DA16200 DA16600 Host Interface and AT Command

This document describes how to use Host interfaces and AT commands in DA16200 and DA16600.

Contents

Contents **1**
Figures **3**
Tables **3**
1. Terms and Definitions **5**
2. References **6**
3. Host Interface **7**
3.1 UART Host Interface 7
3.1.1 PIN MUX Configuration 7
3.2 SPI Host Interface 7
3.2.1 PIN MUX Configuration 7
3.2.2 SPI Protocol 8
3.2.3 AT Command – Sequences and Structures 11
3.2.4 ESC Command – Sequences and Structures 12
3.2.5 Header Format 14
3.2.6 SPI Definition and Structure 15
3.3 SDIO Host Interface 15
3.3.1 PIN MUX Configuration 15
3.3.2 SDIO Protocol 15
3.3.3 AT Command – Sequences and Structures 18
3.3.4 ESC Command – Sequences and Structures 20
3.3.5 SDIO Definition and Structures for Implementation 21
4. AT Commands **22**
4.1 Overview 22
4.2 Add AT Command Feature in SDK 22
4.2.1 Execute AT Commands on SPI 22
4.2.2 Execute AT Commands on SDIO 23
4.3 AT Command Format 23
4.3.1 Basic Command Format 23
4.3.2 Extended Command Format 24
4.3.3 Response Format 24
5. AT Command Sets **26**
5.1 Basic Function Commands 26
5.2 Network Function Commands 38
5.3 Wi-Fi Function Commands 45
5.4 Wi-Fi Function Commands for WPA3 63
5.5 Wi-Fi Function Commands for WPA Enterprise 65

5.5.1 WPA-Enterprise Connection Example 67
5.6 Advanced Function Commands 68
5.6.1 MQTT Commands 68
5.6.2 HTTP-Client Commands 91
5.6.3 HTTP-Server Commands 97
5.6.4 WebSocket-Client Commands 98
5.6.5 OTA Commands 99
5.6.6 Zeroconf Commands 106
5.6.7 Provision Commands Over Wi-Fi 108
5.6.8 Bluetooth® LE Commands 110
5.6.9 Transfer Function Commands 111
5.6.10 RF Test Function Commands 123
5.6.11 System and Peripheral Function Commands 129
6. AT Command Example 139
6.1 Data Transfer Test 139
6.1.1 TCP Server Socket Test 139
6.1.2 TCP Client Socket Test 140
6.1.3 UDP Socket Test 141
Appendix A License Information 143
Appendix B HTTP API Return Values 144
B.1 Return Value as Defined By NetX Duo HTTP 144
B.2 Return Value as Defined by LWIP HTTP 144
Appendix C User UART Configuration 146
C.1 How to Run AT Command on UART2 146
C.2 User UART Configuration 146
C.3 Use Case 147
Appendix D DA16200/DA16600 Cipher Suites 148
Appendix E Reason Code for Wi-Fi Connection Failure or Disconnection 150
Appendix F Fast Reconnect Function 153
F.1 Technical Overview 153
Appendix G Bluetooth® LE Coexistence Feature 154
Appendix H Wi-Fi Passive-Scan 155
H.1 Passive-Scan with Specified Channel and Scan-Time Limit 155
H.2 Passive-Scan Result 155
H.3 Passive-Scan Stop 155
H.4 Passive-Scan Sequence 155
Appendix I Detailed Error Codes for AT Command 157
Appendix J AT Command Development Environment Configuration 168
J.1 How to Connect DA16200/DA16600 Board 168
J.2 Configure Serial Port for UART 169
J.3 Configuration for MCU Wake-Up (Optional) 170
7. Revision History 170

Figures

Figure 1: Basic format - SPI....8
Figure 2: Write sequence....9
Figure 3: Write operation structure....9
Figure 4: Read sequence....10
Figure 5: Read operation structure....10
Figure 6: AT command sequence....11
Figure 7: AT command structure....12
Figure 8: ESC command sequence....13
Figure 9: ESC command structure....13
Figure 10: SPI signals for write request....14
Figure 11: SPI signals for read response....14
Figure 12: Basic format - SDIO....16
Figure 13: Write sequence....16
Figure 14: Structure....17
Figure 15: Read sequence....18
Figure 16: Structure....18
Figure 17: AT command sequence....19
Figure 18: Structure....19
Figure 19: ESC command sequence....20
Figure 20: Structure....20
Figure 21: Example sequence to initiate AT command through SDIO interface....23
Figure 22: Example sequence to initiate MQTT protocol with DPM....81
Figure 23: Procedure to send MQTT messages....82
Figure 24: Procedure to process MQTT messages....83
Figure 25: Broker console - CleanSession=1 Connection....85
Figure 26: Broker console - CleanSession=0 Connection....85
Figure 27: Hercules – TCP client socket setting....140
Figure 28: Hercules – TCP server socket setting....141
Figure 29: Hercules – UDP socket setting....142
Figure 30: Passive-scan result....156
Figure 31: AT command development environment....168
Figure 32: Check COM ports on device manager....169
Figure 33: Initial setup to start with AT command....170
Figure 34: GPIO wake-up....170

Tables

Table 1: PIN MUX configuration - UART....7
Table 2: PIN MUX configuration - SPI....7
Table 3: SPI address list....8
Table 4: SPI CMD format....8
Table 5: PIN MUX configuration -SDIO....15
Table 6: Address list....16
Table 7: Basic function command list....26
Table 8: Initiation response list....37
Table 9: Network function command list....38
Table 10: Certificate commands....44
Table 11: Wi-Fi function command list....45
Table 12: Wi-Fi function response list....62
Table 13: WPA3-related Wi-Fi function command list....63
Table 14: WPA-enterprise Wi-Fi function commands....65
Table 15: WPA-enterprise network function command....67
Table 16: MQTT configuration command list....68
Table 17: MQTT operation command list....74
Table 18: MQTT optional configuration commands....76
Table 19: MQTT response list....78
Table 20: CleanSession and QoS matrix in message Rx....86

Table 21: CleanSession and QoS matrix in message Tx 86
Table 22: HTTP-Client command list 91
Table 23: HTTP-Client response list 95
Table 24: HTTP-Server command list 97
Table 25: WebSocket-Client command list 98
Table 26: Web Socket-Client response list 98
Table 27: OTA command list 99
Table 28: OTA response list 104
Table 29: OTA response code list 104
Table 30: Zerocont command list 106
Table 31: Provision command list 109
Table 32: atcmd_provision_stat enumeration 109
Table 33: Bluetooth® LE command list 110
Table 34: Socket command list 111
Table 35: Socket connection response list 114
Table 36: Data transmission command 115
Table 37: Data reception responses 118
Table 38: Secure socket command list 120
Table 39: RF test command list 123
Table 40: RF test examples 128
Table 41: SPI command list 129
Table 42: OTP command list 130
Table 43: OTP address for XTAL offset 131
Table 44. Memory size by XTAL offset 131
Table 45: XTAL command list 131
Table 46: Flash dump command list 132
Table 47: GPIO command list 132
Table 48: LED command list 134
Table 49: PWM command list 135
Table 50: ADC command list 135
Table 51: I2C command list 137
Table 52: Sleep command list 137
Table 53: CALWL command list 138
Table 54: Return value as defined by Netx Duo HTTP 144
Table 55: Return value as defined by LWIP HTTP 144
Table 56: DA16200/DA16600 cipher suites 148
Table 57: AT command error codes 157

1. Terms and Definitions

AP	Access Point
ASCII	American Standard Code for Information Interchange
AT	Attention
CA	Certificate Authority
CCMP	Counter Mode Cipher Block Chaining Message Authentication Code Protocol
CID	Client ID
CMD	Command
COM	Communication Port
CRC	Cyclic Redundancy Check
CW	Continuous Wave
DHCP	Dynamic Host Configuration Protocol
DPM	Dynamic Power Management
EAP	Extensible Authentication Protocol
ESS	Extended Service Set
FW	Firmware
GPIO	General Purpose Input Output
HTTP	Hypertext Transfer Protocol
ICMP	Internet Control Message Protocol
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet Protocol
JSON	JavaScript Object Notation
LMAC	Low MAC
MAC	Medium Access Control
MCU	Micro Controller Unit
MQTT	Message Queuing Telemetry Transport
NVRAM	Non-Volatile RAM
OTA	Over the Air
OTP	One-Time Programmable
OWE	Opportunistic Wireless Encryption
PBC	Push Button Connection
PC	Personal Computer
PER	Packet Error Rate
PSK	Pre-Shared Key
QoS	Quality of Service
RTC	Real-Time Clock
RTOS	Real-Time Operating System
RTS	Request to Send
RX	Receive
SAE	Simultaneous Authentication of Equals
SDK	Software Development Kit
SDIO	Secure Digital Input Output
SNTP	Simple Network Time Protocol
SPI	Serial Peripheral Interface
SSID	Service Set Identifier
STA	Station
TCP	Transport Control Protocol

TIM	Traffic Indication Map
TKIP	Temporal Key Integrity Protocol
TLS	Transport Layer Security
TX	Transmit
UART	Universal Asynchronous Receiver-Transmitter
UDP	User Datagram Protocol
USB	Universal Serial Bus
URL	Universal Resource Locator
WEP	Wired Equivalent Privacy
WLAN	Wireless Local Area Network
WMM	Wi-Fi Multimedia
WPA	Wi-Fi Protected Access version 1
WPA2	Wi-Fi Protected Access version 2
WPS	Wi-Fi Protected Setup
XTAL	Crystal

2. References

[1] UM-WI-056, DA16200 DA16600 FreeRTOS Getting Started Guide, User Manual, Renesas Electronics
[2] UM-WI-046, DA16200 DA16600 FreeRTOS SDK Programmer Guide, User Manual, Renesas Electronics
[3] DA16200 Datasheet, Renesas Electronics
[4] UM-WI-004, DA16200 AT GUI Tool, User Manual, Renesas Electronics
[5] UM-WI-042, DA16200 DA16600 Provisioning Mobile App for Android/iOS
[6] UM-WI-030, DA16200 DA16600 DPM User Manual
[7] UM-WI-011, DA16200 DA16600 Mass Production User Manual

3. Host Interface

This application note describes how an external processor system (referred to as External Host) communicates with a DA16200 via SPI and SDIO physical interface protocols. This document also includes the AT Command Protocol to be used with the External Host.

3.1 UART Host Interface

3.1.1 PIN MUX Configuration

DA16200 can use two interfaces, UART1 and UART2, and DA16600 can use only UART2. UART1 can be assigned to GPIOA[1:0], GPIOA[3:2], GPIOA[5:4], or GPIOA[7:6], and UART2 is to GPIOA[11:10] or GPIOC[7:6]. UART1 can use the hardware flow control via GPIOA[5:4] PIN, but there is no available PIN for UART2 hardware flow control.

For example:

- Assign GPIOA[5:4] as UART1 interface
  _da16x_io_pinmux(PIN_CMUX, CMUX_UART1d); // GPIOA_4 : UART1 TXD, and GPIOA_5 : UART1 RXD
- Assign GPIOC[7:6] as UART2 interface
  _da16x_io_pinmux(PIN_UMUX, UMUX_UART2GPIO); // GPIOC_6 : UART2 TXD, GPIOC_7 : UART2 RXD, GPIOC_8(GPIO)

Table 1: PIN MUX configuration - UART

UART interface	GPIO	Signal name
UART1	GPIOA[0] (Note 1)	TXD
	GPIOA[1]	RXD
	GPIOA[2]	TXD
	GPIOA[3]	RXD
	GPIOA[4]	TXD
	GPIOA[5]	RXD
	GPIOA[6]	TXD
	GPIOA[7]	RXD
UART1 H/W flow control	GPIOA[4]	RTS
	GPIOA[5]	CTS
UART2	GPIOA[10]	TXD
	GPIOA[11]	RXD
	GPIOC[6]	TXD
	GPIOC[7]	RXD

Note 1 See Ref. [3] for more information.

3.2 SPI Host Interface

3.2.1 PIN MUX Configuration

GPIOA[1:0], GPIOA[3:2], GPIOA[7:6], GPIOA[9:8] and GPIOA[11:10] can be assigned to SPI slave interface in DA16200.

For example, assign GPIOA[3:2] and GPIOA[9:8] as SPI slave interface.

- _da16x_io_pinmux(PIN_BMUX, BMUX_SPIs); // GPIOA_2 : CS, GPIOA_3 : CLK
- _da16x_io_pinmux(PIN_EMUX, EMUX_SPIs); // GPIOA_8: MISO, GPIOA_9 : MOSI

Table 2: PIN MUX configuration - SPI

GPIO	Signal name
GPIOA[0]	MISO
GPIOA[1]	MOSI

GPIOA[2]	CS
GPIOA[3]	CLK
GPIOA[6]	CS
GPIOA[7]	CLK
GPIOA[8]	MISO
GPIOA[9]	MOSI
GPIOA[10]	MISO
GPIOA[11]	MOSI

3.2.2 SPI Protocol

3.2.2.1 Message Format

The format of the messages sent/received to/from the external processor is the DA16200 protocol format over SPI physical interface. Message format and parameters included in DA16200 are outlined in Figure 1.

Figure 1: Basic format - SPI

3.2.2.1.1 Address

The address list used by External Host is outlined.

Table 3: SPI address list

Address type	Address
General Command (Write Request)	0x50080254
AT Command	0x50080260
Response Command	0x50080258
Buffer Address	Received from slave in response message

3.2.2.1.2 CMD

The format of CMD fields is outlined in Table 4.

Table 4: SPI CMD format

Bit	Field	Description
7	Auto_Inc	1: Internal Address auto-increment 0: Address Fixed (Not used)
6	Read/Write	1: Read 0: Write
5:2	-	Not Used
1:0	CHIP_ID[1:0]	00: CHIP #0 (Default)

3.2.2.1.3 Length

Payload length of the data field.

3.2.2.2 Write Sequence

Host to Slave write operations are performed in three SPI transactions as shown in Figure 2.

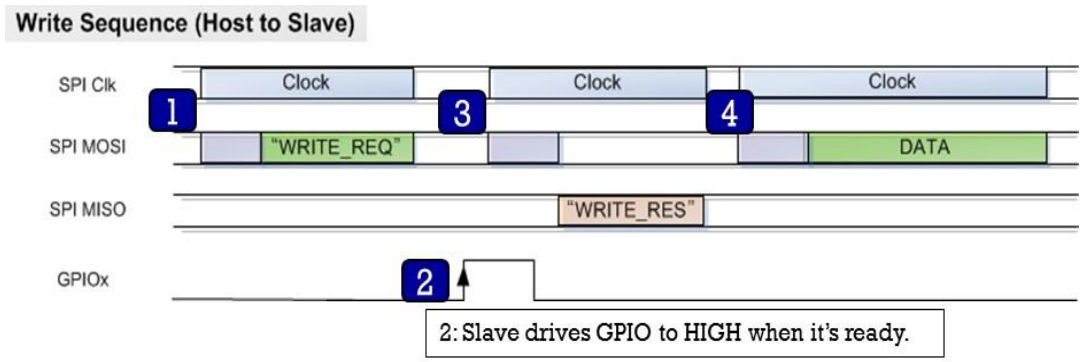


Figure 2: Write sequence

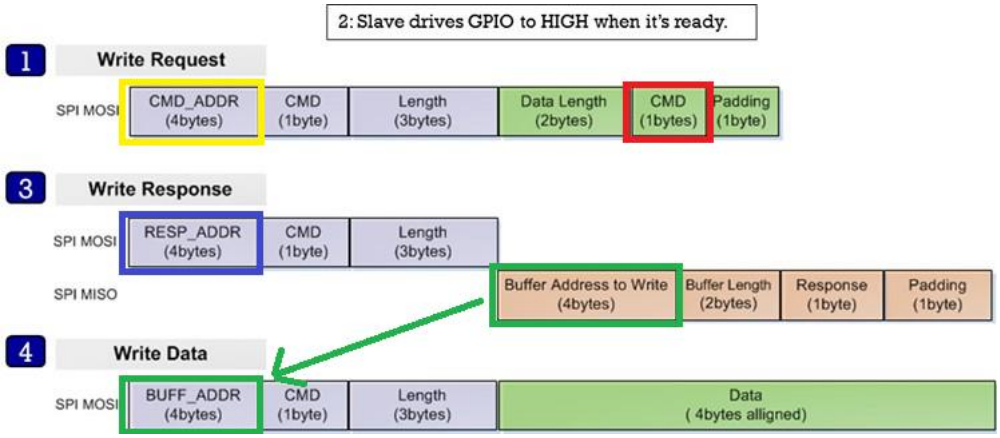


Figure 3: Write operation structure

1. The Host sends a WRITE_REQ command (0x80, red rectangle in Figure 3) to the General Command address (0x50080254) (yellow rectangle in Figure 3).
2. The Host should wait for GPIO interrupt line is High from slave.
3. The Host reads the Write Response message by Response Command address (0x50080258, blue rectangle in Figure 3) and parse it using struct _st_host_response (see Section 3.2.2).
4. The Host sends data to address (BUFF_ADDR) which is received from the Slave in the Write Response message (green rectangle in Figure 3).

NOTE
Buffer Length field contains the length of Data field, and it should be a multiple of 4. The padding field contains number of padded bytes in the Data field due to 4-byte aligned. For example, if the length of the actual data is 11 bytes, the Buffer Length will be 12 (multiples of 4) and Padding field will be 1. Buffer Length (12) = Actual data length (11) + number of padded bytes (1) Host can get the length of actual data using Buffer Length and Padding fields. In addition, the usage of Padding field is applied to the SDK v3.2.8.0 or later.

An interval of several hundred microseconds is required between the “3” and “4” stages. If the interval between the two stages is too short, there is a possibility that two Interrupt Events are recognized as one. The interval depends on the type of application or CPU load. Roughly, when the CPU clock is 120 MHz, an interval of around 300 µs is required.

Example

When the host wants to write 8-byte data (0x8877665544332211) to DA16200:

1. Host sends: (0x50-0x08-0x02-0x54)-(0x80)-(0x00-0x00-0x04)-(0x08-0x00-0x80-0x00)
2. Host waits until GPIO interrupt line is high from DA16200.

3. Host sends (0x50-0x08-0x02-0x58)-(0xC0)-(0x00-0x00-0x08), then reads responses from DA16200.
   Assume the buffer address from Slave is 0x12345678 for easy description.
   Then the read data should be 0x78-0x56-0x34-0x12-0x08-0x00-0x81-0x00.
4. Host sends
   (0x12-0x34-0x56-0x78)-(0x80)-(0x00-0x00-0x08)-(0x11-0x22-0x33-0x44-0x55-0x66-0x77-0x88)
   Note that the payload data is transmitted MSB first and little-big endian system (see Figure 10).

3.2.2.3 Read Sequence and Structure

Figure 4 shows a Slave device transmitting data to the Host when payload is available. This sequence is performed in two SPI transactions.

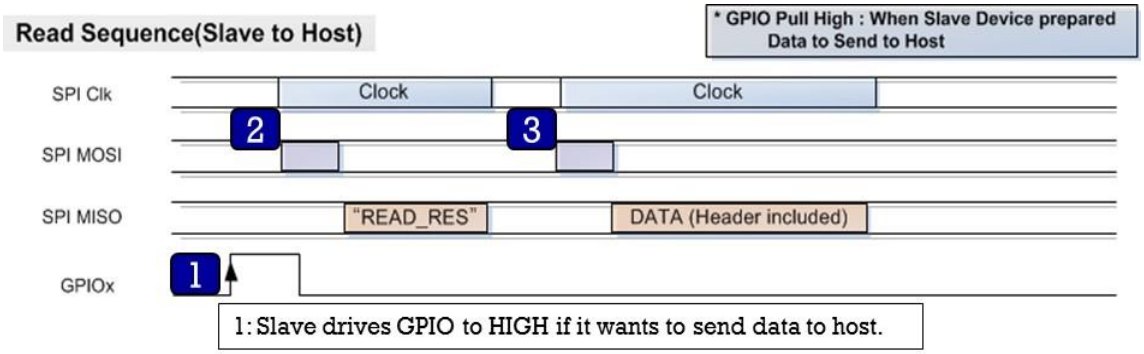


Figure 4: Read sequence

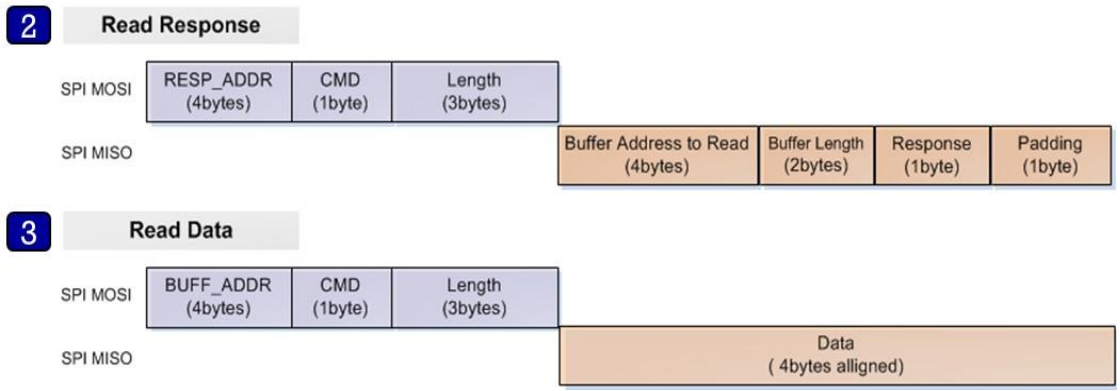


Figure 5: Read operation structure

1. The slave toggles the interrupt line high to inform the Host when data is available.
2. The Host reads the response message from Response Command address (0x50080258, blue rectangle in Figure 5) and parses it using struct _st_host_response (see Section 3.2.6).
3. The Host reads data from address (BUFF_ADDR) which is received from Slave in the response message (green rectangle in Figure 5).

NOTE
Buffer Length field contains the length of Data field, and it should be a multiple of 4. The padding field contains number of padded bytes in the Data field due to 4-byte aligned. For example, if the length of the actual data is 11 bytes, the Buffer Length will be 12 (multiples of 4) and Padding field will be 1. Buffer Length (12) = Actual data length (11) + number of padded bytes (1) Host can get the length of actual data using Buffer Length and Padding fields. In addition, the usage of Padding field is applied to the SDK v3.2.8.0 or later.

There is a 200 ms timeout between reading the response after the interrupt occurs and reading the data after reading the response. If host requires more than 200 ms between each interval, change the timeout value accordingly.

An interval of several hundred microseconds is required between the “2” and “3” stages. If the interval between the two stages is too short, there is a possibility that two Interrupt Events are recognized as one. The interval differs depending on the type of application or CPU load. Roughly, when the CPU clock is 120 MHz, an interval of around 300 µs is required.

Example

1. When the host becomes high on GPIO interrupt line from DA16200, the host sends:

   (0x50-0x08-0x02-0x58)-(0xC0)-(0x00-0x00-0x08), then read response from DA16200.

   Assume the buffer address from Slave is 0x12345678 for easy description and the data length to be sent from DA16200 is 8 bytes.
2. The read data should be 0x78-0x56-0x34-0x12-0x08-0x00-0x83-0x00.
3. Host sends: (0x12-0x34-0x56-0x78)-(0xC0)-(0x00-0x00-0x08), then read data from DA16200.

   Note that the read data is transmitted MSB first and little-big endian system (see Figure 11).

3.2.3 AT Command – Sequences and Structures

AT commands are instructions used to control a modem. AT is the abbreviation of Attention. Every command line starts with AT or at. Start AT is the prefix that informs the modem about the start of a command line. It is not part of the AT command name.
Figure 6 shows how to use the AT Command via SPI on DA16200. This is because AT command uses a predetermined address and the maximum size of data is defined.

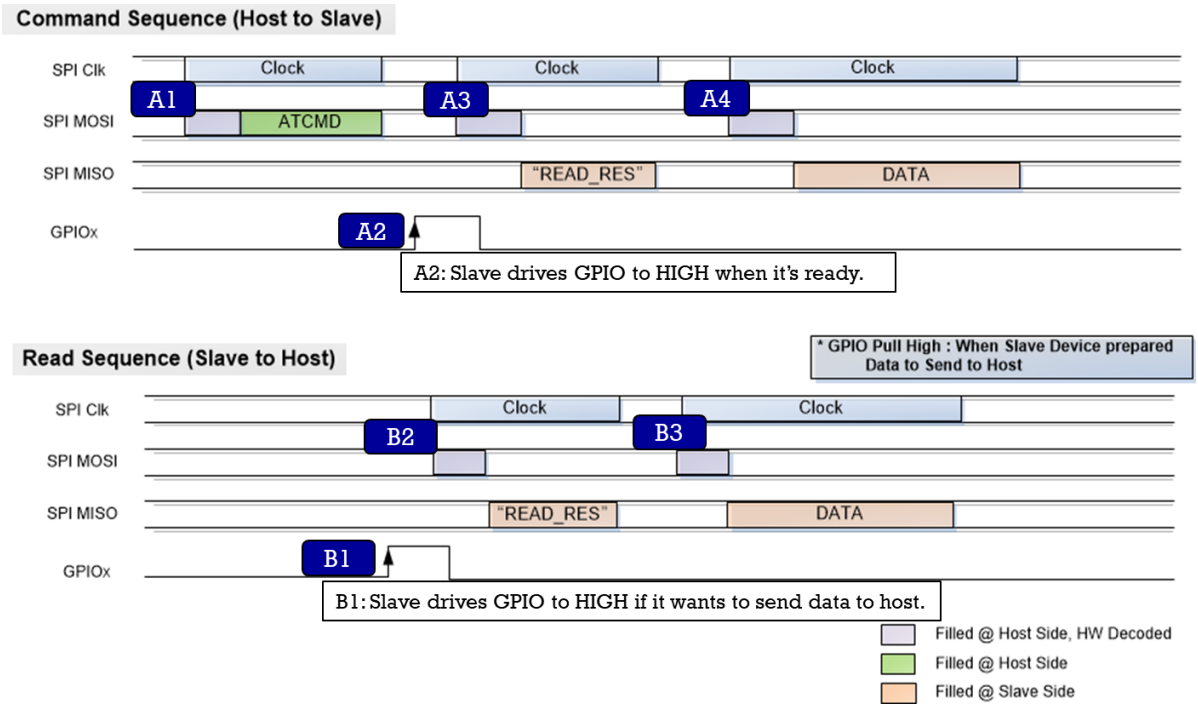


Figure 6: AT command sequence

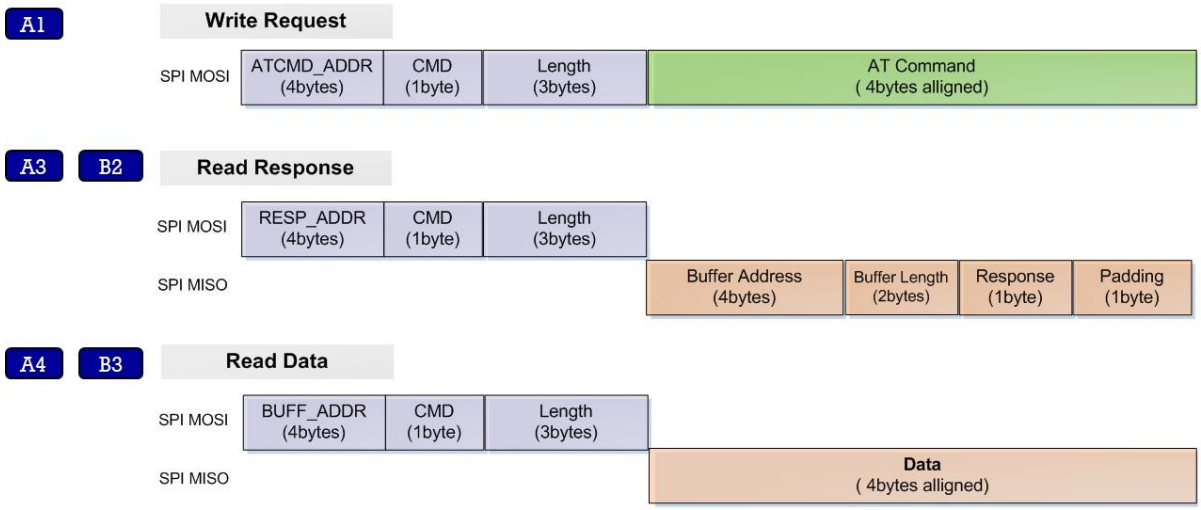


Figure 7: AT command structure

A1: The Host sends an AT command to AT Command address.
A2: The Host waits for GPIO interrupt line to go high.
A3: The Host reads the response message from address and parses it using struct _st_host_response.
A4: The Host reads OK/Error, or data from address (BUF_ADDR), depending on the command type.

Example

- To write AT+VER command to the DA16200, the host sends:
  (0x50-0x08-0x02-0x60)-(0x80)-(0x00-0x00-0x08)-('A'-'T'-'+'-'V'-'E'-'R'-0x00-0x00)
- The read sequence after writing is the same as the example of B1~B3 below.
  Note that the payload data is transmitted MSB first and little-big endian system (see Figure 10).

B1: The Slave toggles high the interrupt line to inform Host when data is available.
B2: The Host reads the response message from Response Command address and parses it using struct _st_host_response.
B3: The Host reads data from address (BUF_ADDR) parsed from the response message.
There is a 200 ms timeout between reading the response after the interrupt occurs and reading the data after reading the response. If host requires more than 200 ms between each interval, change the timeout value accordingly.
An interval of several hundred microseconds is required between the "A3" and "A4" stages, "B2" and "B3" stages. If the interval between the two stages is too short, there is a possibility that two Interrupt Events are recognized as one. The interval differs depending on the type of application or CPU load. Roughly, when the CPU clock is 120 MHz, an interval of around 300 µs is required.

Example

- When the host becomes high on GPIO interrupt line from DA16200, the host sends:
  (0x50-0x08-0x02-0x58)-(0xC0)-(0x00-0x00-0x08), then read response from DA16200.
  Assume the buffer address from Slave is 0x12345678 for easy description and the data length to be sent from DA16200 is 8 bytes.
- The read data should be 0x78-0x56-0x34-0x12-0x08-0x00-0x83-0x00.
- Host sends: (0x12-0x34-0x56-0x78)-(0xC0)-(0x00-0x00-0x08), then read data from DA16200.
  Note that the read data is transmitted MSB first and little-big endian system (see Figure 11).

3.2.4 ESC Command – Sequences and Structures

Figure 8 shows how to use the ESC Command via SPI on DA16200. This is because ESC command uses a predetermined address, and the maximum size of data is defined.

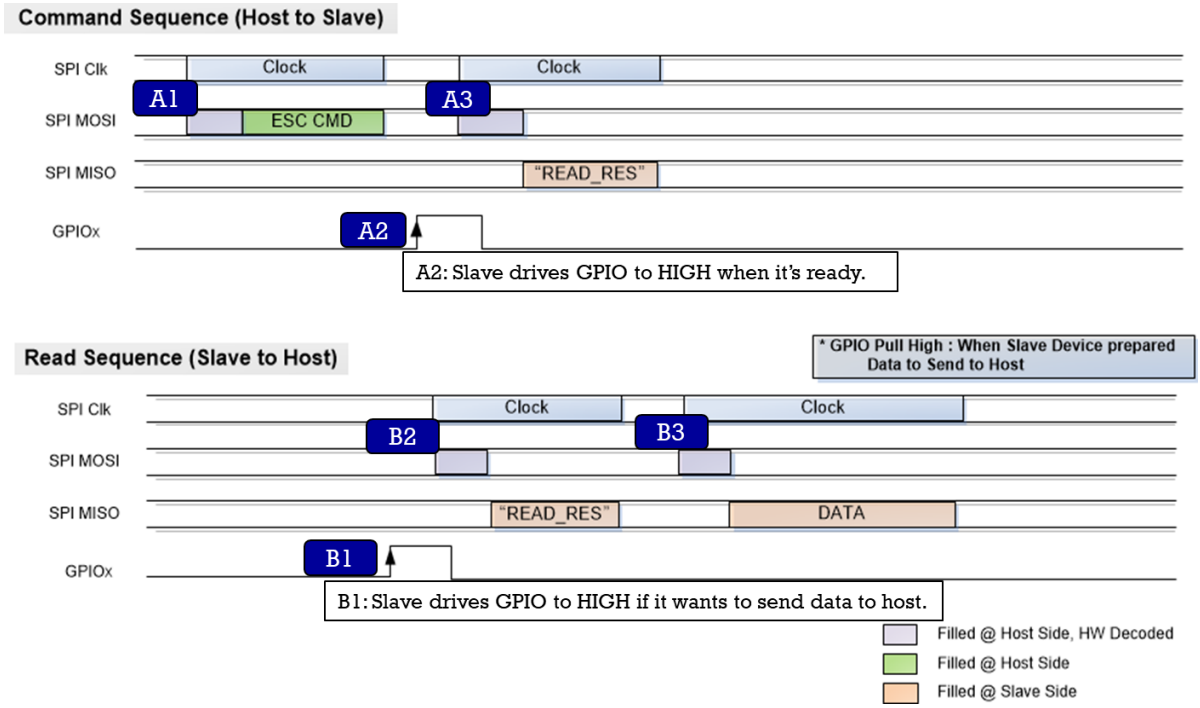


Figure 8: ESC command sequence

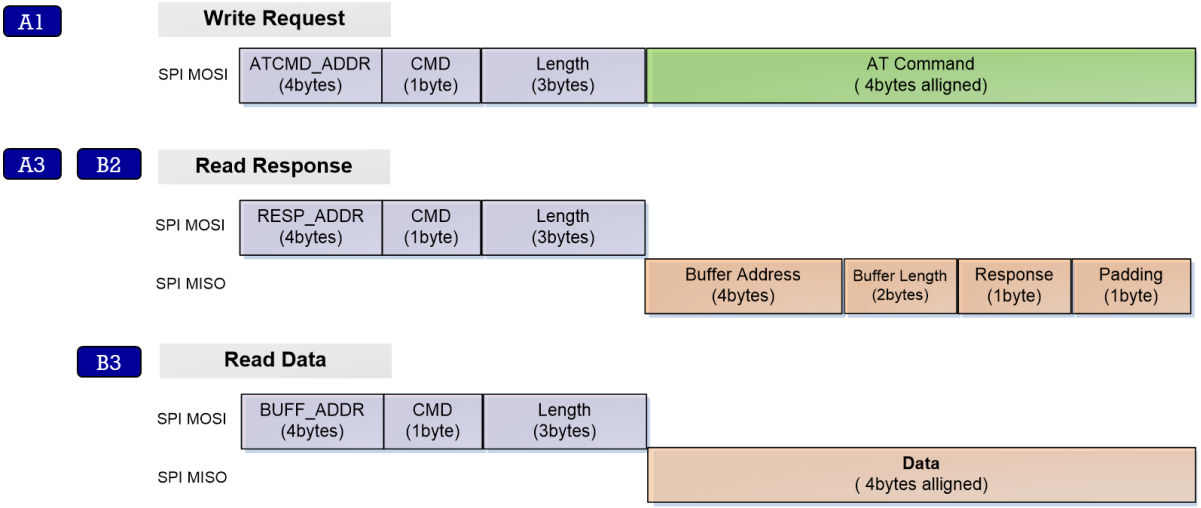


Figure 9: ESC command structure

A1: The Host sends an ESC command to AT Command address.
A2: The Host waits for GPIO interrupt line to go high.
A3: The Host reads the response message from address and parses it using struct _st_host_response. The result for esc command is sent to the host as the response field of struct _st_host_response. The response field is a 1-byte decimal value. A value of 0x20 is a result of OK. All other values are ERROR. And in this case, the value of the buf_address field is read as 0xffffffff, and the value of the host_length field is read as 0x0. Therefore, the subsequent Read Sequence is not required.

Example

- To write <ESC>S010,192.168.0.18,43310,abcde12345 command to the DA16200, the host sends: (0x50-0x08-0x02-0x60)-(0x80)-(0x00-0x00-0x24)-(<ESC>-'S'-'0'-'1'-'0'-','-'1'-'9'-'2'-'.'-'1'-'6'-'8'-'.'-'0'-'.'-'1'-'8'-','-'4'-'3'-'3'-'1'-'0'-','-'a'-'b'-'c'-'d'-'e'-'1'-'2'-'3'-'4'-'5'-0x00)
- The read sequence after writing is the same as the example of B1~B2 below.
  Note that the payload data is transmitted MSB first and little-big endian system (see Figure 10).

B1: The Slave toggles high the interrupt line to inform Host when data is available.
B2: The Host reads the response message from Response Command address and parses it using struct _st_host_response.
B3: The Host reads data from address (BUF_ADDR) parsed from the response message.
There is a 200 ms timeout between reading the response after the interrupt occurs and reading the data after reading the response. If host requires more than 200 ms between each interval, change the timeout value accordingly.
An interval of several hundred microseconds is required between the “B2” and “B3” stages. If the interval between the two stages is too short, there is a possibility that two Interrupt Events are recognized as one. The interval differs depending on the type of application or CPU load. Roughly, when the CPU clock is 120 MHz, an interval of around 300 µs is required.

Example

- When the host becomes high on GPIO interrupt line from DA16200, the host sends: (0x50-0x08-0x02-0x58)-(0xC0)-(0x00-0x00-0x08), then read response from DA16200.
  Assume the buffer address from Slave is 0x12345678 for easy description and the data length to be sent from DA16200 is 8 bytes.
- The read data should be 0x78-0x56-0x34-0x12-0x08-0x00-0x83-0x00.
- Host sends: (0x12-0x34-0x56-0x78)-(0xC0)-(0x00-0x00-0x08), then read data from DA16200.
  Note that the read data is transmitted MSB first and little-big endian system (see Figure 11).

3.2.5 Header Format

3.2.5.1 Write Request (Host to Slave)

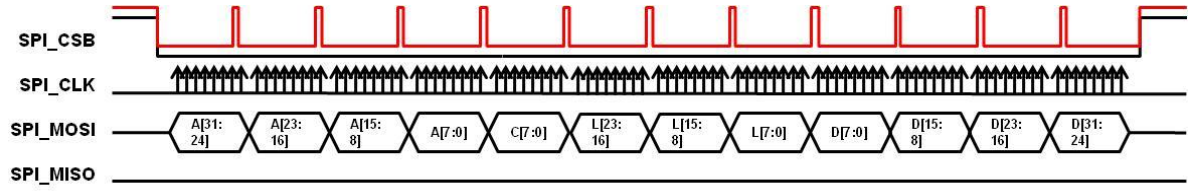
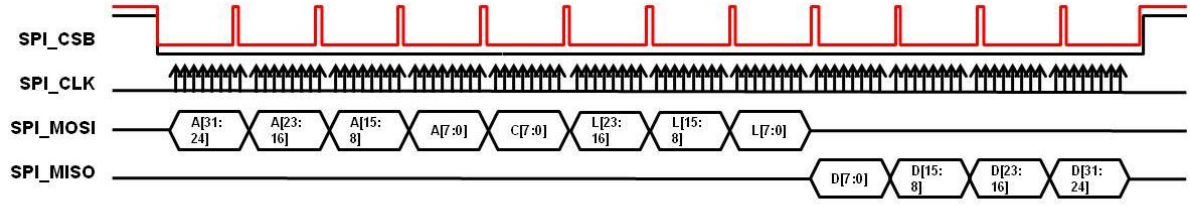


Figure 10: SPI signals for write request

3.2.5.2 Read Response (Slave to Host)



CMD Bit field	Abr.	Description
7	Auto_Inc	1 = internal Address auto-increment 0 = address fixed (not used)
6	Read/Write	1 = Read 0 = Write
5:2		Not used
1:0	CHIP_ID[1:0]	00 = Chip #0 (default)

Figure 11: SPI signals for read response

3.2.6 SPI Definition and Structure

3.2.6.1 SPI Definition

```
#define HOST_MEM_WRITE_REQ      (0x80)
#define HOST_MEM_WRITE_RES      (0x81)
#define HOST_MEM_READ_REQ               (0x82)
#define HOST_MEM_READ_RES               (0x83)
#define FC9K_GEN_CMD_ADDR               (0x50080254)  // Address to Write Command
#define FC9K_RESP_ADDR          (0x50080258)  // Address to Read Response
#define FC9K_ATCMD_ADDR         (0x50080260)  // Address to Send AT Command
```

3.2.6.2 SPI Response Structure

```
typedef struct _st_host_response
{
    u32 buf_address;
    u16 host_length;
    u8  resp;
    u8  dummy;
} st_host_response;
```

3.2.6.3 SPI Request Structure

```
typedef struct _st_host_request
{
    u16 host_write_length;
    u8  host_cmd;
    u8  dummy;
} st_host_request;
```

3.3 SDIO Host Interface

3.3.1 PIN MUX Configuration

SDIO slave is assigned to GPIOA[9:4] in DA16200. For interruption, the D1 port of SDIO can be used, but in some cases, GPIO can also be used.
The following pin mux initialization codes are used for SDIO interface.

```
_da16x_io_pinmux(PIN_CMUX, CMUX_SDs);          // GPIOA_4 : CMD, GIPOA_5 : CLK
_da16x_io_pinmux(PIN_DMUX, DMUX_SDs);          // GPIOA_6 : D3, GPIOA_7 : D2
_da16x_io_pinmux(PIN_EMUX, EMUX_SDs);          // GPIOA_8 : D1, GPIOA_9 : D0
```

If GPIO is used as interruption instead of SDIO D1, the following PAD Mux Setting is additionally required.

```
_da16x_io_pinmux(PIN_FMUX, FMUX_GPIO);         // GPIOA_10 : GPIO, GPIOA_11 : GPIO
```

Table 5: PIN MUX configuration -SDIO

GPIO	Signal name
GPIOA[4]	CMD
GPIOA[5]	CLK
GPIOA[6]	D3
GPIOA[7]	D2
GPIOA[8]	D1
GPIOA[9]	D0

3.3.2 SDIO Protocol

3.3.2.1 Message Format

The format of the messages sent/received to/from the external processor is the DA16200 protocol format over SDIOI physical interface. The format and the parameters included are outlined in Figure 12. For more information about SDIO header configuration, see SDIO Specification. When using SDIO protocol of DA16200, data should be aligned in units of 4-byte length.

Figure 12: Basic format - SDIO

3.3.2.1.1 Address (Included in header)

The address list used by External Host is outlined in Table 6.

Table 6: Address list

Address type	Address
General Command (Write Request)	0x50080254
AT Command	Received from slave in initial stage
Response Command	0x50080258
Buffer Address	Received from slave in response message

3.3.2.1.2 Length (Included in header)

Payload length to follow.

3.3.2.1.3 Interrupt

According to SDIO Specification, Slave can cause Interrupt to Host by using D1 port. However, if the host cannot receive an interrupt using the D1 port, it must use the GPIO to generate the interrupt. The interrupt line used in the sequence diagram of this document means that the D1 port or GPIO is used.

3.3.2.2 Write Sequence

Host to Slave write operations are performed in three SDIO transactions.

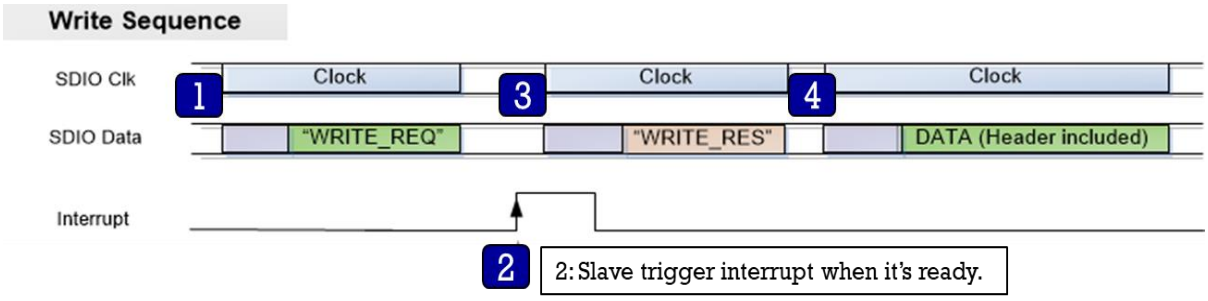


Figure 13: Write sequence

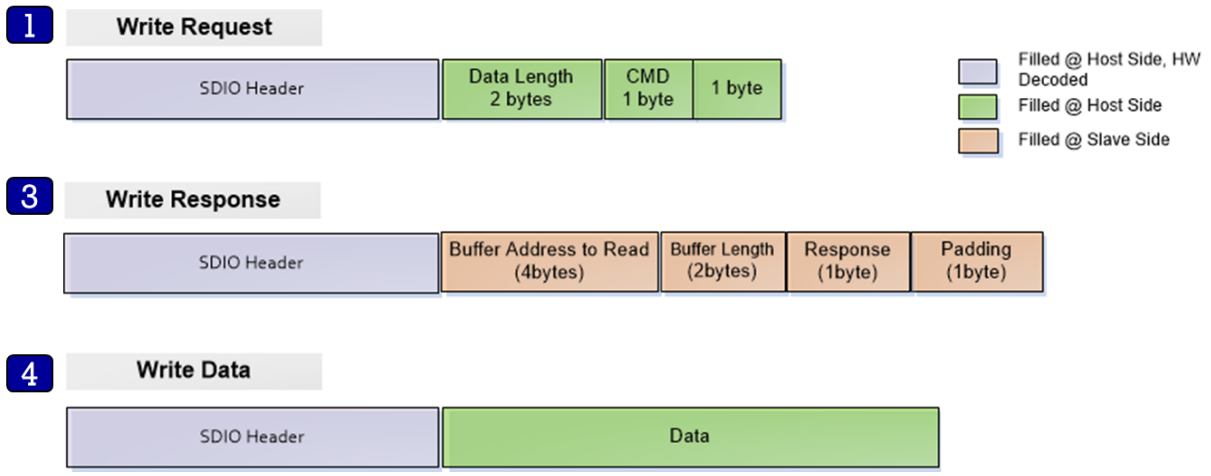


Figure 14: Structure

1. The Host sends a WRITE_REQ command (0x80, red rectangle in Figure 3) to the General Command address (0x50080254).
2. The Host should wait for Interrupt from slave.
3. The Host reads the Write Response message by Response Command address (0x50080258) and parse it using struct _st_host_response (see Section 3.2.6).
4. The Host sends data to address (BUFF_ADDR) which is received from the Slave in the Write Response message (green rectangle in Figure 14).

There is a 200 ms timeout between reading the response after the interrupt occurs and reading the data after reading the response. If the host requires more than 200 ms between each interval, change the timeout value accordingly.
An interval of several hundred microseconds is required between the “3” and “4” stages. If the interval between the two stages is too short, there is a possibility that two Interrupt Events are recognized as one. The interval differs depending on the type of application or CPU load. Roughly, when the CPU clock is 120 MHz, an interval of around 300 µs is required.

Example
When the host wants to write 8-byte data (0x88776655,0x44332211) to DA16200:

- Host sends:
  (SDIO Write-0x50080254, 4 bytes) - (0x08-0x00-0x80-0x00)
- Host waits for interrupt triggering from DA16200.
- Host sends:
  (SDIO Read-0x50080258, 8 bytes) then read response from DA16200.
  Assume the buffer address from Slave is 0x12345678 for easy description.
  Then the read data should be 0x78-0x56-0x34-0x12-0x08-0x00-0x81-0x00.
- Host sends:
  (SDIO Write-0x12345678, 8 bytes)-(0x55-0x66-0x77-0x88-0x11-0x22-0x33-0x44)

3.3.2.3 Read Sequence and Structure

Figure 15 shows a Slave device transmitting data to the Host when payload is available. This sequence is performed in a two SDIO transaction.

