User Manual

DA16200 AT Command

UM-WI-003

Abstract

This document describes the use of the AT commands to configure the DA16200.

Contents

Figures

Tables

1 Terms and Definitions

AP	Access Point
ASCII	American Standard Code for Information Interchange
AT	Attention
CA	Certificate Authority
CID	Client ID
CMD	Command
COM	Communication Port
CRC	Cyclic Redundancy Check
CW	Continuous Wave
DHCP	Dynamic Host Configuration Protocol
DPM	Dynamic Power Management
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
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 memory
PBC	Push Button Connection
PC	Personal Computer
PER	Packet Error Rate
QoS	Quality of Service
RTC	Real Time Clock
RTOS	Real Time Operating System
RTS	Request to Send
RX	Receive
SDK	Software Development Kit
SLIB	System Library
SNTP	Simple Network Time Protocol
SPI	Serial Peripheral Interface
SSID	Service Set Identifier
STA	Station
TCP	Transport Control Protocol
TIM	Traffic Indicator Map
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
WPS	Wi-Fi Protected Setup
XTAL	Crystal

2 References

[1] DA16200 EVK User Manual, Dialog Semiconductor

3 Introduction

Configuration and control of the DA16200 is 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 AT Command Development Environment Configuration

4.1 How to Connect the DA16200 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.

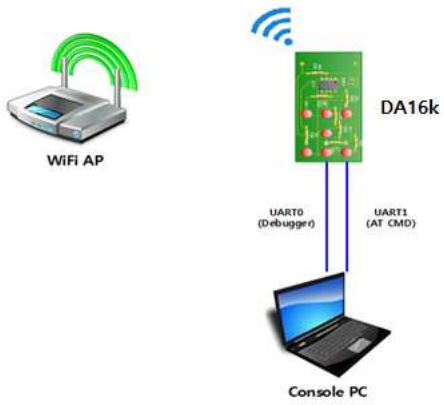


Figure 1: AT Command Development Environment

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 2) provides a UART interface for debugging or firmware download between the PC

and the DA161200. The second (higher number, COM70 in Figure 2) is used for AT COMMAND as shown in Figure 2.

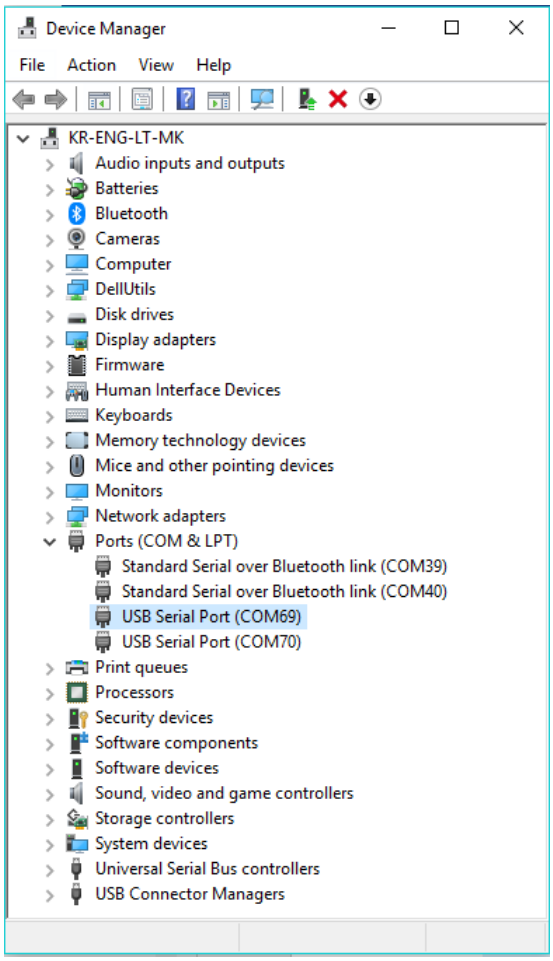


Figure 2: Check COM Ports on Device Manager

4.2 Configure the Serial Port for UART

On a Windows Host, the utility **Tera Term** is used to connect to the DA16200 EVK [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 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 EVK to the PC.
2. Make sure that the PC discovered the two serial ports in Windows Device Manager as shown in Figure 2. 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** dialog:
   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 3. Select the higher COM port number as discovered in step 2.

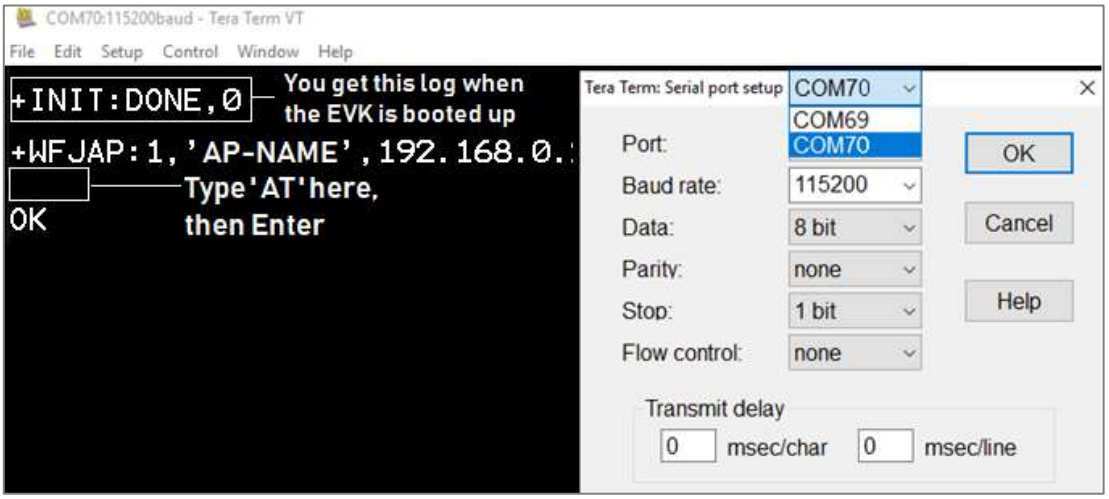


Figure 3: Initial Setup to Get Started with "AT" Command

4.3 Configuration for MCU Wake-Up (Optional)

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

To use the MCU wake-up feature, connect pin GPIO_11 of the DA16200 to the wake-up pin on the MCU. Then, when the DA16200 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.

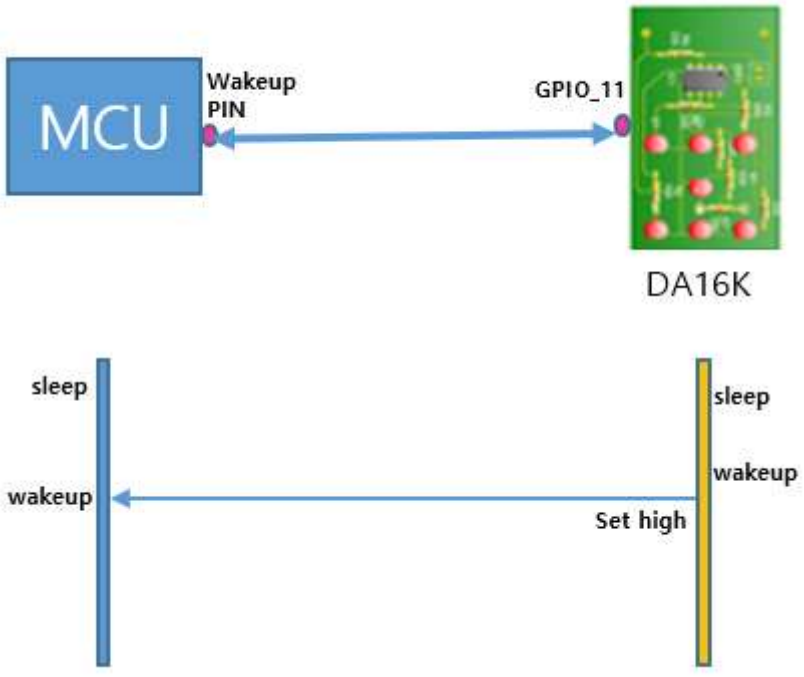


Figure 4: GPIO Wake-Up

5 AT Command Format

5.1 Basic Command Format

- Write CMD

Basic command execution.

ATXX

For example: `ATZ`

`OK`

- Read CMD

Get the parameter values of the command.

ATXX=?

For example: `ATQ=?`

Display result on

`OK`

5.2 Extended Command Format

- Write CMD

Extended command execution.

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 comma(s) or single quote(s) is/are used in the parameter, then a single quote is required before and after.

For example: `AT+WFJAP='ssid,comma'single-quote',3,1,'password'`

`OK`

NOTE
The use of the phrase ', (single-quote and comma next to each other) in the parameter is prohibited.

- Read CMD

Get the parameter values of the command.

AT+XXX=?

For example: `AT+NWIP=?`

```
+ANIP:172.16.0.17,255.255.255.0,172.16.0.1
```

`OK`

5.3 Response Format

- Start-up response

The AT command response when DA16200 is reset.

<CR><LF>+INIT:DONE,<mode><CR><LF>

The AT command response when DA16200 wakes up from DPM sleep.

- 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>

- Normal response

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

- Error response

<CR><LF>ERROR:<error code><CR><LF>

- 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-CMD 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**/**ERROR**) and *extended* response (also known as **Operation Result**).

- Error response codes

Table 1: Error Response Codes

Code	Description
-1	Unknown command
-2	Insufficient parameter
-3	Too many parameters
-4	Wrong parameter value
-5	Unsupported function
-6	Not connected to an AP
-7	No result
-8	Response buffer overflow
-9	Function is not configured
-10	Command timeout
-11	NVRAM write failure
-12	Retention memory write failure
-99	Unknown error

6 Basic Function Commands

Table 2: Basic Function Command List

Command	Parameters	Description
?	(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 - Restore to Factory mode (NVRAM clean) ATE - Command echo ATQ - Result Codes On/Off AT+RESTART - System Restart --- *Middle omission* --- AT+TRSAVE - Save current status of all session === User AT-CMD ====== OK *Note* ● Enabled by default in the SDK	
help	(none)	Same with the “?” command.
	Example Help *Note* ● Enabled by default in the SDK	
AT+	(none)	Show AT command list.

Command	Parameters	Description
	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 command.
	Example ATZ Display result on Echo off OK *Note* ● Enabled by default in the SDK	
ATF	(none)	DA16200 factory reset. Response: "+INIT:DONE,0"
	Example ATF +INIT:DONE,0 *Note* ● Enabled by default in the SDK ● All NVRAM parameters including Wi-Fi profile (Soft-AP or STA) settings are removed, DUT restarts, and “+INIT:DONE,0” will be received	
ATE	(none)	ECHO on/off.
	Example ATE Echo on OK ATE Echo off OK *Note* ● Enabled by default in the SDK	

Command	Parameters	Description
ATQ	(none)	Set displaying result on/off.
	Example ATQ Display result off *Note* ● Enabled by default in the SDK	
ATB	<baudrate> [[,<databits>] [,<parity>] [,<stopbits>] [,<flow control>]]	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 <flow control>: [optional], 0 (Default)/1
	Example 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 to MROM mode, AT command is not available, therefore, MCU needs to force POR booting or enter 'boot' command via UART0 console	
AT+VER	(none)	Get version info. For SDK V2.x.x.x Response: +VER:<ramlib version>,<ptim version>,<main version> For SDK V3.x.x.x Response: +VER:<main version>

Command	Parameters	Description
	Example AT+VER +VER:RTOS-GEN01-01-14648-000000,SLIB-GEN01-01-14645-000000 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	

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td rowspan="2">AT+DEFAP</td><td>(none)</td><td>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)</td></tr>
<tr><td colspan="2">Example
AT+DEFAP
OK

+INIT:DONE,1
Note
● Enabled by default in the SDK
● Default configuration:
○ SSID: DA16200_ 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</td></tr>
<tr><td rowspan="3">AT+BIDX</td><td><idx></td><td>Set Boot index.
<idx>: Boot index (0 or 1).
Response: OK or ERROR</td></tr>
<tr><td>?</td><td>Get the current Boot index.
Response: +BIDX:<0|1></td></tr>
<tr><td colspan="2">Example
AT+BIDX=?
+BIDX:0
OK

AT+BIDX=1
OK

AT+BIDX=?
+BIDX:1
OK
Note
● Enabled by default in the SDK
● System restart requires to reflect boot index. “AT+RESTART” command can be used to restart the system</td></tr>
<tr><td>AT+DPM</td><td><dpm>
[,<nvm_only>]</td><td>Set DPM on/off. To make DPM mode (On/Off) take effect, the system restart is needed.
<dpm>: 0 (Off), 1 (On).
<nvm_only>: 1 (write dpm mode to nvram only, and not reboot), 0 or not specified (change dpm mode and reboot)
Response: OK or ERROR</td></tr>
</table>

