User Manual

DA16200/DA16600 ThreadX SDK Startup Guide

UM-WI-037

Abstract

This document describes how to develop a Wi-Fi application with DA16200 and DA16600.

Contents

Figures

Tables

1 Terms and Definitions

DPM	Dynamic Power Management
UART	Universal Asynchronous Receiver-Transmitter
RTC	Real-Time Clock
WPS	Wi-Fi Protected Setup
SDK	Software Development Kit
CLI	Command Line Interface
EVK	Evaluation Kit
IAR	Abbreviation of Ingenjörsfirman Anders Rundgren (Anders Rundgren Engineering Company)
RF	Radio Frequency
UDP	User Datagram Protocol
WPA	Wi-Fi Protected Access

2 References

[1] UM-WI-023, DA16200_EVK_User_Manual, User Manual, Dialog Semiconductor
[2] UM-WI-002, DA16200_SDK_Programmer_Guide, User Manual, Dialog Semiconductor
[3] UM-WI-007, DA16200_Example_Application_Manual, User Manual Dialog Semiconductor
[4] UM-WI-010, DA16200_MQTT_Programmer_User_Manual, User Manual, Dialog Semiconductor
[5] UM-WI-005, DA16200_DPM_Manager_User_Manual, User Manual, Dialog Semiconductor
[6] AN-WI-002_DA16200_BT_Coexist_Application_Note, Application Note, Dialog Semiconductor
[7] UM-WI-028, DA16200_SoftAP_User_Provisioning, User Manual, Dialog Semiconductor
[8] UM-WI-029, DA16200_WiFi_Connection_Notification, User Manual, Dialog Semiconductor
[9] DA16200_OTA_Update_MCU_Firmware_Dev., Dialog Semiconductor
[10] UM-WI-018, DA16600 Example Application Manual, Dialog Semiconductor

3 Introduction

The DA16200 is a highly integrated ultra-low power Wi-Fi system on a chip that allows users to develop a complete Wi-Fi solution on a single chip. Using the DA16200 and DA16600 SDK in IAR Embedded Workbench for ARM, users can implement and test Wi-Fi application. This document describes how to set up the development environment for the DA16200/DA16600 EVB and SDK.

4 Register and Download SDK

The latest version of the official SDK can be found on the Dialog website: https://www.dialog-semiconductor.com/products/wi-fi/.

All documentation and SDKs can be found under the **Resources** tab of the specific product page.

NOTE
Some of the content requires registration before it can be downloaded. Register to download the SDK.

5 Structure of SDK

The SDK folder structure is shown in Figure 1. The UM-WI-02 DA16200 SDK Programmer guide describes SDK structures in Section 2.1 of the Ref. [2].

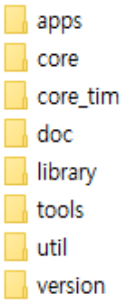

Figure 1: SDK Folder Structure

5.1 SDK and Images

The DA16200 SDK and images are provided. The images are provided for various applications.

These prebuilt images can be used for testing the DA16200 features or can be used directly in a production product without building the SDK.

Table 1: SDK Images

Image	Description
DA16200_IMG_ThreadX_QFN_vx.x.x.x DA16600_IMG_ThreadX_QFN_vx.x.x.x	Default image.
DA16xxx_IMG_ThreadX_ATCMD_QFN_vx.x.x.x DA16xxx_IMG_ThreadX_WPA3_ATCMD_QFN_vx.x.x.x	Image for AT commands.
DA16200_IMG_ThreadX_WPA3_QFN_vx.x.x.x	Image for WPA3 protocol.
DA16200_IMG_ThreadX_Manufacture_QFN_vx.x.x.x	Image for RF test in production.

6 Build SDK and Download Image

6.1 Build the SDK

There is an IAR project file for both the DA16200 SDK and the DA16600 SDK. To build the SDK:

1. Open the IAR workspace file:
   - For DA16200: `apps\da16200\get_started\project\DA16200.eww`
   - For DA16600: `apps\da16600\get_started\project\DA16600.eww`
2. Select all subprojects under the DA16200 or DA16600 project, and then click the **Rebuild All** or **Make**.

NOTE
The DA16200 and DA16600 SDK can only be compiled with *IAR 7.30.4.xxxx tools*. On how to install the IAR tools, see Ref. [2].

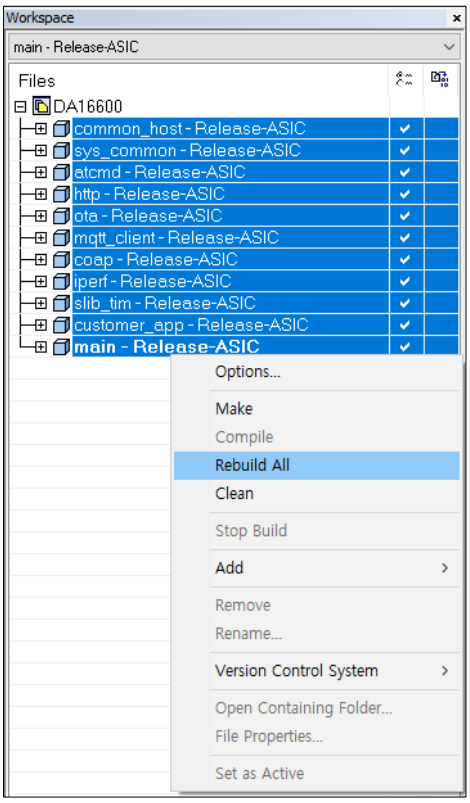

Figure 2: Build the SDK

6.1.1 Build SDK with Another Serial Flash Memory

The DA16200 and DA16600 SDK use the Winbond serial flash memory as default. If another serial flash is used for your target system that is supported by the SDK and can be found in `~\SDK\tools\SBOOT\SFDP` directory, then the configuration file for serial flash `~\SDK\tools\SBOOT\cmconfig\fc9ktpmconfig.cfg` should be changed.

Follow the instruction to build the SDK with another serial flash memory:

1. Remove the `~\SDK\tools\SBOOT\cmconfig\fc9ktpmconfig.cfg` file.
2. Copy and paste the configuration file that matches the serial flash model and partition what you want to use:

   For example, if the Winbond W25Q32JW and 4 MB partition should be used,

   `fc9ktpmconfig.cfg. W25Q32JW(4MB)` ➔ `fc9ktpmconfig.cfg`

6.1.2 Build SDK for 4 MB SFLASH Images

To create 4 MB SFLASH images, follow the instructions:

1. Change the build option for 4 MB partition where is in `~\SDK\apps\da16200\get_started\inc\config_generic_sdk.h`.
   - `#undef __FOR_4MB_SFLASH__` ➔ `#define __FOR_4MB_SFLASH__`
2. Use a proper configuration file the serial flash:
   For example, if you are using Winbond W25Q32JW device,
   `~\SDK\tools\SBOOT\cmconfig\fc9ktpmconfig.cfg.W25Q32JW(4MB)`
   ➔ `~\SDK\tools\SBOOT\cmconfig\fc9ktpmconfig.cfg`

NOTE
Please refer to Appendix C for making 4 MB image in case of the SDK v2.3.4.2 or earlier version.

6.1.3 Build Manufacture SDK

There is a manufacture SDK for DA16200 and DA16600 and it has a function to check/measure RF performance and they are implemented by AT command. So, the SDK can be used in production. The UI-WI-003 AT command User Manual document describes the AT commands.