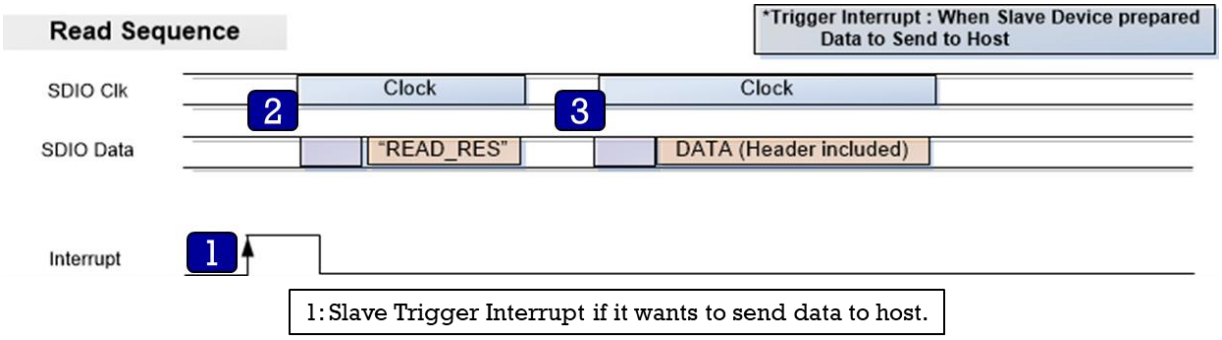


Figure 15: Read sequence

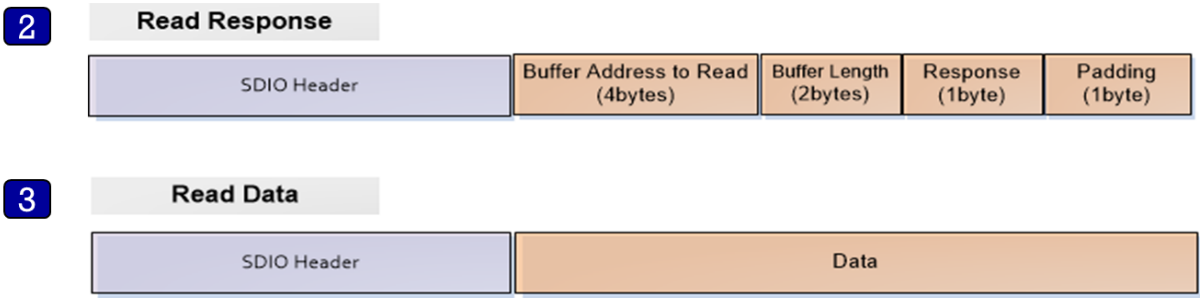


Figure 16: Structure

1. The Slave will trigger interrupts to inform the Host when data is available.
2. The Host reads the response message from Response Command address (0x50080258, blue rectangle in Figure 16), and parse it using struct _st_host_response (see Section 3.2.6).
3. The Host reads data from address (BUFF_ADDR) which is received from Slave in the response message (green rectangle in Figure 16).

There is a 200 ms timeout between reading the response after the interrupt occurs and reading the data after reading the response. If the host requires more than 200 ms between each interval, change the timeout value accordingly.
An interval of several hundred microseconds is required between the “2” and “3” stages. If the interval between the two stages is too short, there is a possibility that two Interrupt Events are recognized as one. The interval differs depending on the type of application or CPU load. Roughly, when the CPU clock is 120 MHz, an interval of around 300 µs is required.

Example

- When the host received interrupt from DA16200,
- Host sends:
  (SDIO Read-0x50080258, 8 bytes) then read response from DA16200.
  Assume the buffer address from Slave is 0x12345678 for easy description and the data length to be sent from DA16200 is 8 bytes. The read data should be 0x78-0x56-0x34-0x12-0x08-0x00-0x83-0x00.
- Host sends:
  (SDIO Read-0x12345678, 8 bytes) then read data from DA16200.

3.3.3 AT Command – Sequences and Structures

AT commands are instructions used to control a modem. AT is the abbreviation of Attention. Every command line starts with AT or at. Note that the starting AT is the prefix that informs the modem about the start of a command line. It is not part of the AT command name.
To use AT commands, read the address of AT (ESC) Command Buffer in the initial stage. Therefore, read the value of address 0x50080264 after SDIO is initialized, and write the command to that address when sending AT command afterwards.
Figure 17 illustrates how to use the AT command through SDIO in DA16200. This is because AT command uses a predetermined address, and the maximum size of data is defined.

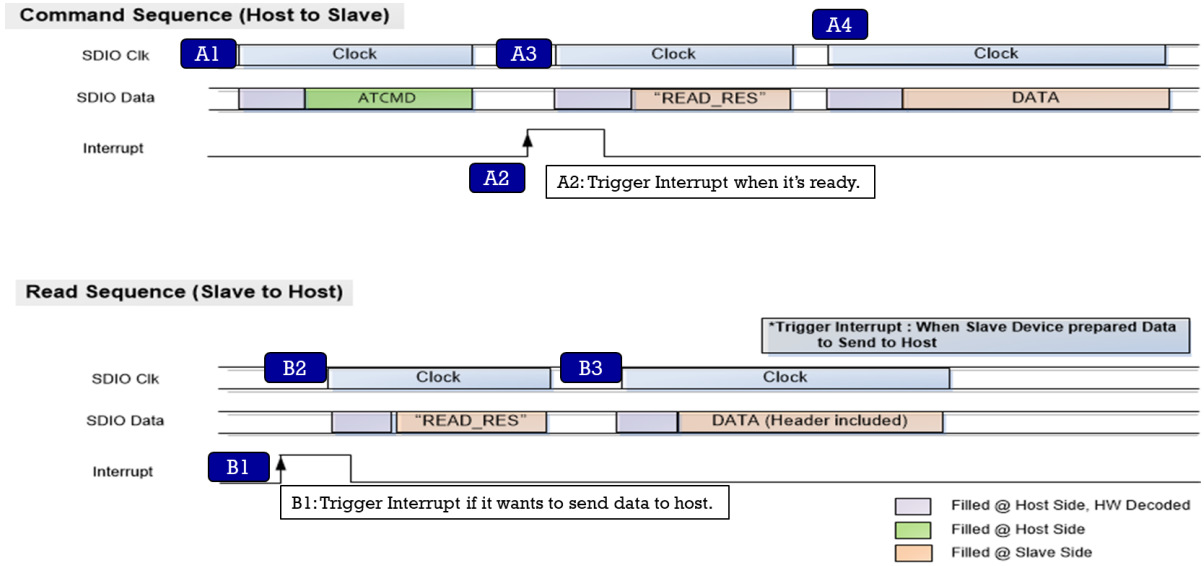


Figure 17: AT command sequence

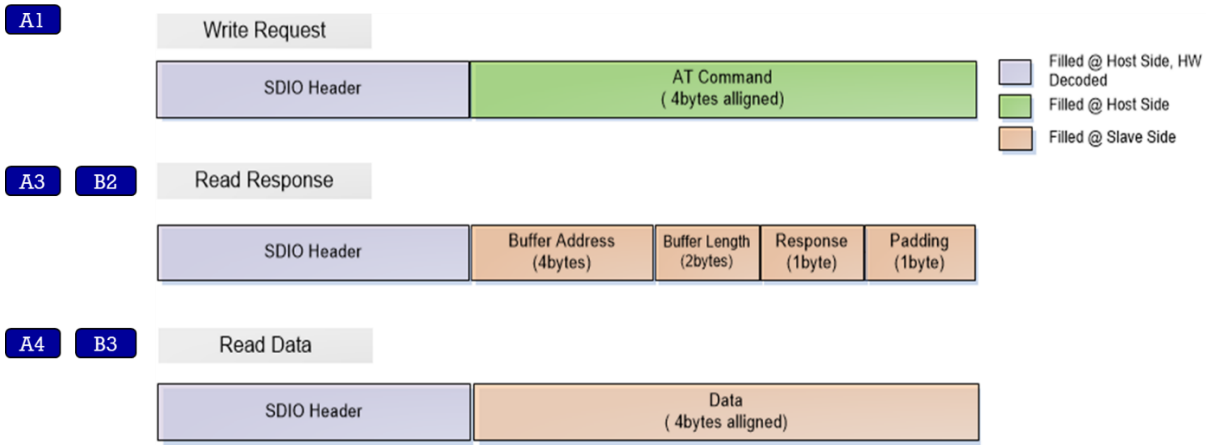


Figure 18: Structure

Descriptions of Figure 17 and Figure 18 are as follows.

A1: The Host sends an AT or ESC command to AT Command address.
A2: The Host waits for interrupt trigger.
A3: The Host reads the response message from address and parses it using.
struct _st_host_response.
A4: The Host reads OK, Error or data from address (BUF_ADDR), depending on the type of command.

B1: The Slave will toggle high the interrupt line to inform Host when data is available.
B2: The Host reads the response message from Response Command address and parses it using struct _st_host_response.
B3: The Host reads data from address (BUF_ADDR) parsed from the response message.
There is a 200 ms timeout between reading the response after the interrupt occurs and reading the data after reading the response. If the host requires more than 200 ms between each interval, change the timeout value accordingly.
An interval of several hundred microseconds is required between the “A3” and “A4” stages, “B2” and “B3” stages. If the interval between the two stages is too short, there is a possibility that two Interrupt Events are recognized as one. The interval differs depending on the type of application or CPU load. Roughly, when the CPU clock is 120 MHz, an interval of around 300 µs is required.

3.3.4 ESC Command – Sequences and Structures

To use ESC commands, read the address of AT (ESC) Command Buffer in the initial stage. Therefore, read the value of address 0x50080264 after SDIO is initialized, and write the command to that address when sending AT command afterwards.

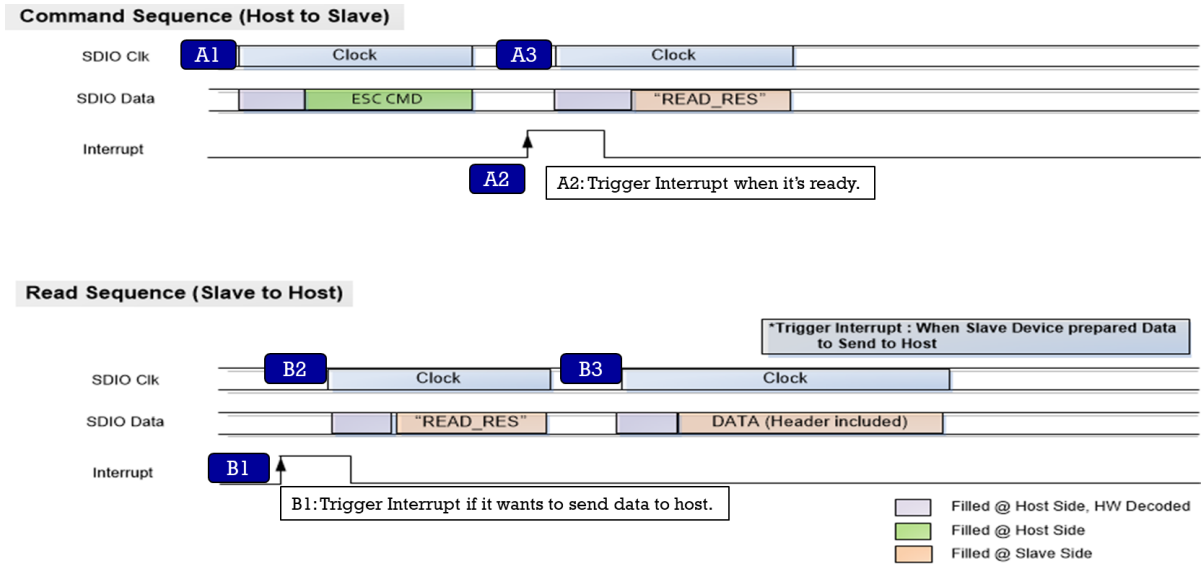


Figure 19: ESC command sequence

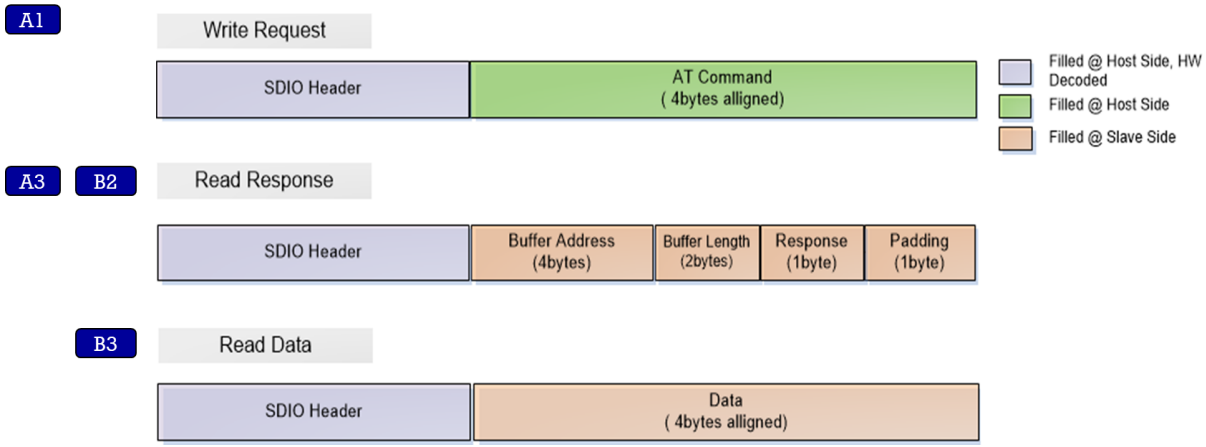


Figure 20: Structure

Descriptions of Figure 19 and Figure 20 are as follows.
A1: The Host sends an ESC command to AT (ESC) Command address.
A2: The Host waits for interrupt trigger.
A3: The Host reads the response message from address and parses it using.
struct _st_host_response. The result for ESC command is sent to the host in the response field of struct _st_host_response. The response field is a 1-byte decimal value. A value of 0x20 is a result of OK. All other values are an ERROR. And in this case, the value of the buf_address field is read as 0xffffffff, and the value of the host_length field is read as 0x0. Therefore, the subsequent Read Sequence is not required.

B1: The Slave will toggle high the interrupt line to inform Host when data is available.
B2: The Host reads the response message from Response Command address and parses it using struct _st_host_response.
B3: The Host reads data from address (BUF_ADDR) parsed from the response message.

There is a 200 ms timeout between reading the response after the interrupt occurs and reading the data after reading the response. If the host requires more than 200 ms between each interval, change the timeout value accordingly.

An interval of several hundred microseconds is required between the “B2” and “B3” stages. If the interval between the two stages is too short, there is a possibility that two Interrupt Events are recognized as one. The interval differs depending on the type of application or CPU load. Roughly, when the CPU clock is 120 MHz, an interval of around 300 µs is required.

3.3.5 SDIO Definition and Structures for Implementation

3.3.5.1 SDIO Definition

```
#define HOST_MEM_WRITE_REQ              (0x80)
#define HOST_MEM_WRITE_RES              (0x81)
#define HOST_MEM_READ_REQ               (0x82)
#define HOST_MEM_READ_RES               (0x83)
#define FC9K_GEN_CMD_ADDR               (0x50080254)  // Address to Write Command
#define FC9K_RESP_ADDR                  (0x50080258)  // Address to Read Response
```

3.3.5.2 SDIO Response Structure

```
typedef struct _st_host_response
{
    u32 buf_address;
    u16 host_length;
    u8  resp;
    u8  dummy;
} st_host_response;
```

3.3.5.3 SDIO Request Structure

```
typedef struct _st_host_request
{
    u16 host_write_length;
    u8  host_cmd;
    u8  dummy;
} st_host_request;
```

4. AT Commands

4.1 Overview

Configuration and control of the DA16200/DA16600 are provided through an ASCII based command string called “AT Command”. AT command is a standard that was originally defined by Hayes Microcomputer for controlling smart modems and is widely used in many products.
AT is an abbreviation of “Attention”, which means to take note of or fix one's sight upon something. An example of an AT command is “ATZ” which instructs a modem to become initialized and return to a state with no command input. An AT command has a very simple structure consisting of a prefix “AT” concatenated with a command string. This is a very convenient method for sending a series of commands over a serial interface such as a UART. Commands may consist of capital letters, lowercase letters, spaces, and some special characters.

4.2 Add AT Command Feature in SDK

This section describes how to include AT command feature in SDK. In SDK, open the file ~/SDK/apps/da16x00/get_started/include/user_main/config_generic_sdk.h using the editor tool and search the string #undef __SUPPORT_ATCMD__.
To enable AT command feature in SDK, change #undef □ #define and save the file. Rebuild the SDK package, and then, newly generated image works as AT command module.

```
//-----------------------
// AT Command Features
//-----------------------

    //
    // Enable/Disable AT command module
    //
    // When enabling this feature, more detailed sub-features are supported.
    // User can check all AT commands in ~/core/system/src/at_cmd/atcmd.c.
    //
    #undef   __SUPPORT_ATCMD__
```

For AT command module, default interface type is UART1 as shown below. If a user wants to use UART2, change #undef __ATCMD_IF_UART2__ to #define __ATCMD_IF_UART2__.

```
  #if defined ( __SUPPORT_ATCMD__ )

    //
    // Default interface of DA16200 EVK is UART1.
    // User can change a type of host-interface among four types listed below.
    //
    #define __ATCMD_IF_UART1__                          // AT command over UART1
    #undef   __ATCMD_IF_UART2__                         // AT command over UART2
    #undef   __ATCMD_IF_SPI__                           // AT command over SPI
    #undef   __ATCMD_IF_SDIO__                          // AT command over SDIO
```

4.2.1 Execute AT Commands on SPI

AT command is configured to use the UART1 interface by default and can be configured to use the SPI interface. To enable the AT commands over SPI interface, modify config_generic_sdk.h as shown below.

```
...
       // AT command service
       #define __SUPPORT_ATCMD__
...
       #if defined ( __SUPPORT_ATCMD__ )
                      #undef   __ATCMD_IF_UART1__              // AT command over UART1
                      #undef __ATCMD_IF_UART2__                // AT command over UART2
                      ...
                      #undef   __USER_UART_CONFIG__    // Support Customer's UART configuration
                      #define __ATCMD_IF_SPI__          // AT command over SPI
                      #undef __ATCMD_IF_SDIO__                 // AT command over SDIO
              #endif  /* __SUPPORT_ATCMD__ */
```

To configure and use the AT commands over SPI interface, see Section 3.2.

4.2.2 Execute AT Commands on SDIO

The AT command can also be configured to use the SDIO interface. To enable the AT commands over SDIO interface, modify config_generic_sdk.h as shown below.

```
...
        // AT command service
        #define __SUPPORT_ATCMD__
...
        #if defined ( __SUPPORT_ATCMD__ )
                        #undef  __ATCMD_IF_UART1__              // AT command over UART1
                        #undef  __ATCMD_IF_UART2__              // AT command over UART2
                        ...
                        #undef  __USER_UART_CONFIG__    // Support Customer's UART configuration
                        #undef  __ATCMD_IF_SPI__                // AT command over SPI
                        #define __ATCMD_IF_SDIO__               // AT command over SDIO
                #endif  /* __SUPPORT_ATCMD__ */
```

To configure the AT commands over SDIO interface, see Section 3.3.

4.2.2.1 Example Sequence for SDIO Interface

An example of the sequence used to initiate a command through the SDIO interface is shown in Figure 21.

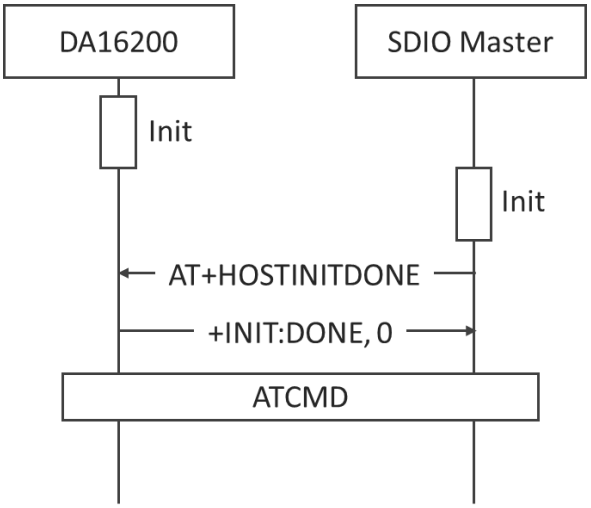


Figure 21: Example sequence to initiate AT command through SDIO interface

When using the SDIO interface, both the DA16200 and the SDIO master devices must be initialized before initiating an AT command. The SDIO master device must send AT+HOSTINITDONE immediately after initialization is completed.

NOTE
For details on how to use the SDK package, see Ref. [2].

4.3 AT Command Format

4.3.1 Basic Command Format

4.3.1.1 Set Command

The set command sets parameters or execute commands.
ATXX
For example:

```
ATZ
OK
```

4.3.1.2 Get Command

The get command queries parameters or get status of the commands.
ATXX=?

For example:
```
ATQ=?
```
Display result on
```
OK
```

4.3.2 Extended Command Format

4.3.2.1 Set Command

The set command sets parameters or execute commands with additional parameters.

AT+XXX=<param1>,<param2>,<param3>,<param4>…<paramN>
For example,
```
AT+NWIP=0,172.16.0.100,255.255.255.0,172.16.0.1
OK
```

If the SSID contains a comma or single quote, the SSID must be enclosed in single quotes.
For example:
```
SSID = MY,SSID'CS
sec = 4
idx = 2
Password = N12345678
```
Is encoded as:
```
AT+WFJAP='MY,SSID'CS',4,2,N12345678'
OK
```

NOTE
The use of a single quote followed by a comma in a parameter is prohibited. For example, AT+WFJAP='MY,SSID',CS',4,2,N12345678 is invalid.

4.3.2.2 Get Command

The get command queries parameters or get status of the commands with additional parameters.
AT+XXX=?
For example:
```
AT+NWIP=?
+ANIP:172.16.0.17,255.255.255.0,172.16.0.1
OK
```

NOTE
Not all commands support the AT+XXX=? query function such as AT+RESTART and ATF. Check the command table for the valid operation of each command.

4.3.3 Response Format

4.3.3.1 Start-up Response

This code is received when DA16200/DA16600 is rebooted.
```
<CR><LF>+INIT:DONE,<mode><CR><LF>
```

The AT command responses when DA16200/DA16600 wakes up from DPM sleep.

4.3.3.2 Basic Response

Basic response gives the command result and is accompanied by a carriage return and a line feed.
```
<CR><LF>+INIT:WAKEUP,<type><CR><LF>
```

4.3.3.3 Normal Response

This code is received for normal operations.
```
<CR><LF>OK<CR><LF>
```

4.3.3.4 Error Response

This code is received when an operation fails for some reason.
```
<CR><LF>ERROR:<error code><CR><LF>
```

4.3.3.5 Extended Response

Extended response gives the command setting values and is followed by a basic response.

<CR><LF>+XXX:[value1],[value2],…

<CR><LF>OK<CR><LF>

NOTE
When an MCU (AT Command Host) waits for a response of a command (for those commands that give extended response as well) to take the next action, it should wait for both normal response (**OK** or **ERROR**) and extended response (also known as **Operation Result**).

Error response codes: See Appendix I.

NOTE
There are major changes in Error response code in SDK v3.2.5.0 and later versions. The examples in this document have been updated based on the changes.

5. AT Command Sets

5.1 Basic Function Commands

Table 7: Basic function command list

Parameter	Description	Conditions
?	(none)	Show AT command usage
	Example ? AT Commands: ? - No example for ? HELP=<command> - Print help message AT - Attention command AT+ - List available commands ATZ - AT command initialize ATF - Delete NVRAM data and certificates, and perform SW reboot ATE - Command echo ATQ - Result Codes On/Off AT+RESTART - System Restart --- Middle omission --- AT+TRSAVE - Save current status of all session === User AT command ====== OK Note: ▪ Enabled by default in the SDK	
help	<cmd_name>	AT command name to query the use of commands
	(none)	Same as the “?” command
	Example HELP AT Commands: ? - No example for ? HELP=<command> - Print help message AT - Attention command AT+	

Parameter	Description	Conditions
	- List available commands ATZ - AT command initialize ATF - Delete NVRAM data and certificates, and perform SW reboot ATE - Command echo ATQ - Result Codes On/Off AT+RESTART - System Restart … --- Middle omission --- OK HELP=ATE ATE - Command echo OK Note: ▪ Enabled by default in the SDK	
AT+	(none)	Show AT command list
	Example AT+ AT AT+ ATZ ATF ATE ATQ AT+RESTART --- Middle omission --- AT+TRSAVE OK Note: ▪ Enabled by default in the SDK	
ATZ	(none)	Initialize AT commands
	Example ATZ Display result on Echo off OK Note: ▪ Enabled by default in the SDK	

Parameter	Description	Conditions
ATF	(none)	DA16200/DA16600, delete nvram data and certificates, and perform SW reboot. Response: "+INIT:DONE,0"
	Example ATF +INIT:DONE,0 Note: ▪ Enabled by default in the SDK ▪ All NVRAM parameters that include Wi-Fi profile (Soft AP or STA) settings are deleted, DUT restarts, and “+INIT:DONE,0” will be received	
ATE	(none)	ECHO on/of
	?	Show Echo status – on/off
	Example ATE Echo on OK ATE AT+NWMQTP Echo off OK ATE=? Echo on OK Note: ▪ Enabled by default in the SDK	
ATQ	(none)	Turn on/off whether to display result code
	?	Show the current status whether to display result codes
	Example ATQ Display results off ATQ=? Display result on OK Note: ▪ Enabled by default in the SDK	
ATB	<baudrate> [[,<databits>] [,<parity>] [,<stopbits>]	Set UART parameters (the main purpose is to change baud rate). <baudrate>: 9600/19200/38400/57600/115200/230400/460800/921600 <databits>: [optional], 5/6/7/8 (Default) <parity>: [optional], n (None, Default)/e (Even)/o (Odd) <stopbits>: [optional], 1 (Default)/2
	?	Show the current baud rate
	Example	

Parameter	Description	Conditions
	ATB=230400 OK Note: ▪ Enabled by default in the SDK ▪ If __USER_UART_CONFIG__ is enabled in SDK (See Appendix C), this command will be disabled	
AT+RESTART	(none)	System restart
	Example AT+RESTART OK +INIT:DONE,0 Note: ▪ Enabled by default in the SDK	
AT+RESET	(none)	System reset. Go to the Boot mode ([MROM] prompt)
	Example AT+RESET OK Note: ▪ Enabled by default in the SDK ▪ Once the system goes into MROM mode, AT command is not available. Therefore, MCU needs to force POR booting or enter 'boot' command via UART0 console	
AT+CHIPNAME	(none)	Get chip name, DA16200 or DA16600
	Example AT+CHIPNAME +CHIPNAME:DA16200 OK Note: ▪ Enabled by default in the SDK	
AT+VER	(none)	Get version information. Response: +VER:<main version>
	Example AT+VER +VER:FRTOS-GEN01-01-xxxxxxxxxx-xxxxxx OK Note: ▪ Enabled by default in the SDK	
AT+SDKVER	(none)	Get the SDK version information. Response: +SDKVER:<major >.<minor >.<revision>.<eng_number> <major>: SDK major number <minor>: SDK minor number <revision>: SDK Revision number <eng_number>: SDK engineering number
	Example	

Parameter	Description	Conditions
	AT+SDKVER +SDKVER:3.2.8.0 OK Note: ▪ Enabled by default in the SDK	
AT+TIME	<date>,<time>	Set the current time. <date>: yyyy-mm-dd <time>: hh:mm:ss Response: OK or ERROR
	?	Get the current time. Response: +TIME:<yyyy-mm-dd> <hh:mm:ss>
	Example AT+TIME=2021-07-15,16:14:30 OK AT+TIME=? +TIME:2021-07-15,16:14:32 OK Note: ▪ Enabled by default in the SDK	
AT+RLT	(none)	Get system running time. Response: +RLT:<days>,<hh:mm.ss>
	Example AT+RLT +RLT:0,01:06.18 OK Note: ▪ Enabled by default in the SDK	
AT+TZONE	<sec>	GMT time zone setting (-43200 ~ 43200). <sec>: Time zone setting parameter Response: OK or ERROR
	?	Get GMT time zone parameter. Response: +TZONE:<sec>
	Example AT+TZONE=? +TZONE:0 OK AT+TZONE=32400 OK AT+TZONE=? +TZONE:32400 OK Note: ▪ Enabled by default in the SDK	

Parameter	Description	Conditions
	▪ The <sec> parameter must be a multiple of 60 seconds. If the value for <sec> is not a multiple of 60 seconds, then the remainder will be discarded	
AT+DEFAP	(none)	All profiles in NVRAM are removed and set up in Soft AP mode with the default configuration. To initialize the Soft AP interface, the system will reboot automatically. Response: OK or ERROR (reboot)
	Example AT+DEFAP OK +INIT:DONE,1 Note: ▪ Enabled by default in the SDK ▪ Default configuration: • SSID: DA16200/DA16600_ XXXXXX (for example, 9FFCF3: the last three hexadecimal values of the board's MAC address) • Authentication: WPA2/CCMP • IP address: 10.0.0.1 • Netmask: 255.255.255.0 • Gateway: 10.0.0.1 • PSK: 12345678 • DHCP server started ○ DHCP range: 10.0.0.2 ~ 10.0.0.11 ○ DHCP DNS: 8.8.8.8 • To query the configuration status, AT+WFSAP and/or AT+NWDHR can be used	
AT+BIDX	<idx>	Set Boot index. <idx>: Boot index (0 or 1) Response: OK or ERROR
	?	Get the current Boot index. Response: +BIDX:<0|1>
	Example AT+BIDX=? +BIDX:0 OK AT+BIDX=1 OK AT+BIDX=? +BIDX:1 OK Note: ▪ Enabled by default in the SDK ▪ System restart is required for changes to take effect. "AT+RESTART" command can be used to restart the system	
AT+DPM	<dpm> [,<nvm_only>]	Set DPM on/off. System restart is required for DPM mode (On/Off) to take effect. <dpm>: 0 (Off), 1 (On) <nvm_only>: 1 (write dpm mode to nvram only, and not reboot),

Parameter	Description	Conditions
		0 or not specified (change dpm mode and reboot) Response: OK or ERROR
	?	Get the current DPM setting. Response: +DPM:<0|1>
	Prerequisite Station mode Example AT+DPM=? +DPM:0 OK AT+DPM=1 ; DPM enabled and system reboots automatically OK +INIT:DONE,0 AT+DPM=1,1 ; DPM enabled without system reboots OK AT+DPM=? +DPM:1 OK Note: ▪ Enabled by default in the SDK ▪ DPM configuration is stored in NVRAM ▪ DA16200/DA16600 is restarted if the “<nvram_only>” parameter is zero or not specified and AT command response is OK • +INIT:DONE,0 message is sent when DA16200/DA16600 boots up • If the usage of the AT command is not valid, then DA16200/DA16600 sends an ERROR message without restarting ▪ If the “nvram_only” parameter is “1”, then restart the system manually using “AT+RESTART” ▪ When DA16200/DA16600 reboots, DA16200/DA16600 tries to connect to the AP if the Wi-Fi connection information is available in the NVRAM • +WFJAP:0,<reason> or +WFJAP:1,'<SSID>',<IP Address> as result of Wi-Fi connection • If Wi-Fi connection fails during boot-up due to some unexpected condition (for example, AP is offline, temporary communication issue with AP, and wrong password is stored), +WFJAP:x may not be sent immediately and takes some time, in which case, wait until +WFJAP:x is received. When a timeout occurs, depending on the application use case, either cancel the connection trial (AT+WFQAP) or retry the connection with the right info (AT+WFJAP/AT+WFJAPA) ▪ If MQTT is configured, DA16200/DA16600 tries to connect to the MQTT broker after a Wi-Fi connection is established. The Operation Result – +NWMQCL:0 or +NWMQCL:1 – is sent over UART1 as a result ▪ DA16200/DA16600 operates DPM if it is set to 1 (TRUE) ○ If Wi-Fi connection is NOT established in DPM mode, DA16200/DA16600 enters the DPM connection retry state ○ DPM connection retry process: ○ While DA16200/DA16600 operates in DPM mode, DA16200/DA16600 executes the process for making connection reestablished if DA16200/DA16600 is in a “disconnected” state with the specified AP for some reason. See the [6] for more details. ○ If the Wi-Fi connection is established but MQTT connection is NOT established (if MQTT is enabled), DA16200/DA16600 tries to connect to the MQTT broker several times and enters DPM Sleep based on MQTT’s abnormal DPM operation	

<table>
<tr><th>Parameter</th><th>Description</th><th>Conditions</th></tr>
<tr><td rowspan="4">AT+DPMKA</td><td><period></td><td>Set DPM keepalive period.
<period>: Keepalive period (millisecond, 0 ~ 600000)
Response: OK or ERROR</td></tr>
<tr><td>?</td><td rowspan="2">Get DPM keepalive period.
Response: +DPMKA=<millisecond></td></tr>
<tr><td>(none)</td></tr>
<tr><td colspan="2">Example
AT+DPMKA
+DPMKA:30000
OK

AT+DPMKA=5000
OK

AT+DPMKA=?
+DPMKA:5000
OK

Note:
▪ Enabled by default in the SDK
▪ The configuration is stored in NVRAM
▪ System restart is required for changes to take effect</td></tr>
<tr><td rowspan="4">AT+DPMTIMWU</td><td><count></td><td>Set DPM TIM wake-up count.
<count>: TIM wake-up count (1 ~ 6000)
Response: OK or ERROR</td></tr>
<tr><td>?</td><td rowspan="2">Get DPM TIM wake-up count.
Response: +DPMTIMWU=<count></td></tr>
<tr><td>(none)</td></tr>
<tr><td colspan="2">Example
AT+DPMTIMWU
+DPMTIMWU:10
OK

AT+DPMTIMWU=20
OK

AT+DPMTIMWU=?
+DPMTIMWU:20
OK

Note:
▪ Enabled by default in the SDK
▪ The configuration is stored in NVRAM
▪ System restart is required for changes to take effect</td></tr>
<tr><td rowspan="4">AT+DPMUSERWU</td><td><time></td><td>Set DPM user wake-up time.
<time>: User wake-up period (millisecond, 0 ~ 86400000)
Response: OK or ERROR</td></tr>
<tr><td>?</td><td rowspan="2">Get DPM user wake-up time.
Response: + DPMUSERWU =<millisecond></td></tr>
<tr><td>(none)</td></tr>
<tr><td colspan="2">Example
AT+DPMUSERWU
+DPMUSERWU:0
OK</td></tr>
</table>

Parameter	Description	Conditions
	AT+DPMUSERWU=300 OK AT+DPMUSERWU=? +DPMUSERWU:300 OK Note: ▪ Enabled by default in the SDK ▪ The configuration is stored in NVRAM ▪ System restart is required for changes to take effect	
AT+CLRDPMSLPEXT	(none)	Set the user application not to enter DPM sleep. Response: OK or ERROR
	Prerequisite DPM enabled Example AT+CLRDPMSLPEXT OK Note: ▪ Enabled by default in the SDK ▪ A host should execute this command within 200 ms after waking up the DA16200/DA16600 through the external wake-up pin, otherwise, DA16200/DA16600 will go into DPM sleep	
AT+SETDPMSLPEXT	(none)	Set the user application ready to enter DPM sleep. Response: OK or ERROR
	Prerequisite DPM enabled Example AT+SETDPMSLPEXT OK Note: ▪ Enabled by default in the SDK ▪ If DA16200/DA16600 is woken up by an external wake-up signal and the “AT+CLRDPMSLPEXT” command is executed, this command should be issued once every job is done. If this command is not run after the job is done, DA16200/DA16600 will not enter DPM sleep	
AT+SETSLEEP2EXT	<period>,<use_retention_memory>	Enter Sleep 2 mode for the period specified. <period>: wake-up timeout, in millisecond. Min. period: 1000 msec. Max. period: 2097151000 (about 24 days) <use_retention_memory>: 1 (retain), 0 (not retain) Response: OK or ERROR
	Example AT+SETSLEEP2EXT=10000,0 OK +INIT:DONE,0 Note: ▪ Enabled by default in the SDK ▪ DA16200/DA16600 can be woken up by RTC_WAKE_UP while in sleep by AT+SETSLEEP2EXT	

<table>
<tr><th>Parameter</th><th>Description</th><th>Conditions</th></tr>
<tr><td></td><td colspan="2">▪ DA16200/DA16600 sends “+INIT:DONE,0” when it wakes up
▪ A value of 0 for the <period> parameter sets the system to wake up only when an RTC_WAKE_UP event occurs
▪ This command should be run in Non-DPM mode only, therefore, if you want to run this command in DPM mode, disable DPM first (AT+DPM=0,1), and run this command. When this command is run in DPM mode enabled, it returns ERROR (-316)
▪ The use of 1 as <use_retention_memory> is obsolete. If you want to use 1 as <use_retention_memory>, use AT+SETSLEEP3EXT command instead</td></tr>
<tr><td rowspan="2">AT+SETSLEEP3EXT</td><td><period></td><td>Enter Sleep 3 mode for the period specified.
<period>: wake-up timeout, in millisecond. Min. period: 1000 msec. Max. period: 2097151000 (about 24 days)</td></tr>
<tr><td colspan="2">Example
AT+SETSLEEP3EXT=10000
OK
+INIT:DONE,0

Note:
▪ Enabled by default in the SDK
▪ Retention memory is ON during Sleep3 mode
▪ DA16200/DA16600 can be woken up by RTC_WAKE_UP while in sleep by AT+SETSLEEP3EXT
▪ DA16200/DA16600 sends “+INIT:DONE,0” or “+INIT:WAKEUP,…” when it wakes up
▪ A value of 0 for the <period> parameter sets the system to wake up only when an RTC_WAKE_UP event occurs
▪ This command can be used in DPM mode or Non-DPM mode</td></tr>
<tr><td rowspan="3">AT+SETSLEEP1EXT</td><td colspan="2">Deprecated</td></tr>
<tr><td><retain_
dpm_memory></td><td>Enter DPM Sleep 2 mode.
<retain_dpm_memory>: 1 (retain), 0 (not retain)
Response: OK or ERROR</td></tr>
<tr><td colspan="2">Example
AT+SETSLEEP1EXT=1
OK
+INIT:DONE,0

Note:
▪ This command is the same as AT+SETSLEEP2EXT with the period set to 0
▪ It recommends to use the AT+SETSLEEP2EXT command instead of this one
▪ Enabled by default in the SDK
▪ DA16200/DA16600 can only be woken up by RTC_WAKE_UP or GPIO which has been assigned as a wake-up source
▪ DA16200/DA16600 sends “+INIT:DONE:0” once it wakes up</td></tr>
<tr><td rowspan="2">AT+GETFASTCONN</td><td>(none)</td><td>Get the Wi-Fi Fast-reconnection mode status value.
Response: +GETFASTCONN:<0|1></td></tr>
<tr><td colspan="2">Example
AT+GETFASTCONN
+GETFASTCONN:0
OK

Note:
▪ Enabled by default in the SDK v3.2.3.0 or later
▪ See Appendix G for “Wi-Fi Fast-reconnect” for the DA16200/DA16600</td></tr>
<tr><td>AT+SETFASTCONN</td><td><flag></td><td>Enable/Disable the Wi-Fi Fast-reconnection mode.
<mode>: 0 (Disable), 1 (Enable)
Response: OK or ERROR</td></tr>
</table>

Parameter	Description	Conditions
	Example AT+SETFASTCONN=1 OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ See Appendix G for “Wi-Fi Fast-reconnect” for the DA16200/DA16600	
AT+MCUWUDONE	(none)	Notify that the MCU wakes up completely. When this command is received, DA16200/DA16600 starts to send messages to the MCU (that is, MCU should send this command immediately after executing “External wakeup”). Response: OK or ERROR
	Example AT+MCUWUDONE OK Note: ▪ Enabled by default in the SDK ▪ When DA16200/DA16600 receives the command, it starts to send messages to the MCU ▪ MCU should send this command immediately when it receives a notification like “+INIT:WAKEUP,UC” ▪ If the “__DPM_TEST_WITHOUT_MCU__” is defined, then MCU does not need to send this command which means it is assumed that MCU is always ready to read a message(response) from DA16200/DA16600	
AT+HOSTINITDONE	(none)	Notify the DA16200 that the MCU has completed initialization (For SDIO interface, the MCU must send this command immediately after initialization.). The DA16200 returns its initialization status as a response. See Table 8. Response: +INIT:DONE,<mode> or +INIT:WAKEUP,<type>)
	Example AT+HOSTINITDONE +INIT:DONE,0 Note: ▪ Enabled by default in the SDK	
AT+DPMABN	<connection_retry_state>	Enable or disable entering sleep mode 3 followed by DPM connection retry process. <connection_retry_state>: 0 (Off), 1 (On) 1: Use the DPM connection retry state. See Ref. [6] for details. 0: Keep trying to scan the AP and make connection established without entering sleep mode 3 Response: OK or ERROR
	?	Get the current DPM connection retry state
	Prerequisite Station mode Example AT+DPMABN=? +DPMABN:0 OK	

Parameter	Description	Conditions
	AT+DPMABN=1 OK Note: ▪ Enabled by default in the SDK v3.2.8.0 or later	
AT+DPMABNWFCCNT	<count>	Set Wi-Fi Connection Retry counts until System enters DPM Abnormal Sleep. <count>: 0 (This feature not used. DPM Abnormal sleep scheme is followed), 1 to 6 (Wi-Fi Connection Retry count) Response: OK or ERROR
	?	Get the current DPM Abnormal Wi-Fi Connection Retry counts set. Response: +DPMABNWFCCNT:<count>
	Example ; If Wi-Fi connection trials are not successful two times in a row, the system goes to DPM Abnormal sleep AT+DPMABNWFCCNT=2 OK AT+DPMABNWFCCNT=? +DPMABNWFCCNT:2 OK Note: ▪ Disabled by default in the SDK ▪ If __WF_CONN_RETRY_CNT_ABN_DPM__ is enabled in the SDK (config_generic_sdk.h), this command will be enabled ▪ The configuration is stored in NVRAM ▪ If the cause of the Wi-Fi connection failure is “Wrong password” input, and if the application wants to cancel the auto-reconnect trial right away, __WIFI_CONN_RETRY_STOP_AT_WK_CONN_FAIL__ should be defined in sys_common_features.h	

Table 8: Initiation response list

Parameter	Description	Conditions
+INIT	DONE,<mode>	DA16200/DA16600 booting is complete: <mode>:0 (STA), 1 (Soft AP) For example: +INIT:DONE,0
	WAKEUP,<type>	DA16200/DA16600 wake-up is complete from DPM SLEEP state. <type> wake-up type ▪ UC: Unicast packet received ▪ NOBCN: No beacon from the connected AP ▪ DEAUTH: Disconnected from the connected AP ▪ EXT: External wakeup ▪ RTC: By a timer registered For example: +INIT:WAKEUP,UC

5.2 Network Function Commands

Table 9: Network function command list

<table>
<tr><th>Parameter</th><th>Description</th><th>Conditions</th></tr>
<tr><td rowspan="4">AT+NWIP</td><td><iface>,<ip_addr>,
<netmask>,<gw></td><td>Set the IP address.

<iface>: WLAN interface. 0 (WLAN0, STA), 1 (WLAN1, Soft AP)
<ip_addr>: IP Address
<netmask>: Subnet mask
<gw>: Gateway
Response: OK or ERROR</td></tr>
<tr><td>?</td><td rowspan="2">Get the IP address of the current WLAN interface.
Response: +NWIP: <iface>,<ip_addr>,<netmask>,<gw></td></tr>
<tr><td>(none)</td></tr>
<tr><td colspan="2">Example
AT+NWIP=0,192.168.0.100,255.255.255.0,192.168.0.1
OK

AT+NWIP
+NWIP:0,192.168.0.100,255.255.255.0,192.168.0.1
OK

At+NWIP=?
+NWIP:0,192.168.0.100,255.255.255.0,192.168.0.1
OK

Note:
▪ Enabled by default in the SDK
▪ In Soft AP mode, after changing IP address, DHCP server pool range should also be updated based on the class of the changed IP address. Use AT+NWDHR to re-define DHCP server pool range after running AT+NWIP
▪ In Soft AP mode, if the IP configuration is changed while the DHCP server is running, then the DHCP server must be restarted using the AT+RESTART or AT+NWDHS=0 > AT+NWDHS=1 command</td></tr>
<tr><td rowspan="4">AT+NWDNS</td><td><dns_ip></td><td>Set the DNS server IP address of STA interface.
<dns_ip>: DNS server IP address
Response: OK or ERROR</td></tr>
<tr><td>?</td><td rowspan="2">Get the DNS server IP address of STA interface.
Response: +NWDNS:<dns_ip></td></tr>
<tr><td>(none)</td></tr>
<tr><td colspan="2">Example
AT+NWDNS=8.8.8.8
OK

AT+NWDNS
+NWDNS:8.8.8.8
OK

Note:
▪ Enabled by default in the SDK
▪ If AT+NWDNS=? is run under DHCP mode, it returns the DNS IP address from DHCP provision data regardless of any DNS IP address set with AT+NWDNS=<dns_ip>. ERROR:-7 (″No result″ or ″Not configured″) can be returned if there is no DHCP provision data existing
▪ If AT+NWDNS=? is run under Static IP mode, it returns the DNS IP address from AT+NWDNS=<dns_ip> that run previously or default one</td></tr>
</table>

Parameter	Description	Conditions
	▪ If AT+NWDNS=<dns_ip> is run under DHCP mode, and the changes to take effect in Static IP mode, it requires a system restart	
AT+NWDNS2	<dns_ip>	Set the 2nd DNS server IP address of STA interface. <dns_ip>: DNS server IP address Response: OK or ERROR
	?	Get the 2nd DNS server IP address of STA interface. Response: +NWDNS2:<dns_ip>
	(none)	
	Example AT+NWDNS2=8.8.8.8 OK AT+NWDNS2 +NWDNS2:8.8.8.8 OK Note: ▪ Enabled by default in the SDK	
AT+NWHOST	<name>	Get the host IP address by name. <name>: Domain name Response: +NWHOST:<ip>
	Example at+nwhost=www. Renesas Electronics-semiconductor.com +NWHOST:54.192.175.64 OK Note: ▪ Enabled by default in the SDK	
AT+NWPING	<iface>,<dst_ip>, <count>	Ping test. <iface>: WLAN interface. 0 (WLAN0), 1 (WLAN1) <dst_ip>: Target IP address <count>: The number of ICMP message transmissions Response: +NWPING:<sent_count>,<recv_count>, <avg_time>,<min_time>,<max_time>
	Example AT+NWPING=0,192.168.0.1,4 +NWPING:4,4,0,0,0 OK Note: ▪ Enabled by default in the SDK	
AT+NWDHC	<dhcpc>	Start/Stop the DHCP client. <dhcpc>: 0 (stop), 1 (start) Response: OK or ERROR
	?	Get the DHCP client status. Response: +NWDHC:<dhcpc>
	(none)	
	Prerequisite DA16200/DA16600 should be connected to AP. Example	

<table>
<tr><th>Parameter</th><th>Description</th><th>Conditions</th></tr>
<tr><td></td><td colspan="2">AT+NWDHC=1
OK

AT+NWDHC
+NWDHC:1
OK

Note:
▪ Enabled by default in the SDK</td></tr>
<tr><td rowspan="4">AT+NWDHCHN</td><td><hostname></td><td>Store the DHCP client host-name.