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td rowspan="2"></td><td>?</td><td>Get the current DPM setting.
Response: +DPM:<0|1></td></tr>
<tr><td colspan="2">
Prerequisite
It works only in station mode.
Example
AT+DPM=?
+DPM:0
OK

AT+DPM=1 ; DPM enables and system restarts automatically.
OK

+INIT:DONE,0

AT+DPM=1,1 ; DPM enable without system restarts.
OK
AT+DPM=?
+DPM:1
OK
Note
● Enabled by default in the SDK
● DPM configuration is stored in NVRAM
● DA16200 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 boots up
○ If the usage of the AT command is not valid, then DA16200 sends an ERROR message without restarting
● If the “nvram_only” parameter is “1”, then restart the system manually using “AT+RESTART”
● When DA16200 reboots, DA16200 tries to connect to the AP if the Wi-Fi connection information is available in the NVRAM
○ +WFJAP:0 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, wrong password is stored, and so on), +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 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 operates DPM if it is set to 1 (TRUE)
○ If Wi-Fi connection is NOT established in DPM mode, DA16200 enters an abnormal DPM operation
○ ** Abnormal DPM operation:
While DA16200 operates in DPM sleep, DA16200 executes an Abnormal DPM operation if DA16200 is in a “disconnected” state with the specified AP for some reason.
</td></tr>
</table>

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td colspan="2">Abnormal DPM works like this:
DA16200 enters sleep with a predefined RTC timer (This is called Abnormal DPM RTC Timer) configured if the AP connection fails. If DA16200 is woken up by the Abnormal DPM RTC timer, DA16200 tries to connect to the specified AP within a predefined period and sleeps again for a predefined time. The DA16200 library provides default predefined values for Abnormal DPM, but users can modify the relevant parameters based on their application use case scenarios
○ If the Wi-Fi connection is established but MQTT connection is NOT established (if MQTT is enabled), DA16200 tries to connect to the MQTT broker several times and enters DPM Sleep based on MQTT's abnormal DPM operation</td></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 to take effect</td></tr>
<tr><td rowspan="3">AT+DPMTIMWU</td><td><count></td><td>Set DPM TIM wake-up count.
<count>: TIM wake-up count (1 ~ 65535).
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>
</table>

Command	Parameters	Description
	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 to take effect	
AT+DPMUSERWU	<time>	Set DPM user wake-up time. <time>: User wake-up period (second, 0 ~ 86400). Response: OK or ERROR
	?	Get DPM user wake-up time. Response: + DPMUSERWU =<second>
	(none)	
	Example AT+DPMUSERWU +DPMUSERWU:0 OK 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 to take effect	
AT+CLRDPMSLPEXT	(none)	Set the user application not to enter DPM sleep. Response: OK or ERROR

Command	Parameters	Description
	Prerequisite It works in DPM mode. *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 through the external wake-up pin, otherwise, DA16200 will go into DPM Sleep	
AT+SETDPMSLPEXT	(none)	Set the user application ready to enter DPM sleep. Response: OK or ERROR
	Prerequisite It works in DPM mode. *Example* AT+SETDPMSLPEXT OK *Note* ● Enabled by default in the SDK ● If DA16200 is woken up by an external wake-up signal and the "AT+CLRDPMSLPEXT" command is executed, this command should be issued once every job you wanted to do is done. If this command is not run after the job is done, DA16200 will not enter DPM Sleep	
AT+SETSLEEP2EXT	<period>,<retain_dpm_memory>	Enter DPM Sleep 2 mode for the period specified. <period>: wake-up timeout, in second(s) <retain_dpm_memory>: 1 (retain), 0 (not retain) Response: OK or ERROR
	Example AT+SETSLEEP2EXT=10,1 OK +INIT:DONE,0 *Note* ● Enabled by default in the SDK ● DA16200 can be woken up by RTC_WAKE_UP while in sleep by AT+SETSLEEP2EXT ● DA16200 sends "+INIT:DONE:0" when it wakes up	
AT+SETSLEEP1EXT	<retain_dpm_memory>	Enter DPM Sleep 1 mode. <retain_dpm_memory>: 1 (retain), 0 (not retain) Response: OK or ERROR

Command	Parameters	Description
	Example AT+SETSLEEP1EXT=1 OK +INIT:DONE,0 *Note* ● Enabled by default in the SDK ● DA16200 can only be woken up by RTC_WAKE_UP or GPIO which has been assigned as a wake-up source ● DA16200 sends “+INIT:DONE:0” once it wakes up	
AT+MCUWUDONE	(none)	Notify that the MCU wakes up completely. After this command is received, DA16200 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 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	
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>

Command	Parameters	Description
	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 (sys_common_features.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 3: Initiation Response List

Response	Parameters	Description
+INIT	DONE,<mode>	DA16200 booting is complete. <mode>: 0 (STA), 1 (Soft-AP) For example: +INIT:DONE,0
	WAKEUP,<type>	DA16200 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

7 Network Function Commands

Table 4: Network Function Command List

Command	Parameters	Description
AT+NWIP	<iface>,<ip_addr>, <netmask>,<gw>	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
	?	Get the IP address of the current WLAN interface. Response: +NWIP: <iface>,<ip_addr>,<netmask>,<gw>
	(none)	
	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	
AT+NWDNS	<dns_ip>	Set the DNS server IP address of STA interface. <dns_ip>: DNS server IP address. Response: OK or ERROR
	?	Get the DNS server IP address of STA interface. Response: +NWDNS:<dns_ip>
	(none)	
	Example AT+NWDNS=8.8.8.8 OK AT+NWDNS +NWDNS: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.dialog-semiconductor.com +NWHOST:54.192.175.64 OK *Note* ● Enabled by default in the SDK	

Command	Parameters	Description
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 This command is valid only after DA16200 is connected to an AP. *Example* AT+NWDHC=1 OK AT+NWDHC +NWDHC:1 OK *Note* ● Enabled by default in the SDK	
AT+NWDHR	<start_ip>,<end_ip>	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
	?	Get an IP address range of the DHCP server. Response: +NWDHR:<start_ip>,<end_ip>
	(none)	
	Prerequisite *Use this command in 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	
AT+NWDHLT	<lease_time>	Set an IP lease time (in seconds) of the DHCP server. <lease_time>: IP lease time (from 60 sec to 86400 sec) Response: OK or ERROR
	?	Get an IP lease time of the DHCP server.

Command	Parameters	Description
	(none)	Response: +NWDHLT:<lease_time>
	Prerequisite Use this command in Soft-AP mode. *Example* AT+NWDHLT=1800 OK AT+NWDHLT +NWDHLT:1800 OK *Note* ● Enabled by default in the SDK	
AT+NWDHS	<dhcpd>	Start/Stop DHCP server. <dhcpd>: 0 (stop), 1 (start). Response: OK or ERROR
	<dhcpd>, <start_ip>,<end_ip>, <lease_time>, <dns_ip>	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). <dns_ip>: DNS server IP address (optional). Response: OK or ERROR
	?	Get the DHCP client status. Response: +NWDHS:<dhcpd>
	(none)	
	Prerequisite Use this command in Soft-AP mode. *Example* AT+NWDHS=1 OK AT+NWDHS=1,10.0.0.2,10.0.0.10,1800,168.126.63.1 OK AT+NWDHS +NWDHS:1,10.0.0.2,10.0.0.10,1800,168.126.63.1 OK *Note* ● Enabled by default in the SDK	
AT+NWSNS AT+NWSNS1 AT+NWSNS2	<server_ip>	Set the SNTP server IP address/domain name. <server_ip>: SNTP server IP address/domain name. Response: OK or ERROR
	?	Get the SNTP server IP address. Response: +NWSNS:<sntp>
	(none)	

Command	Parameters	Description
	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 fails to synchronize the system time with a server ● If not specified, default SNTP servers are tried	
AT+NWSNUP	<period>	Set the SNTP client update period (in seconds). <period>: SNTP client update period (from 60 sec to 131072 sec) Response: OK or ERROR
	?	Get the SNTP client update period. Response: +NWSNUP:<period>
	(none)	
	Example AT+NWSNUP=86400 OK 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)	

Command	Parameters	Description
	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 reboots. <sntp>=0 removes “auto start” flag from NVRAM, so DA16200 does not try to sync time when rebooted	
AT+NWTLSV	<ver>	Set the TLS version for WPA-Enterprise. <ver>: TLS Version. 0 (TLSv 1.0), 1 (TLSv1.1), 2 (TLSv1.2). Response: OK or ERROR
	? (none)	Get the current TLS version for WPA-Enterprise. Response: +NWTLSV=<ver>
	Example AT+NWTLSV=? +NWTLSV=2 ; default version AT+NWTLSV=2 OK *Note* ● Enabled by default in the SDK	
AT+NWCCRT	(none)	Check if certificates exist. There are two sets of certificates: ● Set #1: for MQTT Root CA (bit 2)/Cert (bit 1)/Key (bit 0) ● Set #2: for HTTPS client for OTA Root CA (bit 5)/Cert (bit 4)/Key (bit 3) For example: if DA16200 has the Root CA and Cert in Set #1, the return value is 6. Response: +VER:<cert>
	Example AT+NWCCRT +NWCCRT:6 ; MQTT OK 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

Command	Parameters	Description
	Example AT+NWDCRT OK *Note* ● Enabled by default in the SDK	

Table 5: Certificate Command

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 two 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) <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	

8 Wi-Fi Function Commands

Table 6: Wi-Fi Function Command List

Command	Parameters	Description
AT+WFMODE	<mode>	Set the Wi-Fi mode. <mode>: 0 (STA), 1 (Soft-AP). 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 ● To take effect, a system restart is required	
AT+WFMAC	<mac>	Write a user MAC address in the NVRAM. Response: OK or ERROR
	?	Get the current MAC address of the activated WLAN interface. Response: +WFMAC:<mac>
	(none)	
	Prerequisite The last digit should be an even number to be a valid MAC address. *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* ● Enabled by default in the SDK ● A user MAC address is stored in NVRAM and a system restart is required to take effect ● DA16200 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 you read the MAC address in Soft-AP mode, it becomes the MAC address you wrote + 1	

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></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+WFMAC=?
+WFMAC: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 to take effect
● DA16200 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">Prerequisite
The last digit should be an even number to be a valid MAC address.
Example
AT+WFOTP=EC:9F:0D:90:00:48
OK
AT+WFMAC=?
+WFMAC:EC:9F:0D:90:00:48
OK
Note
● Enabled by default in the SDK
● An OTP MAC address is stored in OTP and system restart is required 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 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 you read the MAC address in Soft-AP mode, it becomes the MAC address you wrote + 1</td></tr>
<tr><td>AT+WFSTAT</td><td>(none)</td><td>Get Wi-Fi configuration.
Response: +WFSTAT:<Wi-Fi interface><var>…</td></tr>
</table>

Command	Parameters	Description
	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 status/mode	
AT+WFPBC	(none)	Run the WPS PBC method. Response: OK or ERROR

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td colspan="2">Prerequisite
A router should support WPS and PBC.
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
● The existing connection, if any, will be lost when this command is run</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">Prerequisite
An AP should support WPS PIN.
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</td></tr>
<tr><td>AT+WFCWPS</td><td>(none)</td><td>Cancel WPS (both PBC and PIN).
Response: OK or ERROR</td></tr>
</table>

Command	Parameters	Description
	Prerequisite WPS should be in progress. *Example* AT+WFCWPS OK *Note* ● Enabled by default in the SDK ● An error will be returned if WPS is not in progress	
AT+WFCC	<code>	Set a country code. <code>: Country code (defined by ISO 3166-1 alpha-2 standard). Such as KR, US, JP, CH, and so on. Response: OK or ERROR
	?	Get the current country code. Response: AT+WFCC=<code>
	(none)	
	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 returns an error code that is -4	
AT+WFRSSI	(none)	Get the current RSSI value. Response: +RSSI: -34

Command	Parameters	Description
	Prerequisite Connection with an AP should be established. *Example* AT+WFRSSI +RSSI:-25 OK AT+WFRSSI=? +RSSI:-25 OK *Note* ● Enabled by default in the SDK ● DA16200 will respond "+RSSI:NOT_CONN" with error(-7) if the connection is not established	
AT+WFSCAN	(none)	Scan APs. Response: +WFSCAN:<bssid><\t><frequency><\t><signal strength><\t><flag><\t><ssid><LF>…
	Example AT+WFSCAN +WFSCAN:70:5d:cc:32:15:32 2422 -30 [WPA2-PSK-CCMP][WPS][ESS] IPTIME_N604BLACK_ERIC 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	
AT+WFJAP	<ssid>,<sec>[,<hidden>] (sec=0)	Connect to an AP. <ssid>: AP SSID. <sec>: Security protocol. 0 (OPEN), 1 (WEP), 2 (WPA), 3 (WPA2), 4 (WPA+WPA2). <idx>: Key index for WEP. 0~3 <enc>: Encryption. 0 (TKIP), 1 (AES), 2 (TKIP+AES). <key>: Passphrase. <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>'] <OPS_RESULT> : 1 (SUCCESS), 0 (FAILED) <SSID>: The SSID will be surrounded by single quotation mark <IP_ADDRESS>: Assigned IP address and format is xxx.xxx.xxx.xxx
	<ssid>,<sec>, <idx>,<key>[,<hidden>] (sec=1)	
	<ssid>,<sec>, <enc>,<key>[,<hidden>] (sec=2|3|4)	
	?	Get the AP provisioning information.