There is an IAR workspace file on DA16200 and DA16600 Manufacture SDK. To build the SDK:

1. Open the IAR workspace file:
   - For DA16200: `~\SDK\apps\da16200\manufacture\project\DA16200_Manufacture.eww`
   - For DA16600: `~\SDK\apps\da16600\manufacture\project\DA16600_Manufacture.eww`
2. Select all subprojects under the DA16xx workspace, and then click **Rebuild All** or **Make**.

6.1.4 Build SDK for fcCSP Package

The DA16200 SDK provides a QFN package type Ram library as default but it also has a ram library for the fcCSP package that can be used for SDK build. And, there are two Tx power modes with the fcCSP package, and they can be distinguished by filename extensions which are fcCSP_LP and fcCSP_NP.

They will be a part of the SLIB firmware of DA16200 after SDK build.

To create a RAM Library image for the fcCSP package with the DA16200 SDK, follow the instructions below:

1. fcCSP Compile feature: `~\SDK\apps\da16200\get_started\inc\sys_common_features.h`
   `#undef FOR_FCCSP_SDK__` ➔ `#define __FOR_FCCSP_SDK__`
2. Low TX power
   `~\SDK\tools\SBOOT\image\DA16xxx_slib_ramlib.bin.fcCSP_LP`
   ➔ `~\SDK\tools\SBOOT\image\DA16xxx_slib_ramlib.bin`
   `~\SDK\tools\SBOOT\image\DA16xxx_slib_ramlib.rtm.fcCSP_LP`
   ➔ `~\SDK\tools\SBOOT\image\DA16xxx_slib_ramlib.rtm`
   `~\SDK\apps\da16200\get_started\inc\sys_common_features.h`
   `#define __FCCSP_LOW_POWER__ // Low TX power`
3. Normal TX power
   `~\SDK\tools\SBOOT\image\DA16xxx_slib_ramlib.bin.fcCSP_NP`
   ➔ `~\SDK\tools\SBOOT\image\DA16xxx_slib_ramlib.bin`
   `~\SDK\tools\SBOOT\image\DA16xxx_slib_ramlib.rtm.fcCSP_NP`
   ➔ `~\SDK\tools\SBOOT\image\DA16xxx_slib_ramlib.rtm`


```
~\SDK\apps\da16200\get_started\inc\sys_common_features.h
#undef __FCCSP_LOW_POWER__ // Normal TX power
```

An SoC package type can be checked through the “SDK version” information that is shown during boot like “V2.x.x.x CSP LP” or “V2.x.x.x CSP NP”.

```
****************************************************
*            DA16200 SDK Information
* ------------------------------------------------
*
* - CPU Type         : Cortex-M4 (80MHz)
* - OS Type          : ThreadX 5.7
* - Serial Flash     : 4 MB
* - SDK Version      : V2.4.0.0 CSP LP
* - F/W Version      : RTOS-GEN01-01-14648-000000
*                    : SLIB-GEN01-01-14645-000000
* - F/W Build Time   : Jul 15 2021 16:35:59
* - Boot Index       : 0
*
****************************************************
```

Figure 3: Boot Logo with fcCSP LP Image

```
****************************************************
*            DA16200 SDK Information
* ------------------------------------------------
*
* - CPU Type         : Cortex-M4 (80MHz)
* - OS Type          : ThreadX 5.7
* - Serial Flash     : 4 MB
* - SDK Version      : V2.4.0.0 CSP NP
* - F/W Version      : RTOS-GEN01-01-14648-000000
*                    : SLIB-GEN01-01-14645-000000
* - F/W Build Time   : Jul 15 2021 16:59:17
* - Boot Index       : 0
*
****************************************************
```

Figure 4: Boot Logo with fcCSP NP Image

6.2 Download an Image

You can download the firmware image using a **.ttl** macro file in **Tera Term**. The **.ttl** macro files are located in the image package with the name *Download_xxxx.ttl*. See Ref. [1].

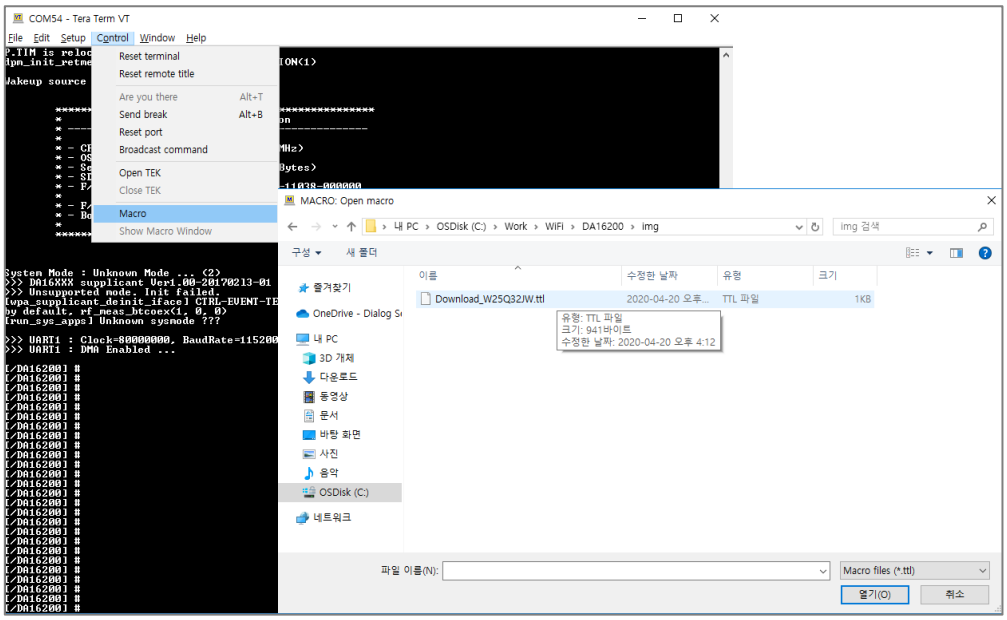

Figure 5: Download an Image

NOTE
There is a boot index information in the sflash address of 0x9000. If you use a macro, it will be erased to make the region for boot index 0. If you download an image manually, you must check the region you download is matched with the boot index.

7 Sample Project Testing

The sample project is in `~\SDK\apps\common\examples` folder and has the folder structure as Figure 6. And, each sample project has two directories that are `project` and `src`. An `img` directory will be created as a build result. See HTML file (`~\SDK\doc\html\index.html`) and Ref. [3].

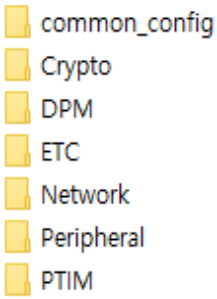

Figure 6: Sample Project Folder Structure

The DPM samples are like the normal samples except they have additional logic for the DPM control.

Each sample exists in each project independently, so you can build and test independently with other sample images that also exist independently.

7.1 Test the Sample Project

There are three workspace files in each sample project directory and one of them can be chosen to task a test:

- `DA16200_doorbell_ref_sample.eww`
- `DA16200_sample.eww`
- `DA16600_sample.eww`

To test a sample project:

1. Run `project\DA16xxx_sample.eww` in a project.
2. Build a sample and download images located in the `img` folder of the project.

7.2 Reference Test of UDP

See Ref. [3].

All sample projects are in `~\SDK\apps\common\examples` directory.

