligence development purpose which results, whether directly or indirectly, in the creation or development of an automated device, program, tool, algorithm, process, methodology, product and/or other output. Unauthorized use or disclosure of this Material or the information contained herein is strictly prohibited, and You agree to indemnify Qualcomm Technologies, its affiliates and licensors for any damages or losses suffered by Qualcomm Technologies, its affiliates and/or licensors for any such unauthorized uses or disclosures of this Material, in whole or part.

Qualcomm Technologies, its affiliates and/or licensors retain all rights and ownership in and to this Material. No license to any trademark, patent, copyright, mask work protection right or any other intellectual property right is either granted or implied by this Material or any information disclosed herein, including, but not limited to, any license to make, use, import or sell any product, service or technology offering embodying any of the information in this Material.

THIS MATERIAL IS BEING PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, WHETHER EXPRESSED, IMPLIED, STATUTORY OR OTHERWISE. TO THE MAXIMUM EXTENT PERMITTED BY LAW, QUALCOMM TECHNOLOGIES, ITS AFFILIATES AND/OR LICENSORS SPECIFICALLY DISCLAIM ALL WARRANTIES OF TITLE, MERCHANTABILITY, NON-INFRINGEMENT, FITNESS FOR A PARTICULAR PURPOSE, SATISFACTORY QUALITY, COMPLETENESS OR ACCURACY, AND ALL WARRANTIES ARISING OUT OF TRADE USAGE OR OUT OF A COURSE OF DEALING OR COURSE OF PERFORMANCE. MOREOVER, NEITHER QUALCOMM TECHNOLOGIES, NOR ANY OF ITS AFFILIATES AND/OR LICENSORS, SHALL BE LIABLE TO YOU OR ANY THIRD PARTY FOR ANY EXPENSES, LOSSES, USE, OR ACTIONS HOWSOEVER INCURRED OR UNDERTAKEN BY YOU IN RELIANCE ON THIS MATERIAL.

Certain product kits, tools and other items referenced in this Material may require You to accept additional terms and conditions before accessing or using those items.

Technical data specified in this Material may be subject to U.S. and other applicable export control laws. Transmission contrary to U.S. and any other applicable law is strictly prohibited.

Nothing in this Material is an offer to sell any of the components or devices referenced herein.

This Material is subject to change without further notification.

In the event of a conflict between these Terms of Use and the Website Terms of Use on www.qualcomm.com, the *Qualcomm Privacy Policy* referenced on www.qualcomm.com, or other legal statements or notices found on prior pages of the Material, these Terms of Use will control. In the event of a conflict between these Terms of Use and any other agreement (written or click-through, including, without limitation any non-disclosure agreement) executed by You and Qualcomm Technologies or a Qualcomm Technologies affiliate and/or licensor with respect to Your access to and use of this Material, the other agreement will control.

These Terms of Use shall be governed by and construed and enforced in accordance with the laws of the State of California, excluding the U.N. Convention on International Sale of Goods, without regard to conflict of laws principles. Any dispute, claim or controversy arising out of or relating to these Terms of Use, or the breach or validity hereof, shall be adjudicated only by a court of competent jurisdiction in the county of San Diego, State of California, and You hereby consent to the personal jurisdiction of such courts for that purpose.

2) Trademark and Product Attribution Statements.

Qualcomm is a trademark or registered trademark of Qualcomm Incorporated. Arm is a registered trademark of Arm Limited (or its subsidiaries) in the U.S. and/or elsewhere. The Bluetooth® word mark is a registered trademark owned by Bluetooth SIG, Inc. Other product and brand names referenced in this Material may be trademarks or registered trademarks of their respective owners.

Snapdragon and Qualcomm branded products referenced in this Material are products of Qualcomm Technologies, Inc. and/or its subsidiaries. Qualcomm patented technologies are licensed by Qualcomm Incorporated.

THE DOCUMENTATION ACCOMPANYING THE MATERIALS AND/OR RELEVANT PRODUCTS AND SERVICE OFFERINGS MAY INCLUDE IMPORTANT USE LIMITATIONS. ANY DEVIATIONS FROM APPLICABLE USE LIMITATIONS MAY ADVERSELY IMPACT PERFORMANCE, DURABILITY, QUALITY OR SAFETY. YOU ASSUME ALL RISKS AND LIABILITIES ASSOCIATED WITH ANY DEVIATIONS