Command	Parameters	Description
	(none)	Operation Results: If provisioning data available: +WFJAP:'<SSID>',<sec>,<enc>,'<Passphrase>' If provisioning data is not available: ERROR:-7 (No result)
	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	

Command	Parameters	Description
	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	
AT+WFJAPA	<ssid>[,<key>][,<hidden>]	Connect to an AP. 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. Response: OK or ERROR Operation Results: +WFJAP:<OPS_RESULT>[,'<SSID>','<IP_ADDRESS>'] <OPS_RESULT> : 1 (SUCCESS), 0 (FAILED) <SSID>: The SSID will be surrounded by single quotation mark <IP_ADDRESS>: Assigned IP address and format is xxx.xxx.xxx.xxx
	? (none)	Get the AP provisioning information (SSID and Passphrase only) Operation Results: If provisioning data is available: +WFJAPA:'<SSID>','<Passphrase>' If provisioning data is not available: ERROR:-7 (No result)

Command	Parameters	Description
	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 ● 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	
AT+WFCAP	(none)	Connect to an AP with the current WLAN0 interface configuration. Response: OK or ERROR

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><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:-9 ; No AP profile found

AT+WFCAP

ERROR:-7 ; 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 in NVRAM, DA16200 returns an error (-9)
● If DA16200 is already in connection with an AP, it returns an error (-7)</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 is in connection status.
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>AT+WFSTA</td><td>(none)</td><td>Check Wi-Fi connection.
Response: +WFSTA:<status>
<status> 1 (Connected), 0 (disconnected)</td></tr>
</table>

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td colspan="2">Prerequisite
DA16200 is running in station mode.
Example
AT+WFSTA

+WFSTA:0
OK

AT+WFSTA

+WFSTA:1
OK
Note
● Enabled by default in the SDK
● If DA16200 runs the command in Soft-AP mode, it returns an error (-7)</td></tr>
<tr><td rowspan="3">AT+WFROAP</td><td><roam></td><td>Operate the STA roaming.
<roam>: 1 (run), 0 (stop).
Response: OK or ERROR</td></tr>
<tr><td>?
(none)</td><td>Get the roaming status.
Response: +WFROAP:<roam></td></tr>
<tr><td colspan="2">Prerequisite
DA16200 is running in station mode.
Example
AT+WFROAP=1

OK

AT+WFROAP=0

OK

AT+WFROAP=?

+WFROAP:1
OK
Note
● Enabled by default in the SDK
● This command enables a “simple” roaming. The roaming configuration consists of one parameter called the roaming threshold (which is set by AT+WFROTH. By default it is -65). Let’s assume DA16200 is connected to an AP, and there are other APs that have the same SSID and security settings around DA16200. When DA16200 moves around and if the currently connected AP’s RSSI becomes lower than the threshold specified, it tries to connect to an AP (with the same SSID and security) that has higher RSSI. If the condition is met, DA16200 silently switches to the new AP without disconnection event
● The auto roaming start flag is stored in NVRAM when <roam> is set to 1 and when the system reboots, if the flag is set, roaming behavior is enabled. If <roam> is 0, the roaming flag is removed from NVRAM and roaming is disabled</td></tr>
</table>

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td rowspan="4">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 rowspan="2">Get the STA roaming threshold.
Response: +WFROTH:<rssi></td></tr>
<tr><td>(none)</td></tr>
<tr><td colspan="2">Prerequisite
DA16200 is running in station mode.
Example
AT+WFROTH=-55

OK

AT+WFROTH=?

+WFROTH:-55
OK
Note
● 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</td></tr>
<tr><td rowspan="4">AT+WFDIS</td><td><disabled></td><td>Set the Wi-Fi STA profile unused. If set to 1, DA16200 will not start to connect to the configured AP when rebooting.
<disabled >: 1 (Unused), 0 (Used).
Response: OK or ERROR</td></tr>
<tr><td>?</td><td rowspan="2">Get the status of the Wi-Fi profile.
Response: +WFDIS:<disabled></td></tr>
<tr><td>(none)</td></tr>
<tr><td colspan="2">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 during boot-up procedure which means system restart is required</td></tr>
<tr><td>AT+WFSAP</td><td><ssid>,<sec>,
<ch>,<code>
(sec=0)</td><td>Set up Soft-AP interface.
<ssid>: AP SSID.</td></tr>
</table>

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td rowspan="3"></td><td><ssid>,<sec>,
<enc>,<key>,
<ch>,<code>
(sec=2|3|4)</td><td><sec>: Security protocol. 0 (OPEN), 2 (WPA), 3 (WPA2), 4 (WPA+WPA2).
<enc>: Encryption. 0 (TKIP), 1 (AES), 2 (TKIP+AES).
<key>: Passphrase.
<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</td></tr>
<tr><td>?
(none)</td><td>Get the Soft-AP interface configuration.
Response:
+WFSAP:'<ssid>',<auth>,<enc>,'<key>',<ch>,<code>
Operation Result:
+WFSAP:<ssid> is printed on success</td></tr>
<tr><td colspan="2">Example
AT+WFSAP=DA16200_MY_SSID,0,1,KR

+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

+WFSAP:DA16200_MY_SSID
OK

AT+WFSAP=?

+WFSAP:'DA16200_MY_SSID',3,1,'12345678',1,KR
OK

AT+WFSAP='DA16200,MY_SSID',3,1,'12345678',1,KR

+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)</td></tr>
</table>

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td colspan="2">● The ‘,’(comma) is included in the SSID string, enclose the SSID with a single quotation mark</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 (-99)
● DA16200 returns an error (-99) 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
Run this command in Soft-AP mode.
Example
AT+WFTAP

OK
Note
● Enabled by default in the SDK
● This command is valid while DA16200 is running in Soft-AP mode</td></tr>
<tr><td colspan="3">Additional Note for AT+WFSAP, AT+WFOAP, AT+WFTAP:
Use case example:
For example, in STA mode, in factory reset state
…
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 you want
AT+WFOAP // start Soft-AP if you want</td></tr>
<tr><td>AT+WFRAP</td><td>(none)</td><td>Restart the Soft-AP interface.
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">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 returns an error (-99)</td></tr>
<tr><td rowspan="2">AT+WFLCST</td><td>(none)</td><td>Get connected station information.
Response: +WFLCST:<mac><LF><flags><LF><var>…</td></tr>
<tr><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 returns “+WFLCST:NOT_FOUND”</td></tr>
<tr><td rowspan="3">AT+WFAPWM</td><td><mode></td><td>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</td></tr>
<tr><td>?</td><td rowspan="2">Get IEEE 802.11 Wi-Fi mode of Soft-AP interface.
Response: +WFAPWM:<mode></td></tr>
<tr><td>(none)</td></tr>
</table>

Command	Parameters	Description
	Example AT+WFAPWM=0 OK AT+WFAPWM=1 OK AT+WFAPWM=? +WFAPWM:1 OK *Note* ● Enabled by default in the SDK ● Use this command in Soft-AP mode ● The configuration is stored in NVRAM ● System restart is required to take effect	
AT+WFAPCH	<ch>	Set the operating channel number for the Soft-AP interface. <ch>: Operating channel (from 0 to 14, 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 ● Use this command in Soft-AP mode ● The configuration is stored in NVRAM ● System restart is required 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)	

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td colspan="2">Example
AT+WFAPBI=200

OK

AT+WFAPBI=?

+WFAPBI:200
OK
Note
● Enabled by default in the SDK
● Use this command in Soft-AP mode
● The configuration is stored in NVRAM
● System restart is required to take effect</td></tr>
<tr><td rowspan="4">AT+WFAPUI</td><td><timeout></td><td>Set station disconnection timeout in Soft-AP mode.
<timeout>: Disconnection timeout (sec).
Response: OK or ERROR</td></tr>
<tr><td>?</td><td rowspan="2">Get station disconnection timeout in Soft-AP mode.
Response: +WFAPUI:<timeout></td></tr>
<tr><td>(none)</td></tr>
<tr><td colspan="2">Example
AT+WFAPUI=60

OK
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
● Use this command in Soft-AP mode
● The configuration is stored in NVRAM
● System restart is required to take effect</td></tr>
<tr><td rowspan="3">AT+WFAPRT</td><td><threshold></td><td>Set the AP RTS threshold (octets).
<threshold>: RTS threshold (from 1 to 2347).
Response: OK or ERROR</td></tr>
<tr><td>?</td><td rowspan="2">Get the AP RTS threshold.
Response: +WFAPRT:<threshold></td></tr>
<tr><td>(none)</td></tr>
</table>

Command	Parameters	Description
	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 ● Use this command in Soft-AP mode ● The configuration is stored in NVRAM ● System restart is required 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 There should be a station connected. *Example* AT+WFAPDE=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 ● You can check the MAC address of an STA that you want to send deauthentication 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

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td colspan="2">Prerequisite
There should be a station connected.
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
● You can check the MAC address of an STA that you want to send disassociation 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</td></tr>
<tr><td rowspan="4">AT+WFWMM</td><td><wmm></td><td>Set WMM on/off.
<wmm>: 0 (off), 1 (on).
Response: OK or ERROR</td></tr>
<tr><td>?</td><td rowspan="2">Get the WMM status.
Response: +WFWMM:<wmm></td></tr>
<tr><td>(none)</td></tr>
<tr><td colspan="2">Prerequisite
Run this command in 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 a WMM info element. WMM enables QoS on the AC category
● The configuration is stored in NVRAM
● System restart is required to take effect</td></tr>
<tr><td rowspan="2">AT+WFWMP</td><td><wmmps></td><td>Set WMM-PS (WMM Power Save) on/off.
<wmmps>: 0 (off), 1 (on).
Response: OK or ERROR</td></tr>
<tr><td>?</td><td>Get the WMM-PS status.
Response: +WFWMP:<wmmps></td></tr>
</table>

Command	Parameters	Description
	Prerequisite Run this command in 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	

Table 7: Wi-Fi Function Response List

Response	Parameters	Description
+WFJAP	<result>,<ssid>,<ip>	The result of AP connection in STA mode. (The result of AT+WFJAP or AT+WFJAPA or AT+WFCAP.) <result>: 0 (failed), 1 (succeeded). <ssid>: SSID of the AP when succeeded. <ip>: IP address of the station when succeeded. For example: +WFJAP:0 The Wi-Fi connection is not established. +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>	Disconnected from the AP. <reserved>: 0 For example: +WFDAP:0
+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.

8.1 Wi-Fi Function Commands for WPA3

You can configure DA16200 as WPA3 Station or WPA3 Soft-AP with a special DA16200 SDK where the WPA3 feature is enabled. By default, WPA3 is not enabled in DA16200 Generic SDK. Please ask Dialog if you need this specific SDK.

Syntax of all the Wi-Fi function commands is the same as described in Table 6 apart from the following commands where you need to specify WPA3 specific parameters.