From now on, it is referred to as SAMPLE_DIR.

7.2.1 Block Diagram

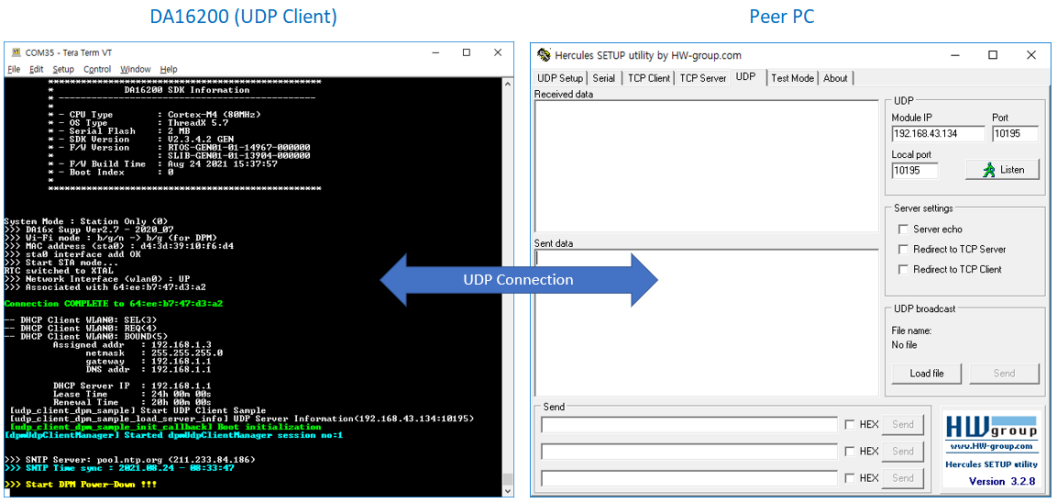

Figure 7: UDP Block Diagram

This example is described based on the Hercules utility. You can download it from https://www.hw-group.com/software/hercules-setup-utility.

7.2.2 DA16200 Setup Procedure for the UDP Client

The project file is in `SAMPLE_DIR\Network\UDP_Client_DPM`.

To set up DA16200 for the UDP Client:

1. Build `SAMPLE_DIR\Network\UDP_Client_DPM\project\DA16200_sample.eww`.
2. Download an image from `SAMPLE_DIR\Network\UDP_Client_DPM\img`.
3. Set the Station mode for testing the DPM mode by the CLI command. Do not enable the DPM mode during setup.

```
[/DA16200] #setup
```

4. Set up the UDP server information (Server IP and Server Port: the peer PC is the UDP server).

```
/* A server IP and port number should be checked as the IP and port below is
just example. */
[/DA16200] # nvram.setenv UDPC_SERVER_IP 192.168.43.134
[/DA16200] # nvram.setenv UDPC_SERVER_PORT 10195
[/DA16200] # dpm on
```

After DPM is on, the DA16200 restarts automatically.

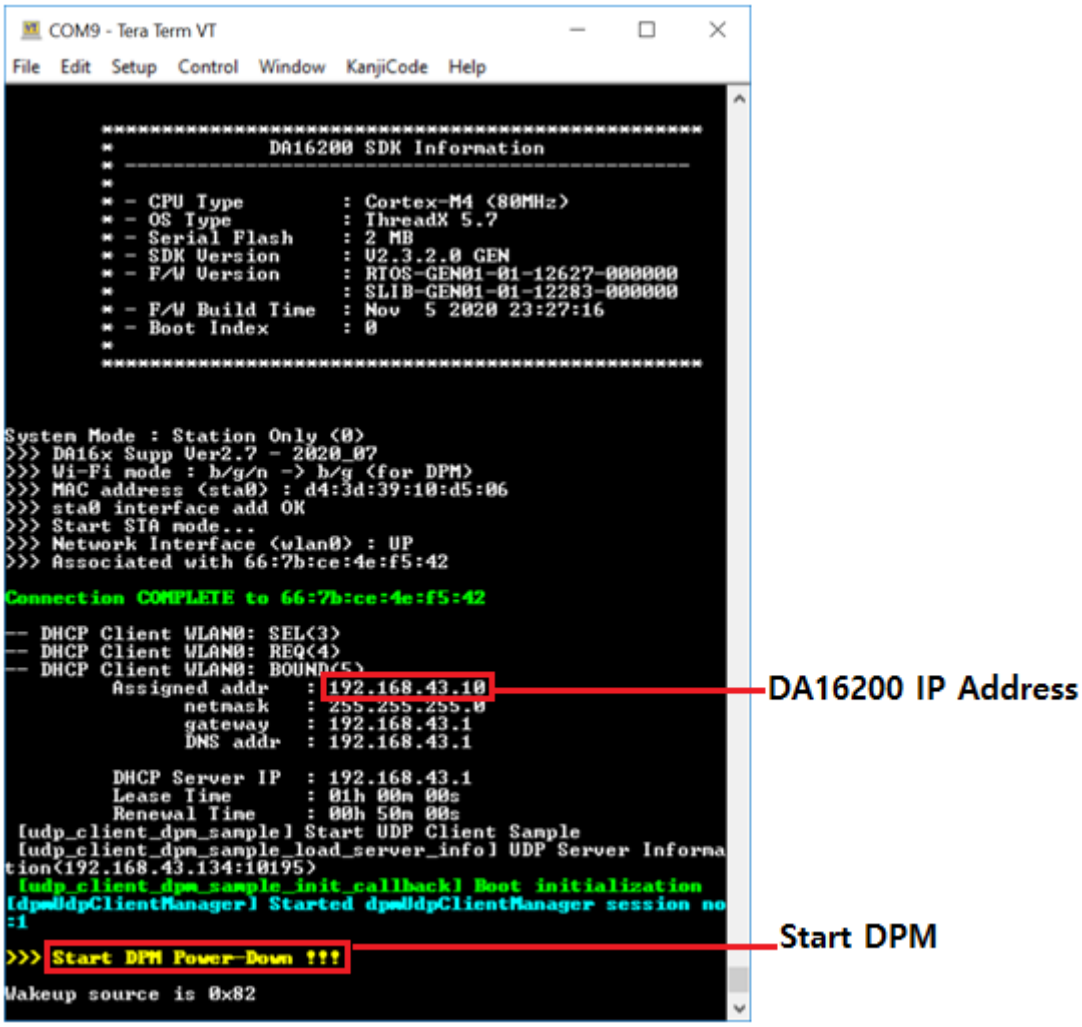

Figure 8: DA16200 Setup for the UDP Client

7.2.3 Peer PC (Hercules) Setup Procedure for the UDP Server

To set up peer PC for the UDP Server:

1. Run Hercules.
2. Select UDP (a).
3. Set the Client IP address and the UDP Port (DA16200 is the UDP Client) (b).
4. Set the UDP Server port (c).

5. Open the UDP connection (d).
6. Send the data (e).
   - The sent data is displayed under **Sent data** (f).
   - The received data is displayed under **Received data** (g).