<hostname> DHCP client host-name
Response: OK or ERROR</td></tr>
<tr><td>?</td><td rowspan="2">Get the DHCP client host-name which is stored by user
Response: +NWDHCHN=<hostname></td></tr>
<tr><td>(none)</td></tr>
<tr><td colspan="2">Example
at+nwdhchn=TEST_DHCP*HOSTNAME
ERROR:-615
at+nwdhchn=TEST-DHCP-HOSTNAME
OK

Note:
▪ Enabled by default in the SDK
▪ The hostname can contain only uppercase letters (A-Z), lowercase letters (a-z), numbers (0-9), and minus sign (-)</td></tr>
<tr><td rowspan="2">AT+NWDHCHNDEL</td><td>(none)</td><td>Delete DHCP client host-name which was stored by user</td></tr>
<tr><td colspan="2">Example
at+nwdhchndel
OK

Note:
▪ Enabled by default in the SDK</td></tr>
<tr><td rowspan="4">AT+NWDHR</td><td><start_ip>,<end_ip></td><td>Set an IP address range of the DHCP server.

<start_ip>: Starting IP address assigned by the DHCP server
<end_ip>: Ending IP address assigned by the DHCP server
Response: OK or ERROR</td></tr>
<tr><td>?</td><td rowspan="2">Get an IP address range of the DHCP server.
Response: +NWDHR:<start_ip>,<end_ip></td></tr>
<tr><td>(none)</td></tr>
<tr><td colspan="2">Prerequisite
Soft AP mode

Example
AT+NWDHR=10.0.0.2,10.0.0.11
OK

AT+NWDHR
+NWDHR:10.0.0.2,10.0.0.11
OK

Note:
▪ Enabled by default in the SDK</td></tr>
</table>

Parameter	Description	Conditions
	▪ DHCP server restart (AT+RESTART or AT+NWDHS=0 > AT+NWDHS=0) is required for changes to take effect	
AT+NWDHLT	<lease_time>	Set an IP lease time (in seconds) of the DHCP server. <lease_time>: IP lease time (from 60 to 86400 seconds) Response: OK or ERROR
	?	Get an IP lease time of the DHCP server. Response: +NWDHLT:<lease_time>
	(none)	
	Prerequisite Soft AP mode Example AT+NWDHLT=1800 OK AT+NWDHLT +NWDHLT:1800 OK Note: ▪ Enabled by default in the SDK ▪ DHCP server restart (AT+RESTART or AT+NWDHS=0 > AT+NWDHS=0) is required for changes to take effect	
AT+NWDHS	<dhcpd>	Start/Stop DHCP server. <dhcpd>: 0 (stop), 1 (start) Response: OK or ERROR
	<dhcpd>, <start_ip>,<end_ip>, <lease_time>	Start the DHCP server with options. <dhcpd>: 1 (start) <start_ip>: Starting IP address for the DHCP client <end_ip>: Ending IP address for the DHCP client <lease_time>: IP lease time (optional, in second, default is 1800) Response: OK or ERROR
	?	Get the DHCP client status. Response: +NWDHS:<dhcpd>
	(none)	
	Prerequisite Soft AP mode Example AT+NWDHS=1 OK AT+NWDHS=1,10.0.0.2,10.0.0.10,1800 OK AT+NWDHS +NWDHS:1,10.0.0.2,10.0.0.10,1800 OK Note: ▪ Enabled by default in the SDK	

<table>
<tr><th>Parameter</th><th>Description</th><th>Conditions</th></tr>
<tr><td>AT+NWDHIP</td><td>(none)</td><td>Show the information of the DHCP Client(s) connected.

Response: OK or ERROR
When the response is OK, the following response comes first before OK.
+NWDHIP:<mac_addr_1>,<ip_addr_1>;<mac_addr_2>,<ip_addr_2>;….</td></tr>
<tr><td></td><td colspan="2">Example
// One DHCP client is in the connected state.
AT+NWDHIP
+NWDHIP:80:35:c1:79:c1:da,10.0.0.2
OK

// Two DHCP clients are in the connected state.
AT+NWDHIP
+NWDHIP:80:35:c1:79:c1:da,10.0.0.2;b4:f1:da:b4:27:11,10.0.0.3
OK

// No DHCP client exists.
AT+NWDHIP
ERROR:-622 // Clients are not connected

Note:
▪ Enabled by default in the SDK
▪ Use this command when DHCP server is running</td></tr>
<tr><td>AT+NWSNS
AT+NWSNS1
AT+NWSNS2</td><td><server_ip></td><td>Set the SNTP server IP address/domain name.

<server_ip>: SNTP server IP address/domain name
Response: OK or ERROR</td></tr>
<tr><td></td><td>?
(none)</td><td>Get the SNTP server IP address.
Response: +NWSNS:<sntp></td></tr>
<tr><td></td><td colspan="2">Example
AT+NWSNS=8.8.8.8
OK

AT+NWSNS
+NWSNS:8.8.8.8
OK

Note:
▪ Enabled by default in the SDK
▪ Up to three SNTP servers can be specified by users; an SNTP server is contacted in round robin manner if DA16200/DA16600 fails to synchronize the system time with a server
If not specified, default SNTP server will be used</td></tr>
<tr><td>AT+NWSNUP</td><td><period></td><td>Set the SNTP client update period (in seconds).

<period>: SNTP client update period (from 60 to 129600 seconds)
Response: OK or ERROR</td></tr>
<tr><td></td><td>?
(none)</td><td>Get the SNTP client update period.
Response: +NWSNUP:<period></td></tr>
<tr><td></td><td colspan="2">Example
AT+NWSNUP=86400
OK</td></tr>
</table>

Parameter	Description	Conditions
	AT+NWSNUP +NWSNUP:86400 OK Note: ▪ Enabled by default in the SDK	
AT+NWSNTP AT+NWSNTP1 AT+NWSNTP2	<sntp>	Start/Stop the SNTP Client. <sntp>: 0 (stop), 1 (start) Response: OK or ERROR
	<sntp>, <server_ip>, <period>	Start the SNTP client with options. <sntp>: 1 (start) <server_ip>: SNTP server IP address (or domain) <period>: SNTP client update period (optional, second, default is 86400) Response: OK or ERROR
	?	Get the SNTP status. Response: +NWSNTP:<sntp>
	(none)	
	Example AT+NWSNTP=0 OK AT+NWSNTP=1,pool.ntp.org,86400 OK AT+NWSNTP +NWSNTP:1,pool.ntp.org,86400 OK Note: ▪ Enabled by default in the SDK ▪ If <sntp> is 1, SNTP client is started immediately and tries to do time sync with the server specified. <sntp>=1 also enables ″auto start″ of the SNTP client if DA16200/DA16600 reboots. <sntp>=0 removes ″auto start″ flag from NVRAM, so DA16200/DA16600 does not try to sync time when rebooted	
AT+NWCCRT	(none)	Check if certificates exist. There are three sets of certificates: ▪ Set #1: for MQTT Root CA (bit 2)/Cert (bit 1)/Key (bit 0)/DH param (bit 9) ▪ Set #2: for HTTPS client for OTA Root CA (bit 5)/Cert (bit 4)/Key (bit 3)/DH param (bit 10) ▪ Set #3: for WPA Enterprise Root CA (bit 8)/Cert (bit 7)/Key (bit 6)/DH param (bit 11) For example: if DA16200/DA16600 has the Root CA and Cert in Set #1, the return value is 6. Response: +VER:<cert>
	Example AT+NWCCRT +NWCCRT:6 ; MQTT OK	

Parameter	Description	Conditions
	AT+NWCCRT +NWCCRT:56 ; HTTPS OK Note: ▪ Enabled by default in the SDK	
AT+NWDCRT	(none)	Delete all TLS certificates including private key. Response: OK or ERROR
	Example AT+NWDCRT OK Note: ▪ Enabled by default in the SDK	

Table 10: Certificate commands

Escape sequence	Parameters	Description
<ESC>C	<cert_id>,<content><ETX>	Store certificate or private key. <ESC>C: To enter certificate input mode, type in <ESC>(0x1B) and C keys together <cert_id>: Certificate ID There are three sets of certificates: ▪ Set #1: for MQTT 0 (Root CA)/1 (Client Certificate)/2 (Private Key) ▪ Set #2: for HTTPS client for OTA 3 (Root CA)/4 (Client Certificate)/5 (Private Key) ▪ Set #3: for WPA Enterprise 6 (Root CA)/7 (Client Certificate)/8 (Private Key) <content>: Certificate data. Copy and paste cert ascii text. Max length is 2048 <ETX>: Indication of the end of content (Ctrl+C, 0x03) Response: OK or ERROR For example: <ESC>C1,----- BEGIN CERTIFICATE -----MIIodknvfano923nf/ ...<ETX>
	Example <ESC>C0,Root CA<ETX> OK <ESC>C1,Client CA<ETX> OK <ESC>C2,Provate Key<ETX> OK Note: ▪ Enabled by default in the SDK	
<ESC>Cert	<module>, <certificate type>, <mode>[, <format>, <length>, <content>]	Store or delete a certificate/CA/private key/DH params. <ESC>CERT: To enter certificate input mode <module>: Module ID. 0 - MQTT, 1 - HTTPs client for OTA, 2 - WPA Enterprise <certificate type>: Certificate type, 0 - CA certificate, 1 - Certificate, 2 - Private key, 3 - DH params <mode>: Input mode. 0 - Store, 1 - Deletion

Escape sequence	Parameters	Description
		<format>: Certificate format, 0 - DER, 1 - PEM if mode is 0 (Store) <length>: Length of certificate if mode is 0 (Store) <content>: Certificate data if mode is 0 (Store) Response: OK or ERROR For example: <ESC>CERT,0,1146,----- BEGIN CERTIFICATE -----MIIDFDCCAf...
	Example <ESC>CERT,0,0,0,1,980,-----BEGIN CERTIFICATE-----… OK <ESC>CERT,0,1,0,1,990,-----BEGIN CERTIFICATE-----… OK <ESC>CERT,0,2,0,1,31 AT+WFPBC 0,-----BEGIN EC PRIVATE KEY-----… OK <ESC>CERT,0,3,0,1,432,-----BEGIN DH PARAMETERS-----… OK Note: ▪ Enabled by default in the SDK	

5.3 Wi-Fi Function Commands

Table 11: Wi-Fi function command list

Command	Parameters	Description
AT+WFMODE	<mode>	Set the Wi-Fi mode. <mode>: 0 (STA), 1 (Soft AP), 2 (Concurrent Mode) Response: OK or ERROR
	?	Get the current Wi-Fi mode. Response: +WFMODE:<mode>
	(none)	
	Example AT+WFMODE=0 ; Set Station mode OK AT+WFMODE=1 ; Set Soft AP mode OK AT+WFMODE ; Get current Wi-Fi mode +WFMODE:1 OK AT+WFMODE=? ; Get current Wi-Fi mode +WFMODE:1 OK Note: ▪ Enabled by default in the SDK ▪ Wi-Fi mode is stored in NVRAM ▪ System restart is required for changes to take effect To use the concurrent mode, complete the following steps: • AT+DEFAP	

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td colspan="2">• AT+WFMODE=2
• AT+RESTART
• AT+WFJAPA=SSID,PASSWORD (or AT+WFJAPA=SSID,X,X,PASSWORD)</td></tr>
<tr><td rowspan="4">AT+WFMAC</td><td><mac></td><td>Write a user MAC address in the NVRAM.
Response: OK or ERROR</td></tr>
<tr><td>?</td><td rowspan="2">Get the current MAC address of the activated WLAN interface.
Response: +WFMAC:<mac></td></tr>
<tr><td>(none)</td></tr>
<tr><td colspan="2">Example
AT+WFMAC=EC:9F:0D:9F:FA:64
OK

AT+WFMAC=?
+WFMAC:EC:9F:0D:9F:FA:64
OK

AT+WFMAC
+WFMAC:EC:9F:0D:9F:FA:64
OK

AT+WFMAC=? ; In Soft AP mode
+WFMAC:EC:9F:0D:9F:FA:65
OK

Note:
▪ The last digit should be an even number to be a valid MAC address
▪ Enabled by default in the SDK
▪ A user MAC address is stored in NVRAM and a system restart is required for changes to take effect
▪ DA16200/DA16600 provides three types of the MAC address and the priority is in the following order: Spoofing MAC address, User MAC address, OTP MAC address
▪ When reading the MAC address in Soft AP mode, it becomes the MAC address that was written + 1</td></tr>
<tr><td rowspan="2">AT+WFSPF</td><td><mac></td><td>Write the spoofing MAC address in the NVRAM
Response: OK or ERROR</td></tr>
<tr><td colspan="2">Example
AT+WFSPF=EC:9F:0D:90:00:48
OK

AT+WFSPF=?
+WFSPF:EC:9F:0D:90:00:48
OK

Note:
▪ Enabled by default in the SDK
▪ Either odd or even number last digit of MAC address is accepted. Use this command only in STA mode
▪ A spoofing MAC address is stored in NVRAM and a system restart is required for changes to take effect
▪ DA16200/DA16600 provides three types of the MAC address and the priority is in the following order: Spoofing MAC address, User MAC address, OTP MAC address
▪ The AT+WFMAC=? command can be used to read back the spoofing MAC address as this command does not support query</td></tr>
<tr><td rowspan="2">AT+WFOTP</td><td><mac></td><td>Write the MAC address in the OTP memory.
Response: OK or ERROR
The MAC address written in the OTP is used as WLAN0 MAC address and MAC address + 1 will be used as WLAN1 MAC address</td></tr>
<tr><td colspan="2">Example
AT+WFOTP=EC:9F:0D:90:00:48</td></tr>
</table>

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td colspan="2">OK

AT+WFMAC=?
+WFOTP:EC:9F:0D:90:00:48
OK

Note:
▪ The last digit should be an even number to be a valid MAC address
▪ Enabled by default in the SDK
▪ An OTP MAC address is stored in OTP and system restart is required for changes to take effect
▪ An old MAC address in the OTP will be invalidated if it exists
▪ There are four MAC address slots available in OTP. It is possible to write the OTP MAC address four times in total at the production
▪ DA16200/DA16600 provides three types of the MAC address and the priority is in the following order: Spoofing MAC address, User MAC address, OTP MAC address
▪ The AT+WFMAC=? command can be used to read back the OTP MAC address as this command does not support query
▪ When reading the MAC address in Soft AP mode, it becomes the MAC address that was written + 1</td></tr>
<tr><td>AT+WFSTAT</td><td>(none)</td><td>Get Wi-Fi configuration.
Response: +WFSTAT:<Wi-Fi interface><var>…</td></tr>
<tr><td></td><td colspan="2">Example
AT+WFSTAT

+WFSTAT:sta0
mac_address= ec:9f:0d:9f:fa:64
wpa_state=DISCONNECTED
disconnect_reason=0

OK

AT+WFSTAT

+WFSTAT:softap1
mac_address=ec:9f:0d:9f:fa:65
wpa_state=DISCONNECTED
disconnect_reason=0

OK

AT+WFSTAT

+WFSTAT:sta0
mac_address=ec:9f:0d:9f:fa:64
bssid=70:5d:cc:32:15:32
ssid=MY_AP_SSID
id=0
mode=STATION
key_mgmt=WPA2-PSK
pairwise_cipher=CCMP
group_cipher=CCMP
channel=3
wpa_state=COMPLETED

OK

Note:
▪ Enabled by default in the SDK
▪ A response can be different depending on the current DA16200/DA16600 status or mode</td></tr>
<tr><td>AT+WFPBC</td><td>(none)</td><td>Run the WPS PBC method.
Response: OK or ERROR</td></tr>
</table>

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td colspan="2">Example
AT+WFPBC

OK
+WFJAP:1,'MY_APS_SSID',192.168.0.3

Note:
▪ Enabled by default in the SDK
▪ A WPS button can be pressed after issuing the command
▪ A router should support WPS and PBC</td></tr>
<tr><td rowspan="3">AT+WFPIN</td><td><pin></td><td rowspan="2">Run the WPS PIN method.

<pin>: PIN (eight digits)
(none): Generate a random PIN
Response: +WFPIN:<pin>
OK or ERROR</td></tr>
<tr><td>(none)</td></tr>
<tr><td>?</td><td>Get the current PIN.
Response: +WFPIN:<pin></td></tr>
<tr><td colspan="2">Example
AT+WFPIN=13557799
+WFPIN:13557799
OK

AT+WFPIN
+WFPIN:36269112 ; Generate random number.
OK

AT+WFPIN=?
+WFPIN:36269112
OK

Note:
▪ Enabled by default in the SDK
▪ An AP should support WPS PIN</td></tr>
<tr><td rowspan="2">AT+WFCWPS</td><td>(none)</td><td>Cancel WPS (both PBC and PIN).
Response: OK or ERROR</td></tr>
<tr><td colspan="2">Prerequisite
WPS should be in progress.

Example
AT+WFCWPS
OK

Note:
▪ Enabled by default in the SDK
▪ Return error if WPS is not in progress</td></tr>
<tr><td rowspan="3">AT+WFCC</td><td><code></td><td>Set a country code.

<code>: Country code (defined by ISO 3166-1 alpha-2 standard) such as KR, US, JP, and CH
Response: OK or ERROR</td></tr>
<tr><td>?</td><td rowspan="2">Get the current country code.
Response: AT+WFCC=<code></td></tr>
<tr><td>(none)</td></tr>
</table>

Command	Parameters	Description
	Example AT+WFCC=KR OK AT+WFCC +WFCC:KR OK AT+WFCC=? +WFCC:KR OK Note: ▪ Enabled by default in the SDK ▪ A country code is stored in the NVRAM ▪ A country code consists of two characters ▪ If a country is invalid, DA16200/DA16600 returns an error code that is –113 ▪ System restart is required for changes to take effect ▪ If this command is run in Soft AP mode with a new country code and the operating channel range of the new country does not cover the operating channel currently set, the operating channel is automatically switched to channel 1	
AT+WFRSSI	(none)	Get the current RSSI value. Response: +RSSI: -34
	Prerequisite DA16200/DA16600 should be connected to AP. Example AT+WFRSSI +RSSI:-25 OK …. (if there is no connection to an AP) AT+WFRSSI +RSSI:NOT_CONN ERROR:-400 Note: ▪ Enabled by default in the SDK ▪ DA16200/DA16600 will respond ″+RSSI:NOT_CONN″ with error (-400) if the connection is not established	
AT+WFSCAN	(none)	Scan Aps. Response: +WFSCAN:<bssid><\t><frequency><\t><signal strength><\t><flag><\t><ssid><LF>…
	Prerequisite The country code should be set via AT+WFCC. Example AT+WFSCAN +WFSCAN:70:5d:cc:32:15:32 2422 -30 [WPA2-PSK-CCMP][WPS][ESS] IPTIME_N604BLACK_ERIC	

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td colspan="2">b4:a9:4f:62:39:46 2422 -32 [WPA-PSK-CCMP+TKIP][WPA2-PSK-CCMP+TKIP][WPS][ESS] SK_WiFiGIGA3943

OK

Note:
▪ Enabled by default in the SDK
▪ An SSID can be missed in case of hidden AP</td></tr>
<tr><td>AT+WFPSCAN</td><td><channel time limit>,
<ch>..<ch></td><td><channel time limit>: Channel scan time limit (should be more than 30000 microsecond)
<ch>: Carrier frequency (from 0 to14)
Response: BSSID Wi-Fi_Channel RSSI SSID Security Type</td></tr>
<tr><td></td><td colspan="2">Prerequisite
The country code should be set via AT+WFCC.
Station mode

Example
AT+WFPSCAN=120000,1,3,5

70:5d:cc:8b:49:8e 2412 -47 Gen_Port_*.5_AP [WPA2-PSK-CCMP][WPS][ESS]
72:5d:cc:c0:9a:c4 2412 -47 IPTIME_A3004NS-M_Bell [WPS][ESS]
…
+PSCAN:TIMEOUT

Note:
▪ Enabled by default in the SDK v3.2.3.0 or later
▪ Multiple parameters can be typed in <ch> as example (′0′ means all channel)</td></tr>
<tr><td>AT+WFPCDTMIN</td><td><bssid>,
<min_threshold></td><td><bssid>: BSSID
<min_threshold>: minimum threshold (from -10 to -100)
Response: OK or ERROR</td></tr>
<tr><td></td><td>?
(none)</td><td>Get the current condition</td></tr>
<tr><td></td><td colspan="2">Prerequisite
Station mode

Example
AT+WFPCDTMIN=72:5d:cc:d0:82:bc,-80
OK

AT+WFPCDTMIN
+WFPCDTMIN: 72:5d:cc:d0:82:bc,-80
OK

Note:
▪ Enabled by default in the SDK v3.2.3.0 or later</td></tr>
<tr><td>AT+WFPCDTMAX</td><td><bssid>,
<max_threshold></td><td><bssid>: BSSID</td></tr>
</table>

Command	Parameters	Description
		<max_threshold>: maximum threshold (from -10 to -100) Response: OK or ERROR
	?	Get the current condition
	(none)	
	Prerequisite Station mode Example AT+WFPCDTMAX=72:5d:cc:c0:82:bc,-20 OK AT+WFPCDTMAX +WFPCDTMAX: 72:5d:cc:c0:82:bc,-20 OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later	
AT+WFPSTOP	(none)	Stop passive scan. Response: OK or ERROR
	Prerequisite Station mode. Example AT+WFPSTOP OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later	
AT+WFJAP	<ssid>,<sec>[,<hidden>] (sec=0|5)	Connect to an AP. <ssid>: AP SSID <sec>: Security protocol. 0 (OPEN), 1 (WEP), 2 (WPA), 3 (WPA2), 4 (WPA+WPA2)), 5 (WPA3 OWE), 6 (WPA3 SAE), 7 (WPA2 RSN & WPA3 SAE) <idx>: Key index for WEP. 0~3 <enc>: Encryption. 0 (TKIP), 1 (AES), 2 (TKIP+AES) <key>: Passphrase. 8 ~ 63 characters are allowed <hidden>: 1 (<ssid> is hidden), 0 or [not specified] (<ssid> is NOT hidden) Response: OK or ERROR Operation Results: +WFJAP:<OPS_RESULT>[,'<SSID>','<IP_ADDRESS>'] +WFJAP:<OPS_RESULT>,<REASON>,[<REASON_CODE>] <OPS_RESULT> : 1 (SUCCESS), 0 (FAILED) If <OPS_RESULT> : 1 <SSID>: The SSID will be surrounded by single quotation mark <IP_ADDRESS>: Assigned IP address and format is xxx.xxx.xxx.xxx If <OPS_RESULT> : 0 <REASON> : well-known reason in text <REASON_CODE> : if < REASON > is OTHER, this shows the reason code
	<ssid>,<sec>, <idx>,<key>[,<hidden>] (sec=1)	
	<ssid>,<sec>, <enc>,<key>[,<hidden>] (sec=2|3|4|6|7)	

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td></td><td>For details about <REASON> or <REASON_CODE>, see Table 12</td></tr>
<tr><td></td><td>?
(none)</td><td>Get the AP provisioning information.

Operation Results:
If provisioning data available:
+WFJAP:’<SSID>’,<sec>,<enc>,’<Passphrase>’
If provisioning data is not available:
ERROR:-410 (No SSID is found)</td></tr>
<tr><td></td><td colspan="2">Example
AT+WFJAP=MY_AP_SSID,0 ; Open security
OK
+WFJAP:1,'MY_AP_SSID',192.168.43.32

AT+WFJAP=MY_AP_SSID,0,1 ; Open security + hidden SSID
OK
+WFJAP:1,'MY_AP_SSID',192.168.43.32

AT+WFJAP=MY_AP_SSID,1,0,12345 ; WEP security
OK
+WFJAP:1,'MY_AP_SSID',192.168.0.7

AT+WFJAP=MY_AP_SSID,1,0,12345,1 ; WEP + hidden AP
OK
+WFJAP:1,'MY_AP_SSID',192.168.0.7

AT+WFJAP=MY_AP_SSID,4,2,N12345678 ; WPA2 security
OK
+WFJAP:1,'MY_AP_SSID',192.168.0.7

AT+WFJAP=MY_AP_SSID,4,2,N12345678,1 ; WPA2 + hidden AP
OK
+WFJAP:1,'MY_AP_SSID',192.168.0.7

AT+WFJAP=?
+WFJAP:'MY_AP_SSID',4,2,'N12345678'
OK

Note:
Enabled by default in the SDK
▪ The host should wait for both command response OK or ERROR and Operation Result ; wait for OK, and +WFJAP:1,'<SSID>',<IP Address> for successful connection
▪ Depending on the network condition, it may take more time to get an Operation Result due to internal connection re-trials
▪ No system reboot happens after running this command
▪ The AP configuration parameters (AP Profile) are stored in NVRAM
▪ WPA3 Personal is enabled by default in the SDK v3.2.5.0 or later, see the examples listed above
▪ If <sec> is set to 6 (WPA3 SAE) or 7 (WPA2 RSN & WPA3 SAE), 1 (AES) is only valid as <enc> because WPA3 SAE allows only CCMP</td></tr>
<tr><td>AT+WFJAPA</td><td><ssid>[,<key>][,<hidden>]</td><td>Connect to an AP.

If <key> exists, security protocol is WPA+WPA2 and encryption is TKIP+AES.</td></tr>
</table>

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td></td><td>if <key> is omitted, security protocol is OPEN.
<hidden>: 1 (<ssid> is hidden), 0 or [not specified] (<ssid> is NOT hidden)
if <hidden> is omitted, <ssid> is not hidden.
<ssid>: AP SSID
<key>: Passphrase. 8 ~ 63 characters are allowed
Response: OK or ERROR

Operation Results:
+WFJAP:<OPS_RESULT>[,'<SSID>','<IP_ADDRESS>']
+WFJAP:<OPS_RESULT>,<REASON>,[<REASON_CODE>]
<OPS_RESULT>: 1 (SUCCESS), 0 (FAILED)
If <OPS_RESULT>: 1
<SSID>: The SSID will be surrounded by single quotation mark
<IP_ADDRESS>: Assigned IP address and format is xxx.xxx.xxx.xxx
If <OPS_RESULT>: 0
<REASON>: Well-known reason in text
<REASON_CODE>: If < REASON > is OTHER, this shows the reason code
For details about <REASON> or <REASON_CODE>, see Table 12.</td></tr>
<tr><td></td><td>?
(none)</td><td>Get the AP provisioning information (SSID and Passphrase only).

Operation Results:
If Wi-Fi connection is success :
+WFJAPA:'<SSID>','<Passphrase>'
If Wi-Fi connection is failed :
ERROR:-425 (No SSID found)</td></tr>
<tr><td></td><td colspan="2">Example
AT+WFJAPA=MY_AP_SSID ; Open security
OK
+WFJAP:1,'MY_AP_SSID',192.168.43.32

AT+WFJAPA=MY_AP_SSID,1 ; Open security + hidden SSID
OK
+WFJAP:1,'MY_AP_SSID',192.168.43.32

AT+WFJAPA=MY_AP_SSID,N12345678 ; WPA2 security
OK
+WFJAP:1,'MY_AP_SSID',192.168.43.32

AT+WFJAPA=MY_AP_SSID,N12345678,1 ; WPA2 security + hidden AP
OK
+WFJAP:1,'MY_AP_SSID',192.168.43.32

AT+WFJAPA=?
+WFJAPA:'MY_AP_SSID','N12345678'
OK

Note:
▪ Enabled by default in the SDK
▪ The host should wait for both command response OK or ERROR and Operation Result ; wait for OK, and +WFJAP:1,'<SSID>',<IP Address> for successful connection</td></tr>
</table>

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td colspan="2">▪ Depending on the network condition, it may take more time to get an Operation Result due to internal connection re-tries
▪ No system reboot is required after running this command
▪ The AP configuration parameters (AP Profile) are stored in NVRAM</td></tr>
<tr><td rowspan="2">AT+WFCAP</td><td>(none)</td><td>Connect to an AP with the current WLAN0 interface configuration.
Response: OK or ERROR</td></tr>
<tr><td colspan="2">Prerequisite
AP profile parameters should exist in NVRAM.

Example
AT+WFCAP
OK
+WFJAP:1,'MY_AP_SSID',192.168.0.7

AT+WFCAP
ERROR:-503 ; Connect to AP fail. (for example, No AP profile found)

AT+WFCAP
ERROR:-460 ; Already connected

Note:
▪ Enabled by default in the SDK
▪ An AP profile can be stored in NVRAM by issuing the ″AT+WFJAPA″ or ″AT+WFJAP″ command
▪ If there is no AP profile found, DA16200/DA16600 returns an error (-503)
▪ If DA16200/DA16600 is already in connection with an AP, it returns an error (-460)</td></tr>
<tr><td rowspan="2">AT+WFQAP</td><td>(none)</td><td>Disconnect from the currently associated AP.
Response: OK or ERROR</td></tr>
<tr><td colspan="2">Prerequisite
DA16200/DA16600 should be connected to AP.

Example
AT+ WFQAP
OK

Note:
▪ Enabled by default in the SDK
▪ No error returns if it has already been disconnected from an AP</td></tr>
<tr><td rowspan="2">AT+WFSTA</td><td>(none)</td><td>Check Wi-Fi connection.
Response: +WFSTA:<status>
<status> 1 (Connected), 0 (disconnected)</td></tr>
<tr><td colspan="2">Prerequisite
Station mode.

Example
AT+WFSTA
+WFSTA:0
OK

AT+WFSTA
+WFSTA:1
OK</td></tr>
</table>

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td colspan="2">Note:
▪ Enabled by default in the SDK
▪ If DA16200/DA16600 runs the command in Soft AP mode, it returns an error (-100)</td></tr>
<tr><td>AT+WFROAP</td><td><roam></td><td>Operate the STA roaming.
<roam>: 1 (run), 0 (stop)
Response: OK or ERROR</td></tr>
<tr><td></td><td>?</td><td rowspan="2">Get the roaming status.
Response: +WFROAP:<roam></td></tr>
<tr><td></td><td>(none)</td></tr>
<tr><td></td><td colspan="2">Prerequisite
Station mode

Example
AT+WFROAP=1
OK

AT+WFROAP=0
OK

AT+WFROAP=?
+WFROAP:1
OK

Note:
▪ Enabled by default in SDK
▪ This command enables ″simple″ roaming. The roaming configuration consists of one parameter called the roaming threshold (set to AT+WFROTH, -65 by). Assume that the DA16200/DA16600 is connected to an AP, and there are other APs that have the same SSID and security settings around the DA16200/DA16600. As the DA16200/DA16600 is not fixed, if the RSSI value of the currently connected AP is lower than the specified threshold, it will try to connect to an AP with a higher RSSI (same SSID and security). If the condition is met, the DA16200/DA16600 will silently switch to the new AP without a disconnection event
▪ The auto roaming start flag is stored in NVRAM when <roam> is set to 1, and roaming operation is enabled if the flag setting is not changed regardless of system reboot. If <roam> is 0, the roaming flag is removed from NVRAM and roaming is disabled
▪ "Simple" roaming is not supported in DPM mode. So, setting DPM mode with command "AT+DPM=1" disables "The auto roaming start flag"</td></tr>
<tr><td>AT+WFROTH</td><td><rssi></td><td>Set the STA roaming threshold.
<rssi>: Roaming threshold value (from 0 to -95 dBm)
Response: OK or ERROR</td></tr>
<tr><td></td><td>?</td><td rowspan="2">Get the STA roaming threshold
Response: +WFROTH:<rssi></td></tr>
<tr><td></td><td>(none)</td></tr>
<tr><td></td><td colspan="2">Prerequisite
Station mode

Example
AT+WFROTH=-55
OK
AT+WFROTH=?

+WFROTH:-55
OK

Note:</td></tr>
</table>

Command	Parameters	Description
	▪ Enabled by default in the SDK ▪ This command writes roaming threshold in NVRAM ▪ When AT+WFROAP=1 is run, the roaming is enabled with the new threshold	
AT+WFDIS	<disabled>	Set the Wi-Fi STA profile unused. If set to 1, DA16200/DA16600 will not start to connect to the configured AP when rebooting. <disabled >: 1 (Unused), 0 (Used) Response: OK or ERROR
	? (none)	Get the status of the Wi-Fi profile. Response: +WFDIS:<disabled>
	Example AT+WFDIS=1 OK AT+WFDIS=? +WFDIS:1 OK Note: ▪ Enabled by default in the SDK ▪ The ″unused″ flag is stored in the NVRAM ▪ The flag affects DA16200/DA16600 during boot-up procedure. System restart is required for changes to take effect	
AT+WFSAP	<ssid>,<sec>, <ch>,<code> (sec=0|5)	Set up Soft AP interface. <ssid>: AP SSID. Max 32 characters are allowed <sec>: Security protocol. 0 (OPEN), 2 (WPA), 3 (WPA2), 4 (WPA+WPA2)), 5 (WPA3 OWE), 6 (WPA3 SAE), 7 (WPA2 RSN & WPA3 SAE) <enc>: Encryption. 0 (TKIP), 1 (AES), 2 (TKIP+AES) <key>: Passphrase. 8 ~ 63 characters are allowed <ch>: Operating channel (optional). Default is 1 or uses the current channel if Soft AP is operating <code>: Country code (optional). If exists, <ch> is essential Response: OK or ERROR
	? (none)	Get the Soft AP interface configuration. Response: +WFSAP:'<ssid>',<auth>,<enc>,'<key>',<ch>,<code> Operation Result: +WFSAP:<ssid> is printed on success
	Example AT+WFSAP=DA16200_MY_SSID,0,1,KR ; Open-Mode +WFSAP:DA16200_MY_SSID OK AT+WFSAP=? +WFSAP:'DA16200_MY_SSID',0,1,KR OK AT+WFSAP=DA16200_MY_SSID,3,1,12345678,1,KR ; WPA2-AES +WFSAP:DA16200_MY_SSID OK	

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td colspan="2">AT+WFSAP=?
+WFSAP:'DA16200_MY_SSID',3,1,'12345678',1,KR
OK

AT+WFSAP='DA16200,MY_SSID',3,2,'12345678',1,KR ; WPA2-AES
+WFSAP:DA16200,MY_SSID
OK

AT+WFSAP=?

+WFSAP:'DA16200,MY_SSID',3,1,'12345678',1,KR
OK

Note:
▪ Enabled by default in the SDK
▪ The Soft AP configuration parameters are stored in NVRAM
▪ If the command is issued in station mode, a reboot is required to start as Soft AP mode. (If the command is issued in Soft AP mode, then no system restart is required)
▪ The ',' (comma) is included in the SSID string and enclose the SSID with a single quotation mark
▪ WPA3 Personal is enabled by default in the SDK v3.2.5.0 or later. See the example
▪ If <sec> is set to 6 (WPA3 SAE) or 7 (WPA2 RSN & WPA3 SAE), 1 (AES) is only valid as <enc> because WPA3 SAE allows only CCMP</td></tr>
<tr><td rowspan="2">AT+WFOAP</td><td>(none)</td><td>Operate Soft AP interface.
Response: OK or ERROR</td></tr>
<tr><td colspan="2">Prerequisite
A Soft AP profile should be stored in NVRAM.

Example
AT+WFOAP
OK

Note:
▪ Enabled by default in the SDK
▪ Run this command in Soft AP mode
▪ If there is no profile in NVRAM, it returns an error (-522)
▪ DA16200/DA16600 returns an error (-522) if it already operates as Soft AP</td></tr>
<tr><td rowspan="2">AT+WFTAP</td><td>(none)</td><td>Stop the Soft AP interface.
Response: OK or ERROR</td></tr>
<tr><td colspan="2">Prerequisite
Soft AP mode

Example
AT+WFTAP
OK

Note:
▪ Enabled by default in the SDK
▪ This command is valid while DA16200/DA16600 is running in Soft AP mode</td></tr>
<tr><td colspan="3">Additional note for AT+WFSAP, AT+WFOAP, AT+WFTAP:

Example:</td></tr>
</table>

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td colspan="3">In STA mode after running ATF command
…
AT+WFSAP=DA16200_OPEN,0 // set up Soft AP
AT+RESTART // reboot to start in the configured Soft AP mode
…
DUT starts as Soft AP
….
AT+WFTAP // stop Soft AP if required
AT+WFOAP // start Soft AP if required</td></tr>
<tr><td>AT+WFRAP</td><td>(none)</td><td>Restart the Soft AP interface.
Response: OK or ERROR</td></tr>
<tr><td></td><td colspan="2">Prerequisite
A profile for Soft AP should be stored in NVRAM.

Example
AT+WFRAP

OK

Note:
▪ Enabled by default in the SDK
▪ This command is valid in Soft AP mode
▪ If it runs in station mode, DA16200/DA16600 returns an error (-100)</td></tr>
<tr><td>AT+WFLCST</td><td>(none)</td><td>Get connected station information.
Response: +WFLCST:<mac><LF><flags><LF><var>…</td></tr>
<tr><td></td><td colspan="2">Example
AT+WFLCST

+WFLCST:a6:f2:7c:d4:53:1c
flags=[AUTH][ASSOC][AUTHORIZED][SHORT_PREAMBLE][WMM][MAYBE_WPS][HT]
aid=1
capability=0x421
listen_interval=10
wifi_mode=802.11n
timeout_next=NULLFUNC POLL
rx_packets=290
tx_packets=4
rx_bytes=29625
tx_bytes=10658
inact_cnt=0
connected_time=20
sta_count=1

OK

AT+WFLCST
+WFLCST:NOT_FOUND
OK

Note:
▪ Enabled by default in the SDK
▪ If there is no station connected, then DA16200/DA16600 returns ″+WFLCST:NOT_FOUND″</td></tr>
</table>

Command	Parameters	Description
AT+WFAPWM	<mode>	Set IEEE 802.11 Wi-Fi mode of Soft AP interface. <mode>: 0 (B/G/N), 1 (G/N), 2 (B/G), 3 (N), 4 (G), 5 (B) Response: OK or ERROR
	?	Get IEEE 802.11 Wi-Fi mode of Soft AP interface. Response: +WFAPWM:<mode>
	(none)	
	Example AT+WFAPWM=0 OK AT+WFAPWM=1 OK AT+WFAPWM=? +WFAPWM:1 OK Note: ▪ Enabled by default in the SDK ▪ The configuration is stored in NVRAM ▪ System restart is required for changes to take effect	
AT+WFAPCH	<ch>	Set the operating channel number for the Soft AP interface. <ch>: Operating channel (from 0 to 13, 0 is auto) Response: OK or ERROR
	?	Get the operating channel number for the Soft AP interface. Response: +WFAPCH:<ch>
	(none)	
	Example AT+WFAPCH=5 OK AT+WFAPCH=? +WFAPCH:5 OK Note: ▪ Enabled by default in the SDK ▪ The configuration is stored in NVRAM ▪ System restart is required for changes to take effect	
AT+WFAPBI	<interval>	Set the AP beacon interval. <interval>: Beacon interval (ms) Response: OK or ERROR
	?	Get the AP beacon interval. Response: +WFAPBI:<interval>
	(none)	
	Example AT+WFAPBI=200 OK AT+WFAPBI=? +WFAPBI:200 OK Note:	

Command	Parameters	Description
	▪ Enabled by default in the SDK ▪ The configuration is stored in NVRAM ▪ System restart is required for changes to take effect	
AT+WFAPUI	<timeout>	Set station disconnection timeout in Soft AP mode. <timeout>: Disconnection timeout (sec) (from 30 to 86400, step = 10) If 0, the default value (300 seconds) is used Response: OK or ERROR
	?	Get station disconnection timeout in Soft AP mode.
	(none)	Response: +WFAPUI:<timeout>
	Example AT+WFAPUI=60UOK AT+WFAPUI=? +WFAPUI:60 OK Note: ▪ Enabled by default in the SDK ▪ Within the specified time, if an STA does not send any frame, Soft AP sends a NULL frame after the timeout is expired to check STA′s inactivity. If no ACK is received from the STA, Soft AP removes the STA ▪ The configuration is stored in NVRAM ▪ System restart is required for changes to take effect	
AT+WFAPRT	<threshold>	Set the AP RTS threshold (octets). <threshold>: RTS threshold (from 1 to 2347) Response: OK or ERROR
	?	Get the AP RTS threshold.
	(none)	Response: +WFAPRT:<threshold>
	Example AT+WFAPRT=2100 OK AT+WFAPRT=? +WFAPRT:2100 OK Note: ▪ Enabled by default in the SDK ▪ If a frame that is bigger than the RTS threshold specified is to be sent, RTS/CTS frames are sent first to avoid collision in the air. By default, the RTS threshold is 2347 ▪ The configuration is stored in NVRAM ▪ System restart is required for changes to take effect	
AT+WFAPDE	<mac>	Send de-authentication frame to the connected station. <mac>: MAC address of the connected station Response: OK or ERROR
	Prerequisite DA16200/DA16600 should be connected to AP. Example AT+WFAPDE=E6:0D:E5:A5:5D:B3 +WFDST:e6:0d:e5:a5:5d:b3	

Command	Parameters	Description
	OK Note: ▪ Enabled by default in the SDK ▪ Use this command in Soft AP mode ▪ Check the MAC address of an STA that needs to send de-authentication frame by using the command “AT+WFLCST” ▪ If the operation is not successful (for example, a wrong MAC address is specified), the operation result (+WFDST:<mac_addr>) does not come	
AT+WFAPDI	<mac>	Send disassociation frame to the connected station. <mac>: MAC address of the connected station Response: OK or ERROR
	Prerequisite DA16200/DA16600 should be connected to AP. Example AT+WFAPDI=E6:0D:E5:A5:5D:B3 +WFDST:e6:0d:e5:a5:5d:b3 OK Note: ▪ Enabled by default in the SDK ▪ Use this command in Soft AP mode ▪ Check the MAC address of an STA that needs to send disassociation frame by using the command “AT+WFLCST” ▪ If the operation is not successful (for example, a wrong MAC address is specified), no operation result (+WFDST:<mac_addr>) are sent	
AT+WFWMM	<wmm>	Set WMM on/off. <wmm>: 0 (off), 1 (on) Response: OK or ERROR
	?	Get the WMM status. Response: +WFWMM:<wmm>
	(none)	
	Prerequisite Soft AP mode Example AT+WFWMM=1 OK AT+WFWMM=? +WFWMM:1 OK Note: ▪ Enabled by default in the SDK ▪ WMM is enabled by default. If WMM is enabled, Beacon/Probe Rsp/Assoc frames will have WMM information. WMM enables QoS on the AC category ▪ The configuration is stored in NVRAM	
AT+WFWMP	<wmmps>	Set WMM-PS (WMM Power Save) on/off. <wmmps>: 0 (off), 1 (on) Response: OK or ERROR
	?	Get the WMM-PS status

Command	Parameters	Description
		Response: +WFWMP:<wmmps>
	Prerequisite Soft AP mode Example AT+WFWMM=0 OK AT+WFWMM=? +WFWMM:0 OK Note: ▪ Enabled by default in the SDK ▪ By default, WMM-PS is disabled. If WMM-PS is enabled, Beacon/Probe Rsp/Assoc Rsp frames sent from Soft AP will have a U-APSD flag set. For WMM and WMM-PS to properly work, the STA should also have WMM and WMM-PS certified ▪ The configuration is stored in NVRAM	

Table 12: Wi-Fi function response list

Response	Parameters	Description
+WFJAP	<result>,<ssid>,<ip> <result>, <well-known-reason> [,<reason_code>]	The result of AP connection in STA mode (The result of AT+WFJAP or AT+WFJAPA or AT+WFCAP). <result>: 0 (failed), 1 (succeeded) For <result>: 1 <ssid>: SSID of the AP when succeeded <ip>: IP address of the station when succeeded For <result>: 0 <well-known-reason>: connection trial failure reason in text format, TIMEOUT / WRONGPWD / ACCESSLIMIT / OTHER TIMEOUT: connection attempt failed after continuous connection attempts WRONGPWD: WPA 4-Way Handshake failed, Pre-shared key (password) may be incorrect ACCESSLIMIT: disconnected because the authorized access number limit has been reached OTHER: other reasons <reason_code>: if <well-known-reason> is OTHER, this field shows which reason caused the connection trial failure See Appendix E For example: +WFJAP:0,TIMEOUT +WFJAP:1,'ap_test',192.168.0.10 // The Wi-Fi connection is established, and the assigned IP address is 192.168.0.10.
+WFDAP	<reserved>, <well-known-reason> [,<reason_code>]	Disconnected from the AP. <reserved>: 0 <well-known-reason>: disconnection reason in text format, AUTH_NOT_VALID / DEAUTH / INACTIVITY / APBUSY / OTHER AUTH_NOT_VALID: Previous authentication no longer valid DEAUTH: De-authenticated as STA is leaving INACTIVITY: Disassociated due to inactivity

Response	Parameters	Description
		APBUSY: Disassociated because AP is unable to handle all currently associated STAs <reason_code> : If <well-known-reason> is OTHER, this field shows which reason caused the disconnection See Appendix E For example: +WFDAP:0,INACTIVITY +WFDAP:0,DEAUTH +WFDAP:0,OTHER,8 ◊ WLAN_REASON_CLASS2_FRAME_FROM_NONAUTH_STA (8)
+WFCST	<mac>	A Wi-Fi station connected in Soft AP mode. <mac>: MAC address of the connected station
+WFDST	<mac>	A Wi-Fi station disconnected in Soft AP mode. <mac>: MAC address of the disconnected station

5.4 Wi-Fi Function Commands for WPA3

Customers and users can configure DA16200/DA16600 as WPA3 STA or WPA3 Soft AP. WPA3 Personal is enabled by default in the SDK v3.2.5.0 or later. For older SDKs, WPA3 is not enabled by default, contact Renesas Electronics. Syntax of all the Wi-Fi function commands is the same as described in Table 13 apart from the following commands where it needs to specify WPA3 specific parameters.

Table 13: WPA3-related Wi-Fi function command list

Command	Parameters	Description
AT+WFJAP	See AT+WFJAP	-
	Example AT+WFJAP=MY_AP_SSID,5 ; WPA3 OWE OK +WFJAP:1,'MY_WPA3_AP_SSID',192.168.0.7 AT+WFJAP=MY_AP_SSID,5,1 ; WPA3 OWE + hidden SSID OK +WFJAP:1,'MY_AP_SSID',192.168.43.32 AT+WFJAP=MY_AP_SSID,6,1,N12345678 ; WPA3 SAE security OK +WFJAP:1,'MY_WPA3_AP_SSID',192.168.0.7 AT+WFJAP=MY_AP_SSID,6,1,12345678,1 ; WPA3 SAE + hidden AP OK +WFJAP:1,'MY_AP_SSID',192.168.0.7 AT+WFJAP=MY_AP_SSID,7,1,N12345678 ; WPA2(RSN)+WPA3 SAE security OK +WFJAP:1,'MY_WPA3_AP_SSID',192.168.0.7 AT+WFJAP=? +WFJAP:'MY_AP_SSID',6,1,'N12345678' OK	
AT+WFJAPA3	<wpa3_flag>,<ssid> [,<key>][,<hidden>]	Connect to an AP which includes WPA3 type. <wpa3_flag>: WPA2/WPA3 AP-type If 1, WPA3-AP. If 0, WPA/WPA2-AP.