Table 8: List of WPA3-Relevant Wi-Fi Function Commands

Command	Parameters	Description
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+WPA3_SAE) <idx>: Key index for WEP. 0~3 <enc>: Encryption. 0 (TKIP), 1 (AES), 2 (TKIP+AES). <key>: Passphrase. <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>'] <OPS_RESULT> : 1 (SUCCESS), 0 (FAILED) <SSID>: The SSID will be surrounded by single quotation mark <IP_ADDRESS>: Assigned IP address and format is xxx.xxx.xxx.xxx
	<ssid>,<sec>, <idx>,<key>[,<hidden>] (sec=1)	
	<ssid>,<sec>, <enc>,<key>[,<hidden>] (sec=2|3|4|6|7)	
	?	Get the AP provisioning information

Command	Parameters	Description
	(none)	Operation Results: If provisioning data available: +WFJAP:'<SSID>',<sec>,<enc>,'<Passphrase>' If provisioning data is not available: ERROR:-7 (No result)

Example

```
AT+WFJAP=MY_AP_SSID,5          ; OWE

OK

+WFJAP:1,'MY_AP_SSID',192.168.43.32


AT+WFJAP=MY_AP_SSID,5,1        ; OWE + hidden SSID

OK

+WFJAP:1,'MY_AP_SSID',192.168.43.32


AT+WFJAP=MY_AP_SSID,6,1,12345678                ; WPA3_SAE


OK

+WFJAP:1,'MY_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,7,1,12345678       ; WPA2+WPA3_SAE

OK

+WFJAP:1,'MY_AP_SSID',192.168.0.7


AT+WFJAP=?

+WFJAP:'MY_AP_SSID',6,1,'N12345678'
OK
```

Note

- Disabled by default in the SDK. To use this command, use WPA3 SDK or Contact Dialog

Command	Parameters	Description
	● 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	
AT+WFJAPA	<is_wpa3>,<ssid>[,<key>] [,<hidden>]	Connect to an AP. For connection to a WPA3 (OWE, SAE, or WPA2+WPA3) AP, give <is_wpa3> 1, otherwise, give 0. If <key> exists, then security protocol is WPA+WPA2 and encryption is TKIP+AES. if <key> is omitted, then security protocol is OPEN (is_wpa3=0) or OWE (is_wpa3=1). <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. Response: OK or ERROR Operation Results: +WFJAP:<OPS_RESULT>[,'<SSID>','<IP_ADDRESS>'] <OPS_RESULT> : 1 (SUCCESS), 0 (FAILED) <SSID>: The SSID will be surrounded by single quotation mark <IP_ADDRESS>: Assigned IP address and format is xxx.xxx.xxx.xxx
	? (none)	Get the AP provisioning information (SSID and Passphrase only) Operation Results: If provisioning data is available: +WFJAPA:'<SSID>','<Passphrase>' If provisioning data is not available: ERROR:-7 (No result)
	Example AT+WFJAPA=1,MY_AP_SSID *; OWE security* OK +WFJAP:1,'MY_AP_SSID',192.168.43.32 AT+WFJAPA=1,MY_AP_SSID,1 *; OWE security + hidden SSID* OK +WFJAP:1,'MY_AP_SSID',192.168.43.32 AT+WFJAPA=1,MY_AP_SSID,N12345678 *; WPA3_SAE security*	

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

+WFJAP:1,'MY_AP_SSID',192.168.43.32

AT+WFJAPA=1,MY_AP_SSID,N12345678,1 ; WPA3_SAE security + hidden AP

OK

+WFJAP:1,'MY_AP_SSID',192.168.43.32

AT+WFJAPA=?

+WFJAPA:'MY_AP_SSID','N12345678'
OK
Note
● Disabled by default in the SDK. To use this command, use WPA3 SDK or Contact Dialog
● 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</td></tr>
<tr><td rowspan="3">AT+WFSAP</td><td><ssid>,<sec>,
<ch>,<code>
(sec=0|5)</td><td rowspan="2">Set up the Soft-AP interface.
<ssid>: AP SSID.
<sec>: Security protocol. 0 (OPEN), 2 (WPA), 3 (WPA2), 4 (WPA+WPA2), WPA3_OWE (5), WPA3_SAE (6), WPA2+WPA3 (7)
<enc>: Encryption. 0 (TKIP), 1 (AES), 2 (TKIP+AES)
<key>: Passphrase.
<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</td></tr>
<tr><td><ssid>,<sec>,
<enc>,<key>,
<ch>,<code>
(sec=2|3|4|6|7)</td></tr>
<tr><td>?
(none)</td><td>Get the Soft-AP interface configuration.
Response:
+WFSAP:'<ssid>',<auth>,<enc>,'<key>',<ch>,<code>
Operation Result:
+WFSAP:<ssid> is printed on success</td></tr>
</table>

Command	Parameters	Description
	Example AT+WFSAP=DA16200_MY_SSID,5,1,KR ;OWE +WFSAP:DA16200_MY_SSID OK AT+WFSAP=? +WFSAP:'DA16200_MY_SSID',5,1,KR OK AT+WFSAP=DA16200_MY_SSID,6,1,12345678,1,KR ; WPA3_SAE +WFSAP:DA16200_MY_SSID OK AT+WFSAP=? +WFSAP:'DA16200_MY_SSID',6,1,'12345678',1,KR OK AT+WFSAP='DA16200,MY_SSID',7,1,12345678,1,KR ; WPA2+WPA3 +WFSAP:DA16200,MY_SSID OK AT+WFSAP=? +WFSAP:'DA16200,MY_SSID',7,1,'12345678',1,KR OK *Note* ● Disabled by default in the SDK. To use this command, use WPA3 SDK or Contact Dialog ● 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, enclose the SSID with a single quotation mark	

9 Advanced Function Commands

9.1 MQTT Commands

Table 9: MQTT 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 should be connected to a network. If not, this command will not work. *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, subscriber topic, and so on are required to be done before issuing this command	
AT+NWMQMSG	<msg>,<topic>	Publish an MQTT message. <msg>: Message to be published (max length: 256 bytes). <topic>: MQTT topic (optional). Response: OK or ERROR
	Prerequisite MQTT client should be enabled. (+NWMQCL:1) *Example* AT+NWMQMSG=Hello world !!! OK AT+NWMQMSG='{"car":"red", "type":"bus"}' OK *Note* ● Enabled by default in the SDK ● If a single quotation is used in a message, surrounded by double quotation marks	
AT+NWMQBR	<ip>,<port>	Set the IP address and the port number of the MQTT Broker. <ip>: Broker's IP address. <port>: Broker's port number. Response: OK or ERROR
	?	Get the IP address and the port number of the MQTT Broker. Response: +NWMQBR:<ip>,<port>
	(none)	

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td colspan="2">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 IP and port configured are stored in the NVRAM
● MQTT restart is required to take the new configuration effect</td></tr>
<tr><td rowspan="4">AT+NWMQQOS</td><td><qos></td><td>Set the MQTT QoS level.
<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 QoS level.
Response: +NWMQQOS:<qos></td></tr>
<tr><td>(none)</td></tr>
<tr><td colspan="2">Example
AT+NWMQQOS=1
OK

AT+NWMQQOS
+NWMQQOS:1
OK
Note
● Enabled by default in the SDK
● MQTT restart is required to take the new configuration effect</td></tr>
<tr><td rowspan="4">AT+NWMQTLS</td><td><tls></td><td>Enable/disable the MQTT TLS function.
<tls>: 1 (enable), 0 (disable)
Response: OK or ERROR</td></tr>
<tr><td>?</td><td rowspan="2">Get MQTT TLS status.
Response: +NWMQTLS:<tls></td></tr>
<tr><td>(none)</td></tr>
<tr><td colspan="2">Prerequisite
Certificate should be stored before enabling the TLS function. Refer to Table 5.
Example
AT+NWMQTLS=1
OK

AT+NWMQTLS
+NWMQQOS:1
OK
Note
● Enabled by default in the SDK
● MQTT restart is required to take the new configuration effect</td></tr>
<tr><td>AT+NWMQTS</td><td><num>,<topic#1>,
<topic#2>,
…</td><td>Set the topic(s) of the MQTT subscriber.
<num>: Number of topics.
<topic#n>: MQTT subscriber topic(s).
Response: OK or ERROR</td></tr>
</table>

Command	Parameters	Description
	?	Get the MQTT subscriber topic(s). Response: +NWMQTS:<num>,<topic#1>,<topic#2>,…
	(none)	
	Example AT+NWMQTS=? ERROR:-7 AT+NWMQTS=1,da16k_sub OK AT+NWMQTS=? +NWMQTS:1,"da16k_sub" OK *Note* ● Enabled by default in the SDK ● "ERROR:-7" means ERR_NO_RESULT ● MQTT restart is required to take the new configuration effect	
AT+NWMQTP	<topic>	Set the topic(s) of the MQTT publisher. <topic>: MQTT publisher topic. Response: OK or ERROR
	?	Get the MQTT publisher topic. Response: +NWMQTP:<topic>
	(none)	
	Example AT+NWMQTP=? ERROR:-7 AT+NWMQTP=da16k_pub OK AT+NWMQTP=? +NWMQTP:da16k_pub OK *Note* ● Enabled by default in the SDK ● "ERROR:-7" means ERR_NO_RESULT ● MQTT restart is required to take the new configuration 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)	

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td colspan="2">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 to take the new configuration effect</td></tr>
<tr><td rowspan="4">AT+NWMQCID</td><td><client_id></td><td>Set the MQTT Client ID.
<client_id>: Client ID.
Response: OK or ERROR</td></tr>
<tr><td>?</td><td rowspan="2">Get the current MQTT Client ID.
Response: +NWMQCID:<client_id></td></tr>
<tr><td>(none)</td></tr>
<tr><td colspan="2">Example
AT+NWMQCID=?
ERROR:-7

AT+NWMQCID=client-1
OK

AT+NWMQCID
+NWMQCID:client-1
OK
Note
● Enabled by default in the SDK
● “ERROR:-7” means ERR_NO_RESULT
● MQTT restart is required to take the new configuration effect</td></tr>
<tr><td rowspan="3">AT+NWMQLI</td><td><name>,<pw></td><td>MQTT login information.
<name>: ID.
<pw>: Password.
Response: OK or ERROR</td></tr>
<tr><td>?</td><td rowspan="2">Get the MQTT login information.
Response: +NWMQLI:<name>,<pw></td></tr>
<tr><td>(none)</td></tr>
</table>

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td colspan="2">Example
AT+NWMQLI=?
ERROR:-7

AT+NWMQLI=da16k_user,12345678
OK

AT+NWMQLI
+NWMQLI:da16k_user,12345678
OK
Note
● Enabled by default in the SDK
● “ERROR:-7” means ERR_NO_RESULT
● MQTT restart is required to take the new configuration effect</td></tr>
<tr><td rowspan="4">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">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 to take the new configuration 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>
</table>

Command	Parameters	Description
	Example AT+NWMQWILL=? ERROR:-7 AT+NWMQWILL=da16k_will,bye,0 OK AT+NWMQWILL +NWMQWILL:da16k_will,bye,0 OK *Note* ● Enabled by default in the SDK ● "ERROR:-7" means ERR_NO_RESULT ● MQTT restart is required to take the new configuration 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	
AT+NWMQTT	<ip>,<port>, <sub_topic>, <pub_topic>, <qos>,<tls>, <username>, <password>	Run the MQTT Client with options. After entering this command, system will reboot automatically. At reboot, DA16200 tries to connect to the MQTT broker after the Wi-Fi connection is successfully established. <ip>: Broker's IP address, <port>: Broker's port number. <sub_topic>: MQTT subscriber topic. <pub_topic>: MQTT publisher topic. <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

Command	Parameters	Description
	Prerequisite MQTT client should be disabled (+NWMQCL:0) before running this command *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	

The table below shows optional MQTT configuration commands for the MQTT brokers that require TLS ALPN, SNI, or Cipher Suite info from MQTT Client at the connection stage. These commands are not enabled by default.

To enable these commands, build SDK with `__MQTT_TLS_OPTIONAL_CONFIG__` defined in sys_common_features.h

Table 10: 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) before running this command *Example* AT+NWMQALPN=? ERROR:-7 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* ● Disabled by default in the SDK ● If __MQTT_TLS_OPTIONAL_CONFIG__ is enabled in the SDK (sys_common_features.h), this command will be enabled ● "ERROR:-7" means ERR_NO_RESULT	

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td rowspan="3">AT+NWMQSNI</td><td><sni></td><td>Set TLS SNI for MQTT.
<sni>: Server Name Indication. Maximum length of SNI is 64
Response: OK or ERROR</td></tr>
<tr><td>?</td><td>Get the TLS SNI that has been set.
Response: +NWMQSNI:<sni></td></tr>
<tr><td colspan="2">Prerequisite
MQTT client should be disabled (+NWMQCL:0) before running this command
Example
AT+NWMQSNI=?
ERROR:-7

AT+NWMQSNI=a38a9rhiu3roqb-ats.myserver.com
OK

AT+NWMQSNI
+NWMQSNI: a38a9rhiu3roqb-ats.myserver.com
OK
Note
● Disabled by default in the SDK
● If __MQTT_TLS_OPTIONAL_CONFIG__ is enabled in the SDK (sys_common_features.h), this command will be enabled
● “ERROR:-7” means ERR_NO_RESULT</td></tr>
<tr><td rowspan="3">AT+NWMQCSUIT</td><td><cipher suite 1>,
<cipher suite 2>,
…</td><td>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</td></tr>
<tr><td>?</td><td>Get TLS cipher suites that have been set
Response: +NWMQSNI:<number of cipher suites>,<cipher suite 1>,<cipher suite 2>,…</td></tr>
<tr><td colspan="2">Prerequisite
MQTT client should be disabled (+NWMQCL:0) before running this command
Example
AT+NWMQCSUIT=?
ERROR:-7

AT+NWMQCSUIT=c024,c023,c00a,c009,c00d,c032
OK

AT+NWMQCSUIT
+NWMQCSUIT:6,c024,c023,c00a,c009,c00d,c032
OK
Note
● Disabled by default in the SDK
● If __MQTT_TLS_OPTIONAL_CONFIG__ is enabled in the SDK (sys_common_features.h), this command will be enabled
● “ERROR:-7” means ERR_NO_RESULT
● The hex pre-fix “0x” should be removed when one is typed
● DA16200 does not support all the cipher suites due to memory limitation. Contact Dialog if you want to use other cipher suites that are not specified in Appendix D</td></tr>
</table>