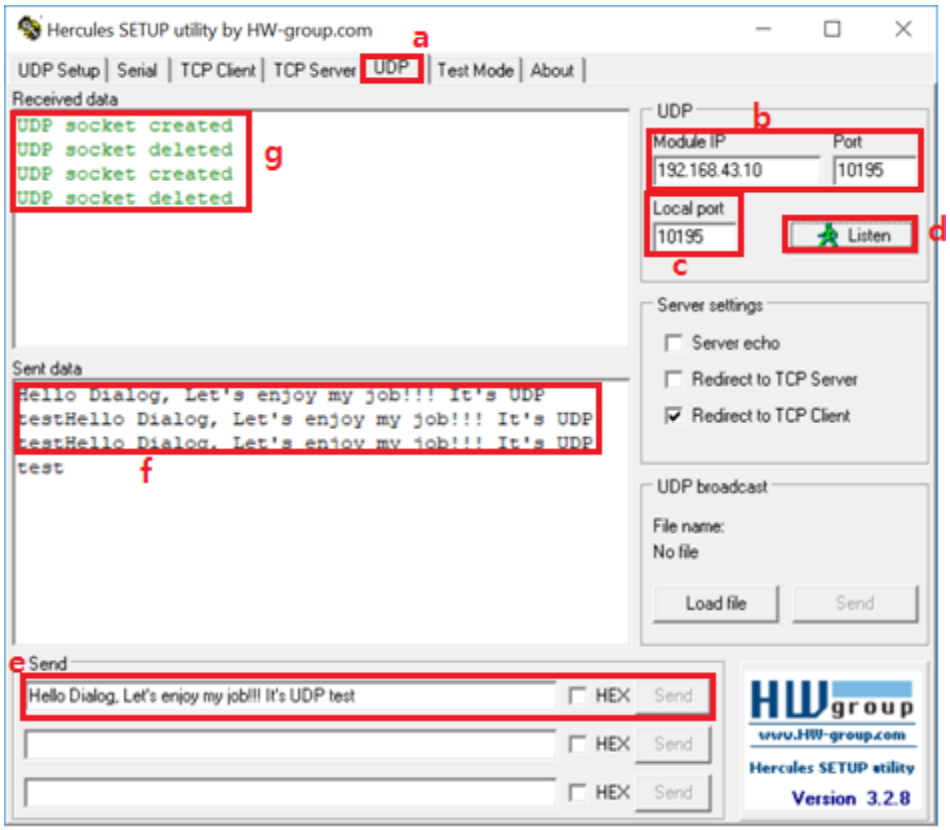

Figure 9: Peer PC Setup for the UDP Client

7.2.4 Test Sequence

The test sequence for UDP includes the following:

1. UDP Server sends the data (a).
2. UDP Server displays the sent data (b).
3. UDP Client wakes up by RTC timer(c).
4. UDP Client receives the data (d).
5. UDP Client sends the received data (e).
6. UDP Server receives the data (f).

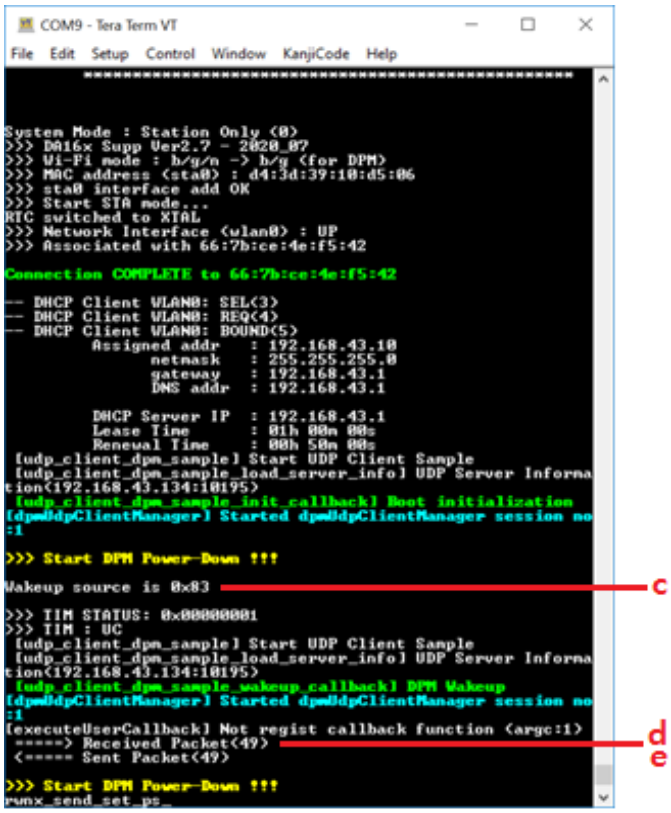

Figure 10: The UDP Client

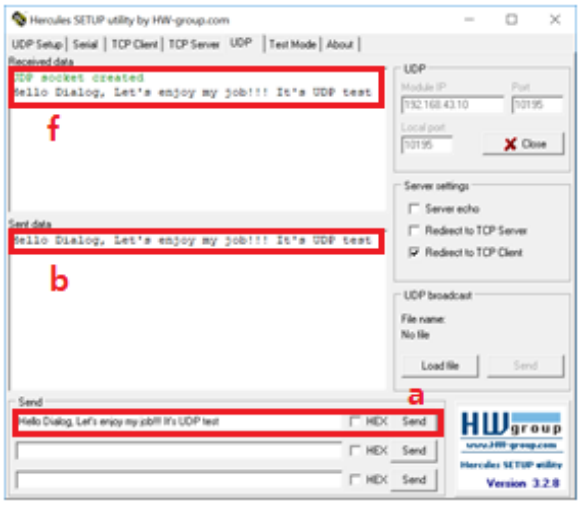

Figure 11: The UDP Server

7.3 Reference Test of TCP

See Ref. [3].

7.3.1 TCP Block Diagram

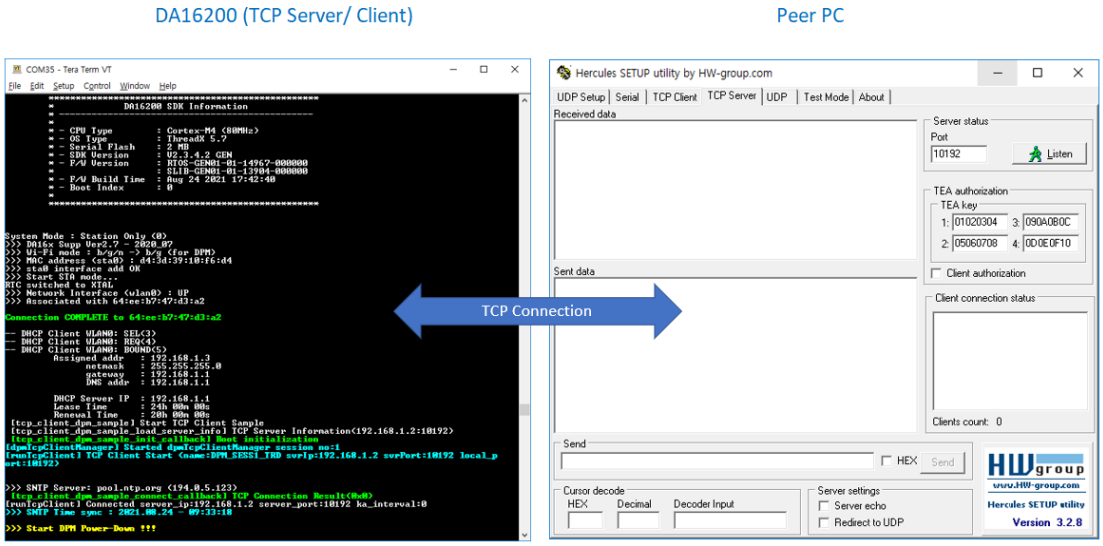

Figure 12: TCP Block Diagram

7.3.1.1 DA16200 Setup Procedure for the TCP Client

The project file is in `SAMPLE_DIR\Network\TCP_Client_DPM`.