Command	Parameters	Description
		If <key> exists, security protocol is WPA+WPA2 and encryption is TKIP+AES. If <key> is omitted, security protocol is OPEN. <hidden>: 1 (<ssid> is hidden), 0 or [not specified] (<ssid> is NOT hidden) If <hidden> is omitted, <ssid> is not hidden. <ssid>: AP SSID <key>: Passphrase. 8 ~ 63 characters are allowed Response: OK or ERROR Operation Results: +WFJAP:<OPS_RESULT>[,'<SSID>','<IP_ADDRESS>'] +WFJAP:<OPS_RESULT>,<REASON>,[<REASON_CODE>] <OPS_RESULT> : 1 (SUCCESS), 0 (FAILED) If <OPS_RESULT> : 1 <SSID>: The SSID will be surrounded by single quotation mark <IP_ADDRESS>: Assigned IP address and format is xxx.xxx.xxx.xxx If <OPS_RESULT> : 0 <REASON> : well-known reason in text <REASON_CODE> : if < REASON > is OTHER, this shows the reason code. For an explanation of <REASON> or <REASON_CODE>, see Table 12
	? (none)	Get the AP profile information (SSID and Passphrase only). Operation Results: If Wi-Fi connection is success : +WFJAP:'<SSID>','<Passphrase>' If Wi-Fi connection is failed: ERROR:-425 (No SSID found)
	Example AT+WFJAPA3=0,WPA2_AP_SSID ; Open-Mode OK +WFJAP:1,'WPA2_AP_SSID',192.168.0.2 AT+WFJAPA3=0,WPA2_AP_SSID,1 ; Open-Mode + hidden SSID OK +WFJAP:1,'WPA2_AP_SSID',192.168.0.2 AT+WFJAPA3=0,WPA2_AP_SSID,N12345678 ; WPA2 security OK +WFJAP:1,'WPA2_AP_SSID',192.168.0.2 AT+WFJAPA3=0,WPA2_AP_SSID,N12345678,1 ; WPA2 security + hidden AP OK +WFJAP:1,'WPA2_AP_SSID',192.168.0.2 AT+WFJAPA3=1,WPA3_AP_SSID ; WPA3-OWE OK +WFJAP:1,'WPA3_AP_SSID',192.168.0.2 AT+WFJAPA3=1,WPA3_AP_SSID,N12345678 ; WPA3-SAE security OK	

Command	Parameters	Description
	+WFJAP:1,'WPA3_AP_SSID',192.168.0.2 AT+WFJAPA3=1,WPA3_AP_SSID,N12345678,1 ; WPA3-SAE + hidden AP OK +WFJAP:1,'WPA3_AP_SSID',192.168.0.2 AT+WFJAPA3=? +WFJAPA3:'MY_AP_SSID','N12345678' OK Note: ▪ Enabled by default in the SDK ▪ The host should wait for both command response OK or ERROR and Operation Result ; wait for OK, and +WFJAPA:1,'<SSID>',<IP Address> for successful connection ▪ Depending on the network condition, it may take more time to get an Operation Result due to internal connection re-trials ▪ No system reboot happens after running this command ▪ The AP configuration parameters (AP Profile) are stored in NVRAM ▪ WPA3 Personal is enabled by default in the SDK v3.2.5.0 or later	
AT+WFSAP	See AT+WFSAP	-
	Example AT+WFSAP=DA16200_MY_SSID,5,1,KR ; WPA3 OWE +WFSAP:DA16200_MY_SSID OK AT+WFSAP=DA16200_MY_SSID,7,1,12345678,1,KR ; WPA2 RSN & WPA3 SAE, AES +WFSAP:DA16200_MY_SSID OK AT+WFSAP=? +WFSAP:'DA16200_MY_SSID',7,1,'12345678',1,KR OK	

5.5 Wi-Fi Function Commands for WPA Enterprise

AT commands of DA16200 provide Wi-Fi commands that can be used as STA in WPA-Enterprise environment. To connect to the WPA-Enterprise AP, the DA16200 need to have profile information for the WPA-Enterprise AP and user account information.

Table 14: WPA-enterprise Wi-Fi function commands

Command	Parameters	Description
AT+WFENTAP	<ssid>,<auth>,<enc>,<phase1>,<phase2>[,<hidden>] (phase1=0|1|2|4)	Create Enterprise profile to NVRAM. <ssid>: Enterprise AP SSID <auth>: Authentication mode for WAP-Enterprise. 8 (WPA-EAP), 9 (WPA2-EAP), 10 (WPA/WPA2-EAP). <enc>: Encryption Type. 0 (TKIP), 1 (AES), 2 (TKIP+AES) <phase1>: Phase #1 EAP type. 0 (Mixed), 1 (PEAP0), 2 (PEAP1), 3 (FAST), 4 (TTLS), 5 (TLS) <pahse2>: Phase #2 EAP type. 0 (Mixed), 1 (MSCHAPV2), 2 (GTC) <hidden>: 1 (<ssid> is hidden), 0 or [not specified] (<ssid> is NOT hidden) Response: OK or ERROR Operation Results: +WFENTAP:<SSID>
	<ssid>,<auth>,<enc>,<phase1>[,<hidden>] (phase1=3|5)	

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td rowspan="3"></td><td>?
(none)</td><td>Get the WPA-Enterprise configuration.
Response:
+WFENTAP:<ssid>,<auth>,<enc>,<phase1>,<pahse2></td></tr>
<tr><td colspan="2">Example
AT+WFENTAP=MY_AP_SSID,10,2,0 ; Phase#1 Mixed mode
OK
+WFENTAP:'MY_AP_SSID'

AT+WFENTAP=MY_AP_SSID,10,2,0,0 ; Phase#1, #2 Mixed mode
OK
+WFENTAP:'MY_AP_SSID'

AT+WFENTAP=MY_AP_SSID,10,2,0,0,1 ; Phase#1, #2 Mixed mode
OK + hidden AP
+WFENTAP:'MY_AP_SSID'

Note:
▪ Enabled by default in the SDK v3.2.3.0 or later
▪ WPA-Enterprise profile is stored in NVRAM
▪ “EAP-FAST” type for <phase_1> is not supported in SDK 3.2.3.0 and earlier SDK
▪ If <phase1> is set to 3 (FAST) or 5 (TLS), <phase2> is not allowed
▪ Need user account to connect to WPA-Enterprise AP. See “AT+WFENTLI” command
▪ System restart is required for changes to take effect</td></tr>
<tr><td rowspan="3">AT+WFENTLI</td><td><id>[,<pw>]</td><td><id>: Login-ID for WPA-Enterprise user account
<pw>: Login-Password for WPA-Enterprise user account

Response: OK or ERROR</td></tr>
<tr><td>?
(none)</td><td>Get current saved User-ID / Password for WPA-enterprise user account.

Response: OK or ERROR
Operation Result: +WFENTLI:<id>,<pwd></td></tr>
<tr><td colspan="2">Example
AT+WFENTLI='USER_ACCOUNT_ID','USER_ACCOUNT_PWD'
OK

AT+WFENTLI
+WFENTLI='USER_ACCOUNT_ID','USER_ACCOUNT_PWD'

AT+WFENTLI=?
+WFENTLI='USER_ACCOUNT_ID','USER_ACCOUNT_PWD'

Note:
▪ Enabled by default in the SDK
▪ User account ID and PASSWORD are stored in NVRAM
▪ System restart is required for changes to take effect</td></tr>
</table>

Table 15: WPA-enterprise network function command

Command	Parameters	Description
AT+NWTLSV	<ver>	Set the minimum accepted TLS protocol version when Phase#1 EAP type is TTLS or TLS of WPA Enterprise. The maximum accepted TLS protocol version is TLSv1.2. For example, If TLSv1.0 is setup, the version of TLS session can be TLSv1.0, TLSv1.1 or TLSv1.2. <ver>: Enterprise AP SSID. 0 <TLSv1.0>, 1 <TLSv1.1>, 2 <TLSv1.2> Response: OK or ERROR
	? (none)	Get current saved TLS version. Response: +NWTLSV:<tls_version>
	Example AT+NWTLSV=NEW_TLS_VER OK AT+NWTLSV +NWTLSV:2 OK AT+NWTLSV=? +NWTLSV:2 OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ TLS Version number is stored as internal value ▪ Default minimum accepted TLS protocol version is TLSv1.2 ▪ System restart is required for changes to take effect	

5.5.1 WPA-Enterprise Connection Example

Create WPA-Enterprise profile and restart the DA16200/DA16600 to start Wi-Fi connection. For all cases, WPA-Enterprise user account information is needed.

```
Case #1, "Mixed" mode for EAP-type.
In this case, Encryption-type is configured as "Mixed" mdoe.
EAP-type and Encryption type are selected automatically.
AT+WFENTAP='WPA-Ent-AP-SSID',10,2,0
AT+WFENTLI='WPA-Ent_User_ID','WPA-Ent_PWD'
AT+RESTART

Case #2, "Mixed" mode for EAP-type and Encryption type.
EAP-type and Encryption type are selected automatically.
AT+WFENTAP='WPA-Ent-AP-SSID',10,2,0,0
AT+WFENTLI='WPA-Ent_User_ID','WPA-Ent_PWD'
AT+RESTART

Case #3, in case of PEAP0 and MSCHAPV2 for WPA-Enterprise.
AT+WFENTAP='WPA-Ent-AP-SSID',10,2,1,1
AT+WFENTLI='WPA-Ent_User_ID','WPA-Ent_PWD'
AT+RESTART

Case #4, "Mixed" mode for EAP-type and set with TLS v1.0.
In this case, Encryption-type is configured as "Mixed" mdoe.
EAP-type and Encryption type are selected automatically.
AT+NWTLSV=1
AT+WFENTAP='WPA-Ent-AP-SSID',10,2,0
AT+WFENTLI='WPA-Ent_User_ID','WPA-Ent_PWD'
AT+RESTART
```

5.6 Advanced Function Commands

5.6.1 MQTT Commands

The commands in Table 16 are for configuring MQTT Client parameters. Restart the MQTT client for the configuration to take effect after running the commands.

NOTE
In DPM mode, stop the MQTT Client (AT+NWMQCL=0) first before running the configuration commands. If any configuration commands are running in DPM mode without stopping MQTT Client, it returns ERROR:-635.

Table 16: MQTT configuration command list

Command	Parameters	Description
AT+NWMQBR	<host_name>,<port>	Set the host name (or IP address) and the port number of the MQTT Broker. <host_name>: Broker's domain name, or IP address <port>: Broker's port number Response: OK or ERROR
	? (none)	Get the host name or IP address and the port number of the MQTT Broker. Response: +NWMQBR:<host_name>,<port>
	Prerequisite MQTT client should be disabled (+NWMQCL:0). Example AT+NWMQBR=192.168.0.65,1884 OK AT+NWMQBR=? +NWMQBR:192.168.0.65,1884 OK Note: ▪ Enabled by default in the SDK ▪ The broker host name (or IP address) and port configured are stored in the NVRAM ▪ MQTT restart is required for the new configuration to take effect	
AT+NWMQQOS	<qos>	Set the MQTT QoS level. <qos>: 0 (at most once), 1 (at least once), 2 (exactly once) Response: OK or ERROR
	? (none)	Get the MQTT QoS level. Response: +NWMQQOS:<qos>
	Prerequisite MQTT client should be disabled (+NWMQCL:0). Example AT+NWMQQOS=1 OK AT+NWMQQOS +NWMQQOS:1 OK Note: ▪ Enabled by default in the SDK ▪ MQTT restart is required for the new configuration to take effect	

Command	Parameters	Description
AT+NWMQTLS	<tls>	Enable/disable the MQTT TLS function. <tls>: 1 (enable), 0 (disable) Response: OK or ERROR
	?	Get MQTT TLS status. Response: +NWMQTLS:<tls>
	(none)	
	Prerequisite Certificate should be stored. See Table 10. MQTT client should be disabled (+NWMQCL:0). Example AT+NWMQTLS=1 OK AT+NWMQTLS +NWMQQOS:1 OK Note: ▪ Enabled by default in the SDK ▪ MQTT restart is required for the new configuration to take effect	
AT+NWMQCS	<clean_session>	Set clean session mode. <clean_session>: 1(session cleared), 0(session retained) Response: OK or ERROR
	?	Get clean session status. Response: +NWMQCS:<clean_session>
	(none)	
	Prerequisite MQTT client should be disabled (+NWMQCL:0). Example AT+NWMQCS=1 OK AT+NWMQCS=? +NWMQCS:1 OK Note: ▪ Enabled by default in SDK v3.2.3.0 ▪ To disable/enable this feature, use __MQTT_CLEAN_SESSION_MODE_SUPPORT__ in config_generic_sdk.h ▪ If __MQTT_CLEAN_SESSION_MODE_SUPPORT__ is not defined, mqtt client always connects to a mqtt broker in CleanSession=1 mode ▪ MQTT re-connection is required for this new configuration to take effect ▪ See MQTT Example: Using CleanSession=0	
AT+NWMQTS	<num>,<topic#1>, <topic#2>, …	Set the topic(s) of the MQTT subscriber. <num>: Number of topics <topic#n>: MQTT subscriber topic(s). Max topic length = 64 Response: OK or ERROR
	?	Get the MQTT subscriber topic(s). Response: +NWMQTS:<num>,<topic#1>,<topic#2>,…
	(none)	
	Prerequisite MQTT client should be disabled (+NWMQCL:0).	

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td colspan="2">Example
AT+NWMQTS=?
ERROR:-654

AT+NWMQTS=1,da16k_sub
OK

AT+NWMQTS=?
+NWMQTS:1,”da16k_sub”
OK

Note:
▪ Enabled by default in the SDK
▪ Return ″ERROR:-654″ when there is no subscriber topic set
▪ After this command is run, the previously configured subscriber topic(s) is(are) cleared and set to the new one(s)
▪ MQTT restart is required for the new configuration to take effect</td></tr>
<tr><td>AT+NWMQATS</td><td><topic></td><td>Add the specified topic to MQTT configuration</td></tr>
<tr><td></td><td colspan="2">Prerequisite
MQTT client should be disabled (+NWMQCL:0).

Example
AT+NWMQATS=ABCD
OK

AT+NWMQTS=?
+NWMQTS:1,"ABCD"
OK

Note:
▪ Enabled by default in the SDK
▪ Query command (AT+NWMQATS=?) not supported. If AT+NWMQATS=? Is run, ″?″ is added as a topic</td></tr>
<tr><td>AT+NWMQDTS</td><td><topic></td><td>Delete the specified topic from MQTT configuration</td></tr>
<tr><td></td><td colspan="2">Prerequisite
MQTT client should be disabled (+NWMQCL:0).

Example
AT+NWMQTS=?
+NWMQTS:2,"ABCD",”EFGH”
OK

AT+NWMQDTS=ABCD
OK

AT+NWMQTS=?
+NWMQTS:1,”EFGH”
OK

Note:
▪ Enabled by default in the SDK</td></tr>
</table>

Command	Parameters	Description
AT+NWMQTP	<topic>	Set a topic of the MQTT publisher. <topic>: MQTT publisher topic Response: OK or ERROR
	?	Get the MQTT publisher topic. Response: +NWMQTP:<topic>
	(none)	
	Prerequisite MQTT client should be disabled (+NWMQCL:0). Example AT+NWMQTP=? ERROR:-662 AT+NWMQTP=da16k_pub OK AT+NWMQTP=? +NWMQTP:da16k_pub OK Note: ▪ Enabled by default in the SDK ▪ ″ERROR:-662″ means ″No Publish topic exists″ ▪ MQTT restart is required for the new configuration to take effect ▪ There is one slot for storing a publish topic so that the stored topic is replaced with new one if the AT command is re-issued	
AT+NWMQV311	<use_v311>	Use MQTT protocol v3.1.1. Default is v3.1. <use_v311>: 1 (v3.1.1) / 0 (v3.1)
	?	Shows the MQTT protocol version currently set
	Prerequisite MQTT client should be disabled (+NWMQCL:0). Example AT+NWMQV311=? +NWMQV311:0 OK AT+NWMQV311=1 OK AT+NWMQV311=? +NWMQV311:1 OK Note: ▪ Enabled by default in the SDK ▪ MQTT restart is required for the new configuration to take effect	
AT+NWMQPING	<period>	Set MQTT ping period. <period>: Ping period (second) Response: OK or ERROR
	?	Get the current MQTT ping period. Response: +NWMQPING:<period>
	(none)	
	Prerequisite	

Command	Parameters	Description
	MQTT client should be disabled (+NWMQCL:0). Example AT+NWMQPING=? +NWMQPING:600 OK AT+NWMQPING=300 OK AT+NWMQPING +NWMQPING:300 OK Note: ▪ Enabled by default in the SDK ▪ MQTT restart is required for the new configuration to take effect	
AT+NWMQCID	<client_id>	Set the MQTT Client ID. <client_id>: Client ID Response: OK or ERROR
	?	Get the current MQTT Client ID. Response: +NWMQCID:<client_id>
	(none)	
	Prerequisite MQTT client should be disabled (+NWMQCL:0). Example AT+NWMQCID=? +NWMQCID:da16x_CCA4 //generate a default cid if there is no cid stored in NVRAM AT+NWMQCID=client-1 OK AT+NWMQCID +NWMQCID:client-1 OK Note: ▪ Enabled by default in the SDK ▪ MQTT restart is required for the new configuration to take effect	
AT+NWMQLI	<name>,<pw>	MQTT login information. <name>: ID <pw>: Password Response: OK or ERROR
	?	Get the MQTT login information. Response: +NWMQLI:<name>,<pw>
	(none)	
	Prerequisite MQTT client should be disabled (+NWMQCL:0). Example AT+NWMQLI=? ERROR:-673	

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td colspan="2">AT+NWMQLI=da16k_user,12345678
OK

AT+NWMQLI
+NWMQLI:da16k_user,12345678
OK

Note:
▪ Enabled by default in the SDK
▪ ″ERROR:-673″ means ″No user name exists″
▪ MQTT restart is required for the new configuration to take effect</td></tr>
<tr><td rowspan="3">AT+NWMQAUTO</td><td><auto></td><td>Enable/Disable auto-start of MQTT Client at reboot.
<auto>: 1 (Enable), 0 (Disable)</td></tr>
<tr><td>?</td><td rowspan="2">Get the MQTT Client’s auto start configuration status.
Response: +NWMQAUTO:<auto></td></tr>
<tr><td>(none)</td></tr>
<tr><td colspan="2">Prerequisite
MQTT client should be disabled (+NWMQCL:0).

Example
AT+NWMQAUTO=?
+NWMQAUTO:0
OK

AT+NWMQAUTO=1
OK

AT+NWMQAUTO
+NWMQAUTO:1
OK

Note:
▪ Enabled by default in the SDK
▪ Default is 0 (disable)
▪ MQTT restart is required for the new configuration to take effect</td></tr>
<tr><td rowspan="3">AT+NWMQWILL</td><td><topic>,<msg>,<qos></td><td>Set MQTT Will message.
<topic>: Will topic
<msg>: Will message
<qos>: Will QoS. 0 (at most once), 1 (at least once), 2 (exactly once)
Response: OK or ERROR</td></tr>
<tr><td>?</td><td rowspan="2">Get the MQTT Will message.
Response: +NWMQWILL:<topic>,<msg>,<qos></td></tr>
<tr><td>(none)</td></tr>
<tr><td colspan="2">Prerequisite
MQTT client should be disabled (+NWMQCL:0).

Example
AT+NWMQWILL=?
ERROR:-664
Or
ERROR:-665

AT+NWMQWILL=da16k_will,bye,0
OK</td></tr>
</table>

Command	Parameters	Description
		AT+NWMQWILL +NWMQWILL:da16k_will,bye,0 OK Note: ▪ Enabled by default in the SDK ▪ ″ERROR:-664 or -665″ means topic or message is missing ▪ MQTT restart is required for the new configuration to take effect
AT+NWMQDEL	(none)	Reset the MQTT configurations. Response: OK or ERROR
	Prerequisite MQTT client should be disabled (+NWMQCL:0). Example AT+NWMQDEL OK Note: ▪ This command will reset all MQTT configurations ▪ If the MQTT client is running, run this command after the MQTT client is disabled by AT+NWMQCL=0	

Table 17: MQTT operation command list

Command	Parameters	Description
AT+NWMQCL	<mqtt_client>	Enable/disable the MQTT client. <mqtt_client>: 0 (disable), 1 (enable) Response: OK or ERROR
	?	Get the MQTT client status. Response: +NWMQCL:<mqtt_client>
	(none)	
	Prerequisite DA16200/DA16600 should be connected to AP. Example AT+NWMQCL=1 OK AT+NWMQCL +NWMQCL:1 OK Note: ▪ Enabled by default in the SDK ▪ If the system restarts, then the MQTT client is not started automatically as this command is just to start/stop the MQTT client ▪ Setting MQTT configuration parameters such as MQTT broker IP, port number, and subscriber topic, are required to be done before issuing this command	
AT+NWMQMSG	<msg>,<topic>	Publish an MQTT message. <msg>: Message to be published <topic>: MQTT topic (optional) Response: OK or ERROR Operation Results:

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td></td><td>Send Success: +NWMQMSGSND:1
Send Failure: +NWMQMSGSND:0,<err_code></td></tr>
<tr><td></td><td colspan="2">Prerequisite
MQTT client should be enabled (+NWMQCL:1).

Example
AT+NWMQMSG=Hello world !!!
OK
+NWMQMSGSND:1

AT+NWMQMSG='{"car":"red", "type":"bus"}'
OK
+NWMQMSGSND:1

AT+NWMQMSG=Hello
OK
+NWMQMSGSND:0,-6 // Send failed due to mqtt is not in connected state

Note:
▪ Enabled by default in the SDK v3.2.3.0 or later
▪ If a single quotation is used in a message, surrounded by double quotation marks
▪ In the default AT command image, the maximum total combined string length allowed for <msg> + <topic> should be less than or equal to 2066. So, if it needs to send a max length <msg> (2048 bytes long) with an explicit <topic> specified, the <topic> length should be 18 characters long or less. If it needs to send a message with the maximum length topic allowed (which is 64), send 2002 bytes <msg> in maximum
▪ About Operation Results (+NWMQMSGSND:1 or +NWMQMSGSND:0,<err_code>):
• Depending on network condition, a message publishing transaction (Qos 0, 1, 2) may take some time if network condition is not good
• A new async response +NWMQMSGSND:1 or +NWMQMSGSND:0 that comes after ″OK″ indicates either completion of a publish transaction or failure
• In CleanSession=1 mode, if mqtt is disconnected, the host can immediately get +NWMQMSGSND:0, but in CleanSession=0 mode, +NWMQMSGSND:0 is not sent but instead the transaction resumes when mqtt client re-connects. See MQTT Example: Using CleanSession=0</td></tr>
<tr><td>AT+NWMQUTS</td><td><topic></td><td>Unsubscribe from the specified topic</td></tr>
<tr><td></td><td colspan="2">Prerequisite
MQTT client should be enabled (+NWMQCL:1).

Example
AT+NWMQUTS=ABCD
OK

Note:
▪ Enabled by default in the SDK (available from SDK v3.2.3.0 or later)
▪ This command should be run while the MQTT client is in a connected state with Broker</td></tr>
<tr><td>AT+NWMQTT</td><td><host_name>,<port>,
<sub_topic>,
<pub_topic>,
<qos>,<tls>,
<username>,
<password></td><td>Run the MQTT Client with options. After entering this command, system will reboot automatically. At reboot, DA16200/DA16600 tries to connect to the MQTT broker after the Wi-Fi connection is successfully established.
<host_name>: Broker’s domain name or IP address
<port>: Broker’s port number
<sub_topic>: MQTT subscriber topic
<pub_topic>: MQTT publisher topic</td></tr>
</table>

Command	Parameters	Description
		<qos>: MQTT QoS level. <tls>: Enable/disable MQTT TLS. 1 (enable), 0 (disable) <username>: Login ID (optional) <password>: Login password (optional) Response: OK or ERROR
	Prerequisite MQTT client should be disabled (+NWMQCL:0). Example AT+NWMQTT=192.168.0.65,1884,da16k_sub,da16k_pub,0,0 ; Below are logs after DA16200 reboot +INIT:DONE,0 +WFJAP:1,'test_ap_ssid',192.168.0.88 +NWMQCL:1 Note: ▪ Enabled by default in the SDK ▪ After the system reboot, operation result is sent, see ″+NWMQCL″ response	

Table 18 shows optional MQTT configuration commands for the MQTT brokers that require TLS ALPN, SNI, or Cipher Suite information from MQTT Client at the connection stage. These commands are enabled by default in SDK v3.2.3.0 or later.

Table 18: MQTT optional configuration commands

Command	Parameters	Description
AT+NWMQALPN	<num>, <alpn#1>, <alpn#2>, <alpn#3>	Set the TLS ALPN protocol name for MQTT. <num>: Number of ALPNs. Maximum number of ALPN is three <alpn#n>: TLS ALPN protocol name. Maximum length of each ALPN protocol name is 24 Response: OK or ERROR
	?	Get the TLS ALPN(s) that have been set. Response: +NWMQALPN:<num>,<alpn#1>,<alpn#2>,<alpn#3>
	Prerequisite MQTT client should be disabled (+NWMQCL:0). Example AT+NWMQALPN=? ERROR:-644 AT+NWMQALPN=2,alpn-protrol-name-an,alpn-protocol-name-ax OK AT+NWMQALPN +NWMQALPN:2,"alpn-protrol-name-an","alpn-protocol-name-ax" OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __MQTT_TLS_OPTIONAL_CONFIG__ is enabled in the SDK, this command will be enabled ▪ ″ERROR:-644″ means ″No ALPN is set″	
AT+NWMQSNI	<sni>	Set TLS SNI for MQTT. <sni>: Server Name Indication. Maximum length of SNI is 64

Command	Parameters	Description
		Response: OK or ERROR
	?	Get the TLS SNI that has been set. Response: +NWMQSNI:<sni>
	Prerequisite MQTT client should be disabled (+NWMQCL:0). Example AT+NWMQSNI=?AERROR:-648 AT+NWMQSNI=a38a9rhiu3roqb-ats.myserver.com OK AT+NWMQSNI +NWMQSNI: a38a9rhiu3roqb-ats.myserver.com OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __MQTT_TLS_OPTIONAL_CONFIG__ is enabled in the SDK, this command will be enabled ▪ “ERROR:-648” means “No SNI is set”	
AT+NWMQCSUIT	<cipher suite 1>, <cipher suite 2>, …	Set TLS Cipher suites <cipher suite>: A hex decimal value of cipher suite. See Appendix D. Maximum number of cipher suites is 17 Response: OK or ERROR
	?	Get TLS cipher suites that have been set Response: +NWMQSNI:<number of cipher suites>,<cipher suite 1>,<cipher suite 2>,…
	Prerequisite MQTT client should be disabled (+NWMQCL:0). Example AT+NWMQCSUIT=? ERROR:-650 or ERROR:-651 AT+NWMQCSUIT=c024,c023,c00a,c009,c00d,c032 OK AT+NWMQCSUIT +NWMQCSUIT:6,c024,c023,c00a,c009,c00d,c032 OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __MQTT_TLS_OPTIONAL_CONFIG__ is enabled in the SDK, this command will be enabled ▪ ″ERROR:-650 or -651″ means ″No CSUIT info is set″ ▪ The hex pre-fix ″0x″ should be removed when one is typed ▪ DA16200/DA16600 does not support all the cipher suites due to memory limitation. Contact Renesas Electronics for using other cipher suites that are not specified in Appendix D	

Table 19: MQTT response list

Response	Parameters	Description
+NWMQCL	<result> [,TOO_LONG_MSG_RX]	The result of the MQTT client connection. <result>: 0 (disconnected), 1 (connected) For example: +NWMQCL:1 If MQTT connection to the MQTT broker is successfully established +NWMQCL:0 If the MQTT connection is NOT successfully established +NWMQCL:0,TOO_LONG_MSG_RX Can be sent when the MQTT is disconnected by unsupported message length only if the current mqtt connection is with clean_session=0 and qos greater than or equal to 1.
	Example ; When MQTT connection to the MQTT broker is successfully established +NWMQCL:1 ; When MQTT broker is down, +NWMQCL:0 Note: ▪ DA16200/DA16600 Debug Console log (UART0) • When MQTT connection to the MQTT broker is successfully established, DA16200/DA16600 sends ″+NWMQCL:1″ message to MCU (or AT command console). At this moment, the following log appears: >>> MQTT Client connection OK • When the MQTT broker is down, DA16200/DA16600 sends ″+NWMQCL:0″ message to MCU (or AT command console) after retrying connection. In console log, the following logs appears: *Failed to receive pkt. (0x38)* *Failed to read pkt(0x7880)* *MQTT Client disconnected ... (state=6)* *[SUB] REQ mqtt_restart (count=1)* *Connecting FAIL (0x38)* *Unable to connect (The connection was refused.)* *…* *[SUB] REQ mqtt_restart (count=5)* *Connecting FAIL (0x38)* *Unable to connect (The connection was refused.)* *[SUB] MAX Retry (Retry Cnt=6).* ▪ To get this Operation Result, it may take more time if the DA16200/DA16600 connection retrial happens depending on the test network condition ▪ This message is also sent after AT+NWMQTT is run or if any MQTT configuration command is run and then the system is restarted • DA16200/DA16600 restarts if the AT command format is OK ○ +INIT:DONE,0 message is sent as DA16200/DA16600 boots up ○ If usage of the AT command is not valid, DA16200/DA16600 sends an ERROR message without restarting • DA16200/DA16600 tries to connect to the AP after the reboot ○ +WFJAP:0,<reason> or +WFJAP:1,'<SSID>',<IP Address> as result of the Wi-Fi connection ○ If the Wi-Fi connection information such as SSID or key is NOT stored correctly in the DA16200/DA16600 NVRAM, +WFJAP:x response is NOT sent and the MQTT connection is NOT attempted as well. Because the MQTT connection needs a successful Wi-Fi connection first	

<table>
<tr><th>Response</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td colspan="2">

DA16200/DA16600 tries to connect to the MQTT broker after the Wi-Fi connection is established. The MQTT broker information is stored in NVRAM. Connection result
+NWMQCL:0 or +NWMQCL:1 – is sent over UART1 as a result

▪ NWMQCL:0 is sent when the Wi-Fi Link goes down due to the following conditions

Sometimes, the host can get an unsolicited +NWMQCL:0 message when DPM Wakeup under poor signal condition with the AP connected. An example is when Wi-Fi connection with AP becomes unstable – such as when DPM Keepalive fails, Beacon loss detected. In these cases, as STA, DA16200 / DA16600 tries to get “Wi-Fi link” down and up to reconnect with AP (while doing this, Wi-Fi is disconnected and then re-connected). MQTT client, when this kind of situation is detected, will try to re-connect to Broker after forcing disconnection. When MQTT disconnection happens, +NWMQCL:0 is sent to the host. On receipt of this unsolicited message, the host should wait for +NWMQCL:1

▪ +NWMQCL:0,TOO_LONG_MSG_RX

If the current mqtt_client connection with Broker is configured with clean_session=0 and qos is greater than or equal to 1, +NWMQCL:0,TOO_LONG_MSG_RX can be sent to the host (exceptional case). This message indicates that mqtt_client is disconnected by receiving a message that exceeds the message length limit (2048) of da16x mqtt client. What the host should do, in this situation, is configure clean_session to 1 and connect to Broker to delete the message, and then disconnect from Broker and reconnect with clean_session=0 again to start over (a kind of Recovery). According to MQTT Spec, Broker keeps sending a message that has not been ACKed by the client. In this case, the long message (valid for Broker, but not valid for da16x mqtt client) can repeatedly be sent to da16x mqtt client unless connection with clean_session=1 is made from da16x mqtt client

This response message is enabled by default in SDK v3.2.3.0
Example recovery flow on receipt of +NWMQCL:0,TOO_LONG_MSG_RX

<pre>...
// mqtt client is disconnected by receiving a message with unsupported length
+NWMQCL:0,TOO_LONG_MSG_RX
AT+NWMQCS=1 // set clean_session=1
OK
// connect to Broker (to clear the invalid long message)
AT+NWMQCL=1
OKtable

+NWMQCL:1
AT+NWMQCL=0 // disconnect from Broker
+NWMQCL:0

OK
AT+NWMQCS=0 // set clean_session=0
OK
AT+NWMQCL=1 // connect to Broker with clean_session=0
OK

+NWMQCL:1</pre>
</td></tr>
<tr><td rowspan="2">+NWMQMSG</td><td><msg>,<topic>,<length></td><td>Received the MQTT message.

<msg>: Message data
<topic>: Received topic
<length>: Message length</td></tr>
<tr><td colspan="2">Example
; When DA16200/DA16600 receives a message from the MQTT publisher, the following message will be sent from DA16200/DA16600 to AT command console:

+NWMQMSG:Hello world!!!!,da16k_sub,15</td></tr>
</table>

Response	Parameters	Description
	Note: ▪ MQTT client is in a connected state with the broker (+NWMQCL:1)	

5.6.1.1 MQTT Client Connection Example

Configure the parameters and start the MQTT Client (After Wi-Fi Connection):
AT+NWMQBR=172.16.0.1,1884
AT+NWMQTS=1,da16k_sub
AT+NWMQTP=da16k_pub
AT+NWMQAUTO=1 (Optional, if DPM mode is used, setting this parameter is needed)
AT+NWMQCL=1
If the connection is successful, the following is shown:
+NWMQCL:1
If DA16K receives a PUBLISH from a broker, the following is shown:
+NWMQMSG:Hello World,da16k,11
DA16K can send a PUBLISH to a broker. Type the following command:
AT+NWMQMSG='Hello I'm DA16K'

5.6.1.2 MQTT TLS Connection Example

Configure the MQTT parameters:AT+NWMQBR=172.16.0.1,8883
AT+NWMQTS=1,da16k_sub
AT+NWMQTP=da16k_pub
AT+NWMQTLS=1
AT+NWMQAUTO=1 (Optional, if DPM mode is used, setting this parameter is needed)
To check the validity of a certificate, the DA16K should set the exact current time:
AT+TIME=yyyy-mm-dd,hh:mm:ss
And store the certificate and private key if needed. (See <ESC>C in Table 10)
After all settings are made, start the client:
AT+NWMQCL=1

5.6.1.3 MQTT Example with DPM

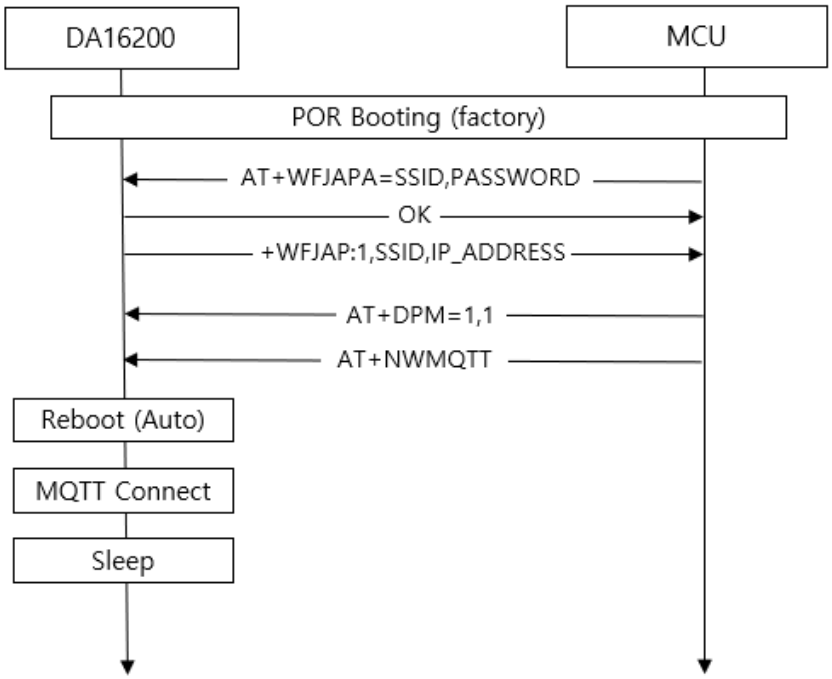


Figure 22: Example sequence to initiate MQTT protocol with DPM

Figure 22 is an example sequence to initiate the MQTT protocol with DPM in the DA16200/DA16600.
In the normal BOOT state, connect to an AP (AT+WFJAPA) and change its run mode to DPM mode (AT+DPM=1,1 □ optional parameter '1' means writing DPM mode to NVRAM and does not reboot. To make DPM mode take effect, a reboot is required).
To configure the MQTT connection information, enter command AT+CLRDPMSLPEXT and type the following as an example:

```
AT+NWMQTT=test.mosquitto.org,1883,sub_topic,pub_topic,0,0
```

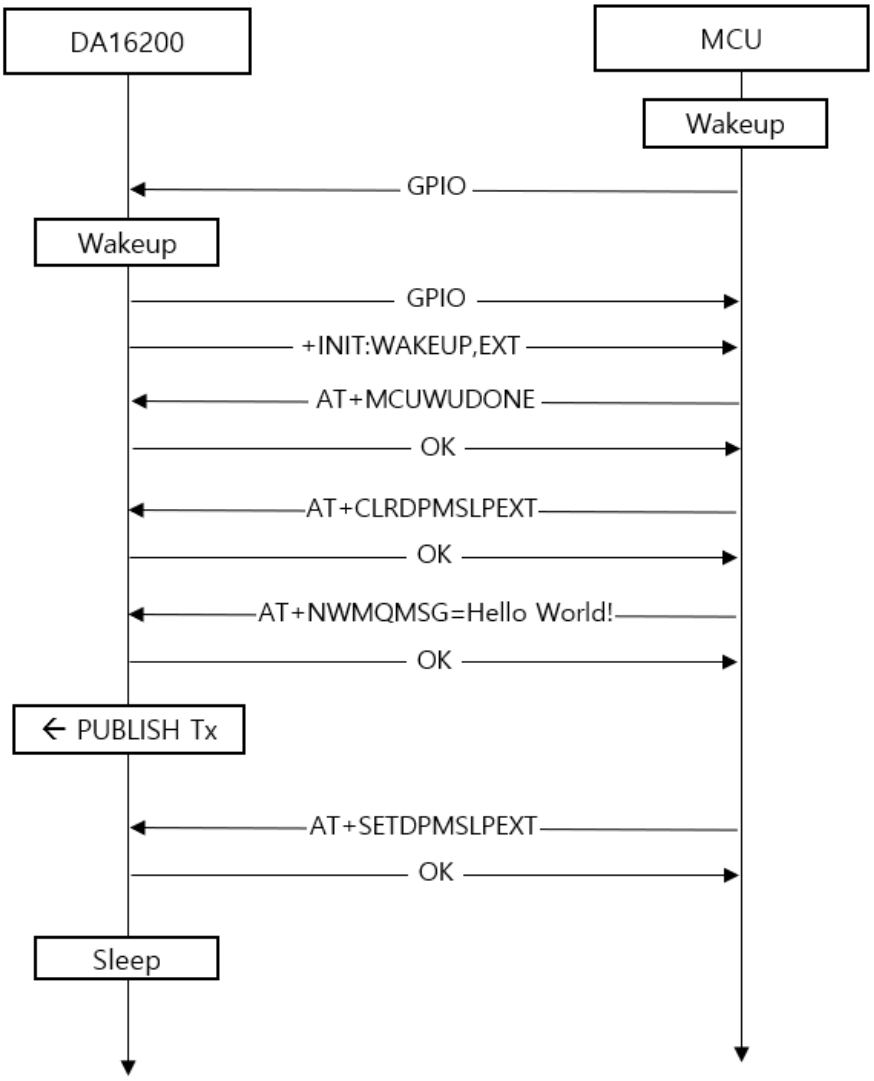


Figure 23: Procedure to send MQTT messages

Figure 23 shows the procedure to send an MQTT message in Sleep mode. When MCU wakes up the DA16200/DA16600, the response +INIT:WAKEUP,EXT is sent. The MCU sends the command AT+MCUWUDONE to inform that MCU is ready to operate. To prevent that the DA16200/DA16600 enters DPM Sleep mode, MCU should send command AT+CLRDPMSLPEXT before an MQTT PUBLISH is sent. To make the DA16200/DA16600 enter DPM Sleep mode again, send a PUBLISH with command AT+NWMQMSG, and then enter command AT+SETDPMSLPEXT.

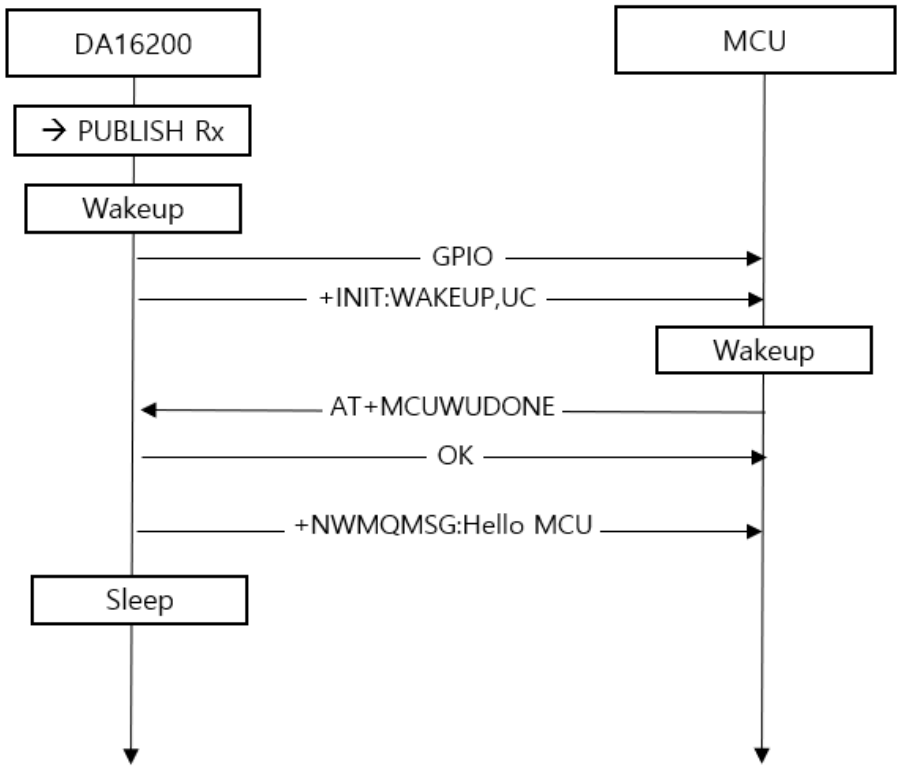


Figure 24: Procedure to process MQTT messages

Figure 24 shows how to process a received MQTT message while in Sleep mode. When the DA16200/DA16600 wakes up by a PUBLISH message from an MQTT broker, the response +INIT:WAKEUP,UC is sent. The MCU sends the AT+MCUWUDONE to inform that it is ready to operate. Next, the DA16200/DA16600 sends the received PUBLISH to the MCU and enters DPM Sleep mode again.

5.6.1.4 MQTT Example: Changing Subscription Topic when Running

Assume that the Wi-Fi/MQTT connection is configured properly and DPM is set to 1 (TRUE). Below is the recommended sequence. Note that the double quotation marks are used.

1. Trigger RTC_WAKE_UP Event (by MCU)
2. Wait for "+INIT:WAKEUP,EXT" Response. Send AT+MCUWUDONE, and wait for “OK”
3. Run "AT+CLRDPMSLPEXT" command
4. Wait for "OK" response
5. loop running "AT+NWMQCL=?"
   a. if responses are "+NWMQCL:0" and "OK"
   b. then, goto E. to run "AT+NWMQCL=?" command
   c. else if responses are "+NWMQCL:1" and "OK"
   d. then, goto next, F.
   e. else if response is "ERROR:x"
   f. then, Run "AT+SETDPMSLPEXT"
   g. Wait for "OK" response
   h. return
6. Run "AT+NWMQCL=0"
7. Wait for "+NWMQCL:0" and "OK" response
8. Run "AT+NWMQTS=<New MQTT Subscription Topic>"
9. Wait for "OK" response
10. Run "AT+RESTART"
11. Wait for "+INIT:DONE,0" response
12. Wait for "+WFJAP:1,'<SSID>',<IP ADDRESS>"
13. Wait for "+NWMQCL:1" response

5.6.1.5 MQTT Example: Reading Subscription Topic when Running

Assume that the Wi-Fi/MQTT connection is configured properly and DPM is set to 1 (TRUE). The reading of the MQTT publishing topic would be similar. Below is the recommended sequence. Note that the double quotation marks are used.

1. Trigger RTC_WAKE_UP Event
   Wait for "+INIT:WAKEUP,EXT" Response. Send AT+MCUWUDONE, and wait for "OK"
2. Run "AT+CLRDPMSLPEXT" command
3. Wait for "OK" response
4. Run "AT+NWMQTS=?"
5. Wait for "+NWMQTS:<MQTT Subscription Topic>" and "OK" response
   Note that there are possibilities to receive the ERROR response if the format of the command has some errors.
6. Run "AT+SETDPMSLPEXT"
7. Wait for "OK" response

Assume that the Wi-Fi/MQTT connection is configured properly and DPM is set to 1 (TRUE). The reading of the MQTT publishing topic would be similar. Below is the recommended sequence. Note that the double quotation marks are used.

Trigger RTC_WAKE_UP Event
1. Wait for "+INIT:WAKEUP,EXT" Response. Send AT+MCUWUDONE, and wait for “OK”
2. Run "AT+CLRDPMSLPEXT" command
3. Wait for "OK" response
4. Run "AT+NWMQTS=?"
5. Wait for "+NWMQTS:<MQTT Subscription Topic>" and "OK" response
6. Run "AT+SETDPMSLPEXT"

5.6.1.6 MQTT Example: Using CleanSession=0

5.6.1.6.1 CleanSession=0 Mode

When an MQTT Client (Mqttc onward) establishes connection with an MQTT Broker (Broker onward), there are two types of session: CleanSession=1 and CleanSession=0.
CleanSession=1: default session type. when Broker gets a connect request from an Mqttc that tries to connect with an option “CleanSession=1” (which is default config on DA16x), Broker treats the connection as a “new”

session. If there is any existing session associated with the same client_id found, Broker clears that previous session and creates a new one with the client_id.

CleanSession=0: when Broker gets a connect request from an Mqttc that tries to connect with an option “CleanSession=0”, Broker tries to find a session (session data) with the same client_id first. If it finds one, it keeps using that session for the new Mqttc.

While Mqttc is in operation with Broker, there may be times when the TCP connection gets unstable and disconnected (for example, mqtt ping failed) which may cause some messages that had been published to Broker at that specific disconnected time may not be delivered to a subscriber. If new messages (with QoS > 0) are published to Broker and for sessions that have been configured in “CleanSession=0”, Broker retains and re-send them when the Mqttc is re-connected. Mqttc (if CleanSession=0 is enabled) also should retain the state of the unfinished / unacked messages until reconnection.

```
1647307743: New connection from 192.168.0.2 on port 8883.
1647307743: New client connected from 192.168.0.2 as da16x_D9CC (c1, k60).
1647307743: Sending CONNACK to da16x_D9CC (0, 0)
1647307743: Received SUBSCRIBE from da16x_D9CC
1647307743:     SUB_TOPIC (QoS 2)
1647307743: da16x_D9CC 2 SUB_TOPIC
1647307743: Sending SUBACK to da16x_D9CC
```

Figure 25: Broker console - CleanSession=1 Connection

```
1647307743: New connection from 192.168.0.2 on port 8883.
1647307743: New client connected from 192.168.0.2 as da16x_D9CC (c1, k60).
1647307743: Sending CONNACK to da16x_D9CC (0, 0)
1647307743: Received SUBSCRIBE from da16x_D9CC
1647307743:     SUB_TOPIC (QoS 2)
1647307743: da16x_D9CC 2 SUB_TOPIC
1647307743: Sending SUBACK to da16x_D9CC
```

Figure 26: Broker console - CleanSession=0 Connection

Even with CleanSession=0 connection, Broker does not maintain session data if MQTT is disconnected in the following cases.

- If a new message is published with QoS 0 after MQTT is disconnected
- If MQTTC connection QoS is 0

DA16x supports CleanSession=0 mode in the following way.