Table 11: MQTT Response List

Response	Parameters	Description
+NWMQCL	<result>	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. Until you get an **Operation Result**, it may take more time if the DA16200 connection retrial happens depending on your 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. 1. DA16200 restarts if the AT command format is **OK**. a. +INIT:DONE,0 message is sent as DA16200 boots up. b. If usage of the AT command is not valid, DA16200 sends an **ERROR** message without restarting. 2. DA16200 tries to connect to the AP after the reboot. a. +WFJAP:0 or +WFJAP:1,'<SSID>',<IP Address> as result of the Wi-Fi connection. b. If the Wi-Fi connection information such as SSID or key is NOT stored correctly in the DA16200 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. 3. DA16200 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.
	Prerequisite Sent only when MQTT Client is run *Example* ; When MQTT connection to the MQTT broker is successfully established +NWMQCL:1 ; When MQTT broker is down, +NWMQCL:0 *Note* ● DA16200 Debug Console log (UART0) ○ When MQTT connection to the MQTT broker is successfully established, DA16200 sends “+NWMQCL:1” message to MCU (or AT command console). At this moment, you can see the following log: *>>> MQTT Client connection OK* ○ When the MQTT broker is down, DA16200 sends “+NWMQCL:0” message to MCU (or AT command console) after retrying connection. In console log, you can see the following logs: *Failed to receive pkt. (0x38)* *Failed to read pkt(0x7880)* *MQTT Client disconnected ... (state=6)* *[SUB] REQ mqtt_restart (count=1)*	

Response	Parameters	Description
	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).*	
+NWMQMSG	<msg>,<topic>,<length>	Received the MQTT message. <msg>: Message data. <topic>: Received topic. <length>: Message length.
	Prerequisite MQTT client is in a connected state with the broker. (+NWMQCL:1) MQTT publisher is ready to send a message. *Example* ; When DA16200 receives a message from the MQTT publisher, the following message will be sent from DA16200 to AT command console: +NWMQMSG:Hello world!!!!,da16k_sub,15 *Note* <msg> : Hello world!!!! <topic> : da16k_sub <length> : 15 (length of “Hello world!!!!”)	

9.1.1 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'
```

9.1.2 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. (<ESC>C in Section 7)

After all settings are made, start the client:

```
AT+NWMQCL=1
```

9.1.3 MQTT Example with DPM

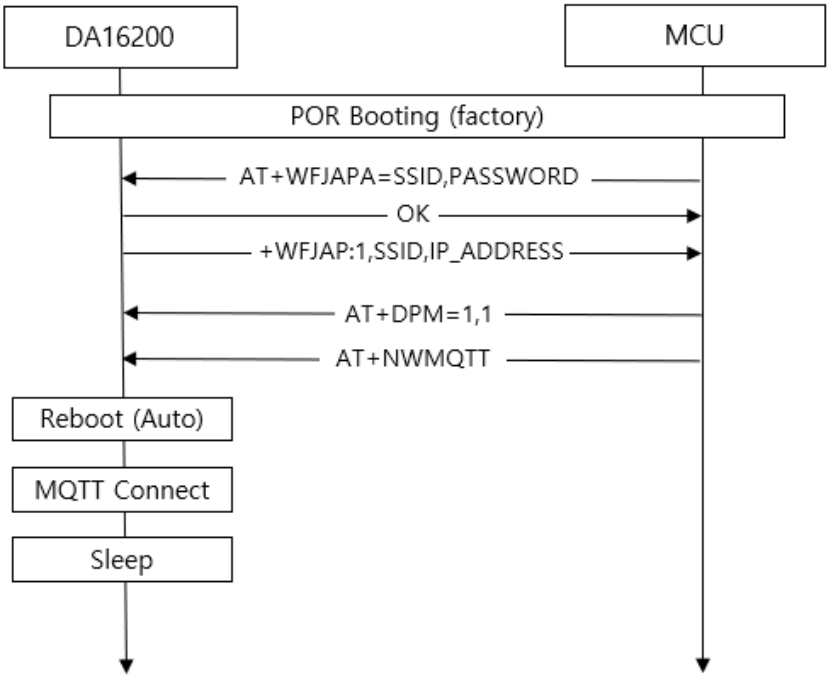


Figure 5: Example Sequence to Initiate MQTT Protocol with DPM

Figure 5 is an example sequence to initiate the MQTT protocol with DPM in the DA16200.

In the normal BOOT state, connect to an AP (AT+WFJAPA) and change the DA16200 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
```

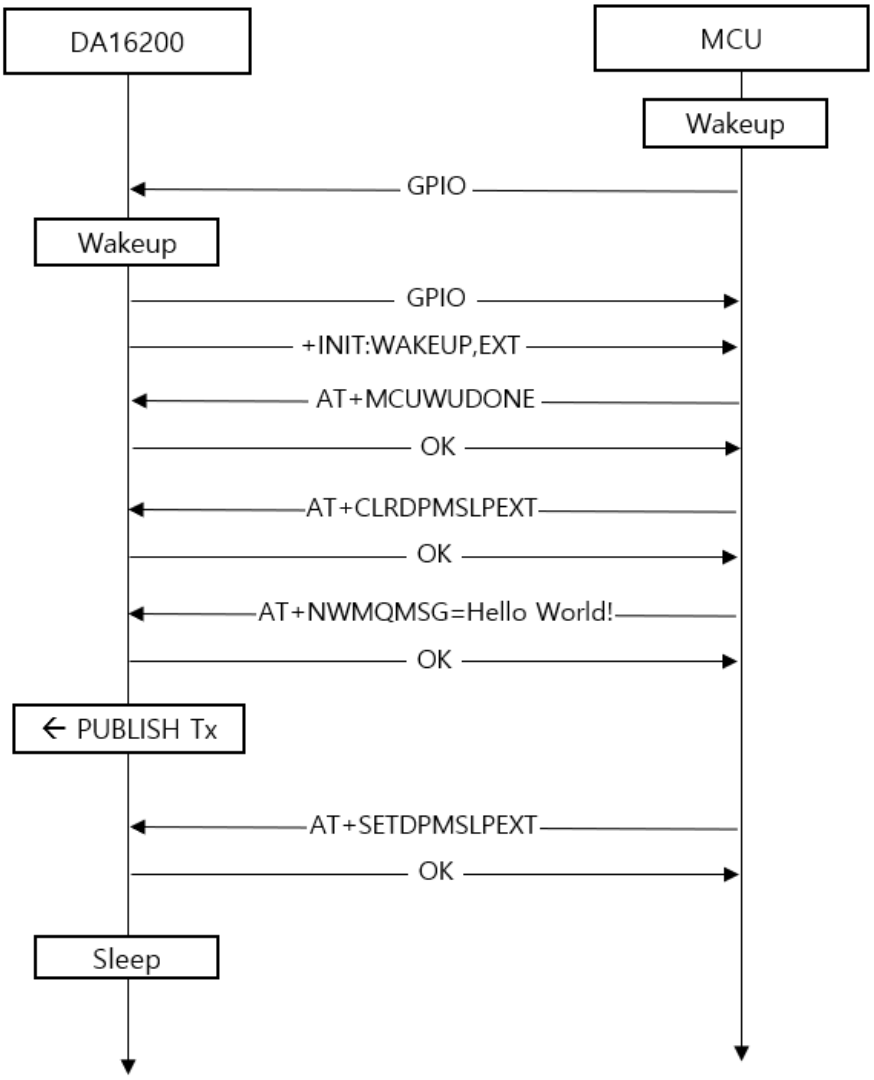


Figure 6: Procedure to Send MQTT Messages

Figure 6 shows the procedure to send an MQTT message in Sleep mode.

When MCU wakes up the DA16200, 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 enters DPM Sleep mode, MCU should send command AT+CLRDPMSLPEXT before an MQTT PUBLISH is sent. To make the DA16200 enter DPM Sleep mode again, send a PUBLISH with command AT+NWMQMSG, and then enter command AT+SETDPMSLPEXT.

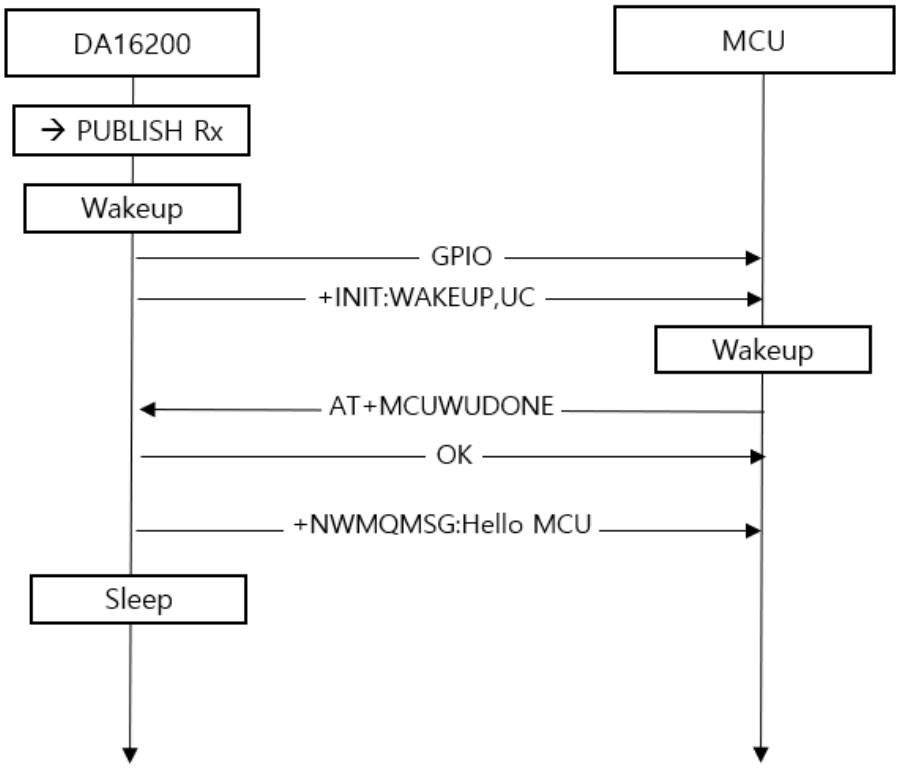


Figure 7: Procedure to Process MQTT Messages

Figure 7 shows the procedure to process an MQTT message received while in Sleep mode.

When the DA16200 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 sends the received PUBLISH to the MCU and enters DPM Sleep mode again.

9.1.4 MQTT Example: Changing Subscription Topic when Running

Let's 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.

```
A. Trigger RTC_WAKE_UP Event (by MCU)
B. Wait for "+INIT:WAKEUP,EXT" Response. Send AT+MCUWUDONE, and wait for "OK"
C. Run "AT+CLRDPMSLPEXT" command
D. Wait for "OK" response
E. loop running "AT+NWMQCL=?"
      E.1 if responses are "+NWMQCL:0" and "OK"
      E.2      then, goto E. to run "AT+NWMQCL=?" command
      E.3 else if responses are "+NWMQCL:1" and "OK"
      E.4      then, goto next, F.
      E.5 else if response is "ERROR:x"
      E.6      then, Run "AT+SETDPMSLPEXT"
      E.7            Wait for "OK" response
      E.8            return
F. Run "AT+NWMQCL=0"
G. Wait for "+NWMQCL:0" and "OK" response
H. Run "AT+NWMQTS=<New MQTT Subscription Topic>"
I. Wait for "OK" response
J. Run "AT+RESTART"
K. Wait for "+INIT:DONE,0" response
L. Wait for "+WFJAP:1,'<SSID>',<IP ADDRESS>"
M. Wait for "+NWMQCL:1" response
```

9.1.5 MQTT Example: Reading Subscription Topic when Running

Let's 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 you need to consider that you can get the ERROR response if that format of the command may
    have some error.
6.  Run "AT+SETDPMSLPEXT"
7.  Wait for "OK" response
```

Let's 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
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.  Run "AT+NWMQTS=?"
6.  Wait for "+NWMQTS:<MQTT Subscription Topic>" and "OK" response
7.  Run "AT+SETDPMSLPEXT"
```

9.2 HTTP-Client Commands

Table 12: HTTP-Client Command List

Command	Parameters	Description
AT+NWHTC	<url>,<method>(,<msg>)	Start the HTTP client with options. <url>: HTTP server address. <method>: GET, POST or PUT. <msg>: Request message for POST and PUT methods.
	Prerequisite Target system should be connected to an AP. HTTP server should exist. *Example* AT+NWHTC=https://httpbin.org/get,get +NWHTC:HTTP/1.1 200 OK AT+NWHTC=https://httpbin.org/post,post,HTTP-Client POST method sample test! +NWHTC:HTTP/1.1 200 OK AT+NWHTC=https://httpbin.org/put,put,HTTP-Client PUT method sample test! +NWHTC:HTTP/1.1 200 OK *Note* ● Disabled by default in the SDK ● If __SUPPORT_HTTP_CLIENT__ is enabled in SDK, this command will be enabled.	