To set up the DA16200 for the TCP Client:

1. Build `SAMPLE_DIR\Network\TCP_Client_DPM\project\DA16200_sample.eww`.
2. Download an image from `SAMPLE_DIR\Network\TCP_Client_DPM\img`.
3. Set the Station mode for testing the DPM mode by the CLI command. Do not enable the DPM mode during setup.

```
[/DA16200] #setup
```

4. Set up the TCP Server information (Server IP and Server Port: peer PC is the TCP Server).

```
[/DA16200] # nvram.setenv TCPC_SERVER_IP 192.168.1.2
[/DA16200] # nvram.setenv TCPC_SERVER_PORT 10192
[/DA16200] # dpm on
```

After DPM is on, the DA16200 restarts automatically.

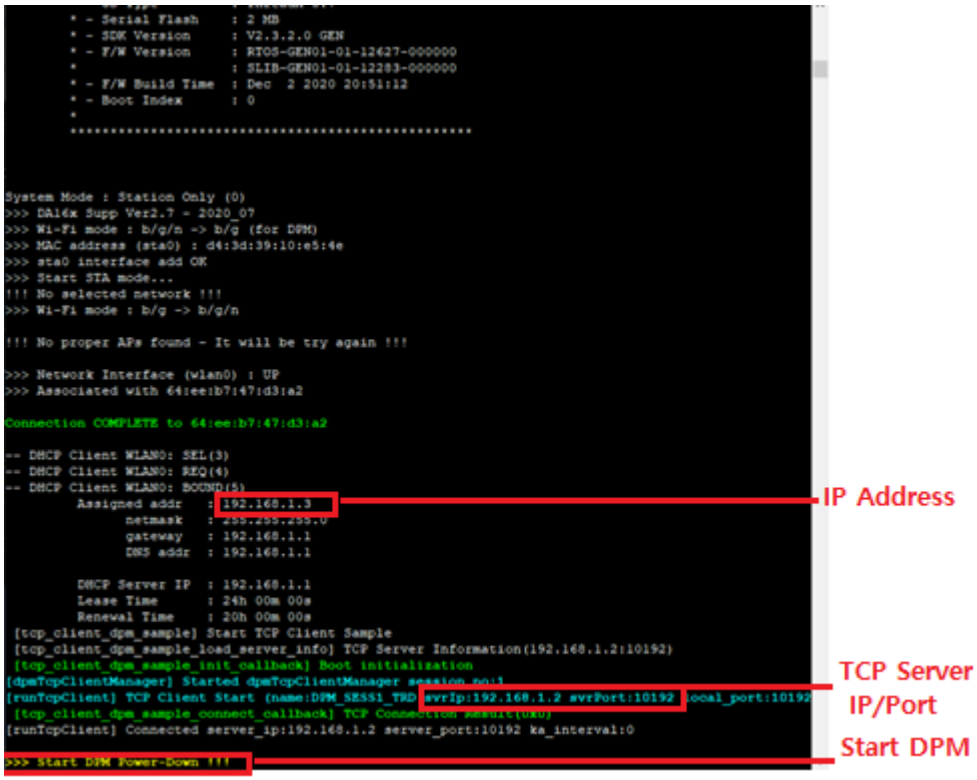

Figure 13: DA16200 Setup for the TCP Client

7.3.1.2 Peer PC (Hercules) Setup Procedure for the TCP Client

To set up the peer PC (Hercules) for the TCP Client:

1. Run Hercules.
2. Select the TCP Client (a).
3. Set the TCP Server IP Address and Port to connect (b and c).
4. Test the TCP connection (d).
5. Send the data (e).
6. The connection information, the sent data, and received data are displayed under **Received/Sent data** (f).

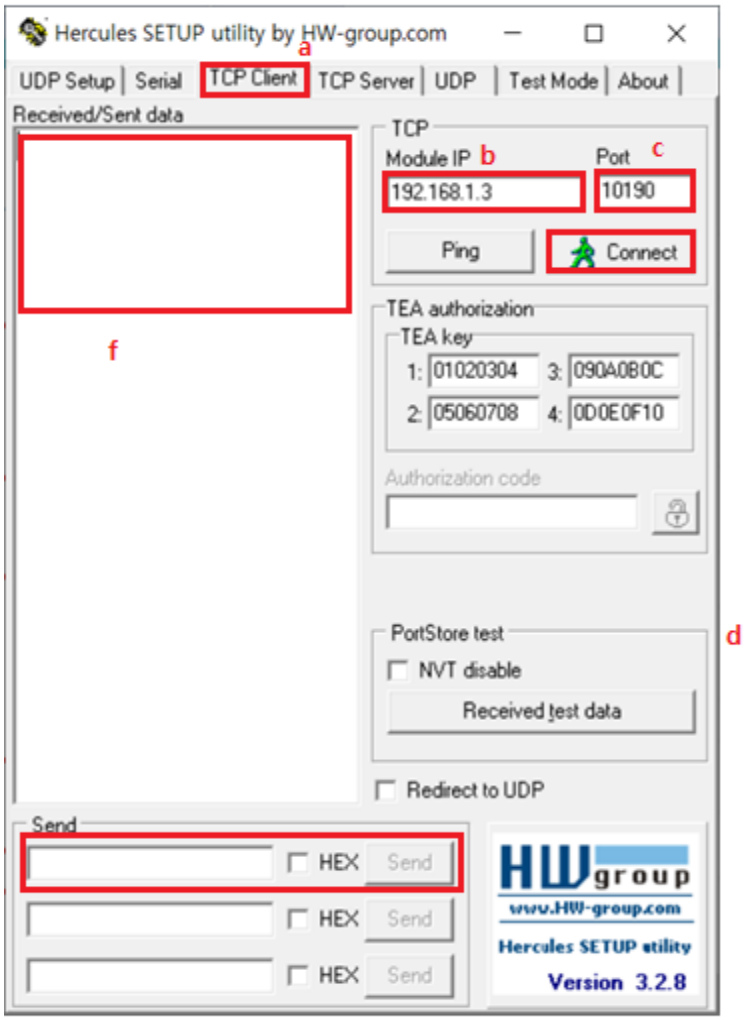

Figure 14: Peer PC Setup for the TCP Client

7.3.1.3 Test Sequence

The test sequence for TCP includes the following:

1. Restart DA16200, and then click the **Connect** button.
2. TCP Client shows the TCP Connection information (a).
3. TCP Client sends the data (b).
4. TCP Server displays the sent data (in pink color) (c).
5. TCP Client wakes up by RTC timer (d).
6. TCP Client receives the data (e).