- “CleanSession=0 feature” is enabled by default in SDK v3.2.3.0 or later
- If a customer application decides that QoS 1 or Qos 2 and CleanSession=0 is used in their application, the message (payload) size (both Tx and Rx) should be pre-decided (because there is limitation in dpm user pool size). By default, 100 bytes are defined

  #define MQTT_MSG_TBL_PRESVD_MAX_PLAYLOAD_LEN 100
- Depending on application’s expected maximum payload size while operation, other value can be defined
- DPM User Pool has limited amount (about 8K in total) in the system
- Check available "free" DPM User Pool size first (by using the console command "dpm user_pool"), and then calculate the max payload length and message number for the application if needed
- The default configuration (payload_len: 100, max_count: 10) allocates about 1.9 kB of dpm user pool (Check mq_msg_tbl_presvd_t for detail)
- Search the following compiler options in config_generic_sdk.h

  //max payload length of a preserved message

  #define MQTT_MSG_TBL_PRESVD_MAX_PLAYLOAD_LEN 100

  // max number of preserved messages

  #define MQTT_MSG_TBL_PRESVD_MAX_MSG_CNT 10
- Supported command to set CleanSession mode: AT+NWMQCS=<1|0>

CleanSession and QoS Matrix Table for PUBLISH Rx

Table 20: CleanSession and QoS matrix in message Rx

Subscriber			Unacked message delivery (After MQTT reconnection)	QoS (Effective actual)	Publisher message's QoS
Case	Clean Session	QoS			
1	1	0	X	0	0
2	1	1	X	0	0
3	1	2	X	0	0
4	1	0	X	0	1
5	1	1	X	1	1
6	1	2	X	1	1
7	1	0	X	0	2
8	1	1	X	1	2
9	1	2	X	2	2
10	0	0	X	0	0
11	0	1	X	0	0
12	0	2	X	0	0
13	0	0	X	0	1
14	0	1	O	1	1
15	0	2	O	1	1
16	0	0	X	0	2
17	0	1	O	1	2
18	0	2	O	2	2

With CleanSession=1, no unacked message delivery happens when MQTT reconnect happens (marked as x)
With CleanSession=0, only case 14, 15, 17, and 18 makes message redelivery happen for messages that had been delivered to Broker while the MQTTC was offline (marked as O).

CleanSession and QoS Matrix Table for PUBLISH Tx

Expectation 1	Application assumes that it failed to send a message and waits until mqtt gets re-connected.
Behavior 1	Application sends messages again.
Expectation 2	Application assumes that it failed to send a message but will resume sending the message when mqtt re-connected.
Behavior 2	Application simply waits as mqtt send the message automatically.

Table 21: CleanSession and QoS matrix in message Tx

Publisher			Expectation if MQTT gets disconnected (while QoS 1/2 message is not fully acked or QoS 0 Send is being sent)	
Case	Clean Session	QoS		
1	1	0	Expectation 1	Behavior 1
2	1	1	Expectation 1	Behavior 1
3	1	2	Expectation 1	Behavior 1
4	0	0	Expectation 1	Behavior 1
5	0	1	Expectation 2	Behavior 2
6	0	2	Expectation 2	Behavior 2

When publishing a message from DA16x, the application's expectation and action/behavior may be different if CleanSession=0 and QoS 1 or 2 are used in some specific cases.
In normal network conditions, there is no difference in message send behavior between CleanSession=0 and CleanSession=1.
In some abnormal cases where QoS 1/2's ACK message (PUBACK, PUBREC, PUBREL, or PUBCOMP) get lost due to some bad network conditions (which can cause Mqttc re-connection), CleanSession=0 can recover the previous message state and resume the communication with Broker.

However, if CleanSession=1 is used, when Mqttc is disconnected, it can safely re-transmit the message when Mqttc is reconnected. Depending on use cases of applications / host applications, either approach (CleanSession=0 or CleanSession=1) can be utilized.

5.6.1.6.2 How to Connect with CleanSession=0

```
AT+NWMQQOS=2

OK

AT+NWMQCS=0

OK

AT+NWMQCL=1

OK


+NWMQCL:1
```

To activate “CleanSession=0 support mode” in DA16x, QoS should be 1 or 2 and CleanSession option should be set to 0. If either option (CleanSession and QoS) is not set as above, CleanSession=0 support mode is disabled.

5.6.1.6.3 How to Restart CleanSession=0 Test

If it needs to re-test (fresh new test) with CleanSession=0 mode, depending on the previous session type, it may need Broker to clear the previous session.

The reason is that since an Mqttc connects with CleanSession=0, Broker does not delete the session data until the Mqttc re-connects with CleanSession=1.

Case 1: Previous session is CleanSession=1 and need to restart a new CleanSession=0 test

```
AT+NWMQCL=0

OK

AT+NWMQCS=0

OK

AT+NWMQCL=1

OK


+NWMQCL:1
```

Case 2: Previous session is CleanSession=0 and need to re-test another CleanSession=0 test run.

```
AT+NWMQCL=0
+NWMQCL:0

OK
AT+NWMQCS=1
OK
AT+NWMQCL=1
OK

+NWMQCL:1
AT+NWMQCL=0
+NWMQCL:0

OK
AT+NWMQCS=0
OK
AT+NWMQCL=1
OK

+NWMQCL:1
```

5.6.1.6.4 PUBLISH Rx Test Steps

Test steps are as follows under non-DPM and DPM mode.

Non-DPM Mode:

- DA16x: connect to Broker
- Publisher: send one or two messages
- DA16x: check if the messages are received
- DA16x: disconnect from Broker
- Publisher: send one or two messages (let say msg_A)
- DA16x: reconnect to Broker
- DA16x: check if msg_A (sent while DA16x is offline) is received.

DPM Mode:

- DA16x: connect to Broker. Enter DPM Sleep
- Publisher: send one or two messages
- DA16x: check if the messages are received
- DA16x: turn off AP. Do not turn on AP, but wait for the mqtt keep alive period (to make sure Broker recognizes the Mqttc disconnection)
- Publisher: send one or two messages (let say msg_A)
- DA16x: turn on AP. Wait until DA16x is connected to AP
- DA16x: reconnected to AP and check if msg_A (sent while DA16x is offline) is received.

NOTE

- Mosquitto broker (Broker), mosquitto publisher (Publisher), and DA16x (Subscriber) are used for the test.
- Message length from publisher should be less than or equal to 100. If longer messages are sent, they may not be restored properly when mqtt is reconnected.

5.6.1.6.5 PUBLISH Rx Test Steps - Example 1 (Non-DPM)

Below are the test steps for case 15 (non-DPM mode).

[DA16x] connect Mqttc with CleanSession=0 and QoS 2.

```
AT+NWMQQOS=2
OK
AT+NWMQCS=0
OK
AT+NWMQCL=1
OK

+NWMQCL:1
```

[Other Publisher] publish messages.

```
C:\mosquitto>mosquitto_pub -h 192.168.0.230 -p 8883 --cafile cas.pem --cert wifiuser.pem --key
wifiuser.key --tls-version tlsv1 --insecure -q 2 -t SUB_TOPIC -m "Hello_q2"

C:\mosquitto>mosquitto_pub -h 192.168.0.230 -p 8883 --cafile cas.pem --cert wifiuser.pem --key
wifiuser.key --tls-version tlsv1 --insecure -q 2 -t SUB_TOPIC -m "Hello_q2_2"
```

[DA16x] check the messages are successfully received.

```
+NWMQMSG:Hello_q2,SUB_TOPIC,8


+NWMQMSG:Hello_q2_2,SUB_TOPIC,10
```

[DA16x] disconnect from Broker.

```
AT+NWMQCL=0
+NWMQCL:0

OK
```

[Other Publisher] publish two messages (while DA16x is in disconnected state)

```
C:\mosquitto>mosquitto_pub -h 192.168.0.230 -p 8883 --cafile cas.pem --cert wifiuser.pem --key
wifiuser.key --tls-version tlsv1 --insecure -q 2 -t SUB_TOPIC -m "Hello_q2_3"

C:\mosquitto>mosquitto_pub -h 192.168.0.230 -p 8883 --cafile cas.pem --cert wifiuser.pem --key
wifiuser.key --tls-version tlsv1 --insecure -q 2 -t SUB_TOPIC -m "Hello_q2_4"
```

[DA16x] reconnect to Broker and check if the two messages that had been published while DA16x is in disconnected state are received successfully.

```
AT+NWMQCL=1
OK

+NWMQCL:1

+NWMQMSG:Hello_q2_3,SUB_TOPIC,10

+NWMQMSG:Hello_q2_4,SUB_TOPIC,10
```

5.6.1.6.6 PUBLISH Rx Test Steps - Example 2 (DPM)

Below are the test steps for case 18 (DPM mode). Note that mosquitto broker and mosquitto publisher are used for test.

[DA16x] Connect with CleanSession=0 and QoS 2.

```
AT+NWMQQOS=2
OK
AT+NWMQCS=0
OK
AT+RESTART
OK

+INIT:DONE,0

+WFJAP:1,'SYN_TEST_AP',192.168.1.195

+ATPROV=STATUS 0

+NWMQCL:1
```

[Other Publisher] Publish messages.

```
C:\mosquitto>mosquitto_pub -h 192.168.0.230 -p 8883 --cafile cas.pem --cert wifiuser.pem --key
wifiuser.key --tls-version tlsv1 --insecure -q 2 -t SUB_TOPIC -m "Hello_q2_1"

C:\mosquitto>mosquitto_pub -h 192.168.0.230 -p 8883 --cafile cas.pem --cert wifiuser.pem --key
wifiuser.key --tls-version tlsv1 --insecure -q 2 -t SUB_TOPIC -m "Hello_q2_2"
```

[DA16x] check the messages are successfully received.

```
+INIT:WAKEUP,UC

+ATPROV=STATUS 0

+NWMQMSG:Hello_q2_1,SUB_TOPIC,10

+INIT:WAKEUP,UC

+ATPROV=STATUS 0

+NWMQMSG:Hello_q2_2,SUB_TOPIC,10
```

NOTE

At wakeup time, the host should send AT+CLRDPMSLPEXT to get +NWMQMSG after which AT+SETDPMSLPEXT should be sent by the host to let DA16x enter DPM Sleep.

[DA16x] Turn OFF AP.

```
+INIT:WAKEUP,NOBCN

+ATPROV=STATUS 0

+WFDAP:0,INACTIVITY
...
```

[Broker] make sure Mqttc is disconnected.

```
...
1647405247: Socket error on client da16x_D9CC, disconnecting.
...
```

[Other Publisher] publish two messages (while DA16x is in disconnected state).

```
C:\mosquitto>mosquitto_pub -h 192.168.0.230 -p 8883 --cafile cas.pem --cert wifiuser.pem --key wifiuser.key --tls-version
tlsv1 --insecure -q 2 -t SUB_TOPIC -m "Hello_q2_3"

C:\mosquitto>mosquitto_pub -h 192.168.0.230 -p 8883 --cafile cas.pem --cert wifiuser.pem --key wifiuser.key --tls-version
tlsv1 --insecure -q 2 -t SUB_TOPIC -m "Hello_q2_4"
```

[DA16x] Turn ON AP

[DA16x] Wait until AP is connected and see whether “hello_qos_3” and “hello_qos_4” are received.

```
+INIT:DONE,0

+WFJAP:1,'SYN_TEST_AP',192.168.1.195

+ATPROV=STATUS 0

+NWMQCL:1

+NWMQMSG:Hello_q2_3,SUB_TOPIC,10

+NWMQMSG:Hello_q2_4,SUB_TOPIC,10
```

5.6.1.6.7 PUBLISH Tx Test Steps

Test steps are as follows:

- DA16x: connect to Broker
- DA16x: send a messages
- DA16x: check if the message send is successful

NOTE
Message length from DA16200/DA16600 should be less than or equal to 100 bytes for case 5 and 6 configuration. Sending longer messages returns failure. For cases other than case 5 or 6, message length limit is 2048 bytes.

5.6.1.6.8 PUBLISH Tx Test Steps – Example

Below are the test steps for case 6 (non-DPM mode).

```
AT+NWMQQOS=2
OK
AT+NWMQCS=0
OK
AT+NWMQCL=1
OK

+NWMQCL:1
AT+NWMQMSG=hello_q2
OK

+NWMQMSGSND:1
```

5.6.2 HTTP-Client Commands

Table 22: HTTP-Client command list

Command	Parameters	Description
AT+NWHTC	<url>,<method>(,<body>)	Start the HTTP client with options. <url>: HTTP server address <method>: GET, POST or PUT <body>: Body for POST and PUT methods, omitted for other methods. For JSON type, enclose the message with the single quotation - <'body'>
	Prerequisite DA16200/DA16600 should be connected to AP. Example 1 AT+NWHTC=http://httpbin.org/get,get OK Example 2 AT+NWHTC=http://httpbin.org/post,post,HTTP-Client POST method sample test! OK For JSON type,	

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td colspan="2">AT+NWHTC=http://httpbin.org/post,post,'{ username: "aaa", password: "1234"}'
OK

Note:
▪ Enabled by default in the SDK v3.2.3.0 or later
▪ If __SUPPORT_HTTP_CLIENT_FOR_ATCMD__ is enabled in the SDK, this command will be enabled</td></tr>
<tr><td></td><td><url>,message,<'header +body'></td><td>Users can directly input header and body as plain text.
Line feeds and carriage returns are inserted as \r\n.
<url>: HTTP server address
message: Use the message as a fixed option (not the http method)
<'header+body'>: Enter a plain text string in the form of 'header+body'</td></tr>
<tr><td></td><td colspan="2">Prerequisite
DA16200/DA16600 should be connected to AP.

Example 1 : GET method request (header)
AT+NWHTC=http://httpbin.org/get,message,'GET /get HTTP/1.1\r\nHost: httpbin.org\r\nConnection: Close\r\n\r\n'
OK

Example 2 : POST method request (header+body)
AT+NWHTC=http://httpbin.org/post,message,'POST /postHTTP/1.1\r\nHost: httpbin.org\r\nAccept: */*\r\nContent-Length: 10\r\nConnection: Close\r\n\r\nHelloWorld\r\n'
OK
For JSON type,
AT+NWHTC=http://httpbin.org/post,message,'POST /postHTTP/1.1\r\nHost: httpbin.org\r\nContent-Type: application/json\r\nContent-Length: 40\r\nConnection: Close\r\n\r\n{ username: "aaa", password: "1234"}\r\n'
OK

Note:
▪ Enabled by default in the SDK v3.2.3.0 or later
▪ If __SUPPORT_HTTP_CLIENT_FOR_ATCMD__ is enabled in the SDK, this command will be enabled</td></tr>
<tr><td>AT+NWHTCH</td><td><url>,<method>(,<msg>)</td><td>AT+NWHTCH with H appended after AT+NWHTC.
All parameters and functions are exactly the same as AT+NWHTC.
The difference is that data size is inserted in front of the received data and transmitted.

Start the HTTP client with options.
<url>: HTTP server address
<method>: GET, POST or PUT
<msg>: Request message for POST and PUT methods</td></tr>
<tr><td></td><td colspan="2">Prerequisite
DA16200/DA16600 should be connected to AP.

Example 1
AT+NWHTCH=https://httpbin.org/get,get
OK

+NWHTCDATA:225,HTTP/1.1 200 OK
Date: Fri, 02 Dec 2022 01:17:30 GMT
Content-Type: application/json</td></tr>
</table>

Command	Parameters	Description

	Content-Length: 297
	Connection: close
	Server: gunicorn/19.9.0
	Access-Control-Allow-Origin: *
	Access-Control-Allow-Credentials: true
	
	+NWHTCDATA:297,{
	"args": {},
	"headers": {
	"Accept": "*/*",
	"Host":"httpbin.org",
	"User-Agent": "lwIP/2.1.2 (http://savannah.nongnu.org/projects/lwip)",
	"X-Amzn-Trace-Id": "Root=1-6389522a-4ceffbc701b0e20f348c5ecc"
	},
	"origin": "124.50.108.25",
	"url": "https://httpbin.org/get"
	}
	
	+NWHTCSTATUS:0
	
	Example 2
	AT+NWHTCH=https://httpbin.org/post,post,HTTP-Client POST method sample test!
	OK
	
	+NWHTCDATA:225,HTTP/1.1 200 OK
	Date: Fri, 02 Dec 2022 01:25:38 GMT
	Content-Type: application/json
	Content-Length: 426
	Connection: close
	Server: gunicorn/19.9.0
	Access-Control-Allow-Origin: *
	Access-Control-Allow-Credentials: true
	
	+NWHTCDATA:426,{
	
	"args": {},
	"data": "HTTP-Client POST method sample test!"
	,
	"files": {},
	"form": {},
	"headers": {
	"Accept": "*/*",
	"Content-Length": "36",
	"Host": "httpbin.org",
	"User-Agent": "lwIP/2.1.2 (http://savannah.nongnu.org/projects/lwip)",
	"X-Amzn-Trace-Id": "Root=1-63895412-4f92fb296482283c68e2155f"
	},
	"json": null,
	"origin": "124.50.108.25",
	"url": "https://httpbin.org/post"
	}	

Command	Parameters	Description
	+NWHTCSTATUS:0 Note: ▪ Enabled by default in the SDK v3.2.5.0 or later ▪ If __SUPPORT_HTTP_CLIENT_FOR_ATCMD__ is enabled in the SDK, this command will be enabled	
AT+NWHTCSNI	<sni>	Set the server name indication. <sni>: Server name
	Example AT+NWHTCSNI=httpbin.org OK Note: ▪ Enabled by default in the SDK ▪ It must be set up before connecting to the server	
AT+NWHTCALPN	<alpn_number>,<alpn1>, <alpn2>,<alpn3>	Set the application layer protocol negotiation. <alpn_number>: Number of alps <alpn1>: First alpn <alpn2>: Second alpn <alpn3>: Third alpn
	Example AT+NWHTCALPN=1,http/1.1 OK Note: ▪ Enabled by default in the SDK ▪ It must be set up before connecting to the server	
AT+NWHTCSNIDEL	(none)	Delete the saved SNI
	Example AT+NWHTCSNIDEL OK Note: ▪ Enabled by default in the SDK ▪ It must be set up before connecting to the server	
AT+NWHTCALPNDEL	(none)	Delete all saved ALPNs
	Example AT+NWHTCALPNDEL OK Note: ▪ Enabled by default in the SDK ▪ It must be set up before connecting to the server	
AT+NWHTCTLSAUTH	<tls_auth_mode>	Set the certificate verification mode. #define MBEDTLS_SSL_VERIFY_NONE 0 #define MBEDTLS_SSL_VERIFY_OPTIONAL 1 #define MBEDTLS_SSL_VERIFY_REQUIRED 2
	Example AT+NWHTCTLSAUTH=1 OK Note:	

Command	Parameters	Description
	▪ Enabled by default in the SDK ▪ It must be set up before connecting to the server	

Table 23: HTTP-Client response list

Command	Parameters	Description
+ NWHTCSTATUS	<status>	Return status along with the received payload according to the requested method. <status>: 0x00 is success See Appendix B For example: +NWHTCSTATUS:0x00
+NWHTCDATA	<data size,>	Insert size information of data received only from AT+NWHTCH command. An integer data size and a comma are inserted, and the actual received data is after that. For example, +NWHTCDATA:426,

5.6.2.1 HTTP-Client Connection Example

GET method request:

```
AT+NWHTC=https://httpbin.org/get,get
OK
HTTP/1.1 200 OK
Date: Tue, 07 Dec 2021 01:19:49 GMT
Content-Type: application/json
Content-Length: 457
Connection: keep-alive
Server: gunicorn/19.9.0
Access-Control-Allow-Origin: *
Access-Control-Allow-Credentials: true

{
  "args": {},
  "headers": {
    "Accept": "*/*",
    "Accept-Encoding": "identity",
    "Accept-Language": "ko-KR,Ko;q=0.9,en-US;q=0.8,en;q=0.7",
    "Host": "httpbin.org",
    "User-Agent": "Mozilla/5.0 (windows NT 6.1; Win64; x64) AppleWebkit/537.36 (KHTML, like Gecko)
Chrome/64.0.3282.186 Safari/537.36",
    "X-Amzn-Trace-Id": "Root=1-61aeb6b5-67d7324c112a7f1631adcc72"
  },
  "origin": "124.50.108.25",
  "url": "https://httpbin.org/get"
}

+NWHTCSTATUS:0x00
```

POST method request:

```
AT+NWHTC=https://httpbin.org/post,post,HTTP-Client POST method sample test!
OK
HTTP/1.1 200 OK
Date: Tue, 07 Dec 2021 01:25:05 GMT
Content-Type: application/json
Content-Length: 586
Connection: keep-alive
Server: gunicorn/19.9.0
Access-Control-Allow-Origin: *
Access-Control-Allow-Credentials: true

{
  "args": {},
  "data": "HTTP-Client POST method sample test!",
  "files": {},
  "form": {},
  "headers": {
    "Accept": "*/*",
    "Accept-Encoding": "identity",
    "Accept-Language": "ko-KR,Ko;q=0.9,en-US;q=0.8,en;q=0.7",
    "Content-Length": "36",
    "Host": "httpbin.org",
    "User-Agent": "Mozilla/5.0 (windows NT 6.1; Win64; x64) AppleWebkit/537.36 (KHTML, like Gecko)
Chrome/64.0.3282.186 Safari/537.36",
    "X-Amzn-Trace-Id": "Root=1-61aeb7f1-341bbb8c3f3d6bc7484370e2"
  },
  "json": null,
  "origin": "124.50.108.25",
  "url": "https://httpbin.org/post"
}

+NWHTCSTATUS:0x00
```

PUT method request:

```
AT+NWHTC=https://httpbin.org/put,put,HTTP-Client PUT method sample test!
OK
HTTP/1.1 200 OK
Date: Tue, 07 Dec 2021 02:04:19 GMT
Content-Type: application/json
Content-Length: 584
Connection: keep-alive
Server: gunicorn/19.9.0
Access-Control-Allow-Origin: *
Access-Control-Allow-Credentials: true

{
  "args": {},
  "data": "HTTP-Client PUT method sample test!",
  "files": {},
  "form": {},
  "headers": {
    "Accept": "*/*",
    "Accept-Encoding": "identity",
    "Accept-Language": "ko-KR,Ko;q=0.9,en-US;q=0.8,en;q=0.7",
    "Content-Length": "35",
    "Host": "httpbin.org",
    "User-Agent": "Mozilla/5.0 (windows NT 6.1; Win64; x64) AppleWebkit/537.36 (KHTML, like Gecko)
Chrome/64.0.3282.186 Safari/537.36",
    "X-Amzn-Trace-Id": "Root=1-61aec123-4c3c5d390c6b31992bb803be"
  },
  "json": null,
  "origin": "124.50.108.25",
  "url": "https://httpbin.org/put"
}

+NWHTCSTATUS:0x00
```

5.6.3 HTTP-Server Commands

Table 24: HTTP-Server command list

Command	Parameters	Description
AT+NWHTS	<flag>	Start or stop the HTTP server depending on the option. <start>: 1 (start), 0 (stop) Response: OK or ERROR
	Prerequisite DA16200/DA16600 should be connected to AP. Example AT+NWHTS=1 OK Note: ▪ Enabled by default in the SDK	
AT+NWHTSS	<flag>	Start or stop the HTTPS server depending on the option. <start>: 1 (start), 0 (stop) Response: OK or ERROR
	Prerequisite DA16200/DA16600 should be connected to AP. Example AT+NWHTSS=1 OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later	

Command	Parameters	Description
	▪ If __SUPPORT_HTTP_SERVER_FOR_ATCMD__ is enabled in the SDK, this command will be enabled	

5.6.3.1 HTTP/HTTPS-Server Start Example

HTTP start:

```
AT+NWHTS=1
```

HTTPS start:

```
AT+NWHTSS=1
```

5.6.4 WebSocket-Client Commands

Table 25: WebSocket-Client command list

Command	Parameters	Description
AT+NWWSC	<operation>,<uri> (<msg>)	Start the WebSocket client with options. <operation> connect, send, or disconnect <uri>: WebSocket server address <msg>: Request message
	Prerequisite DA16200/DA16600 should be connected to AP. Example 1 AT+NWWSC=connect,ws://192.168.86.182:8080 +NWWSC:1 Example 2 AT+NWWSC=send,Send Message Test OK Example 3 AT+NWWSC=disconnect +NWWSC:0 Note: ▪ If __SUPPORT_WEBSOCKET_CLIENT__ is enabled in SDK, this command will be enabled	

Table 26: Web Socket-Client response list

Response	Parameters	Description
+NWWSC	<status>(,<opcode>,<received msg length>,<received msg>)	Return status along with the received payload. <status> 0 is disconnected. 1 is connected. <opcode> Continuation Frame : 0x00 Text Frame : 0x01 Binary Frame : 0x02 Close Frame : 0x08 Ping Frame : 0x09 Pong Frame : 0x0a Example 1: +NWWSC:1 Example 2: +NWWSC:0 Example 3: +NWWSC:1,1,12,Test Message

5.6.5 OTA Commands

Table 27: OTA command list

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td>AT+NWOTADWSTART</td><td><fw_type>,<uri>[,<fw_name>]</td><td>Start downloading firmware from an OTA server.

<fw_type>: Set the type of FW to be downloaded
<uri>: Server URL where a FW exists
<fw_name>: Optional. The maximum input size of fw_type is 7 bytes. MCU_FW will be stored by default if not specified. (Only for MCU FW)
Response: +NWOTADWSTART:0x00</td></tr>
<tr><td></td><td colspan="2">Prerequisite
DA16200/DA16600 should be connected to AP.

Example
; RTOS download
AT+NWOTADWSTART=rtos,https://server/DA16200_RTOS-GEN01-01-1111-000000.img
OK
+NWOTADWSTART:0x00

; Bluetooth® LE FW download (DA16600 only)
AT+NWOTADWSTART=ble_fw,https://server/ble_firmware.img
OK
+NWOTADWSTART:0x00

; MCU FW download
AT+NWOTADWSTART=mcu_fw,https://server/mcu_firmware.img
OK
+NWOTADWSTART:0x00

; MCU FW download (Enter the name of MCU FW within 8 characters. Default is "MCU_FW")
AT+NWOTADWSTART=mcu_fw,https://server/mcu_firmware.img,ver01
OK
+NWOTADWSTART:0x00

; Bluetooth® LE FW download
AT+NWOTADWSTART=ble_fw,https://server/pxr_sr_coex_ext_531_6_0_14_1114_2_ota.img
OK
+NWOTADWSTART:0x00

; Cert Key download:
AT+NWOTADWSTART=cert_key,https://server/ca.pem
OK
+NWOTADWSTART:0x00

Note:
▪ Enabled by default in the SDK
▪ The parameter, fw_type should be lowercase
▪ Bluetooth® LE FW download is available on SDK V3.2.3.0 or later</td></tr>
<tr><td rowspan="2">AT+NWOTARENEW</td><td>(none)</td><td>Reboot with updated FW</td></tr>
<tr><td colspan="2">Prerequisite</td></tr>
</table>

Command	Parameters	Description
	Download RTOS or Bluetooth® LE images. Example AT+NWOTARENEW +NWOTARENEW:0x00 Note: ▪ Enabled by default in the SDK ▪ Will reboot automatically after renewing is completed ▪ Bluetooth® LE image is supported in SDK V3.2.3.0 or later, either RTOS or Bluetooth® LE, or both of them can be supported	
AT+NWOTADWPROG	<fw_type>	FW download progress. <fw_type>: Set a firmware type among rtos, ble_fw, mcu_fw, or cert_key Response: +NWOTADWPROG:100
	Example ; RTOS download progress: AT+NWOTADWPROG=rtos +NWOTADWPROG:100 OK ; MCU FW download progress: AT+NWOTADWPROG=mcu_fw +NWOTADWPROG:100 OK ; Bluetooth® LE FW download progress (DA16600 only): AT+NWOTADWPROG=ble_fw +NWOTADWPROG:100 OK ; Cert Key download progress: AT+NWOTADWPROG=cert_key +NWOTADWPROG:100 OK Note: ▪ Enabled by default in the SDK	
AT+NWOTADWSTOP	(none)	Stop while downloading FW
	Example AT+NWOTADWSTOP OK	
AT+NWOTAFWNAME	(none)	Read a name in the header of the MCU firmware (Only for MCU FW)
	Example AT+NWOTAFWNAME +NWOTAFWNAME:MCU OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __OTA_UPDATE_MCU_FW__ is enabled in the SDK, this command will be enabled	
AT+NWOTAFWSIZE	(none)	Read a size in the header of the MCU firmware (Only for MCU FW)
	Example	

Command	Parameters	Description
	AT+NWOTAFWSIZE +NWOTAFWSIZE:4128 OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __OTA_UPDATE_MCU_FW__ is enabled in the SDK, this command will be enabled	
AT+NWOTAFWCRC	(none)	Read a CRC in the header of the MCU firmware (Only for MCU FW)
	Example AT+NWOTAFWCRC +NWOTAFWCRC:5aa8b6c4 OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __OTA_UPDATE_MCU_FW__ is enabled in the SDK, this command will be enabled	
AT+NWOTAREADFW	<read addr>, <read_size>	Read the MCU firmware as much as the *read_size* from the *read_addr* and transmit it (Only for MCU FW) <*read_addr*>: Hexadecimal without “0x” prefix <*read_size*>: Decimal MCU_FW default address - DA16200 : 0x003A_D000 - DA16600 : 0x003C_2000
	Example AT+NWOTAREADFW=3ad000,128 DA16FMCUÿÿÿÿ12345678901234561234567890123456123456789012345612345678901234561234567890123456123456789012345612345678901234561234567890123456 +NWOTAREADFW:COMPLETE OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __OTA_UPDATE_MCU_FW__ is enabled in the SDK, this command will be enabled	
AT+NWOTATRANSFW	(none)	Transmit the downloaded MCU firmware to the MCU. Transmission will be failed if no header (16 bytes) information exist (Only for MCU FW)
	Example AT+NWOTATRANSFW MCU DA16FMCUÿÿÿÿ12345678901234561234567890123456123456789012345612345678901234561234567890123456123456789012345612345678901234561234567890123456 …… +NWOTATRANSFW:COMPLETE OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __OTA_UPDATE_MCU_FW__ is enabled in the SDK, this command will be enabled	
AT+NWOTAERASEFW	(none)	Erase the MCU firmware stored in a serial flash of DA16200/DA16600. (Only for MCU FW)
	Example AT+NWOTAERASEFW +NWOTAERASEFW:COMPLETE	

Command	Parameters	Description
	OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __OTA_UPDATE_MCU_FW__ is enabled in the SDK, this command will be enabled	
AT+NWOTASETADDR	<sflash_addr>	Designate an address where data can be downloaded within the range of User Area and TLS Certificate Key in the SFLASH area. MCU_FW / CERT_KEY default address - DA16200 : 0x003A_D000 - DA16600 : 0x003C_2000 CERT_KEY should be copied to the operating area if downloaded to the user area, which is the default address. CERT_KEY area address - DA16200 / DA16600 : 0x003A_3000
	Example AT+NWOTASETADDR=3ad000 +NWOTASETADDR:0x00 OK Note: ▪ Enabled by default in the SDK	
AT+NWOTAGETADDR	<fw_type>	Return the value set with NWOTASETADDR. MCU_FW / CERT_KEY default address - DA16200 : 0x003A_D000 - DA16600 : 0x003C_2000
	Example AT+NWOTAGETADDR=mcu_fw +NWOTAGETADDR:3ad000 OK AT+NWOTAGETADDR=cert_key +NWOTAGETADDR:3ad000 OK Note: ▪ Enabled by default in the SDK	
AT+NWOTAREADFLASH	<sflash_addr>,<size>	Read as much as size from *sflash_addr*. MCU_FW default address - DA16200 : 0x003A_D000 - DA16600 : 0x003C_2000
	Example AT+NWOTAREADFLASH=3ad000,128 MCU_FW ?¨ZDA16FMCUÿÿÿÿ12345678901234561234567890123456123456789012345612345678901234561234567890123456123456789012345612345678901234561234567890123456 OK Note: ▪ Enabled by default in the SDK	
AT+NWOTAERASEFLASH	<sflash_addr>,<size>	Delete as much as size from sflash_addr. MCU_FW default address

Command	Parameters	Description
		- DA16200 : 0x003A_D000 - DA16600 : 0x003C_2000
	Example AT+NWOTAERASEFLASH=3ad000,1000 +NWOTAERASEFLASH:COMPLETE OK Note: ▪ Enabled by default in the SDK ▪ It will be erased in 4 kB increments	
AT+NWOTACOPYFLASH	<dest_sflash_addr>,<src_sflash_addr>,<size>	Copy as much as size from src_sflash_addr to dest_sflash_addr. MCU_FW default address - DA16200 : 0x003A_D000 - DA16600 : 0x003C_2000
	Example AT+NWOTACOPYFLASH=3c2000,3ad000,1000 +NWOTACOPYFLASH:COMPLETE OK Note: ▪ Enabled by default in the SDK ▪ It will be copied in 4 kB increments	
AT+NWOTATLSAUTH	<tls_auth_mode>	Set the certificate verification mode. #define MBEDTLS_SSL_VERIFY_NONE 0 #define MBEDTLS_SSL_VERIFY_OPTIONAL 1 #define MBEDTLS_SSL_VERIFY_REQUIRED 2
	Example AT+NWOTATLSAUTH=1 OK Note: ▪ Enabled by default in the SDK v3.2.5.0 or later	
AT+NWOTABYMCU	rtos,<full_size>	Transmit the RTOS stored in the MCU to the DA16200/DA16600 when there is no network access. If the AT+NWOTABYMCU= rtos,<full_size> command is OK, RTOS is transmitted as much as the partial size with tx_size=<partial_size>,<binary data>. Every transmission sends "OK" as a response unless there is an error. If an error occurs or transmission is complete, it responds with +NWOTABYMCU:0x00 including status. Note that only RTOS can be downloaded.
	Example AT+NWOTABYMCU=rtos,1335408 OK tx_size=4096,4643394BDA4F27840000000000000000… OK … tx_size=112,00000000000000000000000001000000… +NWOTABYMCU:0x00 Note: ▪ Enabled by default in the SDK v3.2.5.0 or later ▪ If __OTA_UPDATE_MCU_FW__ is enabled in the SDK, this command will be enabled	

NOTE
When DPM mode is enabled and OTA update is in progress (firmware download is in progress), it does not enter DPM sleep due to SFLASH write operation. After downloading the firmware, the DA16200 resumes to enter DPM sleep mode.

Table 28: OTA response list

Response	Parameters	Description
+NWOTADWSTART	<status>	Return the status of FW download. <status>: 0x00 is success. See Table 29 for other status value. For example: +NWOTADWSTART:0x00
+NWOTARENEW	<status>	Return the status for FW RENEW. <status>: 0x00 is success. See OTA Response Code List for others For example: +NWOTARENEW:0x00
+NWOTADWPROG	<progress>	Return the percentage value (%) of the FW download progress. <progress>: Print download progress in percent For example: +NWOTADWPROG:100
+NWOTADWSTOP	<status>	Return the status of FW download stop. <status>: 0x00 is success. See Table 29 for other status values For example: +NWOTADWSTOP:0x00
+NWOTATRANSFW	COMPLETE or FAIL	Return result of MCU FW transmission. (Only for MCU FW) For example: +NWOTATRANSFW:COMPLETE
+NWOTAFWNAME	<name>	String entered by a user. (Default is MCU_FW) Returns “(NULL)” if there is no MCU FW. (Only for MCU FW)
+NWOTAFWSIZE	<size>	Downloaded MCU FW size. It returns 0 if there is no MCU FW. (Only for MCU FW)
+NWOTAFWCRC	<crc>	Downloaded MCU FW CRC. It returns 0 if there is no MCU FW. (Only for MCU FW)
+NWOTAREADFW	COMPLETE or FAIL	Success: COMPLETE Failure: FAIL (Only for MCU FW)
+NWOTAERASEFW	COMPLETE or FAIL	Success: COMPLETE Failure: FAIL (Only for MCU FW)
+NWOTASETADDR	<status>	<status>: 0x00 is success See Table 29 for other status values.
+NWOTAGETADDR	<sflash_addr>	Return the value of sflash_addr.
(AT+NWOTAREADFLASH)	(Binary)	Return binary data as much as entered SFLASH address and size.
+NWOTAERASEFLASH	COMPLETE or FAIL	Success: COMPLETE Failure: FAIL
+NWOTACOPYFLASH	COMPLETE or FAIL	Success: COMPLETE Failure: FAIL
+NWOTABYMCU	<status>	<status>: 0x00 is success See Table 29 for other status values.

Table 29: OTA response code list

Return value	Description
0x00	Return success.

Return value	Description
0x01	Return fail.
0x02	SFLASH address is wrong.
0x03	FW type is unknown.
0x04	Server URL is unknown.
0x05	FW size is too big.
0x06	CRC is not correct.
0x07	FW version is unknown.
0x08	FW version is incompatible.
0x09	FW not found on the server.
0x0A	Failed to connect to the server.
0x0B	All new FWs have not been downloaded.
0x0C	Failed to allocate memory.
0xA1	Bluetooth® LE FW version is unknown.

5.6.5.1 OTA Download Example

```
RTOS download:
AT+NWOTADWSTART=rtos,https://server/DA16200_RTOS-GEN01-01-1111-000000.img
Bluetooth® LE FW download: (DA16600 only)
AT+NWOTADWSTART=ble_fw,https://server/ble_firmware.img
MCU FW download:
AT+NWOTADWSTART=mcu_fw,https://server/mcu_firmware.img
AT+NWOTADWSTART=mcu_fw,https://server/mcu_firmware.img,ver01
Cert Key download:
AT+NWOTADWSTART=cert_key,https://server/ca.pem
```

5.6.5.2 OTA Download Progress Example

```
RTOS download progress:
AT+NWOTADWPROG=rtos
Bluetooth® LE FW download progress: (DA16600 only)
AT+NWOTADWPROG=ble_fw
MCU FW download progress:
AT+NWOTADWPROG=mcu_fw
Cert Key download progress:
AT+NWOTADWPROG=cert_key
```

5.6.5.3 OTA Renew Example

```
Renew Firmware (reboot with updated FW):
AT+NWOTARENEW
```

5.6.5.4 MCU FW Transport Example

```
MCU FW transmission:
AT+NWOTATRANSFW
Get MCU FW name:
AT+NWOTAFWNAME
Get MCU FW size:
AT+NWOTAFWSIZE
Get MCU FW CRC:
AT+NWOTAFWCRC
Read MCU FW as much as specified size:
AT+NWOTAREADFW=3ad000,128
Delete MCU FW stored in the DA16200/DA16600 SFLASH:
AT+NWOTAERASEFW
```

5.6.5.5 SFLASH User Area Address Setting Example

```
SET ADDR:
AT+NWOTASETADDR=3ad000
GET ADDR:
AT+NWOTAGETADDR=mcu_fw
AT+NWOTAGETADDR=cert_key
```

5.6.5.6 SFLASH READ/COPY/ERASE Example

```
SFLASH Read:
AT+NWOTAREADFLASH=3ad000,128
SFLASH Copy:
AT+NWOTACOPYFLASH=3ad000,0x3c2000,128
SFLASH Erase:
AT+NWOTAERASEFLASH=0x1f2000,128
```

5.6.5.7 TLS Certificate Verification Mode Setting Example

```
SET  MBEDTLS_SSL_VERIFY_NONE:
AT+NWOTATLSAUTH=0
SET  MBEDTLS_SSL_VERIFY_OPTIONAL:
AT+NWOTATLSAUTH=1
SET  MBEDTLS_SSL_VERIFY_REQUIRED:
AT+NWOTATLSAUTH=2
```

5.6.5.8 RTOS by MCU Download Example

```
Initialization:
AT+NWOTABYMCU=rtos,1335408
Data transmission:
tx_size=4096,4643394BDA4F27840000000000000000000…
```

5.6.6 Zeroconf Commands

Table 30: Zerocont command list

Command	Parameters	Description
AT+NWMDNSSTART	<Mode>	Start the mDNS module. The mDNS module is communicated through multicast. DA16200/DA16600 could frequently be changed from/to DPM sleep and wake-up states. It may consume more power. <Mode>: The mode in which the WLAN interface is running, 0 (Station) or1 (Soft AP).
	Prerequisite DA16200/DA16600 should be connected to AP. The host name of mDNS module should be set up. Example AT+NWMDNSSTART=1 OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __SUPPORT_ZERO_CONFIG__ is enabled in the SDK, this command will be enabled.	
	?	Get the string representing the status of mDNS module, “started” or “not started”.
	Example AT+NWMDNSSTART=? +NWMDNSSTART:started	
AT+NWMDNSHNREG	<Host name>	Register the host name in the mDNS module. mDNS supports one configured host name only, to change or set a new mDNS host name. mDNS service must be stopped and started again.

Command	Parameters	Description
		<Host name>: The name of the host to be registered.
	Example AT+NWMDNSHNREG=da16x OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __SUPPORT_ZERO_CONFIG__ is enabled in the SDK, this command will be enabled.	
AT+NWMDNSSRVREG	<Instance name>,<Protocol>,<Port>[,<Text record>]	Register a service in the mDNS module. <Instance name>: The instance name of service to be registered. <Protocol>: The protocol and the type of the service to be registered. <Port>: The port number of the service to be registered. <Text record>: The text record of the service that must be registered and mentioned in “Key=Value” format. Multiple pairs of text records should be separated using a “,”.
	Prerequisite DA16200/DA16600 should be connected to AP. The host name of mDNS module should be set up. Example AT+NWMDNSSRVREG=_WEBAPP,_http,_tcp,80,LIGHT=OFF,FAN=ON OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __SUPPORT_ZERO_CONFIG__ is enabled in the SDK, this command will be enabled.	
AT+NWMDNSSUPDATETXT	<Text record>	Update the text record of a service in the mDNS module. <Text record>: The text record of the service to be updated.
	Example AT+NWMDNSUPDATETXT=LIGHT=OFF,FAN=ON OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __SUPPORT_ZERO_CONFIG__ is enabled in the SDK, this command will be enabled.	
AT+NWMDNSSRVDEREG	(None)	Unregister a service in the mDNS module. Response: OK or ERROR For example: AT+NWMDNSSRVDEREG
	Prerequisite DA16200/DA16600 should be connected to AP. The mDNS module should be running. The service in the mDNS module should be registered. Example AT+NWMDNSSRVDEREG OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __SUPPORT_ZERO_CONFIG__ is enabled in the SDK, this command will be enabled.	

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td rowspan="4">AT+NWMDNSSTOP</td><td>(None)</td><td>Stop the mDNS module.</td></tr>
<tr><td colspan="2">Prerequisite
DA16200/DA16600 should be connected to an AP.
The mDNS module should be running.

Example
AT+NWMDNSSTOP
OK

Note:
▪ Enabled by default in the SDK v3.2.3.0 or later
▪ If __SUPPORT_ZERO_CONFIG__ is enabled in the SDK, this command will be enabled.</td></tr>
<tr><td>?</td><td>Get the string representing the status of the mDNS module, “stopped” or “running”.</td></tr>
<tr><td colspan="2">Example
AT+NWMDNSSTOP
+NWMDNSSTOP:stopped</td></tr>
</table>

5.6.6.1 Zeroconf Example

Configure the parameters and start the mDNS module (After Wi-Fi Connection):

AT+NWMDNSHNREG=da16x

AT+NWMDNSSTART=0

If the mDNS module is started successfully, the following response is shown:

OK

Register a service in the mDNS module:

AT+NWMDNSSRVREG=_WEBAPP,_http._tcp,80,LIGHT=OFF,FAN=ON

If the service is registered successfully, the following response is shown:

OK

Registered service and host name can be discovered by other mDNS services. In this example, Bonjour service (https://support.apple.com/kb/DL999?viewlocale=en_US&locale=zh_TW) on Windows is used to discover them.

To discover DA16200/DA16600’s mDNS, the “-G” option can be used like the following:

```
C:> dns-sd -G v4 da16x.local

Timestamp     A/R Flags if Hostname                Address                                      TTL

18:04:04.474  Add     2 24 da16x.local.             192.168.0.4                                  120
```

To discover the service, the “-L” option can be used like the following:

```
C:>dns-sd -L _WEBAPP _http._tcp local
Lookup _WEBAPP._http._tcp.local
18:04:29.453  _WEBAPP._http._tcp.local. can be reached at da16x.local.:80 (interface 24)
 LIGHT=OFF FAN=ON
```

5.6.7 Provision Commands Over Wi-Fi

The provision commands over Wi-Fi are used for starting to provision procedure and getting provisioning status. To use these commands, __PROVISION_ATCMD__ feature should be enabled in SDK.

Table 31: Provision command list

Command	Parameters	Description
AT+PROVSTART	(none)	Removed all profile data in NVRAM and configure appropriate profile such as Soft AP and concurrent profile, and perform SW reboot. Response: "+INIT:DONE,2"
	Example AT+PROVSTART +INIT:DONE,2 Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __PROVISION_ATCMD__ is enabled in the SDK, this command will be enabled ▪ After restarting, the system will be ready for the provisioning procedure in concurrent mode for the SDK v3.2.6.0 or later, and Soft AP mode for SDK v3.2.4.0 and former versions (the response is “INIT:DONE,1”) ▪ This command supports Wi-Fi based provisioning. See Ref. [5] for provisioning over Bluetooth® LE	
AT+PROVSTAT	(none)	Get status of provisioning. Response: OK or ERROR
	Example AT+PROVSTAT +ATPROV=STATUS 1 OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __PROVISION_ATCMD__ is enabled in the SDK, this command will be enabled ▪ The list of provision status can be found in the thread_atcmd.h. Check the atcmd_provision_stat enumeration	

Table 32: atcmd_provision_stat enumeration

Value	Name	Description
0	ATCMD_PROVISION_IDLE	Not run or finish
1	ATCMD_PROVISION_START	Start
101	ATCMD_PROVISION_SELECTED_AP_SUCCESS	Receive AP information
102	ATCMD_PROVISION_SELECTED_AP_FAIL	
103	ATCMD_PROVISION_WORNG_PW	AP connection fail by the wrong PW
104	ATCMD_PROVISION_NETWORK_INFO	Get Network info. from Mobile App
105	ATCMD_PROVISION_AP_FAIL	AP connection fail
106	ATCMD_PROVISION_DNS_FAIL_SERVER_FAIL	Network connection check
107	ATCMD_PROVISION_DNS_FAIL_SERVER_OK	
108	ATCMD_PROVISION_NO_URL_PING_FAIL	
109	ATCMD_PROVISION_NO_URL_PING_OK	
110	ATCMD_PROVISION_DNS_OK_PING_FAIL_N_SERVER_OK	

Value	Name	Description
113	ATCMD_PROVISION_DNS_OK_PING_N_SERVER_FAIL	
111	ATCMD_PROVISION_DNS_OK_PING_OK	
112	ATCMD_PROVISION_REBOOT_ACK	Reboot after provisioning

While provisioning over Wi-Fi with the mobile application as described in Ref. [5], DA16200 provides responses to MCU regarding provisioning status. See the following examples.

```
+ATPROV=STATUS 0
+ATPROV=STATUS 1
+WFCST:<MAC Address>
+ATPROV=STATUS 101
+WFDST:<MAC Address>
+WFJAP:1,'<SSID>',<IP Address>
```

For DA16600, the following responses are provided to MCU regarding provisioning status while provisioning over Bluetooth® LE . See the following examples.

```
+ATPROV=STATUS 104
+ATPROV=STATUS 101
+WFJAP:1,'<SSID>',<IP Address>
+ATPROV=STATUS 111
+ATPROV=STATUS 112
+INIT:DONE,0
```

NOTE
See Table 12 for details of +WFCST, +WFDST, and +WFJAP response.

5.6.8 Bluetooth® LE Commands

The Bluetooth® LE commands are available when the DA16600 is running the Bluetooth enabled version of the firmware. Select and build the DA16600 project from the SDK to use these commands.