Table 13: HTTP-Client Response List

Response	Parameters	Description
+ NWHTC	<status>	Returns status along with the received payload according to the requested method. <status>: 0x00 is success. See Appendix B. For example: +NWHTCSTATUS:0x00

9.2.1 HTTP-Client Connection Example

GET method request:

```
AT+NWHTC=https://httpbin.org/get,get
```

POST method request:

```
AT+NWHTC=https://httpbin.org/post,post,HTTP-Client POST method sample test!
```

PUT method request:

```
AT+NWHTC=https://httpbin.org/put,put,HTTP-Client PUT method sample test!
```

9.3 HTTP-Server Commands

Table 14: HTTP-Server Command List

Command	Parameters	Description
AT+NWHTS	<flagt>	Start or stop the HTTP server depending on your options. <start>: 1 (start), 0 (stop) Response: OK or ERROR.
	Prerequisite Target system should be connected to an AP. *Example* AT+NWHTS=1 OK *Note* ● Disabled by default in the SDK ● If __SUPPORT_HTTP_SERVER__ is defined in SDK, this command will be enabled	
AT+NWHTSS	<flag>	Start or stop the HTTPS server depending on your options. <start>: 1 (start), 0 (stop) Response: OK or ERROR.
	Prerequisite Target system should be connected to an AP. *Example* AT+NWHTSS=1 OK *Note* ● Disabled by default in the SDK ● If __SUPPORT_HTTP_SERVER__ is defined in SDK, this command will be enabled	

9.3.1 HTTP/HTTPS-Server Start Example

HTTP start:

```
AT+NWHTS=1
```

HTTPS start:

```
AT+NWHTSS=1
```

9.4 OTA Commands

Table 15: OTA Command List

Command	Parameters	Description
AT+NWOTADWSTART	<fw_type>,<uri> (,<fw_name>)	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. It is available if *fw_type* is *mcu_fw*. Maximum input size is eight bytes. MCU_FW will be stored as default if there is no fw_name information. (Only for MCU FW) Response: +NWOTADWSTART:0x00
	Prerequisite Target system should be connected to an AP. OTA server should exist. fw_type should be a lowercase character. *Example* ; RTOS download AT+NWOTADWSTART=rtos,https://server/DA16200_RTOS-GEN01-01-1111-000000.img OK +NWOTADWSTART:0x00 ; SLIB download AT+NWOTADWSTART=slib,https://server/DA16200_SLIB-GEN01-01-1111-000000.img OK +NWOTADWSTART:0x00 ; MCU FW download AT+NWOTADWSTART=other_fw,https://server/mcu_firmware.img OK +NWOTADWSTART:0x00 AT+NWOTADWSTART=other_fw,https://server/mcu_firmware.img,ver01 OK +NWOTADWSTART:0x00 ; Cert Key download: AT+NWOTADWSTART=cert_key,https://server/ca.pem OK +NWOTADWSTART:0x00 *Note* ● Enabled by default in the SDK	
AT+NWOTARENEW	(none)	Reboot with updated FW.

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td colspan="2">Prerequisite
RTOS and SLIB both images should be downloaded in advance.
Example
AT+NWOTARENEW
+NWOTARENEW:0x00
Note
● Enabled by default in the SDK
● Will reboot automatically after renew is done</td></tr>
<tr><td rowspan="2">AT+NWOTADWPROG</td><td><fw_type></td><td>FW download progress.
<fw_type>: SLIB/RTOS.
Response: +NWOTADWPROG:100
(in SDK V3.x.x.x, only RTOS is available)</td></tr>
<tr><td colspan="2">Example
; SLIB download progress:
AT+NWOTADWPROG=slib
+NWOTADWPROG:100
OK

; RTOS download progress:
AT+NWOTADWPROG=rtos
+NWOTADWPROG:100
OK

; MCU FW download progress:
AT+NWOTADWPROG=mcu_fw
+NWOTADWPROG:100
OK

; Cert Key download progress:
AT+NWOTADWPROG=cert_key
+NWOTADWPROG:100
OK
Note
● Enabled by default in the SDK</td></tr>
<tr><td rowspan="2">AT+NWOTADWSTOP</td><td>(none)</td><td>Stop while downloading FW.</td></tr>
<tr><td colspan="2">Example
AT+NWOTADWSTOP
OK</td></tr>
<tr><td rowspan="2">AT+NWOTAFWNAME</td><td>(none)</td><td>Read a name in the header of the MCU firmware (Only for MCU FW)</td></tr>
<tr><td colspan="2">Example
AT+NWOTAFWNAME
+NWOTAFWNAME:MCU
OK
Note
● Disabled by default in the SDK
● If __OTA_UPDATE_MCU_FW__ is enabled in SDK, this command will be enabled</td></tr>
</table>

Command	Parameters	Description
AT+NWOTAFWSIZE	(none)	Read a size in the header of the MCU firmware (Only for MCU FW)
	Example AT+NWOTAFWSIZE +NWOTAFWSIZE:4128 OK *Note* ● Disabled by default in the SDK ● If __OTA_UPDATE_MCU_FW__ is enabled in 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* ● Disabled by default in the SDK ● If __OTA_UPDATE_MCU_FW__ is enabled in 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.
	Example AT+NWOTAREADFW=1f2000,128 DA16FMCUÿÿÿÿ12345678901234561234567890123456123456789012345612345678901234561234567890123456 +NWOTAREADFW:COMPLETE OK *Note* ● Disabled by default in the SDK ● If __OTA_UPDATE_MCU_FW__ is enabled in SDK, this command will be enabled	
AT+NWOTATRANSFW	(none)	Transmit an MCU firmware to MCU through UART1. Transmission will be failed if no header (16 bytes) information exist (Only for MCU FW)
	Example AT+ NWOTATRANSFW MCU DA16FMCUÿÿÿÿ12345678901234561234567890123456123456789012345612345678901234561234567890123456123456789012345612345678901234561234567890123456123456789012345612345678901234 …… + NWOTATRANSFW:COMPLETE OK *Note* ● Disabled by default in the SDK ● If __OTA_UPDATE_MCU_FW__ is enabled in SDK, this command will be enabled	

Command	Parameters	Description
AT+NWOTAERASEFW	(none)	Erase the MCU firmware stored in a serial flash of DA16200. (Only for MCU FW)
	Example AT+ NWOTAERASEFW +NWOTAERASEFW:COMPLETE OK *Note* ● Disabled by default in the SDK ● If __OTA_UPDATE_MCU_FW__ is enabled in SDK, this command will be enabled	
AT+NWOTASETADDR	<sflash_addr>	Data to be downloaded can be designated as an address within the range of *User Area* and *TLS Certificate Key* in the SFLASH area. The default value is 0x003A_D000.
	Example AT+NWOTASETADDR=1f3000 +NWOTASETADDR:0x00 OK *Note* ● Enabled by default in the SDK	
AT+NWOTAGETADDR	<fw_type>	Returns the value set with NWOTASETADDR.
	Example AT+NWOTAGETADDR=mcu_fw +NWOTAGETADDR:1f2000 OK AT+NWOTAGETADDR=cert_key +NWOTAGETADDR:1f2000 OK *Note* ● Enabled by default in the SDK	
AT+NWOTAREADFLASH	<sflash_addr>,<size>	Read as much as size from *sflash_addr*.
	Example AT+NWOTAREADFLASH=1f2000,128 MCU_FW ?¨ZDA16FMCUÿÿÿÿ12345678901234561234567890123456123456789012345612345678901234561234567890123456123456789012345612345678901234561234567890123456 OK *Note* ● Enabled by default in the SDK	
AT+NWOTAERASEFLASH	<sflash_addr>,<size>	Delete as much as size from sflash_addr.

Command	Parameters	Description
	Example AT+NWOTAERASEFLASH=1f2000,1000 +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.
	Example AT+NWOTACOPYFLASH=1f2000,1f3000,1000 +NWOTACOPYFLASH:COMPLETE OK *Note* ● Enabled by default in the SDK ● It will be copied in 4 kB increments	

Table 16: OTA Response List

Response	Parameters	Description
+NWOTADWSTART	<status>	Returns the status of FW download. <status>: 0x00 is success. See Table 17 for other status values. For example: +NWOTADWSTART:0x00
+NWOTARENEW	<status>	Returns the status for FW RENEW. <status>: 0x00 is success. See OTA Response Code List for others. For example: +NWOTARENEW:0x00
+NWOTADWPROG	<progress>	Returns the percentage value (%) of the FW download progress. <progress>: Print download progress in percent. For example: +NWOTADWPROG:100
+NWOTADWSTOP	<status>	Returns the status of FW download stop. <status>: 0x00 is success. See Table 17 for other status values. For example: +NWOTADWSTOP:0x00
+NWOTATRANSFW	COMPLETE or FAIL	Returns 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)

Response	Parameters	Description
+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 17 for other status values.
+NWOTAGETADDR	<sflash_addr>	Returns the value of sflash_addr.
(AT+NWOTAREADFLASH)	(Binary)	Returns 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

Table 17: OTA Response Code List

Return Value	Description
0x00	Return success.
0x01	Return failed.
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. Be sure to download both SLIB and RTOS in the case of SDK V2.x.x.x.
0x0C	Failed to allocate memory.

9.4.1 OTA Download Example

SLIB download: (SDK V2.x.x.x only)

```
AT+NWOTADWSTART=slib,https://server/DA16200_SLIB-GEN01-01-1111-000000.img
```

RTOS download:

```
AT+NWOTADWSTART=rtos,https://server/DA16200_RTOS-GEN01-01-1111-000000.img
```

MCU FW download:

```
AT+NWOTADWSTART=other_fw,https://server/mcu_firmware.img
AT+NWOTADWSTART=other_fw,https://server/mcu_firmware.img,ver01
```

Cert Key download:

```
AT+NWOTADWSTART=cert_key,https://server/ca.pem
```

9.4.2 OTA Download Progress Example

SLIB download progress: (SDK V2.x.x.x only)

```
AT+NWOTADWPROG=slib
```

RTOS download progress:

```
AT+NWOTADWPROG=rtos
```

MCU FW download progress:

```
AT+NWOTADWPROG=mcu_fw
```

Cert Key download progress:

```
AT+NWOTADWPROG=cert_key
```

9.4.3 OTA Renew Example

Renew Firmware (reboot with updated FW):

```
AT+NWOTARENEW
```

9.4.4 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=1f2000,128
```

Delete MCU FW stored in the DA16200 SFLASH:

```
AT+NWOTAERASEFW
```

9.4.5 SFLASH User Area Address Setting Example

SET ADDR:

```
AT+NWOTASETADDR=0x1f2000
```

GET ADDR:

```
AT+NWOTAGETADDR=mcu_fw
AT+NWOTAGETADDR=cert_key
```

9.4.6 SFLASH READ/COPY/ERASE Example

SFLASH Read:

```
AT+NWOTAREADFLASH=0x1f2000,128
```

SFLASH Copy:

```
AT+NWOTACOPYFLASH=0x1f2000,0x1f3000,128
```

SFLASH Erase:

```
AT+NWOTAERASEFLASH=0x1f2000,128
```

9.5 Zeroconf Commands

Table 18: Zeroconf Command List

Command	Parameters	Description
AT+NWMDNSSTART	<Mode>	Start the mDNS module. The mDNS module is communicated through multicast. DA16200 could frequently be changed from/to DPM sleep and wake-up states. It may make power consumption more spend. <Mode>: The mode in which the WLAN interface is running, 0 (Station), 1 (Soft-AP).
	Prerequisite DA16200 in STA mode should connect to an AP. The host name of mDNS module should be set up. *Example* AT+NWMDNSSTART=1 OK *Note* ● Disabled by default in the SDK ● If __SUPPORT_ZERO_CONFIG__ is enabled in 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. <Host name>: The name of the host to be registered.
	Example AT+NWMDNSSHNREG=da16x OK *Note* ● Disabled by default in the SDK ● If __SUPPORT_ZERO_CONFIG__ is enabled in SDK, this command will be enabled	

Command	Parameters	Description
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 in STA mode should connect to an AP. The mDNS module should be running. *Example* AT+NWMDNSSRVREG=_WEBAPP,_http,_tcp,80,LIGHT=OFF,FAN=ON OK *Note* ● Disabled by default in the SDK ● If __SUPPORT_ZERO_CONFIG__ is enabled in 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* ● Disabled by default in the SDK ● If __SUPPORT_ZERO_CONFIG__ is enabled in 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 in STA mode should connect to an AP. The mDNS module should be running. The service in the mDNS module should be registered. *Example* AT+NWMDNSSRVDEREG OK *Note* ● Disabled by default in the SDK ● If __SUPPORT_ZERO_CONFIG__ is enabled in SDK, this command will be enabled	
AT+NWMDNSSTOP	(None)	Stop the mDNS module.

Command	Parameters	Description
	Prerequisite DA16200 in STA mode should connect to an AP. The mDNS module should be running. *Example* AT+NWMDNSSTOP OK *Note* ● Disabled by default in the SDK ● If __SUPPORT_ZERO_CONFIG__ is enabled in SDK, this command will be enabled	
	?	Get the string representing the status of the mDNS module, “stopped” or “running”.
	Example AT+NWMDNSSTOP +NWMDNSSTOP:stopped	

9.5.1 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’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
```