7. TCP Client sends the received data (f).
8. TCP Server receives the data (in black color) (g).

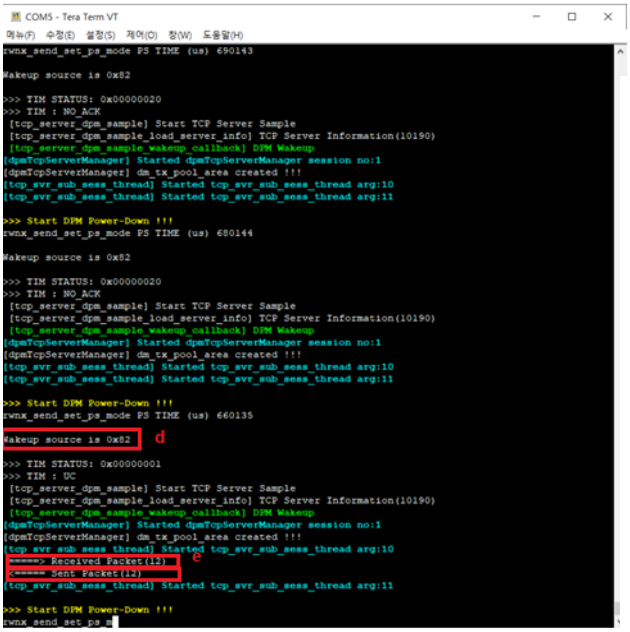

Figure 15: TCP Server

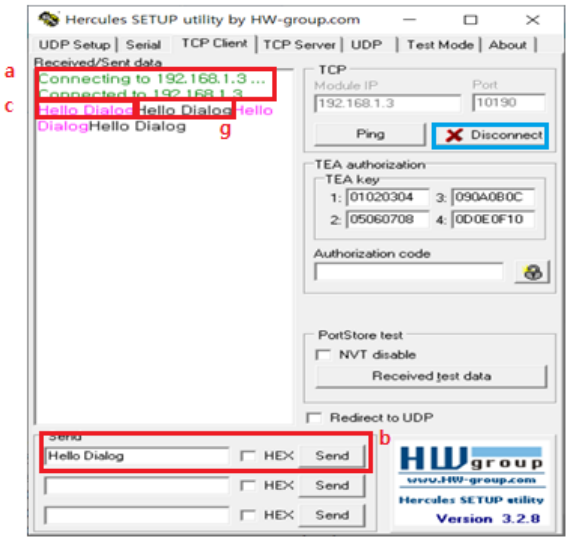

Figure 16: TCP Client

7.4 Reference Test of MQTT

See Ref. [4].

Appendix A Application Features

Users can enable or disable custom features in config_generic_sdk.h.

There are three header files for the configuration:

- `config_generic_sdk.h`: user configuration for the DA16200/DA16600.
- `sys_common_features.h`: system configuration for the DA16200/DA16600.
- `ble_combo_features.h`: Bluetooth® LE configuration for the DA16600.

A.1 config_generic_sdk.h

Table 2: Features in config_generic_sdk.h

Feature	Description
`__SUPPORT_WPS_BTN__`	To support WPS that works with GPIO6.
`__SUPPORT_FACTORY_RESET_BTN__`	To support the **Factory Reset** button that is mapped to GPIO7. If it is detected, the factory reset function executes.
`__SET_BOR_CIRCUIT__`	To enable brown and black out interrupt.
`__SUPPORT_DPM_MANAGER__`	To use DPM manager, enable it then the DPM process runs automatically. See Ref. [5].
`__SUPPORT_CONSOLE_PWD__`	To protect the log display or command through the UART0 console. A password is required to access the console (the default password is da16200 for the DA16200).
`__SUPPORT_SECURITY_HW_INIT__`	To initialize H/W cryptography during boot-up.
`__SUPPORT_BTCOEX__`	Enable to control the RF switching between Wi-Fi and Bluetooth. It is implemented with three-wire logic. See Ref. [6].
`__SUPPORT_DHCP_SVR__`	To run a DHCP server that can be executed in Soft-AP mode.
`__SUPPORT_SNTP_CLIENT__`	To use the SNTP client.
`__SUPPORT_NSLOOKUP__`	To support nslookup network utility.
`__SUPPORT_OTA__`	To support firmware update via OTA. For more information on the OTA sample in SDK, see SAMPLE_DIR\Network\OTA_Update. See Ref. [3].
`__SUPPORT_HTTP_SERVER__`	To use the HTTP server. See the HTTP sample in SDK: SAMPLE_DIR\Network\HTTP_Server.
`__SUPPORT_HTTP_CLIENT__`	To use the HTTP client. See the HTTP sample in SDK: SAMPLE_DIR\Network\HTTP_Client(DPM).
`__SUPPORT_ZERO_CONFIG__`	To support mDNS, DNS-SD, and xmDNS.
`__SUPPORT_MQTT__`	To use MQTT to connect with the server.
`__SUPPORT_UART1__`	To communicate with Host (MCU) through UART. For more information on the UART1 sample in SDK, see SAMPLE_DIR\Peripheral\UART1.
`__SUPPORT_UART2__`	To communicate with Host (MCU) through UART2.

Feature	Description
__SUPPORT_ATCMD__	To enable AT command function that communicates with Host (MCU).
__SUPPORT_WIFI_CONN_CB__	To use the callback with the status when Wi-Fi is connected or disconnected. See Ref. [8].
__SUPPORT_IPERF__	To enable the iperf utility for throughput measurement.
__SUPPORT_PROVISION__	To enable the provisioning feature. See Ref. [7].
__SET_WAKEUP_HW_RESOURCE__	To use RTC_WAKE_UP pin as interrupt while DA16200 is awake. (DA16200 can be woken up from sleep by triggering GPIO signal from external MCU although it is not enabled.)
__FOR_4MB_SFLASH__	To use 4 MB sflash memory map.

A.2 sys_common_features.h

Some features may not affect the built firmware, because some of them are included as a library.

Table 3: Features in sys_common_features.h

Feature	Description
__ENABLE_UART1_CLOCK__	To use the UART1 clock.
__ENABLE_GPIO_CLOCK__	To use GPIO clock.
__SUPPORT_MULTI_IP_IF__	To support multiple IPs.
__SUPPORT_WPA3__ __SUPPORT_IEEE80211W __SUPPORT_WPA3_SAE__ __SUPPORT_WPA3_OWE__	To use WPA3. ● To use PMF ● To use SAE ● To use OWE
__SUPPORT_FAST_CONN_SLEEP_12__	To make the Wi-Fi connection faster when wake-up from sleep1 or 2.
__USER_DPM_ABNORM_WU_INTERVAL__	To coordinate abnormal action when Wi-Fi cannot be connected. sleep duration and wake-up time can be configured.
__OTA_HTTP_CLIENT__	To support OTA to update the DA16200 images. See OTA sample in SDK: SAMPLE_DIR\OTA_Update. See Ref. [3].
__OTA_UPDATE_MCU_FW_	To use the OTA process to upgrade MCU. See Ref. [9].