Table 33: Bluetooth® LE command list

Command	Parameters	Description
AT+BLENAME	<device name>	Change the device name of the DA16600/Bluetooth® LE device. Response: OK or ERROR
	?	Get the device name of the DA16600/Bluetooth® LE device. Response: +BLENAME:<device name>
	(none)	
	Example AT+BLENAME=DA16600-BLE OK AT+BLENAME +BLENAME:DA16600-BLE OK AT+BLENAME=? +BLENAME:DA16600-BLE Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ Enabled getting the device name by default in the SDK v3.2.5.0 or later	
AT+ADVSTOP	(none)	Stop advertising of DA16600/Bluetooth® LE device. Response: OK or ERROR
	Example AT+ADVSTOP OK Note:	

Command	Parameters	Description
	▪ Enabled by default in the SDK v3.2.3.0 or later	
AT+ADVSTART	(none)	Start advertising of DA16600 / Bluetooth® LE device. Response: OK or ERROR
	Example AT+ADVSTART OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later	

5.6.9 Transfer Function Commands

5.6.9.1 Socket Commands

Table 34: Socket command list

Command	Parameters	Description
	<local_port>[,<max allowed connection>]	Open a TCP server socket. <local_port>: Local port number of the socket. <max allowed connection>: It is optional. Set max allowed TCP session. Response: OK (with '+TRCTS:*** ' See Table 35 or ERROR. Async message: CID(+TRTS:<Assigned CID>)
	Prerequisite DA16200/DA16600 should be connected to AP. Example ; Open first TCP server AT+TRTS=10194 +TRTS:0 OK ; Open second TCP server AT+TRTS=10195 +TRTS:3 OK Note: ▪ Enabled by default in the SDK ▪ CID number 0 (TCP server), 1 (TCP client), and 2 (UDP) are pre-assigned numbers. New CID started from 3 is assigned on the order of opening TCP session ▪ The <max allowed connection> parameter is supported by SDK v3.2.5.0 or later ▪ Multiple TCP server and async message with assigned CID are supported by SDK v3.2.5.0 or later. A total of eight sessions can be created for transfer function. But it depends on DA16200/DA16600 SDK configuration. Basically, DA16200/DA16600 can be created eight TCP sockets	
AT+TRTC	<server_ip>, <server_port>[, <local_port>]	Open a TCP client socket and connect to a TCP server. <server_ip>: IP address of TCP server to be accessed <server_port>: Port number of TCP server <local_port>: Local port number of the socket (optional, 0: auto) Response: OK or ERROR Async message: CID(+TRTC:<Assigned CID>)

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td colspan="2">Prerequisite
DA16200/DA16600 should be connected to AP.

Example
AT+TRTC=192.168.0.18,1025,1024
+TRTC:1
OK

AT+TRTC=192.168.0.18,1025,1025
+TRTC:3
OK

Note:
▪ Enabled by default in the SDK
▪ CID number 0 (TCP server), 1 (TCP client), and 2 (UDP) are pre-assigned numbers. New CID started from 3 is assigned on the order of opening TCP session.
▪ Multiple TCP client and async message with assigned CID are able to be supported by SDK v3.2.5.0 or later. Total eight sessions can be created for transfer function. But it depends on DA16200/DA16600 SDK configuration. Basically, DA16200/DA16600 can be created eight TCP sockets</td></tr>
<tr><td rowspan="2">AT+TRUSE</td><td><local_port></td><td>Open a UDP socket.

<local_port>: Local port number of the socket
Response: OK or ERROR
Async message: CID(+TRUSE:<Assigned CID>)</td></tr>
<tr><td colspan="2">Example
AT+TRTALL (optional, run this first if ‘ERROR’ is responded)
AT+TRUSE=10195
+TRUSE:2
OK

AT+TRUSE=10196
+TRUSE:3
OK

Note:
▪ Enabled by default in the SDK
▪ CID number 0 (TCP server), 1 (TCP client), and 2 (UDP) are pre-assigned numbers. New CID started from 3 is assigned on the order of opening TCP session
▪ Multiple UDP sockets and async message with assigned CID are able to be supported by SDK v3.2.5.0 or later. A total of eight sessions can be created for transfer function. But it depends on DA16200/DA16600 SDK configuration. Basically, DA16200/DA16600 can be created eight UDP sockets</td></tr>
<tr><td rowspan="2">AT+TRUR</td><td><remote_ip>,
<remote_port></td><td>Set remote IP and port of the UDP socket.

<remote_ip>: Remote IP address
<remote_port>: Remote port number
Response: OK or ERROR
Async message: CID(+TRUR:2)</td></tr>
<tr><td colspan="2">Example
AT+TRUR=192.168.0.18,1027
+TRUR:2
OK</td></tr>
</table>

Command	Parameters	Description
	Note: ▪ Enabled by default in the SDK ▪ It is only for the CID 2	
AT+TRPRT	<cid>	Get session information by CID. <cid>: Assigned CID Response: <cid>,[TCP|UDP],<remote_ip>,<remote_port>, <local_port>
	Prerequisite A UDP socket should be opened (AT+TRUSE). Example AT+TRPRT=2 +TRPRT:2,UDP,192.168.0.18,10194,10195 OK Note: ▪ Enabled by default in the SDK ▪ CID number 0 (TCP server), 1 (TCP client), and 2 (UDP) are pre-assigned numbers. New CID started from 3 is assigned on the order of opening TCP session	
AT+TRPALL	(none)	Get all session information. Response: <cid>,[TCP|UDP],<remote_ip>,<remote_port>, <local_port><LF>…
	Prerequisite The target system should be connected to any UDP or TCP server. Example AT+TRPALL +TRPALL:2,UDP,192.168.0.18,10194,10195 OK Note: ▪ Enabled by default in the SDK	
AT+TRTRM	<cid> [,<remote_ip> ,<remote_port>]	Close (terminate) a session by CID. If CID is 0 (TCP server), remote_ip, and remote_port are input, the session with the specific remote will be closed. <cid>: Assigned CID <remote_ip>: Remote IP address connected to TCP server. It is only allowed in TCP server <remote_port>: Remote port number connected to TCP server. It is only allowed in TCP server Response: OK or ERROR
	Prerequisite The target system should be connected to any UDP or TCP server. Example AT+TRTRM=2 OK AT+TRTRM=0,192.168.0.18,10194	

Command	Parameters	Description
	OK Note: ▪ Enabled by default in the SDK ▪ The remote_ip and remote_port parameters are only supported in SDK v3.2.3.0 or later	
AT+TRTALL	(none)	Close (terminate) all sessions. Response: OK or ERROR
	Example AT+TRTALL OK Note: ▪ Enabled by default in the SDK	
AT+TRSAVE	(none)	Save status of all sessions to NVRAM. Response: OK or ERROR
	Example AT+TRSAVE OK Note: ▪ Enabled by default in the SDK	
AT+TCPDATAMODE	<mode>	Set the mode of the received TCP data. <mode> 0: text mode (default) 1: hex string mode In text mode, data is returned as an ascii string: “0123ABCD” In hex mode, the data is returned as a hex encoded text string “3031323341424344” Response: OK or ERROR
	Example AT+TCPDATAMODE=1 OK Note: ▪ Enabled by default in the SDK v3.2.2.1 or later	

Table 35: Socket connection response list

Response	Parameters	Description
+TRCTS	<cid>, <remote_ip>, <remote_port>	When sending the AT command (AT+TRTS=40000), receive this response if there is no error. <cid>: Assigned CID for TCP server <remote_ip>: TCP client IP address <remote_port>: TCP client port number
	Example +TRCTS:0,192.168.0.18,41014	
+TRXTS	<cid>, <remote_ip>, <remote_port>	A remote TCP client is disconnected from the TCP server that was opened by AT+TRTS <cid>: Assigned CID for TCP server <remote_ip>: TCP client IP address <remote_port>: TCP client port number
	Example	

Response	Parameters	Description
	; When a remote peer is disconnected +TRXTS:0,192.168.0.18,41014	
+TRXTC	<cid>, <remote_ip>, <remote_port>	The TCP client socket that was opened by AT+TRTC is disconnected. <cid>: Assigned CID for TCP client <remote_ip>: TCP server IP address <remote_port>: TCP server port number
	Example ; When the TCP client socket is disconnected +TRXTC:1,192.168.0.18,1025	

5.6.9.1.1 Data Transfer Commands

Table 36: Data transmission command

Escape sequence	Parameters	Description
<ESC>S	<cid><length>, <remote_ip >, <remote_port >,[<mode>,]<data>	Transmit data through a socket with the CID specified. <ESC>S: To enter data input mode, type in <ESC> (0x1B) and S keys together <cid>: Assigned CID <length>: Data length. Data length can be 0 in only text mode. If this is 0, data is read until "\r" or "\n" is met. In raw mode, data is read until the length <remote_ip>: Remote IP address <remote_port>: Remote port number ▪ For TCP Server, <remote_ip> and <remote_port> of a TCP Client should be given. Maximum four TCP Clients can be connected to the TCP Server ▪ For TCP Client, 0, 0 is given (as the destination is the server) ▪ For UDP: if 0,0 is given, the data is sent to the destination that AT+TRUR has specified. if non-0 <remote_ip> and <remote_port> are given, UDP temporarily sends to the destination <remote_ip> and <remote_port> specifies <mode>: Mode to transmit data in raw or text mode. It is optional. If there is no option, data will be transmitted in text mode. This option is allowed only for UART communication ▪ r: The raw mode is active. In raw mode, Data is read until data length. The data length is specified in <length> parameter ▪ t: The text mode is active. In text mode, the data can be affected if it has unprintable control codes like backspace(0x08) Response: OK or ERROR
	Prerequisite The target system should be connected to any UDP or TCP server/client. Example1 – To send data to TCP client <ESC>S010,192.168.0.18,43110,abcde12345 OK Example2 – To send data to TCP server <ESC>S110,192.168.0.18,1025,abcde12345 OK Example3 – To send data to TCP server with ‘0, 0’ as the destination/server <ESC>S110,0,0,abcde12345 OK Example4 – To send data to UDP receiver	

<table>
<tr><th>Escape sequence</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td colspan="2"><ESC>S210,192.168.0.18,1024,abcde12345
OK

Note:
▪ Enabled by default in the SDK
▪ The maximum length of data depends on TX_PAYLOAD_MAX_SIZE definition. It is defined in atcmd.h for SDK v3.x. TX_PAYLOAD_MAX_SIZE includes all parameters of AT command. Therefore, the maximum length of 'length' parameter depends on length of other parameters
▪ TX_PAYLOAD_MAX_SIZE is defined 4,096 bytes in the SDK v3.2.3.0 or later
▪ For ATCMD over SPI or SDIO:
▪ The result for this command is sent to the host as the “response” field of “struct _st_host_response”. The “response” field is a 1-byte decimal value. A value of 0x20 is a result of “OK”. All other values are an “ERROR”
▪ Recommended to use <ESC>H command if UART baud rate is over 230400 bps. There might be data loss during DA16200/DA16600 parses this AT command</td></tr>
<tr><td rowspan="2"><ESC>M</td><td><cid>,<length>,
<remote_ip >,
<remote_port >,[<mode>,]<data></td><td>Transmit data through a socket with the CID specified.

<ESC>M: To enter data input mode, type in <ESC>(0x1B) and M key together
<cid>: Assigned CID
<length>: Data length. Data length can be 0 in only text mode. If this is 0, data is read until "\r" or "\n" is met. In raw mode, data is read until the length
<remote_ip>: Remote IP address
<remote_port>: Remote port number
▪ For TCP Server, <remote_ip> and <remote_port> of a TCP Client should be given
▪ For TCP Client, 0, 0 is given (as the destination is the server)
▪ For UDP: if 0,0 is given, the data is sent to the destination that AT+TRUR has specified. if non-0 <remote_ip> and <remote_port> are given, UDP temporarily sends to the destination <remote_ip> and <remote_port> specifies
Response: OK or ERROR</td></tr>
<tr><td colspan="2">Prerequisite
The target system should be connected to any UDP or TCP server/client.

Example1 – To send data to TCP client
<ESC>M0,10,192.168.0.18,43110,abcde12345
OK
Example2 – To send data to TCP server
<ESC>M1,10,192.168.0.18,1025,abcde12345
OK
Example3 – To send data to TCP server with ‘0, 0’ as the destination/server
<ESC>M1,10,0,0,abcde12345
OK
Example4 – To send data to UDP receiver
<ESC>M2,10,192.168.0.18,1024,abcde12345
OK

Note:
▪ Enabled by default in the SDK</td></tr>
</table>

<table>
<tr><th>Escape sequence</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td colspan="2">▪ The maximum length of data depends on TX_PAYLOAD_MAX_SIZE definition. It is defined in atcmd.h for SDK v3.x, TX_PAYLOAD_MAX_SIZE includes all parameters of AT command. Therefore, the maximum length of 'length' parameter depends on length of other parameters
▪ TX_PAYLOAD_MAX_SIZE is defined 4,096 bytes in the SDK v3.2.3.0 or later
▪ For ATCMD over SPI or SDIO:
The result for this command is sent to the host as the “response” field of “struct _st_host_response”. The “response” field is a 1-byte decimal value. A value of 0x20 is a result of “OK”. All other values are an “ERROR”
▪ Recommended to use <ESC>H command if UART baud rate is over 230400 bps. There might be data loss during DA16200/DA16600 parses this AT command</td></tr>
<tr><td rowspan="2"><ESC>H</td><td><cid>,<length>,
<remote_ip >,
<remote_port ></td><td>Transmit data through a socket with the CID specified. The host must send data after getting a response OK.

<ESC>H: To enter data input mode, type in <ESC>(0x1B) and H key together
<cid>: Assigned CID
<length>: Data length. Data is read until the length after getting OK response
<remote_ip>: Remote IP address
<remote_port>: Remote port number
▪ For TCP Server, <remote_ip> and <remote_port> of a TCP Client should be given
▪ For TCP Client, 0, 0 is given (as the destination is the server)
▪ For UDP: if 0,0 is given, the data is sent to the destination that AT+TRUR has specified. if non-0 <remote_ip> and <remote_port> are given, UDP temporarily sends to the destination <remote_ip> and <remote_port> specifies
Response: OK or ERROR</td></tr>
<tr><td colspan="2">Prerequisite
The target system should be connected to any UDP or TCP server/client.

Example1 – To send data to TCP client
<ESC>H0,10,192.168.0.18,43110
OK
abcde12345
OK
Example2 – To send data to TCP server
<ESC>H1,10,192.168.0.18,1025
OK
abcde12345
OK
Example3 – To send data to TCP server with ‘0, 0’ as the destination/server
<ESC>H1,10,0,0
OK
abcde12345
OK
Example4 – To send data to UDP receiver
<ESC>H2,10,192.168.0.18,1024
OK
abcde12345
OK

Note:</td></tr>
</table>

Escape sequence	Parameters	Description
	▪ Enabled by default in the SDK ▪ The maximum length of data depends on TX_PAYLOAD_MAX_SIZE definition. It is defined in atcmd.h for SDK v3.x, TX_PAYLOAD_MAX_SIZE includes all parameters of AT command. Therefore, the maximum length of 'length' parameter depends on length of other parameters ▪ TX_PAYLOAD_MAX_SIZE is defined 4,096 bytes in the SDK v3.2.3.0 or later ▪ Not supported over SPI or SDIO	

Table 37: Data reception responses

Response	Parameters	Description
+TRDTS	<cid>, <src_ip>,<src_port>, <length>,<data>	Receive data through TCP server socket. <cid>: Assigned CID <src_ip>: Source IP address <src_port>: Source port number <length>: Data length <data>: Received data
	Example ; When data is sent from a TCP client +TRDTC:1,192.168.0.18,1025,4,test	
+TRDTC	<cid>, <src_ip>,<src_port>, <length>,<data>	Receive data through TCP client socket. <cid>: Assigned CID <src_ip>: Source IP address <src_port>: Source port number <length>: Data length <data>: Received data
	Example ; When TCP client receives data, +TRDTC:1,192.168.0.18,1025,4,test	
+TRDUS	<cid>, <src_ip>,<src_port>, <length>,<data>	Receive data through UDP socket. <cid>: Assigned CID <src_ip>: Source IP address <src_port>: Source port number <length>: Data length <data>: Received data
	Example ; When UDP session receives data, +TRDUS:2,192.168.0.18,10194,4,test	

5.6.9.1.2 Data Transfer with DPM

TCP Server

After a connection to an AP is made in the normal BOOT state, open a TCP server socket, and save the config to NVRAM.

```
AT+TRTS=32000
AT+TRSAVE
```

The TCP server socket that has been opened should be closed before switching to DPM mode.

```
AT+TRTRM=0
```

Change the DA16200/DA16600 state to DPM mode (AT+DPM=1). When the DA16200/DA16600 starts the session on DPM mode successfully, the following is shown:

```
+INIT:DONE,0
+WFJAP:1,'WI-FI_AP',192.168.5.19
+TRPALL:0,TCP,0.0.0.0,0,32000
```

When a TCP client connects to DA16200/DA16600, the following is shown:

+INIT:WAKEUP,UC

To receive +TRCTS message, send AT+MCUWUDONE immediately after “+INIT:WAKEUP,UC”

+TRCTS:0,192.168.0.1,42000

When the DA16200/DA16600 receives a message from a client, the following is shown:

+INIT:WAKEUP,UC

+TRDTS:0,192.168.0.1,42000,10,1234567890

To send a TCP message, send AT+MCUWUDONE immediately after “external wake-up” is triggered (+INIT:WAKEUP,EXT). To prevent that DA16200/DA16600 enters DPM Sleep mode, MCU should send AT+CLRDPMSLPEXT before a message is sent. The DA16200/DA16600 can send data to a TCP client with the command “<ESC>S”. Finally, to enter DPM sleep mode, send “AT+SETDPMSLTEXT”.

```
+INIT:WAKEUP,EXT  // external wake-up
AT+MCUWUDONE
AT+CLRDPMSLPEXT
…
<ESC>S…
…
AT+SETDPMSLPEXT
```

When a TCP client disconnects from DA16200/DA16600, the following is shown:

+INIT:WAKEUP,UC

To receive +RTXTS message, send AT+MCUWUDONE immediately after “+INIT:WAKEUP,UC”

+TRXTS:0,192.168.0.1,42000

TCP Client

After a connection is made to an AP in the normal BOOT state, connect the TCP client of the DA16200/DA16600 to a TCP server and save the config to NVRAM. (To save TCP client config information, the DA16200/DA16600 should connect to the server successfully beforehand.)

AT+TRTC=192.168.5.1,34000

AT+TRSAVE

Before switching to DPM mode, disconnect the TCP Client:

AT+TRTRM=1

Change the DA16200/DA16600 state to DPM mode (AT+DPM=1). When the DA16200/DA16600 starts the session on DPM mode successfully, the following is shown:

+INIT:DONE,0

+WFJAP:1,'WI-FI_AP',192.168.5.19

+TRPALL:1,TCP,192.168.5.1,34000,30000

The procedure to exchange TCP data is the same as in Section 0. When the DA16200/DA16600 receives a message from the server, the following is shown:

+INIT:WAKEUP,UC

+TRDTC:1,192.168.5.1,34000,10,1234567890

UDP Session

After a connection is made to an AP in the normal BOOT state, open a UDP socket and save the config to NVRAM:

AT+TRUSE=48000

AT+TRSAVE

Before switching to DPM mode, disconnect TCP Client:

AT+TRTRM=2

Change the DA16200/DA16600 state to DPM mode. When the DA16200/DA16600 starts the session in DPM mode successfully, the following is shown:

+INIT:DONE,0

+WFJAP:1,'WI-FI_AP',192.168.5.19

+TRPALL:2,UDP,0.0.0.0,0,48000

The procedure to exchange UDP data is the same as in Section 0. When the DA16200/DA16600 receives a message from the server, the following is shown:

+INIT:WAKEUP,UC

+TRDUS:2,192.168.5.23,35000,10,1234567890

5.6.9.2 Secure Socket Commands

Table 38: Secure socket command list

Command	Parameters	Description
AT+TRSSLINIT	<Role>	Initialize the SSL module. DA16200/DA16600 allows to create a module of TLS client. <Role>: The role of SSL, 1 – Client
	Example AT+TRSSLINIT=1 +TRSSLINIT:0 Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __SUPPORT_ATCMD_TLS__ is enabled in the SDK, this command will be enabled	
AT+TRSSLCFG	<CID>,<Configuration ID>,<Configuration value>	Configure SSL connection. <CID>: The CID obtained after issuing the AT+TRSSLINIT command <Configuration ID>: The configuration ID available in the below list of configurations: ▪ 0 - Invalid configuration parameter ▪ 1 – To ser SSL Protocol Version ▪ 2 - To set SSL CA Certificate ▪ 3 - To set SSL Certificate ▪ 6 - To set the SNI ▪ 9 - To enable/disable server validation ▪ 10 - To set the Incoming buffer length ▪ 11 - To set the Outgoing buffer length <Configuration value>: Value to the configuration provided in configuration ID CONF_ID:CONF_VAL ▪ 0 - Invalid ▪ 1 – SSL Protocol Version • 0: TLS Version 1.2 (Default) • 1: TLS Version 1.3 • 2: Compatibility Mode (TLS 1.2/1.3) ▪ 2 - SSL CA Certificate Name ▪ 3 - SSL Certificate Name ▪ 6 - To Set the SNI (supported only for TLS client) ▪ 9 - To enable/disable server validation • 0: Disables server validation (Default) • 1: Enables server validation ▪ 10 - To set the Incoming buffer length ▪ 11 - To set the outgoing buffer length
	Prerequisite CID should be obtained (AT+TRSSLINT). SSL CA certificate and SSL certificate should be set up. Example AT+TRSSLCFG=0,2,CA_CERT OK AT+TRSSLCFG=0,3,CERT OK AT+TRSSLCFG=0,6,da16x OK	

Command	Parameters	Description
	AT+TRSSLCFG=0,9,0 OK AT+TRSSLCFG=0,10,6144 OK AT+TRSSLCFG=0,11,6144 OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __SUPPORT_ATCMD_TLS__ is enabled in the SDK, this command will be enabled	
AT+TRSSLCO	<CID>,<Server IP Address>,<Server Port number>	Connect to an SSL server. <CID>: The CID obtained after issuing the AT+TRSSLINIT command <Server IP Address>: The IP Address of the server to connect. Only supported IPv4 address <Server Port>: The port number of the SSL server to connect
	Prerequisite CID should be obtained (AT+TRSSLINT). Example AT+TRSSLCO=0,192.168.0.11,30000 OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __SUPPORT_ATCMD_TLS__ is enabled in the SDK, this command will be enabled	
AT+TRSSLWR	<CID>,[<Server IP Address>,<Server Port number>,<mode>,]<Data length>,<Data>	Send the data to the SSL server that is already established. <CID>: The CID obtained after issuing the AT+TRSSLINIT command <Server IP Address>: The IP Address of the SSL server is already established. If no input, the IP address would internally be used to the SSL server IP address <Server Port number>: The port number of the SSL server is already established. If no input, the Port number is internally used to the SSL server port number <mode>: Transmit data in raw or text mode. It is optional. If there is no option, data will be transmitted in text mode ▪ r: The raw mode is active. In raw mode, Data is read until data length. The data length is specified in <length> parameter ▪ t: The text mode is active. In text mode, the data can be affected if it has unprintable control codes like backspace(0x08) <Data length>: The length of data to send <Data>: The data to send. The input can be closed by <Ctrl>+C or reaching data length
	Prerequisite CID should be obtained (AT+TRSSLINT). Example AT+TRSSLWR=0,10,0123456789<Ctrl> + C OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later	

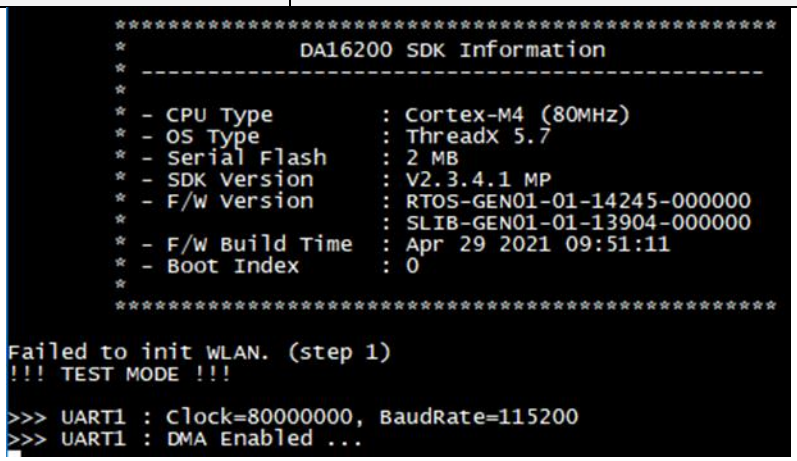
Command	Parameters	Description
	▪ If __SUPPORT_ATCMD_TLS__ is enabled in the SDK, this command will be enabled ▪ The maximum length of data depends on TX_PAYLOAD_MAX_SIZE definition. It is defined in atcmd.h for SDK v3.x. TX_PAYLOAD_MAX_SIZE includes all parameters of AT command. Therefore, the maximum length of 'length' parameter depends on length of other parameters	
AT+TRSSLCL	<CID>	Close the SSL connection. <CID>: The CID obtained after issuing AT+TRSSLINIT command
	Prerequisite CID should be obtained (AT+TRSSLINT). Example AT+TRSSLCL=0 OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __SUPPORT_ATCMD_TLS__ is enabled in the SDK, this command will be enabled	
AT+TRSSLCERT LIST	<Certificate Type>	Show a list of certificates or a list of CA data available in sflash memory. <Certificate Type>: The value of the certificate. 0 – CA Certificates, 1 – Client/Server Certificates
	Example AT+TRSSLCERTLIST=0 +TRSSLCERTLIST=0,CA_CERT Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __SUPPORT_ATCMD_TLS__ is enabled in the SDK, this command will be enabled	
AT+TRSSLCERT STORE	<Certificate Type>, <Sequence>, <Format>, <Name>, [<Data length>,]<Data>	Store a certificate and CA list data in sflash memory. <Certificate Type>: The value of the certificate. 0 – CA Certificates, 1 – Client/Server Certificates <Sequence>: If the value of certificate type is 0 (CA), the number of certificates in the sequence is 1-5. If the certificate type is 1 (Client/Server certificate), then several certificates in a sequence is 1-SSL cert or 2-SSL key <Format>: The value of the CA/Certificate/Key0 – DER, 1 – PEM <Name>: The name of the certificate. While loading certificate and key file separately, the same name should be used in both commands <Data length>: The length of certificate data. If certificate is DER format, data length parameter is mandatory <Data>: The certificate data to be store
	Example AT+TRSSLCERTSTORE=0,1,1,CA_CERT, -----BEGIN CERTIFICATE----- … -----END CERTIFICATE-----<Ctrl>+C OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __SUPPORT_ATCMD_TLS__ is enabled in the SDK, this command will be enabled	
AT+TRSSLCERT DELETE	<Certificate Type>,<Name>	Delete a certificate or CA list data in sflash memory. <Certificate Type>: The type of the certificate. 0 – CA Certificates, 1 – Client/Server Certificates <Name>: The name of the certificate

Command	Parameters	Description
	Example AT+TRSSLCERTDELETE=0,CA_CERT OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __SUPPORT_ATCMD_TLS__ is enabled in the SDK, this command will be enabled	
AT+TRSSLSAVE	(none)	Store the current SSL module's configuration in NVRAM
	Example AT+TRSSLSAVE OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __SUPPORT_ATCMD_TLS__ is enabled in the SDK, this command will be enabled	
AT+TRSSLDELETE	(none)	Delete the stored SSL module's configuration in NVRAM
	Example AT+TRSSLDELETE OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __SUPPORT_ATCMD_TLS__ is enabled in the SDK, this command will be enabled	

5.6.10 RF Test Function Commands

Table 39: RF test command list

Command	Parameters	Description
AT+TMRFNOINIT	<flag>	Set boot mode. <flag>: 0 (normal boot), 1 (RF test mode boot) Response: OK or ERROR
	Example AT+TMRFNOINIT=1 OK AT+RESTART OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ To test RF performance, set the boot mode as RF test mode (AT+TMRFNOINIT=1) and restart the DA16200/DA16600 (AT+RESTART) ▪ After DA16200/DA16600 is restarted, "!!! TEST MODE !!!" log is displayed	

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td colspan="2"> 
</td></tr>
<tr><td>AT+TMLMACINIT</td><td>(none)</td><td>Initialize LMAC (for test mode).
▪ Response: OK or ERROR</td></tr>
<tr><td></td><td colspan="2">Prerequisite
Boot as RF test mode (AT+TMRFNOINIT=1).

Example
AT+TMLMACINIT
OK

Note:
▪ Enabled by default in the SDK v3.2.3.0 or later</td></tr>
<tr><td>AT+RFTESTSTART</td><td>(none)</td><td>Start RF test mode</td></tr>
<tr><td></td><td colspan="2">Prerequisite
Boot as RF test mode (AT+TMRFNOINIT=1).

Example
AT+RFTESTSTART
OK

Note:
▪ Enabled by default in the SDK v3.2.3.0 or later</td></tr>
<tr><td>AT+RFTX</td><td><Ch>,
<BW>,
<numFrames>,
<frameLen>,
<txRate>,
<txPower>,
<destAddr>,
<bssid>,
<htEnable>,
<GI>,
<greenField>,
<preambleType>,
<qosEnable>,
<ackPolicy>,
<scrambler>,
<aifsnVal>,
<ant></td><td>Start RF TX test.

<Ch>: Carrier frequency (2412 ~ 2484 MHz)
<BW>: [0]: Fixed. Carrier bandwidth. 20 MHz fixed
<numFrames>: Number of frames to transmit
<frameLen>: Length of frame (bytes)
<txRate>: Data rate
b1: 11b DSSS 1 Mbps
b2: 11b DSSS 2 Mbps
b5_5: 11b DSSS 5.5 Mbps
b11: 11b DSSS 11 Mbps
g6: 11g 6 Mbps
g9: 11g 9 Mbps
g12: 11g 12 Mbps
g18: 11g 18 Mbps
g24: 11g 24 Mbps
g36: 11g 36 Mbps</td></tr>
</table>

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td></td><td>g48: 11g 48 Mbps
g54: 11g 54 Mbps
n6_5: 11n 6.5 Mbps (7.2 Mbps @Short GI)
n13: 11n 13 Mbps (14.4 Mbps @Short GI)
n19_5: 11n 19.5 Mbps (21.7 Mbps @Short GI)
n26: 11n 26 Mbps (28.9 Mbps @Short GI)
n39: 11n 39 Mbps (43.3 Mbps @Short GI)
n52: 11n 52 Mbps (57.8 Mbps @Short GI)
n58_5: 11n 58.5 Mbps (65 Mbps @Short GI)
n65: 11n 65 Mbps (72.2 Mbps @Short GI)
<txPower>: TX power (0 ~ 15), 0.8 dB step
<destAddr>: MAC address to send packet
<bssid>: BSSID
<htEnable>: N/A
<GI>: [short|long]. Guad interval. 11n mode only
<greenField>: [on|off]. Set greenfield mode on/off
<preambleType>: [short|long]. Preamble type @ DSSS mode
<qosEnable>: [on|off]. MAC header QoS control
<ackPolicy>: [NO|NORM|BA|CBA]
<scrambler>: N/A
<aifsnVal>: [0 ~ 15]. Indicate the AIFS in units of slots after SIFS that HW should wait for before starting backoff, for access category
<ant>: [0]. Fixed
Response: OK or ERROR</td></tr>
<tr><td></td><td colspan="2">Prerequisite
Start RF test mode (AT+RFTESTSTART).

Example
; Tx test with 11N MCS7, 2412 MHz and power grade as ‘0’ (max power)
AT+RFTX 2412,0,0,1000,n65,0
OK

Note:
▪ Enabled by default in the SDK v3.2.3.0 or later</td></tr>
<tr><td>AT+RFTXSTOP</td><td>(none)</td><td>Stop RF TX test</td></tr>
<tr><td></td><td colspan="2">Prerequisite
Start RF TX test (AT+RFTX).

Example
AT+RFTX 2412,0,0,1000,n65,0
OK

AT+RFTXSTOP
OK

AT+RFTX 2442,0,0,1000,n65,0
OK

Note:
▪ Enabled by default in the SDK v3.2.3.0 or later
▪ AT+RFTXSTOP is required before testing other items</td></tr>
<tr><td>AT+RFCWTEST</td><td><Ch>,</td><td>Start CW test.</td></tr>
</table>

Command	Parameters	Description
	<BW>, <txPower>, <ant>, <CWCycle>	<Ch>: Carrier frequency (2412 ~ 2484 MHz) <BW>: [0]: Fixed. Carrier bandwidth. 20 MHz fixed <txPower>: TX power (0 ~ 15), 0.8 dB step <ant>: [0]. Fixed <CWCycle>: 1 MHz fixed Response: OK or ERROR
	Prerequisite Start RF test mode (AT+RFTESTSTART). Example AT+RFCWTEST 2442,0,2 OK CW Tx test with 2442 MHz and power grade as 2 Note: ▪ Enabled by default in the SDK v3.2.3.0 or later	
AT+RFCWSTOP	(none)	Stop CW test. Response: OK or ERROR
	Prerequisite Start RF CW test (AT+RFCWTEST). Example AT+RFCWTEST 2442,0,2 OK AT+RFCWSTOP OK AT+RFCWTEST 2472,0,2 OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ AT+RFCWSTOP is required before testing other items	
AT+RFCONTSTART	<txRate>, <txPower>, <Ch>	Start RF continuous TX test. <txRate>: Data rate. See the AT+RFTX command <txPower>: TX power (0 ~ 15), 0.8 dB step <Ch>: Carrier frequency (2412 ~ 2484 MHz) Response: OK or ERROR
	Prerequisite Start RF test mode (AT+RFTESTSTART). Example ; Continuous Tx test with 11G 54 MHz, 2472 MHz and power grade as 2 AT+RFCONTSTART g54,2,2472 OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later	
	(none)	Stop RF continuous TX test.

Command	Parameters	Description
AT+RFCONTSTOP		Response: OK or ERROR
	Prerequisite Start RF continuous TX test (AT+RFCONTSTART). Example AT+RFCONTSTART g54,2,2412 OK AT+RFCONTSTOP OK AT+RFCONTSTART g54,2,2472 OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later	
AT+RFCHANNEL	<Ch>	Change RF channel for PER test. <Ch>: Carrier frequency (2412 ~ 2484 MHz) Response: OK or ERROR
	Prerequisite Start RF test mode (AT+RFTESTSTART). Example AT+RFCHANNEL 2412 OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later	
AT+RFPERRESET	(none)	Reset PER count. Response: OK or ERROR
	Prerequisite Start RF test mode (AT+RFTESTSTART). Example AT+RFPERRESET OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later	
AT+RFPER	(none)	Display PER state. Indicate a number of Valid packets, FCS Errors packets, PHY Errors packets, and Overflow Errors Response: OK or ERROR
	Prerequisite Start RF test mode (AT+RFTESTSTART). Example AT+RFPER 20 0 0 0 OK Note:	

Command	Parameters	Description
	▪ Enabled by default in the SDK v3.2.3.0 or later	
AT+RFTESTSTOP	(none)	Stop RF test mode.
	Example AT+RFTESTSTOP OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later	

Table 40: RF test examples

Test step	Command	Description
ECHO on	ATE	ECHO ON
Set boot mode	AT+TMRFNOINIT=1	Set boot mode as RF test mode
Restart the DA16200/DA16600	AT+RESTART	Reboot as RF test mode
ECHO on	ATE	ECHO ON
Start RF test mode	AT+RFTESTSTART	Start RF test mode.
Tx Test @11B 1 Mbps	AT+RFTX 2412,0,0,200,b1,0	11B 1 Mbps/Channel 1
	AT+RFTXSTOP	Stop Tx
	AT+RFTX 2442,0,0,200,b1,0	11B 1 Mbps/Channel 7
	AT+RFTXSTOP	Stop Tx
	AT+RFTX 2472,0,0,200,b1,0	11B 1 Mbps/Channel 13
	AT+RFTXSTOP	Stop Tx
Tx Test @11G 54 Mbps	AT+RFTX 2412,0,0,1000,g54,0	11G 54 Mbps/Channel 1
	AT+RFTXSTOP	Stop Tx
	AT+RFTX 2442,0,0,1000,g54,0	11G 54 Mbps/Channel 7
	AT+RFTXSTOP	Stop Tx
	AT+RFTX 2472,0,0,1000,g54,0	11G 54 Mbps/Channel 13
	AT+RFTXSTOP	Stop Tx
Tx Test @11N MCS7	AT+RFTX 2412,0,0,1000,n65,0	11N MCS7/Channel 1
	AT+RFTXSTOP	Stop Tx
	AT+RFTX 2442,0,0,1000,n65,0	11N MCS7/Channel 7
	AT+RFTXSTOP	Stop Tx
	AT+RFTX 2472,0,0,1000,n65,0	11N MCS7/Channel 13
	AT+RFTXSTOP	Stop Tx
Rx Test	AT+RFCHANNEL 2412	Change RF channel to 1
	AT+RFPERRESET	Reset PER count
	AT+RFPER	Display PER state
	AT+RFCHANNEL 2442	Change RF channel to 7
	AT+RFPERRESET	Reset PER count
	AT+RFPER	Display PER state

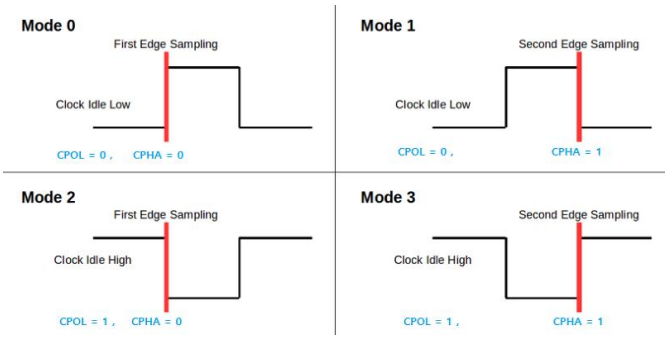
Test step	Command	Description
	AT+RFCHANNEL 2472	Change RF channel to 13
	AT+RFPERRESET	Reset PER count
	AT+RFPER	Display PER state
Stop RF test mode	AT+RFTESTSTOP	Stop RF test mode

NOTE
Renesas Electronics provides the AT-GUI tool to test RF performance easily. The tool and manual are available on the Renesas website (https://www.renesas.com/us/en/products/wireless-connectivity/wi-fi/low-power-wi-fi). See Ref. [4] The 2.4 GHz band is divided into 14 channels at 5 MHz intervals centered at 2.412 GHz, starting with channel 1. The last channel (CH 14) has additional restrictions or cannot be used for use in all regulatory areas. ▪ TX power setting value range: 0x0 ~ 0xB Setting value for unsupported channel: 0xF

5.6.11 System and Peripheral Function Commands

5.6.11.1 SPI Commands

Table 41: SPI command list

Command	Parameters	Description
AT+SPICONF	<clockpol>, <clockpha>	Configure SPI. <clockpol>: Clock polarity [0|1] <clockpha>: Clock phase [0|1]
	Example AT+SPICONF=1,1 OK AT+SPICONF=0,1 OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __SUPPORT_PERI_CMD__ is enabled in the SDK, this command will be enabled ▪ The <clockpol> sets the polarity of the clock signal during the idle state. The idle state is defined as the period when CS is high and transitioning to low at the start of the transmission and when CS is low and transitioning to high at the end of the transmission. The <clockpha> selects the clock phase. Depending on the <clockpha>, the rising or falling clock edge is used to sample the data. The default value of DA16200/DA16600 are clockpol,0 and clockpha 0 

5.6.11.2 OTP Commands

Table 42: OTP command list

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td>AT+UOTPRDASC</td><td><addr>,<cnt></td><td>Read OTP data.
<addr>: OTP address to read 4-byte aligned
<cnt>: Bytes to read
Response: OK or Error
A string of four-bit HEXA value represented by the ASCII code</td></tr>
<tr><td></td><td colspan="2">Example
; Reading 4 bytes at offset h180 (h180 * 4 = h600)
; If data "12345678" is written to 0x600, can read the values
AT+UOTPRDASC=600,4
12345678
OK

Note:
▪ Enabled by default in the SDK v3.2.3.0 or later
▪ If __SUPPORT_PERI_CMD__ is enabled in the SDK, this command will be enabled
▪ Physical OTP offset range of DA16200/DA16600 is h0~h1FF; at each offset, 4 bytes are stored or read
▪ For accessing OTP using this command, a 4-byte aligned address should be given. For example: h0, h4, h8 ...</td></tr>
<tr><td>AT+UOTPWRASC</td><td><addr>,<cnt>,<value></td><td>Write OTP data.

<addr>: OTP address to write 4-byte aligned
<cnt>: Bytes to write
<value>: A string of four-bit HEXA value represented by the ASCII code
Response: OK or Error
Important
For MAC address read or write, AT+WFOTP (write) and AT+WFMAC (read) must be used. Do not use AT+UOTPRDASC or AT+UOTPWRASC for this purpose.
OTP offset from 0x00 ~ 0x2b should not be written as this section is for “secure” boot.</td></tr>
<tr><td></td><td colspan="2">Example
; Writing h12345678 to OTP Address 0x600:
; To write “12345678” data into the 0x600,
AT+UOTPWRASC=600,4,12345678
OK

; To read written Data via UOTPRDASC
AT+UOTPRDASC=600,4
12345678
OK

Note:
▪ Enabled by default in the SDK v3.2.3.0 or later
▪ If __SUPPORT_PERI_CMD__ is enabled in the SDK, this command will be enabled
▪ Physical OTP offset range of DA16200/DA16600 is h0~h1FF; at each offset, 4 bytes are stored or read
▪ For accessing OTP using this command, 4-byte aligned address should be given. For example: h0, h4, h8 ...</td></tr>
</table>

DA16200/DA16600 provides four slots to store MAC addresses and 8 bytes are allocated for each slot.

Table 43: OTP address for XTAL offset

Slot	OTP address	Description	Size (Byte)
MAC Address #0	0x100	MAC Address Low	4
	0x101	MAC Address High	4
MAC Address #1	0x102	MAC Address Low	4
	0x103	MAC Address High	4
MAC Address #2	0x104	MAC Address Low	4
	0x105	MAC Address High	4
MAC Address #3	0x106	MAC Address Low	4
	0x107	MAC Address High	4

DA16200/DA16600 provides two slots to store XTAL offset in the OTP memory. Slot #0 is the primary slot while Slot#1 is for back-up, which is used when overriding Slot #0.

Table 44. Memory size by XTAL offset

Slot	OTP address	Description	Size (Byte)
XTAL Offset #0	0x10A	XTAL Offset #0 value	1
XTAL Offset #1	0x10B	XTAL Offset #1 value	1

5.6.11.3 XTAL Commands

These commands are used for XTAL calibration and the usage is described in DA16200/DA16600 Mass Production Guide. See Ref. [7].

Table 45: XTAL command list

Command	Parameters	Description
AT+XTALWR	<value>	Write XTAL Offset to DA16200/DA16600 system register. <value>: Seven-bits to write [h'1 ~ h'7f] Response: OK or Error
	Example AT+XTALWR=7f OK AT+XTALWR=80 ERROR Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __SUPPORT_PERI_CMD__ is enabled in the SDK, this command will be enabled	
AT+XTALRD	(none)	Read XTAL Offset from DA16200/DA16600 System. Response: <CR><LF><A string of seven-bit HEXA value represented by the ASCII Code><CR><LF>OK<CR><LF> or Error
	Example AT+XTALRD 0x7f OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __SUPPORT_PERI_CMD__ is enabled in the SDK, this command will be enabled	

5.6.11.4 Flash Dump Commands

Table 46: Flash dump command list

Command	Parameters	Description
AT+FLASHDUMP	<address>, <length>	Dump serial flash data. <address>: Start address [h'0 ~ h'3fffff] <length>: Data length [d'] Response: <CR><LF> <dump data> <CR><LF>OK<CR><LF> or Error
	Example ; The following example reads 32 kB from 0x00, (1024*32 = 32768) AT+FLASHDUMP=0,32768 Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __SUPPORT_PERI_CMD__ is enabled in the SDK, this command will be enabled	

5.6.11.5 GPIO Commands

Table 47: GPIO command list

Command	Parameters	Description
AT+GPIOSTART	<port>, <pin >, <direction>	Configure the GPIO pin mux and the direction of a GPIO <port>: GPIO port number ▪ 0: GPIOA ▪ 2: GPIOC <pin>: GPIO pin number. This is a hexadecimal value and indicates a GPIO bitmap ▪ GPIOA: GPIOA0 ~ GPIOA11 ▪ GPIOC: GPIOC6 ~ GPIOC8 <direction>: GPIO pin direction ▪ 0: Set the pin as an input ▪ 1: Set the pin as an output Response: OK or Error
	Example ; To configure GPIOA [3:0] output with using UART interface: ; GPIO (0, 1, 2, 3) is set to binary 1 (0000 0000 0000 1111). AT+GPIOSTART=0,f,1 OK ; To configure GPIOA [3:0] output with using SPI interface: : Avoid reassigning default SPI-pin. ; GPIO (4, 5, 6, 7) is set to binary 1 (0000 0000 1111 0000). AT+GPIOSTART=0,f0,1 OK ; To configure GPIOC [8:6] input: ; GPIO (6, 7, 8) is set to binary 1 (0000 0001 1100 0000). AT+GPIOSTART=2,1c0,0 OK	

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td colspan="2">Note:
▪ Enabled by default in the SDK v3.2.3.0 or later
▪ If __SUPPORT_PERI_CMD__ is enabled in the SDK, this command will be enabled</td></tr>
<tr><td>AT+GPIORD</td><td><port>,
<pin></td><td>Read the GPIO input level.

<port>: GPIO port number
▪ 0: GPIOA
▪ 2: GPIOC
<pin>: GPIO pin number. This is a hexadecimal value and indicates a GPIO bitmap
Response:
<Read value>: [h'0 ~ h'1fff]
OK or Error</td></tr>
<tr><td></td><td colspan="2">Example
; Configure GPIOC[8:6] as output and set to high.
AT+GPIOSTART=2,1c0,1
OK

AT+GPIOWR=2,1c0,1
OK

; Read back the status of the pins:
AT+GPIORD=2,1c0
0x01c0
OK

Note:
▪ Enabled by default in the SDK v3.2.3.0 or later
▪ If __SUPPORT_PERI_CMD__ is enabled in the SDK, this command will be enabled
▪ The read value indicates GPIO bitmap. If a value is 0x1c0, it means GPIO #6, #7, #8 high</td></tr>
<tr><td>AT+GPIOWR</td><td><port>,
<pin>,
<level></td><td>Configures the output level of GPIO pins.

<port>: GPIO port number
▪ 0: GPIOA
▪ 2: GPIOC
<pin>: GPIO pin number. This is a hexadecimal value and indicates a GPIO bitmap
<level>: GPIO output level
▪ 0: Low
▪ 1: High
Response: OK or Error</td></tr>
<tr><td></td><td colspan="2">Prerequisite
Change the direction of GPIO to output (AT+GPIOSTART).