9.6 Provision Commands

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

Table 19: Provision Command List

Command	Parameters	Description
AT+PROVSTART	(none)	Removed all profile data in NVRAM and configure a new station or Soft-AP profile is set. Response: ERROR.
	Example AT+PROVSTART +INIT:DONE,1 *Note* ● Disabled by default in the SDK ● If __PROVISION_ATCMD__ is enabled in the SDK (sys_common_features.h), this command will be enabled ● In case of DA16200, __SUPPORT_FACTORY_RST_APMODE__ should be enabled. For DA16600, __SUPPORT_FACTORY_RST_STAMODE__ should be used. ● After restarting, the system will be ready for the provision procedure	
AT+PROVSTAT	(none)	Get status of provisioning. Response: OK or ERROR.
	Example AT+PROVSTAT +ATPROV=STATUS 1 OK *Note* ● Disabled by default in the SDK ● If __PROVISION_ATCMD__ is enabled in the SDK (sys_common_features.h), 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 20: atcmd_provision_stat enmeration

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	

Value	Name	Description
108	ATCMD_PROVISION_NO_URL_PING_FAIL	
109	ATCMD_PROVISION_NO_URL_PING_OK	
110	ATCMD_PROVISION_DNS_OK_PING_FAIL_N_SERVER_OK	
113	ATCMD_PROVISION_DNS_OK_PING_N_SERVER_FAIL	
111	ATCMD_PROVISION_DNS_OK_PING_OK	
112	ATCMD_PROVISION_REBOOT_ACK	Reboot after provisioning

10 Transfer Function Commands

10.1 Socket Commands

Table 21: Socket Command List

Command	Parameters	Description
AT+TRTS	<local_port>	Open a TCP server socket. <local_port>: Local port number of the socket. Response: OK (with '+TRCTS:*** ', see Table 19) or ERROR
	Prerequisite Target system should be connected to an AP. *Example* AT+TRTS=10194 +TRCTS:0,192.168.0.18,41014 OK *Note* ● Enabled by default in the SDK	
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
	Prerequisite Target system should be connected to an AP and a TCP server should be running (for example, 192.168,0.18 with 1025 port) *Example* AT+TRTC=192.168.0.18,1025,1024 OK *Note* ● Enabled by default in the SDK	
AT+TRUSE	<local_port>	Open a UDP socket. <local_port>: Local port number of the socket. Response: OK or ERROR
	Example AT+TRTALL (optional, run this first if 'ERROR' is responded) AT+TRUSE=10195 OK *Note* ● Enabled by default in the SDK	
AT+TRUR	<remote_ip>, <remote_port>	Set remote IP and port of the UDP socket. <remote_ip>: Remote IP address. <remote_port>: Remote port number. Response: OK or ERROR
	Example AT+TRUR=192.168.0.18,1027 OK *Note* ● Enabled by default in the SDK	

Command	Parameters	Description
AT+TRPRT	<cid>	Get session information by CID. <cid>: 0 (TCP server), 1 (TCP client), 2 (UDP) Response: <cid>,[TCP|UDP],<remote_ip>,<remote_port>, <local_port>
	Prerequisite A UDP socket must be opened first in the target system (AT+TRUSE=10195) *Example* AT+TRPRT=2 +TRPRT:2,UDP,192.168.0.18,10194,10195 OK *Note* ● Enabled by default in the SDK	
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 to get a proper response for this command. The example below is the result after the UDP socket is opened in the target system. *Example* AT+TRPALL +TRPALL:2,UDP,192.168.0.18,10194,10195 OK *Note* ● Enabled by default in the SDK	
AT+TRTRM	<cid>	Close (terminate) a session by CID. <cid>: 0 (TCP server), 1 (TCP client), 2 (UDP) Response: OK or ERROR
	Prerequisite The target system should have any connections to UDP or TCP server. *Example* AT+TRTRM=2 OK *Note* ● Enabled by default in the SDK	
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

Command	Parameters	Description
	Example AT+TRSAVE OK *Note* ● Enabled by default in the SDK	

Table 22: Socket Connection Response List

Response	Parameters	Description
+TRCTS	<cid>, <remote_ip>, <remote_port>	A remote TCP client is connected to the TCP server that was opened by AT+TRTS. <cid>: 0 (TCP Server). <remote_ip>: TCP client IP address. <remote_port>: TCP client port number.
	Prerequisite When you send the AT command (AT+TRTS=40000), you can receive this result if there is no error. *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>: 0 (TCP Server). <remote_ip>: TCP client IP address. <remote_port>: TCP client port number.
	Example ; 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>: 1 (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	

10.1.1 Data Transfer Commands

Table 23: Data Transmission Command

Escape Sequence	Parameters	Description
<ESC>S	<cid><length>, <remote_ip >, <remote_port >,<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>: 0 (TCP server), 1 (TCP client), 2 (UDP) <length>: Data length (If this is 0 then read the command until “\r” or “\n” is met. Maximum length is 2048 bytes, but data will be fragmented by MTU). <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 you give 0,0, 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
	Prerequisite The connections should be set up first to run this command, then press the “ESC” key and type the rest of the string after <ESC>. *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 <ESC>S110,192.168.0.18,1024,abcde12345 OK *Note* ● Enabled by default in the SDK	

Table 24: Data Reception Responses

Response	Parameters	Description
+TRDTS	<cid>, <src_ip>,<src_port>, <length>,<data>	Receive data through TCP server socket. <cid>: 0 (TCP Server). <src_ip>: Source IP address. <src_port>: Source port number. <length>: Data length. <data>: Received data.

Response	Parameters	Description
	Prerequisite A remote TCP client should be connected to the TCP server of your target system. *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>: 1 (TCP Client). <src_ip>: Source IP address. <src_port>: Source port number. <length>: Data length. <data>: Received data
	Prerequisite A TCP server should be set up in another system and the TCP Client of your target system should be connected (AT+TRTC=192.168.0.18,1025,1024), then the TCP server sends the data (test). *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>: 2 (UDP Session). <src_ip>: Source IP address. <src_port>: Source port number. <length>: Data length. <data>: Received data.
	Prerequisite A UDP socket (AT+TRUSE=10194) must be opened first in the target system. Then UDP socket in another system should send the datagram (test). *Example* ; When UDP session receives data, +TRDUS:2,192.168.0.18,10194,4,test	

10.1.2 Data Transfer with DPM

10.1.2.1 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 state to DPM mode (AT+DPM=1). When the DA16200 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, the following is shown:

```
+INIT:WAKEUP,UC
```

```
+TRCTS:0,192.168.0.1,42000
```

When the DA16200 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 enters DPM Sleep mode, MCU should send AT+CLRDPMSLPEXT before a message is sent. The DA16200 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, the following is shown:

```
+INIT:WAKEUP,UC
+TRXTS:0,192.168.0.1,42000
```

10.1.2.2 TCP Client

After a connection is made to an AP in the normal BOOT state, connect the TCP client of the DA16200 to a TCP server and save the config to NVRAM. (To save TCP client config information, the DA16200 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 state to DPM mode (AT+DPM=1). When the DA16200 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 10.1.2.1. When the DA16200 receives a message from the server, the following is shown:

```
+INIT:WAKEUP,UC
+TRDTC:1,192.168.5.1,34000,10,1234567890
```

10.1.2.3 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 state to DPM mode. When the DA16200 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 10.3.1. When the DA16200 receives a message from the server, the following is shown:

```
+INIT:WAKEUP,UC
+TRDUS:2,192.168.5.23,35000,10,1234567890
```

10.2 Secure Socket Commands

Table 25: Secure Socket Command List

Command	Parameters	Description
AT+TRSSLINIT	<Role>	Initialize the SSL module. DA16200 allows to create two SSL modules. <Role>: The role of SSL, 1 – Client.
	Example AT+TRSSLINIT=1 +TRSSLINIT:0 *Note* ● Disabled by default in the SDK ● If __SUPPORT_ATCMD_TLS__ is enabled in the SDK (sys_common_features.h), 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 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 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

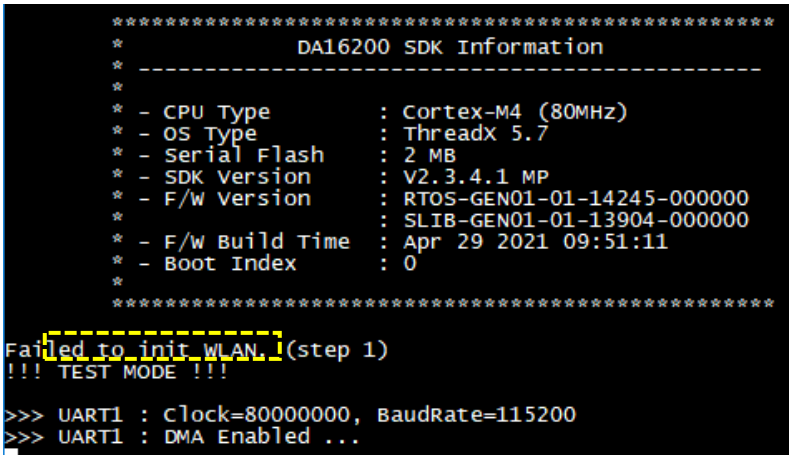
<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></tr>
<tr><td></td><td colspan="2">Prerequisite
The CID obtained after issuing the AT+TRSSLINIT command.
SSL CA certificate and the certificate should be set up to configure them.
Example
0,2,CA_CERT
OK
0,3,CERT
OK
0,6,da16x
OK
0,9,0
OK
0,10,6144
OK
0,11,6144
OK
Note
● Disabled by default in the SDK
● If __SUPPORT_ATCMD_TLS__ is enabled in the SDK (sys_common_features.h), this command will be enabled</td></tr>
<tr><td rowspan="2">AT+TRSSLCO</td><td><CID>,<Server IP Address>,<Server Port number></td><td>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.</td></tr>
<tr><td colspan="2">Prerequisite
The CID obtained after issuing the AT+TRSSLINIT command.
Example
AT+TRSSLCO=0,192.168.0.11,30000
OK
Note
● Disabled by default in the SDK
● If __SUPPORT_ATCMD_TLS__ is enabled in the SDK (sys_common_features.h), this command will be enabled</td></tr>
<tr><td>AT+TRSSLWR</td><td><CID>,<Server IP Address>,<Server Port number>,<Data length>,<Data></td><td>Send the data to the SSL server 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.
<Data length>: The length of data to send.
<Data>: The data to send. Maximum TX payload is 4096 bytes. <Ctrl>+C will close to send data.</td></tr>
</table>

Command	Parameters	Description
	Prerequisite The CID obtained after issuing the AT+TRSSLINIT command. *Example* AT+TRSSLWR=0,10,0123456789<Ctrl> + C OK *Note* ● Disabled by default in the SDK ● If __SUPPORT_ATCMD_TLS__ is enabled in the SDK (sys_common_features.h), this command will be enabled	
AT+TRSSLCL	<CID>	Close the SSL connection. <CID>: The CID obtained after issuing AT+TRSSLINIT command.
	Prerequisite The CID obtained after issuing the AT+TRSSLINIT command. *Example* AT+TRSSLCL=0 OK *Note* ● Disabled by default in the SDK ● If __SUPPORT_ATCMD_TLS__ is enabled in the SDK (sys_common_features.h), this command will be enabled	
AT+TRSSLCERTLIST	<Certificate Type>	Show a list of certificates or a list of CA data available in sflash memory. <Certificate Type>: The type of the certificate. 0 – CA Certificates, 1 – Client/Server Certificates.
	Example AT+TRSSLCERTLIST=0 +TRSSLCERTLIST=0,CA_CERT *Note* ● Disabled by default in the SDK ● If __SUPPORT_ATCMD_TLS__ is enabled in the SDK (sys_common_features.h), this command will be enabled	
AT+TRSSLCERTSTORE	<Certificate Type>, <Sequence>, <Format>, <Name>, <Data>	Store a certificate and CA list data in sflash memory. <Certificate Type>: The type of the certificate. 0 – CA Certificates, 1 – Client/Server Certificates. <Sequence>: If the 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 a number of certificates in a sequence is 1-SSL cert, 2-SSL key. <Format>: The format of the CA/Certificate/Key where: 0: DER, 1: PEM. <Name>: The name of the certificate. While loading certificate and key file separately, the name should be same in both commands. <Data>: The certificate data to be store.