A.3 ble_combo_features.h

Table 4: Features in ble_combo_features.h

Feature	Description
__COMBO_SAMPLE_BLE_PERI_WIFI_SVC__	To use Bluetooth® LE peripheral/Wi-Fi service including provisioning, OTA and gas leak detection example as described in Ref. [10] – sections 3.2, 3.3, and 3.4.
__COMBO_SAMPLE_BLE_PERI_WIFI_SVC_TCP_DPM_SAMPLE__	To use Bluetooth® LE peripheral/Wi-Fi service including provisioning, OTA and DA14531 peripheral example as described in Ref. [10] – sections 3.2, 3.3, and 3.5.
__COMBO_SAMPLE_BLE_PERI_WIFI_SVC_PERIPHERAL_SAMPLE__	To use Bluetooth® LE peripheral/Wi-Fi service including provisioning, OTA and TCP client example as described in Ref. [10] – sections 3.2, 3.3, and 3.6.
__COMBO_SAMPLE_BLE_CENT_SENSOR_GW__	To use Bluetooth® LE Central role function, provisioning, OTA and IoT sensor gateway example as described in Ref. [10] – section 3.2, 3.3, and 3.7.
__GTL_IFC_UART__	Enable Bluetooth® LE UART interface
__SUPPORT_BTCOEX__	Enable BT Coexistence
__SUPPORT_BTCOEX_1PIN__	To use 1 pin for Bluetooth® coexistence instead of 3 pins.
__ENABLE_RTC_WAKE_UP1_INT__ __ENABLE_RTC_WAKE_UP2_INT__	Enable RTC_WAKE_UP1 or 2.
__ENABLE_BLE_WKUP_BEFORE_SEND_MSG__	Enable Bluetooth® LE wake-up before sending GTL message to DA14531.
__DA14531_BOOT_FROM_UART__	Enable to transfer DA14531 FW via UART during boot.
__WIFI_SVC_SECURITY__	Enable Wi-Fi SVC Security, see more details in Ref. [10] – section 4.2.
__WFSVC_ENABLE__	Enable GATT service for Wi-Fi Provisioning.
__LOW_POWER_IOT_SENSOR__	To use Gas leak detection sensor example in Ref. [10] – Section 5.4.
__ENABLE_DPM_FOR_GTL_BLE_APP__	Enable DPM for GTL BLE application.
__SUPPORT_DA14531_GPIO_CONTROL__	To enable DA14531 simple GPIO on/off control.
__SUPPORT_PROVISION__	To enable provisioning by DA14531(BLE)

Appendix B SDK Structure Changes

This document describes changes in SDK structure and IAR projects in DA16200/DA16600 ThreadX SDK v2.4.0.0 to make porting for the user easier compared to the previous version.

B.1 SDK Structure

The SDK v2.4.0.0 has 8 folders:

- **apps**: There are three folders under the apps folder for each project. The project includes IAR project files, applications, and examples for a customer.
- **core**: source codes
- **core_time**: source codes for TIM SDK
- **doc**: user documents (user guides, programmer guides, etc.)
- **library**: to which the pre-compiled lib files (.a) are saved
- **tools**: Build scripts, temporary build artifacts, or environment files
- **util**: utilities for customer
- **version**: version files to include when Image created

B.2 Changes in SDK Structure

The SDK v2.4.0.0 adds the new customer folder for customer code to be developed independently of the SDK core code.

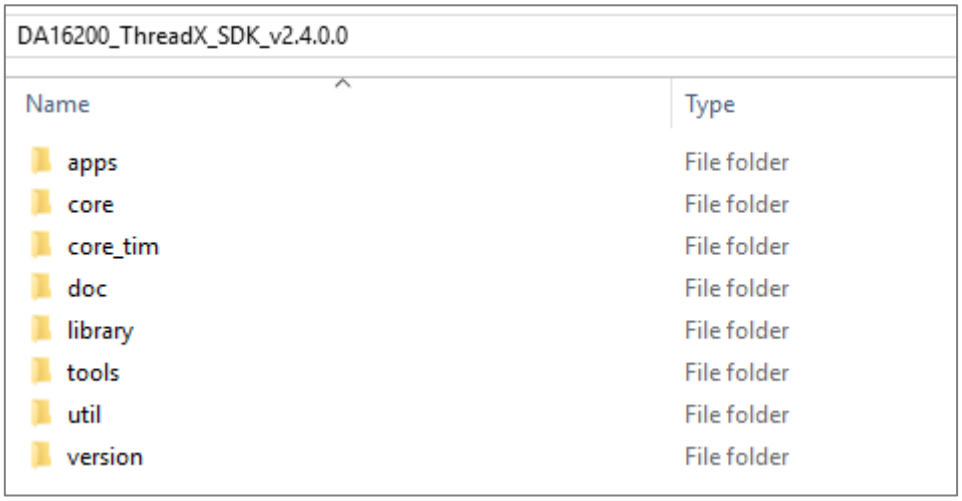

Figure 17: SDK Structure in v2.4.0.0

B.3 Customer Folder

The user application codes of v2.4.0.0 are combined in a new customer folder that is composed of three subfolders:

- **common**: common applications like examples
- **da16200**: user applications for DA16200
- **da16600**: user applications for DA16600

Figure 18: New Customer Folder in v2.4.0.0

B.4 User Application Codes

The user application codes of v2.4.0.0 are distributed in five folders and DA16600 has **ble** folder additionally.

- **ble**: Bluetooth® LE and Wi-Fi combo application (only for DA16600)
- **img**: DA16200 Images
- **inc**: header files for user application
- **lib**: library files (.a)
- **project**: project files to build SDK
- **src**: source files for user application

Figure 19: User Application Code for DA16200 in v2.4.0.0

Figure 20: User Application Code for DA16600 in v2.4.0.0

The user application codes of v2.4.0.0 are combined in `~\SDK\apps\da16200\get_started\src` folder. In case of DA16600 user application code, they can be found in `~\SDK\apps\da16600\get_started\src folder.`

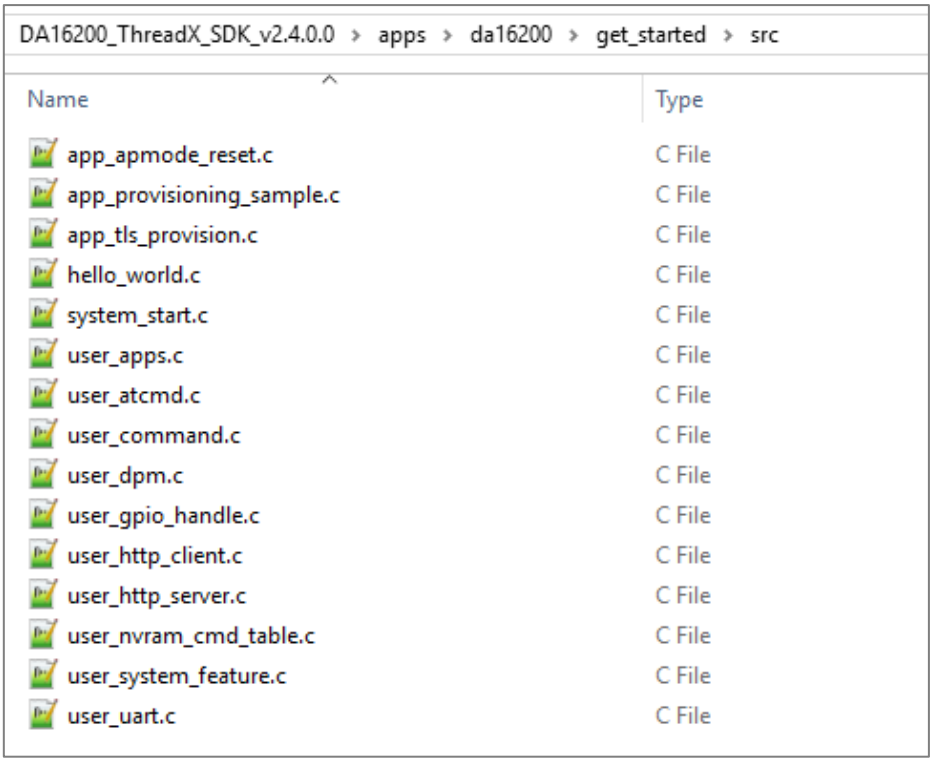

Figure 21: User Application Code in v2.4.0.0

B.5 IAR Project

B.5.1 IAR Workspace File

The IAR IDE workspace file in v2.4.0.0 is `~\SDK\apps\da16200\get_started\project\DA16200.eww` and the `~\SDK\apps\da16600\get_started\project\DA16600.eww` workspace file can be used for DA16600.