Example
; Configure GPIOC[8:6] as output and set to high.
AT+GPIOSTART=2,1c0,1
OK

AT+GPIOWR=2,1c0,1
OK</td></tr>
</table>

Command	Parameters	Description
	Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __SUPPORT_PERI_CMD__ is enabled in the SDK, this command will be enabled	
AT+SAVE_PININFO	(none)	Save pin mux information. Response: OK or Error
	Example AT+SAVE_PININFO OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __SUPPORT_PERI_CMD__ is enabled in the SDK, this command will be enabled ▪ It is to save a current PIN mux configured	
AT+RESTORE_PININFO	(none)	Restore pin mux information. Response: OK or Error
	Example AT+RESTORE_PININFO OK Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __SUPPORT_PERI_CMD__ is enabled in the SDK, this command will be enabled ▪ It is to restore the PIN multiplexing status saved through the AT+SAVE_PININFO command	

5.6.11.6 LED Commands

Table 48: LED command list

Command	Parameters	Description
AT+LEDINIT	<none>	Configure GPIOC_6 (LED1), GPIOC_7 (LED2), and GPIOC_8 (LED3) pins to GPIO output. Response: OK or Error
	Example AT+LEDINIT +OK Note: ▪ Enabled by default in the SDK v3.2.2.1 or later	
AT+LEDCTRL	<port >, <status>	Set LED1/2/3 (GPIOC_6/7/8) pin to output High or Low. <port>: GPIO port number 1: GPIOC_6 2: GPIOC_7 3: GPIOC_8 <status>: LED status off: LED off on: LED on Response: OK or Error
	Example AT+LEDCTRL=1,off OK Note: ▪ Enabled by default in the SDK v3.2.2.1 or later	

5.6.11.7 PWM Commands

Table 49: PWM command list

Command	Parameters	Description
AT+PWMINIT	<none>	Configure GPIOA_10 pin to PWM output. Response: OK or Error
	Example AT+PWMINIT +OK Note: ▪ Enabled by default in the SDK v3.2.2.1 or later ▪ If __ATCMD_IF_UART1__ is enabled in the SDK, this command will be enabled	
AT+PWMSTART	<channel >, <period>, <duty> <mode cycle>	Start PWM output (GPIOA_10) with given period and duty. <channel>: PWM channel, fixed as 0 <period>: period of one clock (microsecond) <duty>: duty as percentage <mode cycle>: fixed as 0 Response: OK or Error
	Example AT+PWMSTART=0,40,50,0 OK Note: ▪ Enabled by default in the SDK v3.2.2.1 or later ▪ If __ATCMD_IF_UART1__ is enabled in the SDK, this command will be enabled	
AT+PWMSTOP	<none>	Stop PWM output. Response: OK or Error
	Example AT+PWMSTOP OK Note: ▪ Enabled by default in the SDK v3.2.2.1 or later ▪ If __ATCMD_IF_UART1__ is enabled in the SDK, this command will be enabled	

5.6.11.8 ADC Commands

Table 50: ADC command list

Command	Parameters	Description
AT+ADCINIT	<none>	Configure GPIOA_0, GPIOA_1, GPIOA_2, GPIOA_3 to analog input pins for ADC. GPIOA_0: ADC channel 0 GPIOA_1: ADC channel 1 GPIOA_2: ADC channel 2 GPIOA_3: ADC channel 3 Response: OK or Error
	Example AT+ADCINIT OK Note:	

Command	Parameters	Description
	▪ Enabled by default in the SDK v3.2.2.1 or later ▪ If __ATCMD_IF_UART1__ is enabled in the SDK, this command will be enabled	
AT+ADCCHEN	<channel >, < resolution >	Enable given ADC channel. <channel>: ADC channel number [0|1|2|3] <resolution>: ADC resolution. fixed as 12-bit Response: OK or Error
	Example AT+ADCCHEN=0,12 OK Note: ▪ Enabled by default in the SDK v3.2.2.1 or later ▪ If __ATCMD_IF_UART1__ is enabled in the SDK, this command will be enabled	
AT+ADCSTART	<divider>	Start ADC function <divider>: divider ADC sampling rate For example, when divider is 1, 1 MHz / (divider (1) + 1) = 500 kHz Response: OK or Error
	Example AT+ADCSTART=1 OK Note: ▪ Enabled by default in the SDK v3.2.2.1 or later ▪ If __ATCMD_IF_UART1__ is enabled in the SDK, this command will be enabled	
AT+ADCREAD	<channel>, <sample count>	Read ADC value. <channel>: ADC channel number [0|1|2|3] <sample count>: count of sample to read Response: <Read value>: [sample count] Response: OK or Error
	Example AT+ADCREAD=0,16 [279 275 269 271 270 268 268 274 274 277 276 269 271 276 264 274] OK Note: ▪ Enabled by default in the SDK v3.2.2.1 or later ▪ If __ATCMD_IF_UART1__ is enabled in the SDK, this command will be enabled	
AT+ADCSTOP	(none)	Stop ADC function. Response: OK or Error
	Example AT+ADCSTOP OK Note: ▪ Enabled by default in the SDK v3.2.2.1 or later ▪ If __ATCMD_IF_UART1__ is enabled in the SDK, this command will be enabled	

5.6.11.9 I2C Commands

Table 51: I2C command list

Command	Parameters	Description
AT+I2CINIT	<none>	Configure GPIOA_8 (I2C_SDA), GPIOA_9 (I2C_SCL) pins to I2C pins. Response: OK or Error
	Example AT+I2CINIT OK Note: ▪ Enabled by default in the SDK v3.2.2.1 or later	
AT+I2CREAD	<slave address>, <register>, <length>	Read values from registers of I2C device. <slave address>: 8-bit slave address of I2C device (hex) <register>,: register value to read (hex) <length>: data length to read (decimal) Response: <Read values> (hex) Response: OK or Error
	Example AT+I2CREAD=d0,10,1 66 OK Note: ▪ Enabled by default in the SDK v3.2.2.1 or later	
AT+ I2CWRITE	<slave address>, <register>, <length>, <values>	Write values to I2C register of I2C device. <slave address>: 8-bit slave address of I2C device (hex) <register>: register value to write (hex) <length>: data length to write (decimal) <values>: data to write (hex) Response: OK or Error
	Example AT+I2CWRITE=d0,10,3,670292 OK Note: ▪ Enabled by default in the SDK v3.2.2.1 or later	

5.6.11.10 Sleep Commands

Table 52: Sleep command list

Command	Parameters	Description
AT+SLEEPMS	<period>	Make DA16200/DA16600 go to Sleep mode 3 and wake up after <period> milliseconds. <period>: Wake-up time in milliseconds. Max period: 2097151000 (about 24 days) Response: OK or Error
	Example AT+SLEEPMS=5000 +INIT:DONE,0	

Command	Parameters	Description
	Note: ▪ Enabled by default in the SDK v3.2.3.0 or later ▪ If __SUPPORT_PERI_CMD__ is enabled in the SDK, this command will be enabled	

5.6.11.11 CALWL Commands

Table 53: CALWL command list

Command	Parameters	Description
AT+CALWR	<gmode_tx_rf_proc>, <txpga_gmode_cal>	Change RF TX GAIN Calibration register for test. <gmode_tx_rf_proc> ▪ 7 bits hexadecimal without “0x” prefix ▪ Offset = bit[5:0] x 0.8 dB ▪ MSB[6] bits 0 then TX gain is decreased ▪ MSB[6] bits 1 then TX gain is increased <txpga_gmode_cal> ▪ 0 : 0 dB offset ▪ 1 : -0.2 dB offset ▪ 2 : -0.4 dB offset 3 : -0.6 dB offset
	Example 1. TX measured +14 dBm and change to +13 dBm AT+CALWR=1,1 OK gmade_tx_rf_proc = 1, -0.8 dB txpga_gmode_cal = 1, -0.2 dB Changed TX Gain: 14 dBm - 0.8 dB - 0.2 dB = 13.0 dBm 2. TX measured +13 dBm and change to +13.6 dBm AT+CALWR=41,3 OK gmade_tx_rf_proc = 41, +0.8 dB txpga_gmode_cal = 1, -0.2 dB Changed TX Gain: 13 dBm + 0.8 dB - 0.2 dB = 13.6 dBm Note: ▪ This setting is not saved to the system and changed when reboot system. ▪ If __SUPPORT_PERI_CMD__ is enabled in the SDK, this command will be enabled	

6. AT Command Example

6.1 Data Transfer Test

This section describes how to test the transfer function commands with a data terminal emulator. Some of the terminal applications to be used for this purpose are:

- Hercules for Windows: https://www.hw-group.com/software/hercules-setup-utility
- Packet Sender for Windows/Linux/MacOS: https://packetsender.com/
- TCP UDP Server & Client for Android: TCP UDP Server & Client - Apps on Google Play
- UDP/TCP/REST Network Utility for iOS: UDP/TCP/REST Network Utility on the App Store

The following sections describe test procedures for socket communication between the DA16200/DA16600 and a PC with Hercules. Run DA16200/DA16600 AT commands on a serial terminal application on the local PC. The terminal must be connected to the UART1 interface of the DA16200/DA16600.

6.1.1 TCP Server Socket Test

1. DA16200/DA16600 AT command:
   a. AT+TRTS=1234 ← Open a TCP server socket with the port number 1234.
2. PC:
   a. Select TCP Client (#1, Figure 27).
   b. Enter the IP address and the port number of DA16200/DA16600 (#2, Figure 27).
   c. Click Connect to connect the socket (#3, Figure 27).
3. DA16200/DA16600 AT command:
   a. +TRCTS:0,192.168.0.2,50166 ← A TCP client socket connected, and IP address is 192.168.0.2 and port is 50166.
4. PC:
   a. Send data (#4, Figure 27).
5. DA16200/DA16600 AT command:
   a. +TRDTS:0,192.168.0.2,50166,24,Renesas IoT WiFi DA16200 ← Received 24 bytes of data: Renesas IoT WiFi DA16200.

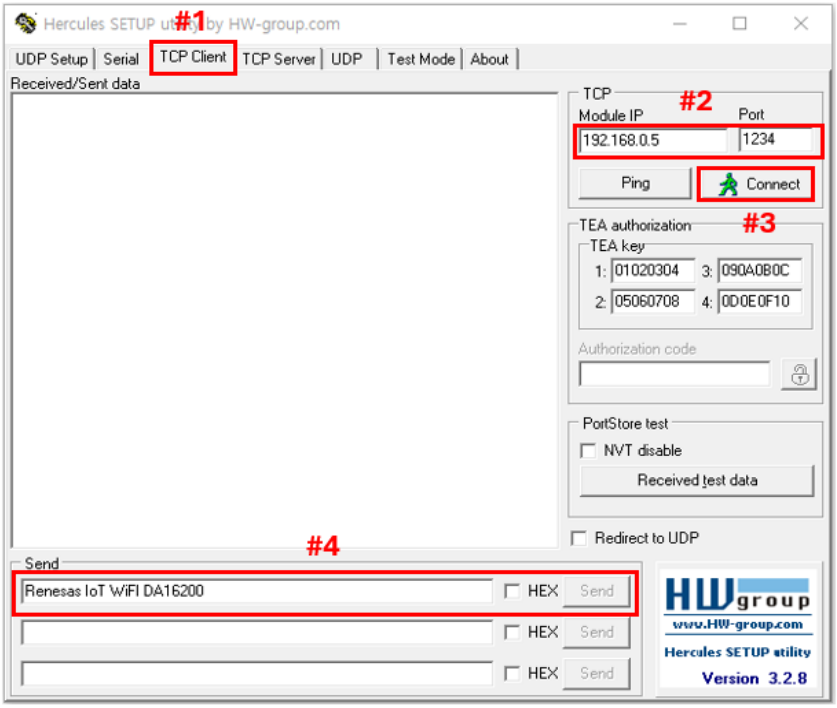


Figure 27: Hercules – TCP client socket setting

6.1.2 TCP Client Socket Test

1. PC:
   a. Select TCP Server (#1, Figure 28).
   b. Enter the port number to be used (#2, Figure 28).
   c. Click Listen to start to Listen (#3, Figure 28).
2. DA16200/DA16600 AT command:
   a. AT+TRTC=192.168.0.2,1234,2300 ← Open a TCP client socket and set the server IP (192.168.0.2), port (1234), and the local port (2300).
   b. <ESC>S18,0,0,12345678 ← Send 8 bytes of data: 12345678.
3. PC:
   a. Receive 8 bytes data.
   b. Send data (#4, Figure 28).
4. DA16200/DA16600 AT command:
   a. `+TRDTC:1,192.168.0.2,1234,24,Renesas IoT WiFi DA16200` ← Received 24 bytes of data: Renesas IoT WiFi DA16200.

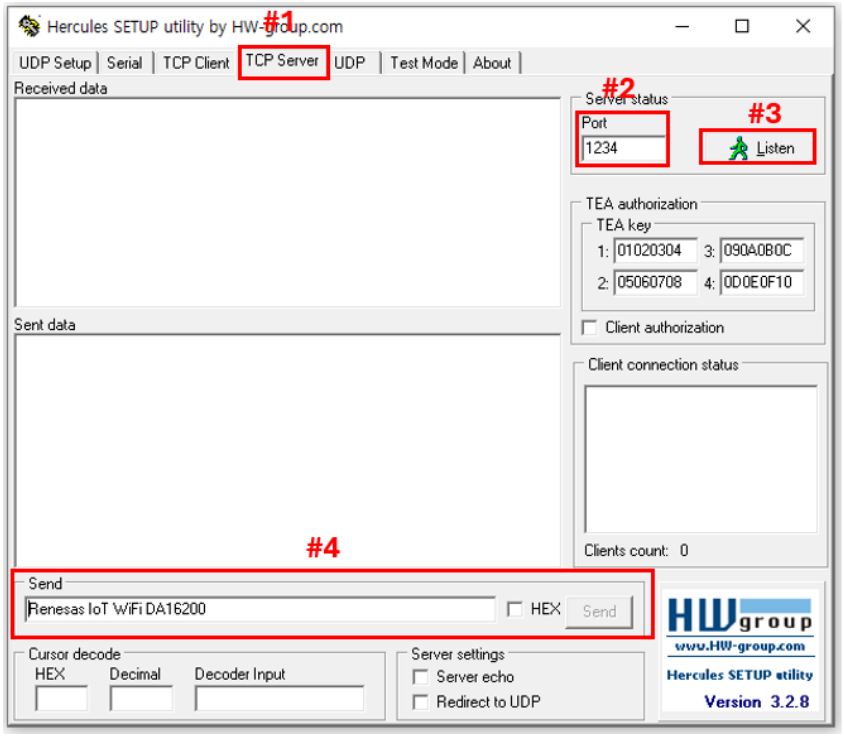


Figure 28: Hercules – TCP server socket setting

6.1.3 UDP Socket Test

1. PC:
   a. Select UDP (#1, Figure 29).
   b. Enter the IP address and port of the counterpart's UDP socket (#2, Figure 29).
   c. Enter the port number to be used and click Listen to open the socket (#3, Figure 29).
   d. Enter data and click Send to transmit (#4, Figure 29).
2. DA16200/DA16600 AT command:
   a. AT+TRUSE=4567 ← Open a UDP socket and set the local port (4567).
   b. AT+TRUR=192.168.0.2,1234 ← Set the remote IP (192.168.0.2) and port (1234).
   c. <ESC>S210,0,0,1234567890 ← Send 10 bytes of data: 1234567890.
   d. +TRDUS:2,192.168.0.2,1234,25,Renesas IoT Wi-Fi DA16200 ← Received 25 bytes of data: Renesas IoT Wi-Fi DA16200.

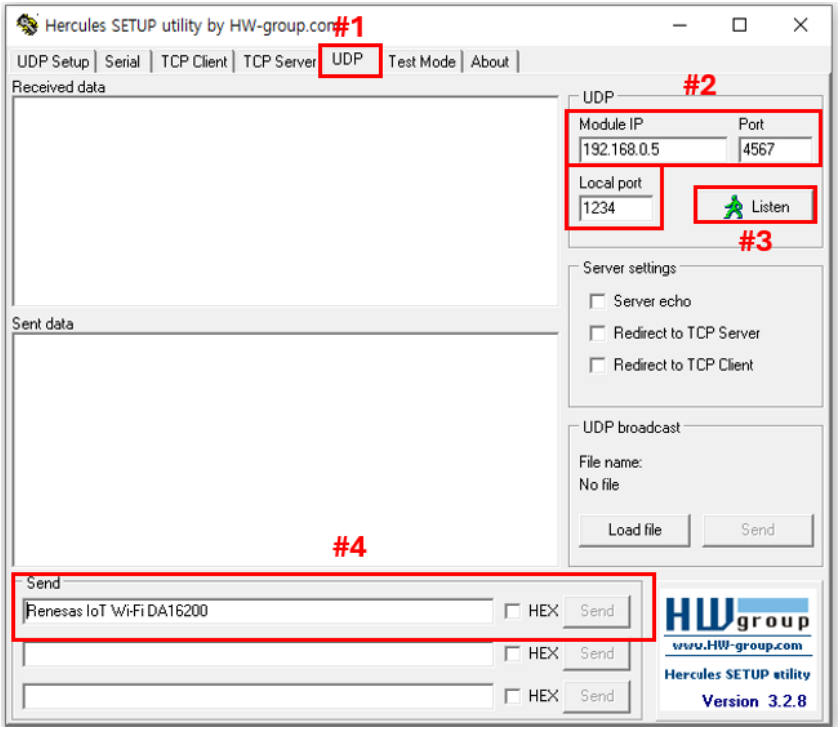


Figure 29: Hercules – UDP socket setting

Appendix A License Information

Mosquitto1.4.14 License

Eclipse Distribution License 1.0

Copyright (c) 2007, Eclipse Foundation, Inc. and its licensors.

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the follow disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of the Eclipse Foundation, Inc.

  Nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO,
THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
POSSIBILITY OF SUCH DAMAGE.

UMAC GPL License

Linux kernel 3.9.0 rc3 version (backport 4.2.6-1)

Appendix B HTTP API Return Values

B.1 Return Value as Defined By NetX Duo HTTP

Table 54: Return value as defined by Netx Duo HTTP

Define	Value	Define	Value
NX_SUCCESS	0x00	NX_RESERVED_CODE1	0x25
NX_NO_PACKET	0x01	NX_SOCKET_UNBOUND	0x26
NX_UNDERFLOW	0x02	NX_NOT_CREATED	0x27
NX_OVERFLOW	0x03	NX_SOCKETS_BOUND	0x28
NX_NO_MAPPING	0x04	NX_NO_RESPONSE	0x29
NX_DELETED	0x05	NX_POOL_DELETED	0x30
NX_POOL_ERROR	0x06	NX_ALREADY_RELEASED	0x31
NX_PTR_ERROR	0x07	NX_RESERVED_CODE2	0x32
NX_WAIT_ERROR	0x08	NX_MAX_LISTEN	0x33
NX_SIZE_ERROR	0x09	NX_DUPLICATE_LISTEN	0x34
NX_OPTION_ERROR	0x0A	NX_NOT_CLOSED	0x35
NX_DELETE_ERROR	0x10	NX_NOT_LISTEN_STATE	0x36
NX_CALLER_ERROR	0x11	NX_IN_PROGRESS	0x37
NX_INVALID_PACKET	0x12	NX_NOT_CONNECTED	0x38
NX_INVALID_SOCKET	0x13	NX_WINDOW_OVERFLOW	0x39
NX_NOT_ENABLED	0x14	NX_ALREADY_SUSPENDED	0x40
NX_ALREADY_ENABLED	0x15	NX_DISCONNECT_FAILED	0x41
NX_ENTRY_NOT_FOUND	0x16	NX_STILL_BOUND	0x42
NX_NO_MORE_ENTRIES	0x17	NX_NOT_SUCCESSFUL	0x43
NX_ARP_TIMER_ERROR	0x18	NX_UNHANDLED_COMMAND	0x44
NX_RESERVED_CODE0	0x19	NX_NO_FREE_PORTS	0x45
NX_WAIT_ABORTED	0x1A	NX_INVALID_PORT	0x46
NX_IP_INTERNAL_ERROR	0x20	NX_INVALID_RELISTEN	0x47
NX_IP_ADDRESS_ERROR	0x21	NX_CONNECTION_PENDING	0x48
NX_ALREADY_BOUND	0x22	NX_TX_QUEUE_DEPTH	0x49
NX_PORT_UNAVAILABLE	0x23	NX_NOT_IMPLEMENTED	0x4A
NX_NOT_BOUND	0x24	NX_NOT_SUPPORTED	0x4B
NX_INVALID_INTERFACE	0x4C	NX_DUPLICATED_ENTRY	0x52
NX_INVALID_PARAMETERS	0x4D	NX_PACKET_OFFSET_ERROR	0x53
NX_NOT_FOUND	0x4E	NX_OPTION_HEADER_ERROR	0x54
NX_CANNOT_START	0x4F	NX_CONTINUE	0x55
NX_NO_INTERFACE_ADDRESS	0x50	NX_PARAMETER_ERROR	0xFF
NX_INVALID_MTU_DATA	0x51		

B.2 Return Value as Defined by LWIP HTTP

Table 55: Return value as defined by LWIP HTTP

Define	Value	Define	Value
ERR_OK	0	ERR_ISCONN	-10
ERR_MEM	-1	ERR_CONN	-11
ERR_BUF	-2	ERR_IF	-12
ERR_TIMEOUT	-3	ERR_ABRT	-13
ERR_RTE	-4	ERR_RST	-14
ERR_INPROGRESS	-5	ERR_CLSD	-15
ERR_VAL	-6	ERR_ARG	-16

Define	Value	Define	Value
ERR_WOULDBLOCK	-7	ERR_UNKNOWN	-17
ERR_USE	-8	ERR_NOT_FOUND	-18
ERR_ALREADY	-9		

Appendix C User UART Configuration

C.1 How to Run AT Command on UART2

AT command is configured to use the UART1 interface by default and can be configured to use the UART2 interface. To configure AT command to use the UART2 interface, modify config_generic_sdk.h as shown in bold below:

```
      // AT command service
      #define __SUPPORT_ATCMD__
...
      #if defined ( __SUPPORT_ATCMD__ )
              #undef __ATCMD_IF_UART1__          // AT command over UART1
              #define __ATCMD_IF_UART2__         // AT command over UART2
              ...
              #undef __USER_UART_CONFIG__ // Support Customer's UART configuration
              #undef __ATCMD_IF_SPI__            // AT command over SPI
              #undef __ATCMD_IF_SDIO__           // AT command over SDIO
       #endif /* __SUPPORT_ATCMD__ */
...
```

C.2 User UART Configuration

There is a feature called User UART Configuration that is enabled by __USER_UART_CONFIG__.
When the SDK is built with __USER_UART_CONFIG__ defined, the UART settings for the AT command interface can be configured. In this case, ATB will not be available.
For example, to run AT command on UART2 with a static baud rate of 230400, the SDK should be configured as shown in bold below.

```
// config_generic_sdk.h
...
      // AT command service
      #define __SUPPORT_ATCMD__
...
      #if defined ( __SUPPORT_ATCMD__ )
              #undef __ATCMD_IF_UART1__   // AT command over UART1
              #define __ATCMD_IF_UART2__  // AT command over UART2
              ...
              #define __USER_UART_CONFIG__ // Support Customer's UART configuration
              ...
       #endif /* __SUPPORT_ATCMD__ */
...

// user_interface.c
...
#if defined ( __USER_UART_CONFIG__ )
```

```
/*
 * Customer configuration for AT command UART
 */
uart_info_t ATCMD_UART_config_info =
{
      UART_BAUDRATE_230400,         /* baud */
      UART_DATABITS_8,              /* bits */
      UART_PARITY_NONE,             /* parity */
      UART_STOPBITS_1,              /* stopbit */
      UART_FLOWCTL_OFF              /* flow control */
};
#endif // __USER_UART_CONFIG__
...
```

With changes above, when DA16200/DA16600 boots, AT command is initialized in baud rate 230400 by default and cannot change at run time.

C.3 Use Case

// __USER_UART_CONFIG__ disabled

- Baud rate (and other parameters) configurable by NVRAM
- ATB available, UART Setting can change at run-time without SDK rebuild
- Example Use case
  - MCU: Run on UART in baud rate 115200
  - MCU: Run ATF
  - DA16200/DA16600: AT command is initialized in 115200
  - MCU: ATB=230400
  - MCU: Now it should change its UART baud rate to 230400 to communicate with DA16200/DA16600

// __USER_UART_CONFIG__ enabled

- AT Command UART's baud rate (and other parameters) is configurable statically
- ATB NOT available
- Example Use Case
  - DA16200/DA16600: DA16200/DA16600 boots and AT command is initialized in 230400 by default now.
  - MCU: Start on UART in baud rate 230400
  - MCU: AT command operation …

Appendix D DA16200/DA16600 Cipher Suites

Table 56: DA16200/DA16600 cipher suites

No.	Cipher suite supported by DA16200/DA16600	Hex code
1	TLS_RSA_WITH_AES_128_CBC_SHA	2F
2	TLS_RSA_WITH_AES_256_CBC_SHA	35
3	TLS_RSA_WITH_AES_128_CBC_SHA256	3C
4	TLS_RSA_WITH_AES_256_CBC_SHA256	3D
5	TLS_RSA_WITH_AES_128_GCM_SHA256	9C
6	TLS_RSA_WITH_AES_256_GCM_SHA384	9D
7	TLS_RSA_WITH_AES_128_CCM	C09C
8	TLS_RSA_WITH_AES_256_CCM	C09D
9	TLS_RSA_WITH_AES_128_CCM_8	C0A0
10	TLS_RSA_WITH_AES_256_CCM_8	C0A1
11	TLS_RSA_WITH_DES_CBC_SHA	9
12	TLS_DHE_RSA_WITH_AES_128_CBC_SHA	33
13	TLS_DHE_RSA_WITH_AES_256_CBC_SHA	39
14	TLS_DHE_RSA_WITH_AES_128_CBC_SHA256	67
15	TLS_DHE_RSA_WITH_AES_256_CBC_SHA256	6B
16	TLS_DHE_RSA_WITH_AES_128_GCM_SHA256	9E
17	TLS_DHE_RSA_WITH_AES_256_GCM_SHA384	9F
18	TLS_DHE_RSA_WITH_AES_128_CCM	C09E
19	TLS_DHE_RSA_WITH_AES_256_CCM	C09F
20	TLS_DHE_RSA_WITH_AES_128_CCM_8	C0A2
21	TLS_DHE_RSA_WITH_AES_256_CCM_8	C0A3
22	TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA	16
23	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA	C011
24	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA	C014
25	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256	C027
26	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384	C028
27	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	C02F
28	TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384	C030
29	TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA	C012
30	TLS_ECDH_RSA_WITH_AES_128_CBC_SHA	C00E
31	TLS_ECDH_RSA_WITH_AES_256_CBC_SHA	C00F
32	TLS_ECDH_RSA_WITH_AES_128_CBC_SHA256	C029
33	TLS_ECDH_RSA_WITH_AES_256_CBC_SHA384	C02A
34	TLS_ECDH_RSA_WITH_AES_128_GCM_SHA256	C031
35	TLS_ECDH_RSA_WITH_AES_256_GCM_SHA384	C032
36	TLS_ECDH_RSA_WITH_3DES_EDE_CBC_SHA	C00D
37	TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA	C009
38	TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA	C00A
39	TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256	C023
40	TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384	C024
41	TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256	C02B
42	TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384	C02C
43	TLS_ECDHE_ECDSA_WITH_AES_128_CCM	C0AC
44	TLS_ECDHE_ECDSA_WITH_AES_256_CCM	C0AD
45	TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8	C0AE
46	TLS_ECDHE_ECDSA_WITH_AES_256_CCM_8	C0AF
47	TLS_ECDHE_ECDSA_WITH_3DES_EDE_CBC_SHA	C008

No.	Cipher suite supported by DA16200/DA16600	Hex code
48	TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA	C004
49	TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA	C005
50	TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA256	C025
51	TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA384	C026
52	TLS_ECDH_ECDSA_WITH_AES_128_GCM_SHA256	C02D
53	TLS_ECDH_ECDSA_WITH_AES_256_GCM_SHA384	C02E
54	TLS_ECDH_ECDSA_WITH_3DES_EDE_CBC_SHA	C003

Appendix E Reason Code for Wi-Fi Connection Failure or Disconnection

SDK v3.x.x.x: core\wifistack\supplicant\src\common\ieee802_11_defs.h.

```
/* Reason codes (IEEE Std 802.11-2016, 9.4.1.7, Table 9-45) */
#define WLAN_REASON_UNSPECIFIED                          1
#define WLAN_REASON_PREV_AUTH_NOT_VALID                  2
#define WLAN_REASON_DEAUTH_LEAVING                       3
#define WLAN_REASON_DISASSOC_DUE_TO_INACTIVITY           4
#define WLAN_REASON_DISASSOC_AP_BUSY                     5
#define WLAN_REASON_CLASS2_FRAME_FROM_NONAUTH_STA        6
#define WLAN_REASON_CLASS3_FRAME_FROM_NONASSOC_STA       7
#define WLAN_REASON_DISASSOC_STA_HAS_LEFT                8
#define WLAN_REASON_STA_REQ_ASSOC_WITHOUT_AUTH           9
/* IEEE 802.11h */
#define WLAN_REASON_PWR_CAPABILITY_NOT_VALID             10
#define WLAN_REASON_SUPPORTED_CHANNEL_NOT_VALID          11
#define WLAN_REASON_BSS_TRANSITION_DISASSOC              12
/* IEEE 802.11i */
#define WLAN_REASON_INVALID_IE                           13
#define WLAN_REASON_MICHAEL_MIC_FAILURE                  14
#define WLAN_REASON_4WAY_HANDSHAKE_TIMEOUT               15
#define WLAN_REASON_GROUP_KEY_UPDATE_TIMEOUT             16
#define WLAN_REASON_IE_IN_4WAY_DIFFERS                   17
#define WLAN_REASON_GROUP_CIPHER_NOT_VALID               18
#define WLAN_REASON_PAIRWISE_CIPHER_NOT_VALID            19
#define WLAN_REASON_AKMP_NOT_VALID                       20
#define WLAN_REASON_UNSUPPORTED_RSN_IE_VERSION           21
#define WLAN_REASON_INVALID_RSN_IE_CAPAB                 22
#define WLAN_REASON_IEEE_802_1X_AUTH_FAILED              23
#define WLAN_REASON_CIPHER_SUITE_REJECTED                24
#define WLAN_REASON_TDLS_TEARDOWN_UNREACHABLE            25
#define WLAN_REASON_TDLS_TEARDOWN_UNSPECIFIED            26
#define WLAN_REASON_SSP_REQUESTED_DISASSOC               27
#define WLAN_REASON_NO_SSP_ROAMING_AGREEMENT             28
#define WLAN_REASON_BAD_CIPHER_OR_AKM                    29
#define WLAN_REASON_NOT_AUTHORIZED_THIS_LOCATION         30
#define WLAN_REASON_SERVICE_CHANGE_PRECLUDES_TS          31
#define WLAN_REASON_UNSPECIFIED_QOS_REASON               32
#define WLAN_REASON_NOT_ENOUGH_BANDWIDTH                 33
#define WLAN_REASON_TDLS_TEARDOWN_UNSPECIFIED            26
/* IEEE 802.11e */
#define WLAN_REASON_DISASSOC_LOW_ACK                     34
#define WLAN_REASON_EXCEEDED_TXOP                        35
#define WLAN_REASON_STA_LEAVING                          36
#define WLAN_REASON_END_TS_BA_DLS                        37
#define WLAN_REASON_UNKNOWN_TS_BA                        38
#define WLAN_REASON_TIMEOUT                              39
#define WLAN_REASON_PEERKEY_MISMATCH                     45
#define WLAN_REASON_AUTHORIZED_ACCESS_LIMIT_REACHED      46
#define WLAN_REASON_EXTERNAL_SERVICE_REQUIREMENTS        47
#define WLAN_REASON_INVALID_FT_ACTION_FRAME_COUNT        48
#define WLAN_REASON_INVALID_PMKID                        49
#define WLAN_REASON_INVALID_MDE                          50
#define WLAN_REASON_INVALID_FTE                          51
#define WLAN_REASON_MESH_PEERING_CANCELLED               52
#define WLAN_REASON_MESH_MAX_PEERS                       53
```

#define WLAN_REASON_MESH_CONFIG_POLICY_VIOLATION	54
#define WLAN_REASON_MESH_CLOSE_RCVD	55
#define WLAN_REASON_MESH_MAX_RETRIES	56
#define WLAN_REASON_MESH_CONFIRM_TIMEOUT	57
#define WLAN_REASON_MESH_INVALID_GTK	58
#define WLAN_REASON_MESH_INCONSISTENT_PARAMS	59
#define WLAN_REASON_MESH_INVALID_SECURITY_CAP	60
#define WLAN_REASON_MESH_PATH_ERROR_NO_PROXY_INFO	61
#define WLAN_REASON_MESH_PATH_ERROR_NO_FORWARDING_INFO	62
#define WLAN_REASON_MESH_PATH_ERROR_DEST_UNREACHABLE	63
#define WLAN_REASON_MAC_ADDRESS_ALREADY_EXISTS_IN_MBSS	64
#define WLAN_REASON_MESH_CHANNEL_SWITCH_REGULATORY_REQ	65
#define WLAN_REASON_MESH_CHANNEL_SWITCH_UNSPECIFIED	66

Appendix F Fast Reconnect Function

When Wi-Fi STA tries to connect to an AP again after waking up from Sleep Mode 2, it needs time to establish a Wi-Fi connection. To shorten the time of reconnection, simply use Fast Reconnect function, which is available without additional setup when AT command feature is enabled.

F.1 Technical Overview

- **Direct Probe Request for Wi-Fi SCAN**

During the Wi-Fi SETUP processing after running ATF command, associated channel number will be saved in NVRAM automatically after success Wi-Fi connect.
After saving the connected channel number to NVRAM, when reconnecting to Wi-Fi is attempted, the total Wi-Fi SCAN time to connect is reduced because only the registered channel number is scanned without performing full-channel scan for Wi-Fi connection.

- **Network Address without DHCP Client Procedure**

The DA16200/DA16600 obtains an IP address by DHCP Client procedure when the first Wi-Fi connection is completed after running ATF command. The DHCP Client operation may take a lot of time in some cases.
To reduce DHCP Client procedure time, the DA16200/DA16600 saves the IP address, subnet mask, gateway address, and DNS address information in NVRAM after a successful DHCP client procedure, and changes to STATIC IP mode internally. STATIC IP mode removes DHCP processing time and improves connection speed.

Appendix G Bluetooth® LE Coexistence Feature

The Bluetooth® LE Coexistence feature is defined as follows:

- DA16200 AT command image: Bluetooth® LE Coexistence feature is disabled
- DA16600 AT command image: Bluetooth® LE Coexistence feature is enabled (3-Pin interface).

If the Bluetooth® LE Coexistence feature or the 1-Pin interface are required, the SDK must be rebuilt. For more details see Ref. [2].

Appendix H Wi-Fi Passive-Scan

A client can use two scanning methods: active and passive. During an active scan, the client radio sends a probe request and receives a probe response from an AP. With a passive scan, the client radio listens for beacons periodically sent by the AP on each channel. A passive scan generally takes more time, since the client must listen and wait for a beacon, than actively probing to find an AP. Another limitation with a passive scan is that if the client does not wait long enough on a channel, then the client may miss an AP beacon.
The DA16200/DA16600 supports active scan and passive scan. It accepts both probe responses and beacons. The DA16200 passive scan consists of frequency and time remaining on the channel, and the result is delivered to the host firmware within 10 ms.

H.1 Passive-Scan with Specified Channel and Scan-Time Limit

The Wi-Fi component should be able to perform a passive Wi-Fi scan that only scans a given list of channels in given time limit. The DA16200 provides passive scan command with ATCMD.

- Related command is AT+WFPSCAN

H.2 Passive-Scan Result

During a passive scan, the Wi-Fi component should be able to report each beacon signal to the host firmware within 10 ms received because of the passive scan. The format which is reported to host firmware is:

- BSSID SSID RSSI Security Type Wi-Fi Channel

H.3 Passive-Scan Stop

The Wi-Fi component should be able to stop an ongoing passive Wi-Fi scan and any power use associated with the scan within 100 ms of receiving a Wi-Fi beacon signal that meets the following criteria. Beacon BSSID matches given pattern AND either of the following:

- Beacon RSSI is greater than the minimum threshold
  OR
- Beacon RSSI is less than the maximum threshold

When it stops scanning by condition, it prints out “+PSCAN:CONDITIONMET”.
Related commands are “AT+WFPCDTMIN”, “AT+WFCDTMAX” and “AT+WFPSTOP”.

H.4 Passive-Scan Sequence

This section describes basic procedure for passive scan between the DA16200/DA16600 and a PC. Run the DA16200 AT commands on a serial terminal application on the local PC. The terminal must be connected to the UART1 interface of the DA16200.

1. Enable ATCMD feature in “config_generic_sdk.h”:
   #define __SUPPORT_ATCMD__
2. Set passive scan condition using ATCMD:
   ATCMD: AT+WFPCDTMIN=72:5d:cc:d0:82:bc,-80
3. Start passive scan using ATMD:
   ATCMD: AT+WFPSCAN=120000,1,3,5
4. Stop passive scan using ATMD:
   ATCMD: AT+WFPSTOP
5. Check passive scan report; see Figure 30.

Figure 30: Passive-scan result

Appendix I Detailed Error Codes for AT Command

Table 57: AT command error codes

Category	Value	Error code	Description
Common	0	AT_CMD_ERR_CMD_OK	OK, no error
	-1	AT_CMD_ERR_UNKNOWN_CMD	Unknown command
	-2	AT_CMD_ERR_INSUFFICENT_ARGS	Insufficient parameter
	-3	AT_CMD_ERR_TOO_MANY_ARGS	Too many parameters
	-4	AT_CMD_ERR_WRONG_ARGUMENTS	Wrong parameter value
	-5	AT_CMD_ERR_NOT_SUPPORTED	Unsupported function
	-6	AT_CMD_ERR_NOT_CONNECTED	Not connected to an AP
	-7	AT_CMD_ERR_NO_RESULT	No result
	-8	AT_CMD_ERR_TOO_LONG_RESULT	Response buffer overflow
	-9	AT_CMD_ERR_INSUFFICENT_CONFIG	Function is not configured
	-10	AT_CMD_ERR_TIMEOUT	Command timeout
	-11	AT_CMD_ERR_NVR_WRITE	NVRAM write failure
	-12	AT_CMD_ERR_RTM_WRITE	Retention memory write failure
	-13	AT_CMD_ERR_SYS_BUSY	System busy
	-14	AT_CMD_ERR_MEM_ALLOC	Memory allocation failure
	-20	AT_CMD_ERR_DATA_TX	Data Tx failure
	-22	AT_CMD_ERR_IP_ADDRESS	IP address get failure
	-100	AT_CMD_ERR_COMMON_SYS_MODE	Wrong system running mode
	-110	AT_CMD_ERR_COMMON_ARG_TYPE	Wrong argument type
	-111	AT_CMD_ERR_COMMON_ARG_RANGE	Argument int-value range error
	-112	AT_CMD_ERR_COMMON_ARG_LEN	Argument value length error
	-113	AT_CMD_ERR_COMMON_WRONG_CC	Wrong country-code
	-114	AT_CMD_ERR_COMMON_WRONG_MAC_ADDR	Wrong MAC address
FLASH	-170	AT_CMD_ERR_SFLASH_READ	SFLASH driver read failure
	-171	AT_CMD_ERR_SFLASH_WRITE	SFLASH drive write failure
	-172	AT_CMD_ERR_SFLASH_ERASE	SFLASH driver erase failure
	-173	AT_CMD_ERR_SFLASH_ACCESS	SFLASH driver access failure
NVRAM	-180	AT_CMD_ERR_NVRAM_READ	NVRAM driver read failure
	-181	AT_CMD_ERR_NVRAM_WRITE	NVRAM driver write failure
	-182	AT_CMD_ERR_NVRAM_ERASE	NVRAM driver erase failure
	-183	AT_CMD_ERR_NVRAM_DIGIT	-
	-184	AT_CMD_ERR_NVRAM_SAME_M AC	-
	-185	AT_CMD_ERR_NVRAM_CANCEL ED	-
	-186	AT_CMD_ERR_NVRAM_INVALID	-
	-187	AT_CMD_ERR_NVRAM_UNKNO WN	-
	-188	AT_CMD_ERR_NVRAM_NOT_SAVED_VALUE	NVRAM name does not exist
Basic	-200	AT_CMD_ERR_BASIC_ARG_NULL_PTR	Not used in AT command module
	-201	AT_CMD_ERR_BASIC_ARG_DATE	Argument “Date” format failure
	-202	AT_CMD_ERR_BASIC_ARG_TIME	Argument “Time” format failure
	-203	AT_CMD_ERR_BASIC_ARG_TIME_ETC	Argument Time value failure
UART	-220	AT_CMD_ERR_UART_INTERFACE	Not defined UART type
	-221	AT_CMD_ERR_UART_BAUDRATE	Argument “BaudRate” failure
	-222	AT_CMD_ERR_UART_DATABITS	Argument “DataBits” failure
	-223	AT_CMD_ERR_UART_PARITY	Argument “Parity” failure

Category	Value	Error code	Description
	-224	AT_CMD_ERR_UART_STOPBIT	Argument “StopBits” failure
	-225	AT_CMD_ERR_UART_FLOWCTRL	Argument “FlowCtrl” failure
	-226	AT_CMD_ERR_UART_BAUDRATE_NV_WR	NVRAM Write failure - Baudrate
	-227	AT_CMD_ERR_UART_DATABITS_NV_WR	NVRAM Write failure - DataBits
	-228	AT_CMD_ERR_UART_PARITY_NV_WR	NVRAM Write failure - Parity
	-229	AT_CMD_ERR_UART_STOPBIT_NV_WR	NVRAM Write failure - StopBit
	-230	AT_CMD_ERR_UART_FLOWCTRL_NV_WR	NVRAM Write failure - FlowCtrl
DPM	-300	AT_CMD_ERR_DPM_MODE_DISABLED	DPM operation is not enabled
	-301	AT_CMD_ERR_DPM_SLEEP_STARTED	DPM sleep function is already running
	-302	AT_CMD_ERR_DPM_FAST_CONN_EN	Fast-connection function is enabled
	-303	AT_CMD_ERR_DPM_USER_RTM_ALLOC	Failed to allocate memory in user area of RTM
	-304	AT_CMD_ERR_DPM_USER_RTM_DUP	Same task name already exists
	-305	AT_CMD_ERR_DPM_MODE_ARG	Wrong argument type: DPM flag
	-306	AT_CMD_ERR_DPM_NVRAM_FLAG_ARG	Wrong argument type: NVRSM flag
	-309	AT_CMD_ERR_DPM_SLP2_PERIOD_TYPE	Wrong argument type: Period
	-310	AT_CMD_ERR_DPM_SLP2_PERIOD_RANGE	Wrong argument value range: Period
	-311	AT_CMD_ERR_DPM_SLP2_RTM_FLAG_ARG	Wrong argument type: RTM flag
	-312	AT_CMD_ERR_DPM_SLP1_RTM_FLAG_RANGE	Wrong argument value range: RTM flag
	-313	AT_CMD_ERR_DPM_SLP1_RTM_FLAG_ARG	Wrong argument type: RTM flag
	-314	AT_CMD_ERR_DPM_SLP1_RTM_FLAG_RANGE	Wrong argument value range: RTM flag
	-315	AT_CMD_ERR_DPM_ABN_ARG	Wrong argument type: DPMABN
	-316	AT_CMD_ERR_DPM_SLP2_DPM_MODE_ENABLED	Wrong system mode: DPM mode
	-317	AT_CMD_ERR_DPM_SLP3_PERIOD_TYPE	Wrong argument t type: Period
	-318	AT_CMD_ERR_DPM_SLP3_PERIOD_RANGE	Wrong argument value range: Period
Wi-Fi	-400	AT_CMD_ERR_WIFI_NOT_CONNECTED	Not connected to AP
	-401	AT_CMD_ERR_WIFI_RUN_MODE_TYPE	Wrong argument type
	-402	AT_CMD_ERR_WIFI_RUN_MODE_RANGE	Wrong argument value range
	-403	AT_CMD_ERR_WIFI_MAC_ADDR	Wrong string type for MAC address
	-404	AT_CMD_ERR_WIFI_WPS_PIN_NUM	Wrong PIN number for WPS connection
	-406	AT_CMD_ERR_WIFI_SCAN_UNSUPPORTED	SCAN command not supported
	-407	AT_CMD_ERR_WIFI_PSCAN_FREQ_RANGE	Wrong argument value range: Frequency

Category	Value	Error code	Description
	-408	AT_CMD_ERR_WIFI_PSCAN_CMAX_RANGE	Wrong argument value: Max RSSI threshold
	-409	AT_CMD_ERR_WIFI_PSCAN_CMIN_RANGE	Wrong argument value: Min RSSI threshold
	-410	AT_CMD_ERR_WIFI_JAP_SSID_NO_VALUE	SSID information not found in NVRAM
	-411	AT_CMD_ERR_WIFI_JAP_SSID_LEN	Too long SSID string (Max length: 32 bytes)
	-412	AT_CMD_ERR_WIFI_JAP_SECU_ARG_TYPE	Wrong argument type: Auth
	-413	AT_CMD_ERR_WIFI_JAP_SECU_ARG_RANGE	Wrong argument value range: Auth
	-414	AT_CMD_ERR_WIFI_JAP_OPEN_TOO_MANY_ARG	Too many arguments for OPEN-mode
	-415	AT_CMD_ERR_WIFI_JAP_OPEN_HIDDEN_TYPE	Wrong argument type (OPEN): Hidden flag
	-416	AT_CMD_ERR_WIFI_JAP_OPEN_HIDDEN_RANGE	Wrong argument value (OPEN): Hidden flag
	-417	AT_CMD_ERR_WIFI_JAP_SECU_HIDDEN_TYPE	Wrong argument type (Security): Hidden flag
	-418	AT_CMD_ERR_WIFI_JAP_SECU_HIDDEN_RANGE	Wrong argument value (Security): Hidden flag
	-419	AT_CMD_ERR_WIFI_JAP_WEP_IDX_TYPE	Wrong argument type: WEP Index
	-420	AT_CMD_ERR_WIFI_JAP_WEP_IDX_RANGE	Wrong argument value range: WEP Index
	-421	AT_CMD_ERR_WIFI_JAP_WEP_KEY_LEN	Wrong argument: WEP key length
	-422	AT_CMD_ERR_WIFI_JAP_WPA_MODE_TYPE	Wrong argument type: Encrypt
	-423	AT_CMD_ERR_WIFI_JAP_WPA_MODE_RANGE	Wrong argument value range: Encrypt
	-424	AT_CMD_ERR_WIFI_JAP_WPA_KEY_LEN	Wrong argument: WPA PSK length
	-425	AT_CMD_ERR_WIFI_JAPA_SSID_NO_VALUE	SSID information not found in NVRAM
	-426	AT_CMD_ERR_WIFI_JAPA_SSID_LEN	Too long SSID string (Max length: 32 bytes)
	-427	AT_CMD_ERR_WIFI_JAPA_PSK_LEN	Wrong argument: WPA PSK length
	-428	AT_CMD_ERR_WIFI_JAPA_WEP_NOT_SUPPORT	Not supported security mode: WEP-mode
	-429	AT_CMD_ERR_WIFI_JAPA_HIDDEN_TYPE	Wrong argument type: Hidden flag
	-430	AT_CMD_ERR_WIFI_JAPA_HIDDEN_RANGE	Wrong argument value range: Hidden flag
	-431	AT_CMD_ERR_WIFI_JAPA_WPA3_MODE_TYPE	Wrong argument type: WPA3 flag
	-432	AT_CMD_ERR_WIFI_JAPA_WPA3_MODE_RANGE	Wrong argument value range: WPA3 flag
	-433	AT_CMD_ERR_WIFI_JAPA_WPA3_HIDDEN_TYPE	Wrong argument type: Hidden flag
	-434	AT_CMD_ERR_WIFI_JAPA_WPA3_HIDDEN_RANGE	Wrong argument value range: Hidden flag
	-435	AT_CMD_ERR_WIFI_ROAP_ROAM_TYPE	Wrong argument type
	-436	AT_CMD_ERR_WIFI_ROAP_ROAM_RANGE	Wrong argument value range

Category	Value	Error code	Description
	-437	AT_CMD_ERR_WIFI_ENTAP_SSID_NO_VALUE	SSID information not found in NVRAM
	-438	AT_CMD_ERR_WIFI_ENTAP_SSID_LEN	Too long SSID string (Max length: 32 bytes)
	-439	AT_CMD_ERR_WIFI_ENTAP_AUTH0_UNSUPPORT	Unsupported security mode
	-440	AT_CMD_ERR_WIFI_ENTAP_ENC0_UNSUPPORT	Unsupported encrypt mode
	-441	AT_CMD_ERR_WIFI_ENTAP_EAP_PHASE1	Unsupported EAP Phase #1 value
	-442	AT_CMD_ERR_WIFI_ENTAP_EAP_PHASE2	Wrong argument value range: EAP Phase #2
	-443	AT_CMD_ERR_WIFI_ENTAP_SECU_MODE	Wrong argument value range: Auth
	-444	AT_CMD_ERR_WIFI_ENTAP_ENC_MODE	Wrong argument value range: Encrypt
	-445	AT_CMD_ERR_WIFI_ENTAP_EAP_MODE	Wrong argument value range: EAP Phase #1
	-446	AT_CMD_ERR_WIFI_ENTAP_EAP_ID_NO_VALUE	Login ID information not found in NVRAM
	-447	AT_CMD_ERR_WIFI_ENTAP_EAP_ID_LEN	Too long ID string (Max length: 64 bytes)
	-448	AT_CMD_ERR_WIFI_ENTAP_EAP_PWD_LEN	Too long PWD string (Max length: 64 bytes)
	-449	AT_CMD_ERR_WIFI_SOFTAP_SSID_NO_VALUE	SSID for Soft AP not found in NVRAM
	-450	AT_CMD_ERR_WIFI_SOFTAP_SECU_MODE	Wrong argument value: Security
	-451	AT_CMD_ERR_WIFI_SOFTAP_ENC_MODE	Wrong argument value range: Encrypt
	-452	AT_CMD_ERR_WIFI_SOFTAP_CH_VALUE_TYPE	Wrong argument type: Channel
	-453	AT_CMD_ERR_WIFI_SOFTAP_CH_VALUE_RANGE	Wrong argument value range: Channel
	-454	AT_CMD_ERR_WIFI_SOFTAP_OPEN_TOO_MANY_ARG	Too many arguments for OPEN-mode
	-455	AT_CMD_ERR_WIFI_SOFTAP_CH_TX_PWR_VALUE	Wrong channel Tx-power value
	-456	AT_CMD_ERR_WIFI_SOFTAP_WEP_NOT_SUPPORT	Unsupported security mode on Soft AP
	-457	AT_CMD_ERR_WIFI_SOFTAP_ENC_MODE_TYPE	Wrong argument type: Encrypt
	-458	AT_CMD_ERR_WIFI_SOFTAP_ENC_MODE_RANGE	Wrong argument value range: Encrypt
	-459	AT_CMD_ERR_WIFI_SOFTAP_PASSKEY_LEN	Too short/long PSK length (Length: 8 ~ 63 bytes)
	-460	AT_CMD_ERR_WIFI_ALREADY_CONNECTED	Wi-Fi session already connected
	-461	AT_CMD_ERR_WIFI_CONCURRENT_NO_PROFILE	Concurrent-mode profile information not found in NVRAM
	-462	AT_CMD_ERR_WIFI_PSCAN_DURATION	Duration value out of range
	-463	AT_CMD_ERR_WIFI_JAPA_WPA3_PSK_LEN	Too short/long PSK length (Length: 8 ~ 63 bytes)
	-464	AT_CMD_ERR_WIFI_SOFTAP_OWE_TOO_MANY_ARG	Too many arguments for OWE

Category	Value	Error code	Description
	-465	AT_CMD_ERR_WIFI_SOFTAP_SSID_LEN	Too long SSID string (Max length: 32 bytes)
	-466	AT_CMD_ERR_WIFI_ENTAP_WPA_HIDDEN_TYPE	Wrong argument type: Hidden flag
	-467	AT_CMD_ERR_WIFI_ENTAP_WPA_HIDDEN_RANGE	Wrong argument value range: Hidden flag
CLI	-500	AT_CMD_ERR_WIFI_CLI_STATUS	Failed to run “cli status” command
	-501	AT_CMD_ERR_WIFI_CLI_SET_NETWORK	Failed to run “cli set_network 0”
	-502	AT_CMD_ERR_WIFI_CLI_SET_NETWORK_HIDDEN	Failed to run “cli set_network 0” w/hidden flag
	-503	AT_CMD_ERR_WIFI_CLI_SELECT_NETWORK	Failed to run “cli select_network 0”
	-504	AT_CMD_ERR_WIFI_CLI_SAVE_CONF	Failed to run “cli save_config”
	-505	AT_CMD_ERR_WIFI_CLI_SAVE_CONF_HIDDEN	Failed to run “cli save_config” w/hidden flag
	-506	AT_CMD_ERR_WIFI_CLI_DISCONNECT	Failed to run “cli disconnect”
	-507	AT_CMD_ERR_WIFI_CLI_DEAUTHENTICATE	Failed to run “cli deauthenticate”
	-508	AT_CMD_ERR_WIFI_CLI_DISASSOCIATE	Failed to run “cli disassociate”
	-510	AT_CMD_ERR_WIFI_CLI_WPS_PBC_ANY	Failed to run “cli wps_pbc any”
	-511	AT_CMD_ERR_WIFI_CLI_WPS_PIN_GET	Failed to run “cli wps_pin get”
	-512	AT_CMD_ERR_WIFI_CLI_WPS_PIN_ANY	Failed to run “cli wps_pin any”
	-513	AT_CMD_ERR_WIFI_CLI_WPS_PIN_NUM	Wrong argument: PIN value (Length: 8 bytes)
	-514	AT_CMD_ERR_WIFI_CLI_WPS_CANCEL	Failed to run “cli wps_cancel”
	-515	AT_CMD_ERR_WIFI_CLI_COUNTRY	Failed to run “cli country”
	-516	AT_CMD_ERR_WIFI_CLI_PSCAN_CH_TL	Failed to run “cli passive_scan chan_time_limit”
	-517	AT_CMD_ERR_WIFI_CLI_PSCAN_STOP	Failed to run “cli passive_scan_stop”
	-518	AT_CMD_ERR_WIFI_CLI_PSCAN_CMAX_GET	Failed to run “cli passive_scan_condition_max”
	-519	AT_CMD_ERR_WIFI_CLI_PSCAN_CMAX_SET	Failed to run “cli passive_scan_condition_max …”
	-520	AT_CMD_ERR_WIFI_CLI_PSCAN_CMIN_GET	Failed to run “cli passive_scan_condition_min”
	-521	AT_CMD_ERR_WIFI_CLI_PSCAN_CMIN_SET	Failed to run “cli passive_scan_condition_min …”
	-522	AT_CMD_ERR_WIFI_CLI_SOFTAP_START	Failed to run “cli ap start”
	-523	AT_CMD_ERR_WIFI_CLI_SOFTAP_STOP	Failed to run “cli ap stop”
	-524	AT_CMD_ERR_WIFI_CLI_SOFTAP_RESTART	Failed to run “cli ap restart”
Network basic	-600	AT_CMD_ERR_NW_NET_IF_NOT_INITIALIZE	Network interface does not initialize
	-601	AT_CMD_ERR_NW_NET_IF_IS_DOWN	Network interface is DOWN
	-602	AT_CMD_ERR_NW_IP_IFACE_TYPE	Wrong argument type: interface
	-603	AT_CMD_ERR_NW_IP_IFACE_RANGE	Wrong argument value range: interface
	-604	AT_CMD_ERR_NW_IP_ADDR_CLASS	Invalid IP address class

Category	Value	Error code	Description
	-605	AT_CMD_ERR_NW_IP_INVALID_ADDR	Invalid IP address type
	-606	AT_CMD_ERR_NW_IP_NETMASK	Invalid Netmask address type
	-607	AT_CMD_ERR_NW_IP_GATEWAY	Invalid Gateway address type
	-608	AT_CMD_ERR_NW_DNS_A_QUERY_FAIL	Failed to get IP address by DNS Query
	-609	AT_CMD_ERR_NW_PING_IFACE_ARG_TYPE	Wrong argument type: Interface
	-610	AT_CMD_ERR_NW_PING_IFACE_ARG_RANGE	Wrong argument value range: Interface
	-611	AT_CMD_ERR_NW_PING_DST_ADDR	Invalid destination IP address
	-612	AT_CMD_ERR_NW_PING_TX_COUNT	Wrong argument: Ping Tx count
DHCP client	-613	AT_CMD_ERR_NW_DHCPC_START_FAIL	Failed to start DHCP client
	-614	AT_CMD_ERR_NW_DHCPC_HOSTNAME_LEN	Too long DHCP hostname (Max length: 32 bytes)
	-615	AT_CMD_ERR_NW_DHCPC_HOSTNAME_TYPE	Wrong format for DHCP hostname
DHCP server	-616	AT_CMD_ERR_NW_DHCPS_START_ADDR_NOT_EXIST	IP pool start-address not found in NVRAM
	-617	AT_CMD_ERR_NW_DHCPS++_END_ADDR_NOT_EXIST	IP pool end-address not found in NVRAM
	-618	AT_CMD_ERR_NW_DHCPS_WRONG_START_IP_CLASS	Invalid start IP address class
	-619	AT_CMD_ERR_NW_DHCPS_WRONG_END_IP_CLASS	Invalid end IP address class
	-620	AT_CMD_ERR_NW_DHCPS_IPADDR_RANGE_MISMATCH	Mismatch IP address class range
	-621	AT_CMD_ERR_NW_DHCPS_IPADDR_RANGE_OVERFLOW	Exceed IP address pool count (Max. 10)
	-622	AT_CMD_ERR_NW_DHCPS_NO_CONNECTED_CLIENT	No connected client information to DHCP server
	-623	AT_CMD_ERR_NW_DHCPS_RUN_FLAG_TYPE	Wrong argument type: dhcpd flag
	-624	AT_CMD_ERR_NW_DHCPS_RUN_FLAG_VAL	Wrong argument value range: dhcpd flag
	-625	AT_CMD_ERR_NW_DHCPS_LEASE_TIME_TYPE	Wrong argument type: dhcpd lease_time
	-626	AT_CMD_ERR_NW_DHCPS_LEASE_TIME_RANGE	Wrong argument value range: dhcpd lease_time
SNTP client	-629	AT_CMD_ERR_NW_SNTP_NOT_SUPPORTED	SNTP client does not supported
	-630	AT_CMD_ERR_NW_SNTP_FLAG_TYPE	Wrong argument type: SNTP flag
	-631	AT_CMD_ERR_NW_SNTP_FLAG_VAL	Wrong argument value range: SNTP flag
	-632	AT_CMD_ERR_NW_SNTP_PERIOD_TYPE	Wrong argument type: SNTP period
	-633	AT_CMD_ERR_NW_SNTP_PERIOD_RANGE	Wrong argument value range: SNTP period
MQTT client	-634	AT_CMD_ERR_NW_MQTT_NOT_CONNECTED	MQTT client is currently not connected
	-635	AT_CMD_ERR_NW_MQTT_NEED_TO_STOP	Need to disconnect the already connected MQTT session
	-636	AT_CMD_ERR_NW_MQTT_UNKNOWN_OP_ID	Input not supported

Category	Value	Error code	Description
	-637	AT_CMD_ERR_NW_MQTT_CLIENT_TASK_START	MQTT client start failed by unknown reason
MQTT broker	-638	AT_CMD_ERR_NW_MQTT_BROKER_NAME_NOT_FOUND	MQTT broker name not found
	-639	AT_CMD_ERR_NW_MQTT_BROKER_PORT_NUM_TYPE	Wrong argument type: MQTT Broker port
	-640	AT_CMD_ERR_NW_MQTT_BROKER_PORT_NUM_RANGE	Wrong argument value range: MQTT Broker port
	-641	AT_CMD_ERR_NW_MQTT_BROKER_NAME_LEN	MQTT Broker name string: max length (MQTT_BROKER_MAX_LEN) exceeded
MQTT TLS	-642	AT_CMD_ERR_NW_MQTT_TLS_TYPE	Wrong argument type: tls
	-643	AT_CMD_ERR_NW_MQTT_TLS_RANGE	Wrong argument value range: tls
	-644	AT_CMD_ERR_NW_MQTT_TLS_ALPN_NOT_EXIST	ALPN information not found in NVRAM
	-645	AT_CMD_ERR_NW_MQTT_TLS_ALPN_COUNT_TYPE	Wrong argument type: count
	-646	AT_CMD_ERR_NW_MQTT_TLS_ALPN_COUNT_RANGE	Wrong argument value range: count (1 ~ 3)
	-647	AT_CMD_ERR_NW_MQTT_TLS_ALPN_NAME_LEN	Too long ALPN name length (Max: 24 bytes)
	-648	AT_CMD_ERR_NW_MQTT_TLS_SNI_NOT_EXIST	SNI information not found in NVRAM
	-649	AT_CMD_ERR_NW_MQTT_TLS_SNI_LEN	Too long SNI string length (Max: 64 bytes)
	-650	AT_CMD_ERR_NW_MQTT_TLS_CSUITE_NUM_NOT_EXIST	CipherSuite count value not found in NVRAM
	-651	AT_CMD_ERR_NW_MQTT_TLS_CSUITE_NOT_EXIST	CipherSuite information not found in NVRAM
	-652	AT_CMD_ERR_NW_MQTT_TLS_CSUITE_NUM_NVRAM_WR	Failed to write Cipher Suit count info to NVRAM
	-653	AT_CMD_ERR_NW_MQTT_TLS_CSUITE_NVRAM_WR	Failed to write Cipher Suit info to NVRAM
MQTT sub-topic	-654	AT_CMD_ERR_NW_MQTT_SUBS_TOPIC_NOT_EXIST	SUB topic does not exist in NVRAM
	-655	AT_CMD_ERR_NW_MQTT_SUBS_TOPIC_NUM_TYPE	Wrong argument type: count
	-656	AT_CMD_ERR_NW_MQTT_SUBS_TOPIC_NUM_RANGE	Wrong argument value range: count (1~4)
	-657	AT_CMD_ERR_NW_MQTT_SUBS_TOPIC_LEN	Too long topic string length (Max: 64 bytes)
	-658	AT_CMD_ERR_NW_MQTT_SUBS_TOPIC_DUP	Duplicate Sub-topic string
	-659	AT_CMD_ERR_NW_MQTT_SUBS_TOPIC_NUM_NVRAM_WR	Failed to write topic count to NVRAM
	-660	AT_CMD_ERR_NW_MQTT_SUBS_TOPIC_NUM_OVERFLOW	Adding a topic exceeds the max topic count (4)
	-661	AT_CMD_ERR_NW_MQTT_SUBS_TOPIC_ALREADY_EXIST	Subscribe topic already exists
MQTT pub-topic	-662	AT_CMD_ERR_NW_MQTT_PUB_TOPIC_NOT_EXIST	PUB topic not found in NVRAM
	-663	AT_CMD_ERR_NW_MQTT_PUB_TOPIC_LEN	Too long topic string length (Max: 64 bytes)
MQTT WILL message	-664	AT_CMD_ERR_NW_MQTT_WILL_TOPIC_NOT_EXIST	WILL topic does not exist in NVRAM
	-665	AT_CMD_ERR_NW_MQTT_WILL_MESSAGE_NOT_EXIST	WILL message not existing in NVRAM

Category	Value	Error code	Description
	-666	AT_CMD_ERR_NW_MQTT_WILL_TOPIC_LEN	Too long WILL topic length (Max: 64 bytes)
	-667	AT_CMD_ERR_NW_MQTT_WILL_MESSAGE_LEN	Too long WILL message length (Max: 64 bytes)
	-668	AT_CMD_ERR_NW_MQTT_WILL_QOS_TYPE	Wrong argument type: qos
	-669	AT_CMD_ERR_NW_MQTT_WILL_QOS_RANGE	Wrong argument value range: qos
MQTT common	-670	AT_CMD_ERR_NW_MQTT_PROTOCOL	Network protocol error occurred with Broker
	-671	AT_CMD_ERR_NW_MQTT_PING_PERIOD_TYPE	Invalid Ping Period value is invalid
	-672	AT_CMD_ERR_NW_MQTT_PING_PERIOD_RANGE	Wrong argument value range: (0 ~ 86400)
	-673	AT_CMD_ERR_NW_MQTT_USERNAME_NOT_EXIST	User name does not exist in NVRAM
	-674	AT_CMD_ERR_NW_MQTT_USERNAME_LEN	Too long username length (Max: 64 bytes)
	-675	AT_CMD_ERR_NW_MQTT_PASSWORD_LEN	Too long password length (Max: 160 bytes)
	-676	AT_CMD_ERR_NW_MQTT_PUB_MESSAGE_LEN	Too long message length (Max: 2048 bytes)
	-677	AT_CMD_ERR_NW_MQTT_PUB_TX_IN_PROGRESS	Previous message Tx is still in progress
HTTP(s) server	-680	AT_CMD_ERR_NW_HTS_TASK_CREATE_FAIL	Failed to create HTTP server task
	-681	AT_CMD_ERR_NW_HTSS_TASK_CREATE_FAIL	Failed to create HTTPs server task
HTTP client	-682	AT_CMD_ERR_NW_HTC_TASK_CREATE_FAIL	Failed to create HTTP client task
	-683	AT_CMD_ERR_NW_HTC_ALPN_CNT_TYPE	Wrong argument type: alpn_number
	-684	AT_CMD_ERR_NW_HTC_ALPN_CNT_RANGE	Wrong argument value range: alpn_number
	-685	AT_CMD_ERR_NW_HTC_ALPN1_STR_LEN	Too long ALPN #1 string length (Max 24 bytes)
	-686	AT_CMD_ERR_NW_HTC_ALPN2_STR_LEN	Too long ALPN #2 string length (Max 24 bytes)
	-687	AT_CMD_ERR_NW_HTC_ALPN3_STR_LEN	Too long ALPN #3 string length (Max 24 bytes)
	-688	AT_CMD_ERR_NW_HTC_SNI_LEN	Too long SNI string length (Max 64 bytes)
Web-Socket client	-689	AT_CMD_ERR_NW_WSC_URL_STR_LEN	Too short URL string length (Min 1 byte)
	-690	AT_CMD_ERR_NW_WSC_INVALID_URL	Invalid URL string
	-691	AT_CMD_ERR_NW_WSC_TASK_ALREADY_EXIST	WebSocket session is already exist
	-692	AT_CMD_ERR_NW_WSC_CB_FUNC_DOES_NOT_EXIST	Not registered user Websocket cb-function
	-693	AT_CMD_ERR_NW_WSC_INVALID_STATE	No connected session to disconnect
	-694	AT_CMD_ERR_NW_WSC_TASK_CREATE_FAIL	Failed to create WebSocket client task
	-695	AT_CMD_ERR_NW_WSC_CLOSE_FAIL	Failed to send “Session-Close” frame

Category	Value	Error code	Description
	-696	AT_CMD_ERR_NW_WSC_SESS_NOT_CONNECTED	No connected session to send message
	-697	AT_CMD_ERR_NW_WSC_UNKNOW_CMD	Unknown WebSocket internal error
OTA	-700	AT_CMD_ERR_NW_OTA_WRONG_FW_TYPE	Wrong argument: fw_type
	-701	AT_CMD_ERR_NW_OTA_DOWN_OK_AND_WAIT_RENEW	Already downloaded
	-702	AT_CMD_ERR_NW_OTA_FLASH_READ_SIZE_TYPE	Wrong argument type: read_addr
	-703	AT_CMD_ERR_NW_OTA_FLASH_COPY_SIZE_TYPE	Wrong argument type: size
	-704	AT_CMD_ERR_NW_OTA_FLASH_ERASE_SIZE_TYPE	Wrong argument type: size
	-705	AT_CMD_ERR_NW_OTA_BY_MCU_INIT	Failed to initialize MCU configuration for OTA
	-706	AT_CMD_ERR_NW_OTA_SET_TLS_AUTH_MODE_NVRAM	Failed to save TLS certificate in NVRAM
	-707	AT_CMD_ERR_NW_OTA_SET_MCU_FW_NAME	Failed to set MCU_FW name. (Max length: 8 bytes)
Zero config	-710	AT_CMD_ERR_NW_MDNS_WRONG_FLAG	Wrong argument: flag of MDNS
	-711	AT_CMD_ERR_NW_MDNS_WRONG_MODE	Wrong argument: mode of MDNS
	-712	AT_CMD_ERR_NW_MDNS_NOT_RUNNING	MDNS is not running
	-713	AT_CMD_ERR_NW_MDNS_ALREADY_RUN	MDNS is already running
	-714	AT_CMD_ERR_NW_MDNS_IN_PROCESS	Progressing Probing and Announcing on MDNS
	-715	AT_CMD_ERR_NW_MDNS_UNKNOW_FAULT	Unknown MDNS internal error
	-716	AT_CMD_ERR_NW_MDNS_START_RUN_MODE_VAL	Invalid interface
	-717	AT_CMD_ERR_NW_MDNS_SOCKET_FAIL	Failed to initialize socket
	-718	AT_CMD_ERR_NW_DNS_SD_NOT_RUNNING	DNS-SD is not running
	-719	AT_CMD_ERR_NW_DNS_SD_ALREADY_RUN	Already DNS-SD is running
	-720	AT_CMD_ERR_NW_DNS_SD_IN_PROCESS	Progressing Probing and Announcing of DNS-SD
	-721	AT_CMD_ERR_NW_DNS_SD_SVC_CREATE_FAIL	Failed to register service of DNS-SD
	-722	AT_CMD_ERR_NW_DNS_SD_SVC_PARAMS	Invalid parameters to register service
	-723	AT_CMD_ERR_NW_DNS_SD_SVC_INST_NAME_NVRAM_WR	Failed to write service name to NVRAM
	-724	AT_CMD_ERR_NW_DNS_SD_SVN_PROTOCOL_NVRAM_WR	Failed to write service protocol to NVRAM
	-725	AT_CMD_ERR_NW_DNS_SD_SVC_PORT_NO_NVRAM_WR	Failed to write service port to NVRAM
	-726	AT_CMD_ERR_NW_DNS_SD_SVC_TEXT_NVRAM_WR	Failed to write service TXT to NVRAM
Transport function (TCP/UDP)	-730	AT_CMD_ERR_TCP_SERVER_LOCAL_PORT_TYPE	Wrong argument: local port of TCP server
	-731	AT_CMD_ERR_TCP_SERVER_MAX_PEER_TYPE	Wrong argument : max allowed peer
	-732	AT_CMD_ERR_TCP_SERVER_TASK_CREATE	Failed to start TCP server
	-733	AT_CMD_ERR_TCP_CLIENT_SVR_PORT_TYPE	Wrong argument: TCP server port of TCP client
	-734	AT_CMD_ERR_TCP_CLIENT_LOCAL_PORT_TYPE	Wrong argument: local port of TCP client
	-736	AT_CMD_ERR_TCP_CLIENT_TASK_CREATE	Failed to start TCP client

Category	Value	Error code	Description
	-737	AT_CMD_ERR_UDP_SESS_LOCAL_PORT_TYPE	Wrong argument: local port of UDP session
	-738	AT_CMD_ERR_UDP_SESS_LOCAL_PORT_RANGE	Invalid range of local port of UDP session
	-739	AT_CMD_ERR_UDP_SESS_TASK_CREATE	Failed to start UDP session
	-740	AT_CMD_ERR_UDP_CID2_SESS_NOT_EXIST	UDP session, CID 2, does not exist
	-741	AT_CMD_ERR_UDP_CID2_ALREADY_EXIST	UDP session, CID 2, already exists
	-742	AT_CMD_ERR_UDP_CID2_SESS_INFO	Invalid UDP session, CID 2, information
	-743	AT_CMD_ERR_UDP_CID2_REMODE_PORT_TYPE	Invalid remote port of UDP session, CID 2
	-744	AT_CMD_ERR_NO_CONNECTED_SESSION_EXIST	No session information
	-745	AT_CMD_ERR_NO_FOUND_REQ_CID_SESSION	No assigned CID to terminate session
	-746	AT_CMD_ERR_CONTEXT_CID_TYPE	Wrong argument type: cid
	-747	AT_CMD_ERR_CONTEXT_DELETE	Failed to terminate session
	-748	AT_CMD_ERR_CONTEXT_TYPE_IS_NOT_TCP_SVR	Wrong CID value: Not TCP server session
	-749	AT_CMD_ERR_CONTEXT_INVALID_SESS_TYPE	Invalid session type to save session information
	-750	AT_CMD_ERR_TRTRM_CID_TYPE	Wrong argument: CID to terminate session
	-751	AT_CMD_ERR_TRTRM_REMOTE_PORT_NUM_TYPE	Wrong argument type: remote_port
	-752	AT_CMD_ERR_TRTRM_TCP_SVR_REMOTE_SESS_DISCON	Failed to disconnect TCP client from TCP server
	-753	AT_CMD_ERR_TCP_SERVER_TERMINATE	Failed to terminate TCP server
	-754	AT_CMD_ERR_TCP_CLIENT_TERMINATE	Failed to terminate TCP client
	-755	AT_CMD_ERR_UDP_SESSION_TERMINATE	Failed to terminate UDP session
	-756	AT_CMD_ERR_MULTI_SESSION_CID_TERMINATE	No assigned CID to terminate session
	-757	AT_CMD_ERR_NO_SESSOIN_TO_SAVE_NVRAM	No session information to save
SSL/TLS	-760	AT_CMD_ERR_SSL_ROLE_NOT_SUPPORT	Not supported role of TLS session
	-761	AT_CMD_ERR_SSL_CONF_CID_TYPE	Wrong argument: CID of TLS session
	-762	AT_CMD_ERR_SSL_CONTEXT_NOT_FOUND	No assigned CID of TLS session
	-763	AT_CMD_ERR_SSL_CONTEXT_ALREADY_EXIST	TLS session is already running to configure
	-764	AT_CMD_ERR_SSL_CONF_ID_NOT_SUPPORTED	Not supported configuration
	-765	AT_CMD_ERR_SSL_SAVE_CLR_ALL_NV	Failed to erase TLS session from NVRAM
	-766	AT_CMD_ERR_SSL_SAVE_FAIL_NV	Failed to save TLS session to NVRAM
	-767	AT_CMD_ERR_SSL_CONF_ID_TYPE	Wrong argument: configuration ID

Category	Value	Error code	Description
	-768	AT_CMD_ERR_SSL_CONF_ID_RANGE	Invalid range of configuration ID
	-769	AT_CMD_ERR_SSL_CONF_CID_CA_CERT	CA certification does not exist for assigned CID
	-770	AT_CMD_ERR_SSL_CONF_CID_CERT	Certification does not exist for assigned CID
	-771	AT_CMD_ERR_SSL_CONF_CID_SNI	Failed to configure SNI of assigned CID
	-772	AT_CMD_ERR_SSL_CONF_CID_SVR_VALID_TYPE	Wrong argument: auth mode of assigned CID
	-773	AT_CMD_ERR_SSL_CONF_CID_SVR_VALID_RANGE	Invalid range of auth mode of assigned CID
	-774	AT_CMD_ERR_SSL_CONF_CID_RX_BUF_LEN	Wrong argument: Rx buffer length of CID
	-775	AT_CMD_ERR_SSL_CONF_CID_TX_BUF_LEN	Wrong argument: Tx buffer length of CID
	-776	AT_CMD_ERR_SSL_CONF_CID_TYPE	Wrong argument: CID to configure TLS session
	-777	AT_CMD_ERR_SSL_CONN_ALREADY_CONNECTED	Already TLS session is connected
	-778	AT_CMD_ERR_SSL_CONN_PORT_NUM_TYPE	Wrong argument: peer_port of TLS client
	-779	AT_CMD_ERR_SSL_CONN_UNKNOWN_HOSTNAME	Unknown hostname to connect TLS server
	-780	AT_CMD_ERR_SSL_CONN_CFG_SETUP_FAIL	Failed to setup TLS client
	-781	AT_CMD_ERR_SSL_CONN_TLS_CLIENT_RUN_FAIL	Failed to connect TLS client
SSL certificate	-782	AT_CMD_ERR_SSL_CERT_TYPE	Wrong argument: type
	-783	AT_CMD_ERR_SSL_CERT_RANGE	Invalid range of certificate type
	-784	AT_CMD_ERR_SSL_CERT_STO_SEQ_TYPE	Wrong argument: sequence type
	-785	AT_CMD_ERR_SSL_CERT_STO_SEQ_RANGE	Invalid range of sequence type
	-786	AT_CMD_ERR_SSL_CERT_STO_FORMAT_TYPE	Wrong argument: format type
	-787	AT_CMD_ERR_SSL_CERT_STO_FORMAT_RANGE	Invalid range of format type
	-788	AT_CMD_ERR_SSL_CERT_STO_ALREADY_EXIST	Already certificate is existed
	-789	AT_CMD_ERR_SSL_CERT_STO_NO_SPACE	Not enough space to save certificate
	-790	AT_CMD_ERR_SSL_CERT_DEL_LIST_NOT_FOUND	Not found certificate to delete
	-791	AT_CMD_ERR_SSL_CERT_MODULE	Invalid module
	-792	AT_CMD_ERR_SSL_CERT_FORMAT	Invalid format
	-793	AT_CMD_ERR_SSL_CERT_LENGTH	Invalid length
	-794	AT_CMD_ERR_SSL_CERT_FLASH_ADDR	Invalid address of sflash memory
	-795	AT_CMD_ERR_SSL_CERT_EMPTY_CERT	No certificate
	-796	AT_CMD_ERR_SSL_CERT_INTERNAL	Internal error
	-797	AT_CMD_ERR_SSL_SESS_TASK_CREATE	Failed to create SSL task
	-999	AT_CMD_ERR_UNKNOWN	Undefined Error

Appendix J AT Command Development Environment Configuration

J.1 How to Connect DA16200/DA16600 Board

This section describes the installation procedure for the drivers, the configuration of the serial port, and all necessary steps to set up and check the connection with the PC.

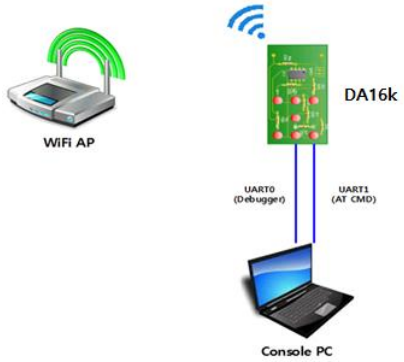


Figure 31: AT command development environment

On first connection to a host PC with Microsoft Windows as operating system, the system will detect several devices and will automatically install all necessary drivers. If the driver is not automatically installed, then get the driver from the following URL: http://www.ftdichip.com/Drivers/CDM/CDM21224_Setup.zip.

There are two virtual COM ports created by the Windows driver. The first COM port (lower number, COM69 in Figure 32) provides a UART interface for debugging or firmware download between the PC and the DA161200. The second (higher number, COM70 in Figure 32 is used for AT command.

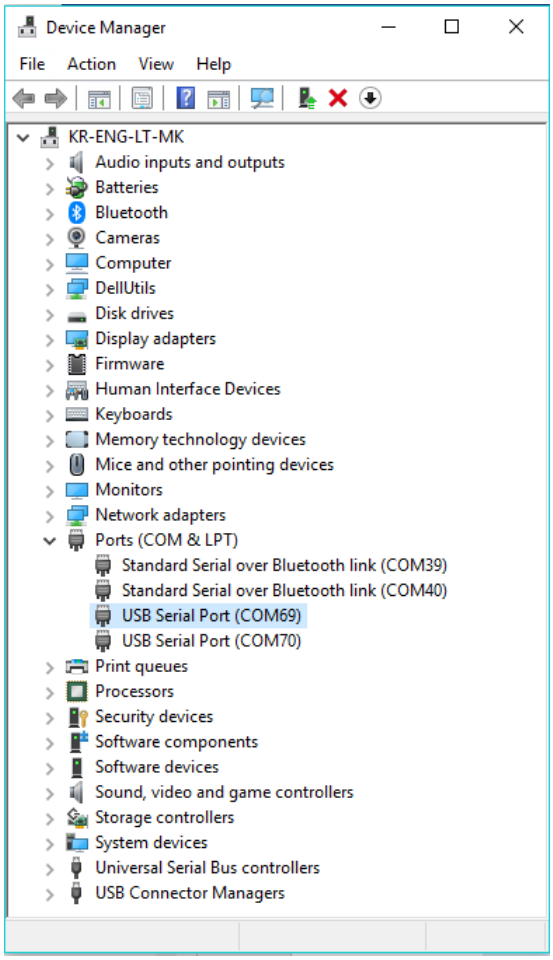


Figure 32: Check COM ports on device manager

J.2 Configure Serial Port for UART

On a Windows Host, the utility Tera Term is used to connect. See Ref. [1].

Tera Term is a free terminal emulator (communication program) that supports multiple communication including serial port connections.

1. Download Tera Term from https://ttssh2.osdn.jp.
2. Run the teraterm-x.yy.exe.
3. Follow the installation wizard.

To make sure that the communication between the DA16200/DA16600 EVK and the host PC is established correctly, check the UART connection between the two nodes. Do the following steps:

1. Use a USB cable to connect the DA16200/DA16600 EVK to the PC.
2. Make sure that the PC discovered the two serial ports in Windows Device Manager as shown in Figure 32. The higher COM port number is connected to UART1.
3. Open Tera Term from the Windows Start menu.
4. In the Tera Term: New connection Renesas Electronics:
   a. Select Serial.
   b. Select the COM Port to use.
   c. Click OK.
5. Select Setup > Serial Port and configure the UART port with the parameters as shown in Figure 33. Select the higher COM port number as discovered in step 2.

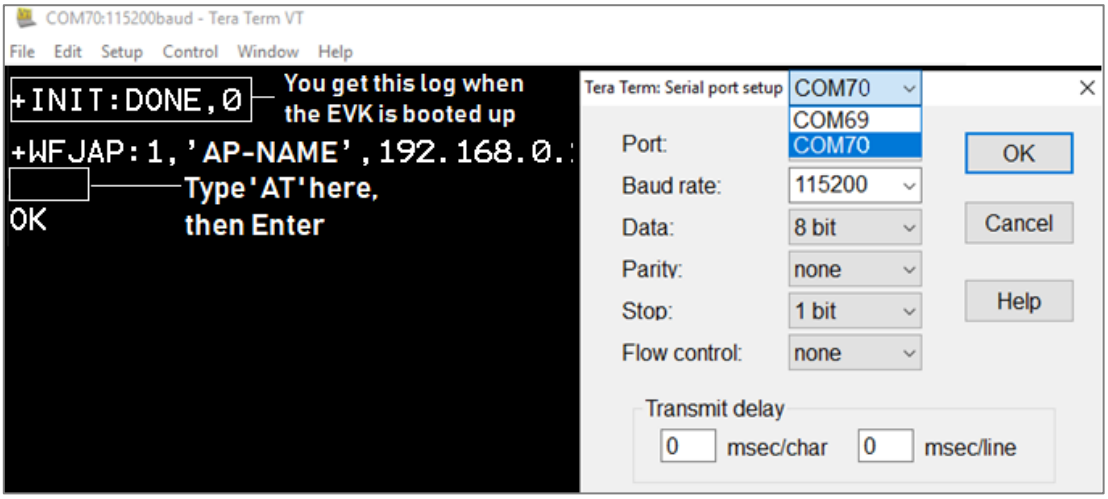


Figure 33: Initial setup to start with AT command

J.3 Configuration for MCU Wake-Up (Optional)

Depending on the application scenarios, both MCU and DA16200/DA16600 may want to be in the SLEEP state and MCU wants to be woken up (by DA16200/DA16600) when DA16200/DA16600 wakes up from DPM Sleep. This can be achieved with the MCU wake-up feature of the DA16200/DA16600.

To use the MCU wake-up feature, connect pin GPIO_11 of the DA16200/DA16600 to the wake-up pin on the MCU. Then, when the DA16200/DA16600 wakes up, GPIO_11 becomes an Output and is set to High (Active High) to trigger the wake-up of the MCU. The wake-up PIN of MCU should be configured to detect the rising edge of GPIO_11 for wake-up.

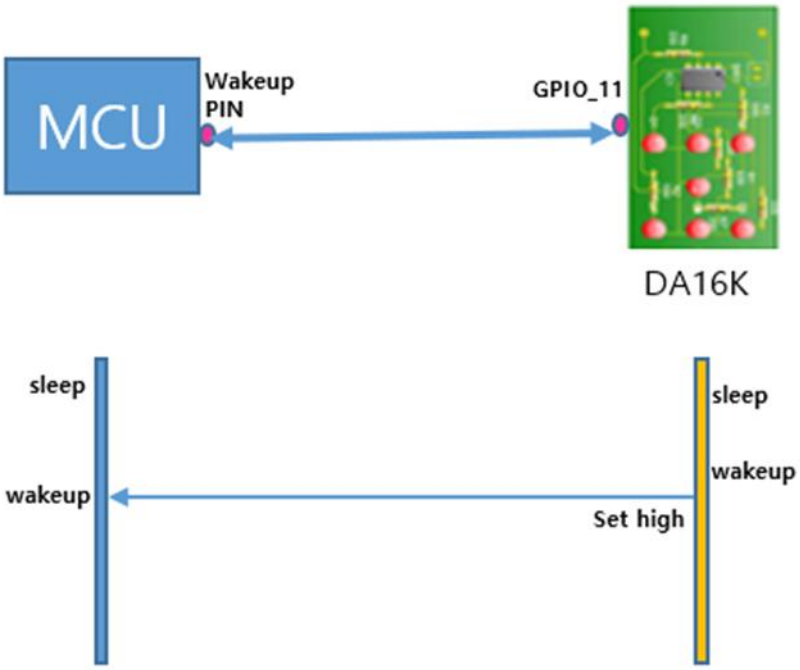


Figure 34: GPIO wake-up

7. Revision History

Revision	Date	Description
3.4	June 21, 2024	▪ Updated the description in AT+PROVSTART and AT+DPMABN
3.3	May 29, 2024	▪ Changed TCP/UDP test tool from IO Ninja to Hercules ▪ Added free TCP/UDP test tool for Linux/MacOS/Android/iOS ▪ Updated AT command error codes ▪ Updated description for ATF/factory reset, AT+WFMODE and AT+NWMQTP ▪ AT+TRSSLCFG: Updated SSL protocol version settings ▪ AT+SETSLEEP2EXT, AT+SETSLEEP3EXT: updated minimum value for <period>
3.2	Sep. 18, 2023	Updated AT command error codes
3.1	Aug. 18, 2023	▪ Updated AT+GPIOSTART and AT+LEDCTRL examples ▪ Added error code to Detailed Error Codes for AT Command ▪ AT+SETSLEEP2EXT, AT+DPMUSERWU: changed the time input unit from seconds to milliseconds ▪ AT+WFENTAP: updated hidden AP settings ▪ AT+TRSSLCERTSTORE: added data length parameter ▪ AT+TRSSLWR: fixed incorrect optional parameter ▪ AT+WFENTLI: fixed incorrect optional parameter ▪ AT+NWCCRT: added DH param for Set #1 and 2, and Set #3 for WPA Enterprise ▪ <ESC>C: added Set #3 for WPA Enterprise ▪ <ESC>CERT: added new AT command in Certificate Command table ▪ AT+SETSLEEP2EXT: description updated ▪ AT+SETSLEEP3EXT added ▪ Updated the usage of padding field in SPI protocol
3.0	Jun. 30, 2023	▪ Updated OTA commands and descriptions in Secure Socket Command List table ▪ AT+WFCC: additional note added ▪ AT+WFJAP, AT+WFSAP: additional note added on <sec> <enc> for WPA3
2.17	Apr. 10, 2023	Corrected examples and descriptions of OTA commands
2.16	Feb. 10,2023	Corrected the range of AT+SETSLEEP2EXT command
2.15	Jan. 27, 2023	Added ESC Command Sequence
2.14	Jan. 12, 2023	Merged user guides and changed the titles. ▪ UM-WI-003, DA16200 DA16600 AT Command ▪ UM-WI-020, DA16200 SPI Host Interface ▪ UM-WI-053, DA16200 SDIO Host Interface
2.13	Dec. 16, 2022	▪ AT+NWHTCH added ▪ AT+NWHTCTLSAUTH added ▪ AT+WFCC: note is added ▪ AT+NWMQMSG: note updated on max length ▪ AT+WFAPUI: description updated ▪ AT+NWIP: note added ▪ AT+WFAPCH: valid range changed ▪ AT+WFWMP: note updated ▪ AT+BLENAME: added to get the BLENAME. ▪ AT+WFJAPA3: added to connect to the WPA3-AP ▪ MQTT Commands re-arranged: split to MQTT configuration command and MQTT operation commands. ▪ Pre-requisite added for MQTT Configuration commands. ▪ Updated DA16200/DA16600 Cipher Suites ▪ New added Appendix J AT commands' error code Typo fixed in the example - AT+WFSPF, AT+WFOTP, <ESC>S
2.12	Aug. 08, 2022	▪ Updated Appendix for running AT command through SDIO or SPI.

Revision	Date	Description
		▪ Added the section of Wi-Fi Function Commands for WPA Enterprise
2.11	Jun. 14, 2022	▪ AT+SDKVER added ▪ Added the Command Format section ▪ Added AT+HOSTINITDONE ▪ Added Appendix E ▪ Added more info on AT+TZONE ▪ Info updated or typo fixed: ATQ, ATB, AT+NWDHIP, AT+WFRSSI ▪ Changed default status by some features are enabled ▪ Added AT+CHIPNAME, AT+CALWR ▪ AT+NWMQMSG : operation response added (+NWMQMSGSND) ▪ Updated +WFJAP:0 and +WFDAP:0 (reason parameter is added) ▪ Updated AT+NWMQBR, AT+NWMQTT ▪ Updated MQTT optional configuration commands (enabled by default in SDK v3.2.3.0) ▪ Added AT+NWMQCS, and CleanSession=0 Guide. ▪ AT+WFJAP / AT+WFJAPA / AT+WFSAP : passphrase range added ▪ AT+TRTS example updated ▪ Added AT+NWMQUTS ▪ Updated Note : AT+NWIP, AT+NWDHR, AT+NWDHLT, AT+NWMQMSG, <ESC>S ▪ +NWMQCL updated ▪ Added Appendix H ▪ Added AT+NWWSC ▪ Added note for <ESC>S : for ATCMD on SPI
2.10	Mar. 22, 2022	▪ Updated logo, disclaimer, and copyright. ▪ Added AT+TCPDATAMODE ▪ Added LED/PWM/ADC/I2C related AT commands
2.9	Dec. 14, 2021	▪ AT+NWMQMSG, AT+NWMQTS: updated max length info on topic and added message ▪ Added AT+NWDNS, HELP, ATE, and ATQ information
2.8	Dec. 08, 2021	▪ Updated OTP Size Reduced (8 kB->2 kB) in OTA section ▪ Updated Data Transfer Commands section ▪ Updated AT+TRSSLWR ▪ Added AT+NWMQATS, AT+NWMQDTS, AT+NWMQV311, and AT+NWDHIP ▪ Added AT+ NWHTCSNI, AT+ NWHTCALPN, AT+NWHTCSNIDEL, and AT+NWHTCALPNDEL
2.7	Nov. 25, 2021	Changed the title
2.6	Oct. 28, 2021	▪ Added AT+WFAPUI: <timeout> valid value range ▪ Updated TCP Client Socket Test section ▪ Updated TCP Server section ▪ Updated: 5th parameter removed from AT+NWDHS ▪ Added: AT+NWDNS2 ▪ AT+XTALRD, AT+FLASHDUMP: <CR><LF> is appended to data
2.5	Sep. 07, 2021	▪ Updated table format of ATCMD (Prerequisite, example, note added) ▪ Added Zeroconf Commands ▪ Added Secure Socket Commands
2.4	Jun. 17, 2021	▪ Updated MQTT commands (MQTT Client Connection Example and MQTT TLS Connection Example) ▪ AT+NWDHDNS deleted (not needed as WAN Port is not available in Soft AP mode) ▪ Added AT+NWMQALPN, AT+NWMQSNI, AT+NWMQCSUIT, AT+SETDPMSLP1EXT, AT+DPMABNWFCCNT

Revision	Date	Description
		▪ Updated AT+WFJAP, AT+WFJAPA : Optional parameter <hidden> added
2.3	Apr. 01, 2021	▪ Added OTA update command ▪ Added support for SDK V3.x.x.x
2.2	Mar. 15, 2021	▪ Added Appendix B HTTP API Return Values ▪ Added Appendix C ▪ Added AT+NWMQAUTO and ATB
2.1	Jan. 13, 2021	Added Wi-Fi Function Commands for WPA3, and updated minor changes
2.0	Dec. 08, 2020	▪ Added additional description on the following commands: ▪ AT+WFSAP, AT+WFOAP, AT+WFTAP, ATF, AT+WFJAPA, AT+NWMQTT, +NWMQCL, AT+DPM ▪ Added new sections: ▪ Added MQTT Example: Changing Subscription Topic while running ▪ Added MQTT Example: Reading Subscription Topic while running
1.9	Nov. 11, 2020	▪ AT+NWCCRT, <ESC>C updated ▪ AT+NWSNS updated ▪ AT+NWHTS updated ▪ AT+NWHTSS updated
1.8	Aug. 18, 2020	▪ Added SNTP commands in Network Function Commands ▪ Added HTTP-client command in HTTP-Client Commands ▪ Added MCU FW update command using OTA in OTA Commands
1.7	Jun. 30, 2020	▪ Added Configuration for MCU Wake-up ▪ Correct typos and wordings
1.6	Apr. 29, 2020	▪ Added AT+WFDIS and AT+SETDPMSLP2EXT ▪ Updated MQTT commands to operate with one-port ▪ Updated to process the comma in the parameters
1.5	Apr. 03, 2020	▪ Added AT+BIDX for changing boot index ▪ Added example code of MQTT commands ▪ Updated RF Test function commands ▪ Updated GPIO commands
1.4	Oct. 21, 2019	▪ Updated Serial Port configuration steps ▪ Removed draft status
1.3	Oct. 15, 2019	▪ Error correction Added explanation to serial program
1.2	Oct. 07, 2019	Editorial review and add code: UM-B-111
1.1	Jul. 25, 2019	Added OTP Memory Address for writing MAC address
1.0	Jul. 03, 2019	Preliminary DRAFT 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

Renesas Electronics' 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

RENESAS ELECTRONICS CORPORATION AND ITS SUBSIDIARIES (“RENESAS”) PROVIDES TECHNICAL SPECIFICATIONS AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT OF THIRD-PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for developers who are designing with Renesas products. You are solely responsible for (1) selecting the appropriate products for your application, (2) designing, validating, and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. Renesas grants you permission to use these resources only to develop an application that uses Renesas products. Other reproduction or use of these resources is strictly prohibited. No license is granted to any other Renesas intellectual property or to any third-party intellectual property. Renesas disclaims responsibility for, and you will fully indemnify Renesas and its representatives against, any claims, damages, costs, losses, or liabilities arising from your use of these resources. Renesas' products are provided only subject to Renesas' Terms and Conditions of Sale or other applicable terms agreed to in writing. No use of any Renesas resources expands or otherwise alters any applicable warranties or warranty disclaimers for these products.

(Disclaimer Rev.1.1 Jan 2024)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu
Koto-ku, Tokyo 135-0061, Japan

www.renesas.com

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

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 www.renesas.com/contact-us/