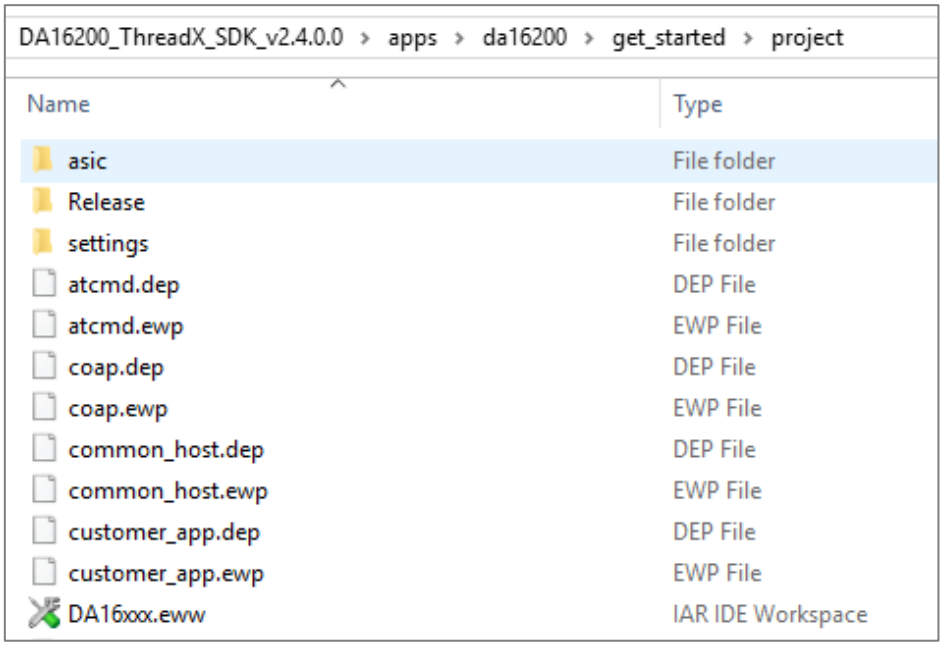

Figure 22: IAR IDE Workspace File in v2.4.0.0

B.5.2 IAR Project Structure

The DA16200 SDK v2.4.0.0 has 11 projects:

- **common_host**: Host interface initialize functions project
- **sys_common:** system common project
- **atcmd:** AT commands project
- **http:** HTTP client and server project
- **ota**: Over the air firmware update project
- **mqtt**: MQTT subscriber and publisher project
- **coap**: CoAP client and server project
- **iperf**: NetX/NetX Duo IPerf project
- **slib_tim**: PTIM project
- **customer_app**: customer application project
- **main**: main project

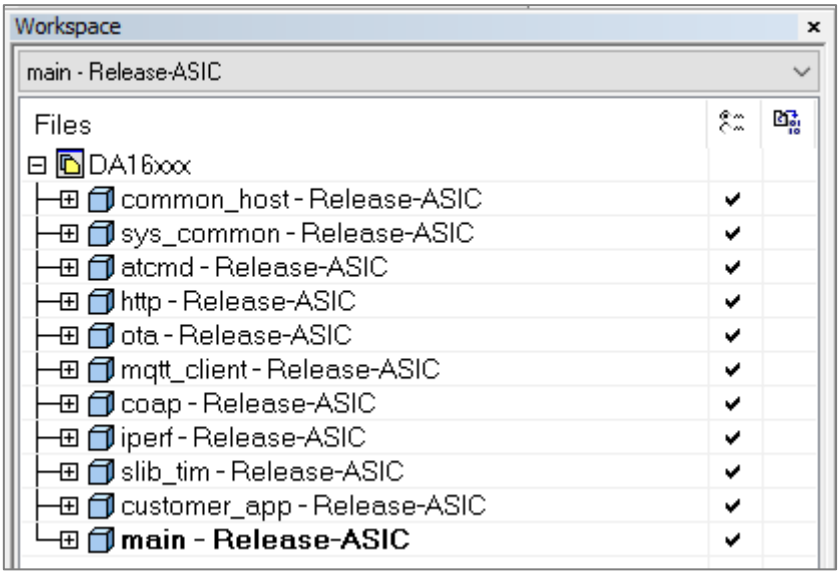

Figure 23: IAR Project Structure in v2.4.0.0

Appendix C Making 4 MB SFLASH Images

The DA16200 SDK basically uses a 2 MB SFLASH memory map. To create an image for a 4 MB SFLASH memory map on the DA16200 SDK, it is required to change some files as in Table 5 and then build the SDK.

This section is for the SDK v2.3.4.2 or earlier version.

Table 5: Files to Create 4 MB SFLAH Memory Map

Items	Description
2nd Bootloader file	~\SDK\tools\SBOOT\image\DA16xxx_ueboot.bin.4MB ➔ ~\ SDK\tools\SBOOT\image\DA16xxx_ueboot.bin
Configuration file	~\SDK\tools\SBOOT\cmconfig\fc9ktpmconfig.cfg.XXXXXX(4MB) ➔ ~\SDK\tools\SBOOT\cmconfig\fc9ktpmconfig.cfg
Load script file	~\SDK\tools\ldscripts\DA16xxx_rtos_cache.icf.4MB ➔ ~\SDK\tools\ldscripts\DA16xxx_rtos_cache.icf
Macro file	~\SDK\tools\macro\da16200_asic_cache.mac.4MB ➔ ~\SDK\tools\macro\da16200_asic_cache.mac
Compile feature	~\SDK\apps\da16200\get_started\inc\config_generic_sdk.h #undef __FOR_4MB_SFLASH__ ➔ #define __FOR_4MB_SFLASH__

Revision History

Revision	Date	Description
1.5	28-Mar-2022	Update logo, disclaimer, copyright.
1.4	02-Sep-2021	Updated structure, path, and features of SDK to cover SDK v2.4. Added how to make Manufacture and fcCSP LP/NP Image. Added Appendix B and C.
1.3	09-Jul-2021	Updated for product version v2.4.0.0. Removed HW section for EVK.
1.2	25-Mar-2021	Modified description for DA16200 V2.3.4.0 SDK update.
1.1	10-Feb-2021	Section 5.1 ● Removed SDK Types. ● Table 1 Removed two SDK images. Section 6.16.1.4 Added a note about boot index. Section 7.2.2 Updated location path. Section 7.3.1.1 Updated location path. Appendix A ● Table 2 Updated features. ● Table 3 Updated features.
1.0	18-Dec-2020	First Release.

Status Definitions

Status	Definition
DRAFT	The content of this document is under review and subject to formal approval, which may result in modifications or additions.
APPROVED or unmarked	The content of this document has been approved for publication.

RoHS Compliance

Dialog Semiconductor's suppliers certify that its products are in compliance with the requirements of Directive 2011/65/EU of the European Parliament on the restriction of the use of certain hazardous substances in electrical and electronic equipment. RoHS certificates from our suppliers are available on request.

Important Notice and Disclaimer

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu

Koto-ku, Tokyo 135-0061, Japan

www.renesas.com

Contact Information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:

https://www.renesas.com/contact/

Trademarks