Command	Parameters	Description
	Example AT+TRSSLCERTSTORE=0,0,0,CA_CERT, -----BEGIN CERTIFICATE----- … -----END CERTIFICATE-----<Ctrl>+C OK *Note* ● Disabled by default in the SDK ● If __SUPPORT_ATCMD_TLS__ is enabled in the SDK (sys_common_features.h), this command will be enabled	
AT+TRSSLCERTDELETE	<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.
	Example AT+TRSSLCERTDELETE=0,CA_CERT OK *Note* ● Disabled by default in the SDK ● If __SUPPORT_ATCMD_TLS__ is enabled in the SDK (sys_common_features.h), this command will be enabled	
AT+TRSSLSAVE	(none)	Store the current SSL module's configuration in NVRAM.
	Example AT+TRSSLSAVE OK *Note* ● Disabled by default in the SDK ● If __SUPPORT_ATCMD_TLS__ is enabled in the SDK (sys_common_features.h), this command will be enabled	
AT+TRSSLDELETE	(none)	Delete the stored SSL module's configuration in NVRAM.
	Example AT+TRSSLDELETE OK *Note* ● Disabled by default in the SDK ● If __SUPPORT_ATCMD_TLS__ is enabled in the SDK (sys_common_features.h), this command will be enabled	

11 RF Test Function Commands

Table 26: 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* ● Disabled by default in the SDK ● To use, manufacture image is needed or contact Dialog ● To test RF performance, set the boot mode as RF test mode (AT+TMRFNOINIT=1) and restart the DA16200 (AT+RESTART) ● After DA16200 is restarted, "!!! TEST MODE !!!" log is displayed 	
AT+TMLMACINIT	(none)	Initialize LMAC (for test mode). Response: OK or ERROR
	Prerequisite Boot as RF test mode (AT+TMRFNOINIT=1) *Example* AT+TMLMACINIT OK *Note* ● Disabled by default in the SDK ● To use, manufacture image is needed or contact Dialog	
AT+RFTESTSTART	(none)	Start RF test mode.

Command	Parameters	Description
	Prerequisite Boot as RF test mode (AT+TMRFNOINIT=1) Restart the DA16200 (AT+RESTART) *Example* AT+RFTESTSTART OK *Note* ● Disabled by default in the SDK ● To use, manufacture image is needed or contact Dialog	
AT+RFTX	<Ch>, <BW>, <numFrames>, <frameLen>, <txRate>, <txPower>, <destAddr>, <bssid>, <htEnable>, <GI>, <greenField>, <preambleType>, <qosEnable>, <ackPolicy>, <scrambler>, <aifsnVal>, <ant>	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 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.

Command	Parameters	Description
		Response: OK or ERROR
	Prerequisite Start RF test mode (AT+RFTESTSTART) *Example* ; Tx test with 11N MCS7, 2412MHz and power grade as '0' (max power) AT+RFTX 2412,0,0,1000,n65,0 OK *Note* ● Disabled by default in the SDK ● To use, manufacture image is needed or contact Dialog	
AT+RFTXSTOP	(none)	Stop RF TX test.
	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* ● Disabled by default in the SDK ● To use, manufacture image is needed or contact Dialog ● AT+RFTXSTOP is required before testing other items	
AT+RFCWTEST	<Ch>, <BW>, <txPower>, <ant>, <CWCycle>	Start CW test. <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* ● Disabled by default in the SDK ● To use, manufacture image is needed or contact Dialog	
AT+RFCWSTOP	(none)	Stop CW test. Response: OK or ERROR

Command	Parameters	Description
	Prerequisite Start RF CW test (AT+RFCWTEST) *Example* AT+RFCWTEST 2442,0,2 OK AT+RFCWSTOP OK AT+RFCWTEST 2472,0,2 OK *Note* ● Disabled by default in the SDK ● To use, manufacture image is needed or contact Dialog ● AT+RFCWSTOP is required before testing other items	
AT+RFCONTSTART	<txRate>, <txPower>, <Ch>	Start RF continuous TX test. <txRate>: Data rate. Refer to 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* ● Disabled by default in the SDK ● To use, manufacture image is needed or contact Dialog	
AT+RFCONTSTOP	(none)	Stop RF continuous TX test. 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* ● Disabled by default in the SDK ● To use, manufacture image is needed or contact Dialog	
AT+RFCHANNEL	<Ch>	Change RF channel for PER test. <Ch>: Carrier frequency (2412 ~ 2484 MHz). Response: OK or ERROR

Command	Parameters	Description
	Prerequisite Start RF test mode (AT+RFTESTSTART) *Example* AT+RFCHANNEL 2412 OK *Note* ● Disabled by default in the SDK ● To use, manufacture image is needed or contact Dialog	
AT+RFPERRESET	(none)	Reset PER count. Response: OK or ERROR
	Prerequisite Start RF test mode (AT+RFTESTSTART) *Example* AT+RFPERRESET OK *Note* ● Disabled by default in the SDK ● To use, manufacture image is needed or contact Dialog	
AT+RFPER	(none)	Display PER state. Indicate a number of Valid packets, FCS Errors packets, PHY Errors packets, Overflow Errors. Response: OK or ERROR
	Prerequisite Start RF test mode (AT+RFTESTSTART) *Example* AT+RFPER 20 0 0 0 OK *Note* ● Disabled by default in the SDK ● To use, manufacture image is needed or contact Dialog	
AT+RFTESTSTOP	(none)	Stop RF test mode.
	Example AT+RFTESTSTOP OK *Note* ● Disabled by default in the SDK ● To use, manufacture image is needed or contact Dialog	

Table 27: 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	AT+RESTART	Reboot as RF test mode
ECHO on	ATE	ECHO ON

Test Step	Command	Description
Start RF test mode	AT+RFTESTSTART	Start RF test mode.
Tx Test @11B 1Mbps	AT+RFTX 2412,0,0,200,b1,0	11B 1Mbps/Channel 1
	AT+RFTXSTOP	Stop Tx
	AT+RFTX 2442,0,0,200,b1,0	11B 1Mbps/Channel 7
	AT+RFTXSTOP	Stop Tx
	AT+RFTX 2472,0,0,200,b1,0	11B 1Mbps/Channel 13
	AT+RFTXSTOP	Stop Tx
Tx Test @11G 54Mbps	AT+RFTX 2412,0,0,1000,g54,0	11G 54Mbps/Channel 1
	AT+RFTXSTOP	Stop Tx
	AT+RFTX 2442,0,0,1000,g54,0	11G 54Mbps/Channel 7
	AT+RFTXSTOP	Stop Tx
	AT+RFTX 2472,0,0,1000,g54,0	11G 54Mbps/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
	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

Dialog provides the AT-GUI tool to test RF performance easily. The tool and manual are available on the Dialog website. (https://www.dialog-semiconductor.com/products/wi-fi/da16200#tab-field_tab_content_resources)

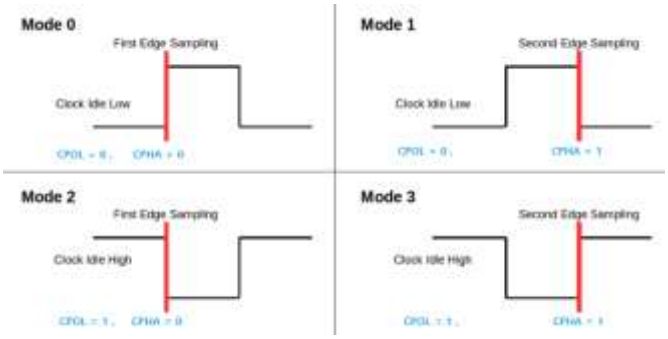
- UM-004 DA16200 AT GUI Tool User Manual
- DA16200 AT GUI Tool

12 System and Peripheral Function Commands

NOTE
Peripheral function commands are disabled in the official AT command binary. To enable, define the feature __SUPPORT_PERI_CMD__ in sys_common_features.h file. These commands are enabled by default in manufacture image.

12.1 SPI Commands

Table 24: 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* ● Disabled by default in the SDK. ● If __SUPPORT_PERI_CMD__ is enabled in the SDK (sys_common_features.h), 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 are clockpol,0 and clockpha 0*  	

12.2 OTP Commands

Table 25: OTP Command List

Command	Parameters	Description
AT+UOTPRDASC	<addr>,<cnt>	Read OTP data. <addr>: OTP address to read four-byte aligned. <cnt>: Bytes to read. Response: OK or Error A string of four-bit HEXA value represented by the ASCII code.
	Example ; Reading four 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* ● Disabled by default in the SDK ● If __SUPPORT_PERI_CMD__ is enabled in the SDK (sys_common_features.h), this command will be enabled ● Physical OTP offset range of DA16200 is h0~h7FF; at each offset, four bytes are stored or read ● For accessing OTP using this command, a four-byte aligned address should be given. For example: h0, h4, h8 ...	
AT+UOTPWRASC	<addr>,<cnt>,<value>	Write OTP data. <addr>: OTP address to write four-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.

Command	Parameters	Description
	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* ● Disabled by default in the SDK ● If __SUPPORT_PERI_CMD__ is enabled in the SDK (sys_common_features.h), this command will be enabled ● Physical OTP offset range of DA16200 is h0~h7FF; at each offset, four bytes are stored or read ● For accessing OTP using this command, a four-byte aligned address should be given. For example: h0, h4, h8 ...	

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

Table 26: OTP Memory Address for Writing MAC Address

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 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 27: Size of Memory by XTAL Offset

Slot	OTP Address	Description	Size (Bytes)
XTAL Offset #0	0x10A	XTAL Offset #0 value	2
XTAL Offset #1	0x10B	XTAL Offset #1 value	2

12.3 XTAL Commands

These commands are used for XTAL calibration and the usage is described in DA16200 Module Production Guide.

Table 28: XTAL Command List

Command	Parameters	Description
AT+XTALWR	<value>	Write XTAL Offset to DA16200 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* ● Disabled by default in the SDK ● If __SUPPORT_PERI_CMD__ is enabled in the SDK (sys_common_features.h), this command will be enabled	
AT+XTALRD	(none)	Read XTAL Offset from DA16200 System. Response: <A string of seven-bit HEXA value represented by the ASCII Code><CR><LF>OK<CR><LF> or Error
	Example AT+XTALWR=7f *Note* ● Disabled by default in the SDK ● If __SUPPORT_PERI_CMD__ is enabled in the SDK (sys_common_features.h), this command will be enabled.	

12.4 Flash Dump Commands

Table 29: 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: <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* ● Disabled by default in the SDK ● If __SUPPORT_PERI_CMD__ is enabled in the SDK (sys_common_features.h), this command will be enabled	

12.5 GPIO Commands

Table 30: GPIO Command List

Command	Parameters	Description
AT+GPIOSTART	<port>, <pin >, <direction>	Configures 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: Sets the pin as an input ● 1: Sets the pin as an output Response: OK or Error
	Example ; To configure GPIOA [3:0] output: ; GPIOs 0,1,2,3 is set to binary 1(0000 0000 0000 1111). AT+GPIOSTART=0,f,1 OK ; To configure GPIOC [8:6] input: ; GPIOs 6, 7, 8 is set to binary 1(0000 0001 1100 0000). AT+GPIOSTART=2,1c0,0 OK *Note* ● Disabled by default in the SDK ● If __SUPPORT_PERI_CMD__ is enabled in the SDK (sys_common_features.h), this command will be enabled	
AT+GPIORD	<port>, <pin>	Reads 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

<table>
<tr><th>Command</th><th>Parameters</th><th>Description</th></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
● Disabled by default in the SDK
● If __SUPPORT_PERI_CMD__ is enabled in the SDK (sys_common_features.h), 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 rowspan="2">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 colspan="2">Prerequisite
The GPIO pin that wants to change output status should be set a direction as output.
Example
; Configure GPIOC[8:6] as output and set to high.
AT+GPIOSTART=2,1c0,1
OK

AT+GPIOWR=2,1c0,1
OK
Note
● Disabled by default in the SDK
● If __SUPPORT_PERI_CMD__ is enabled in the SDK (sys_common_features.h), this command will be enabled</td></tr>
<tr><td>AT+SAVE_PININFO</td><td>(none)</td><td>Save pin mux information.
Response: OK or Error.</td></tr>
</table>

Command	Parameters	Description
	Example AT+SAVE_PININFO OK *Note* ● Disabled by default in the SDK ● If __SUPPORT_PERI_CMD__ is enabled in the SDK (sys_common_features.h), 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* ● Disabled by default in the SDK ● If __SUPPORT_PERI_CMD__ is enabled in the SDK (sys_common_features.h), this command will be enabled ● It is to restore the PIN multiplexing status saved through the AT+SAVE_PININFO command	

12.6 Sleep Commands

Table 31: Sleep Command List

Command	Parameters	Description
AT+*SLEEPMS*	<period>	Make DA16200 go to Sleep mode 3 and wake up after <period> milliseconds. <period>: Wake-up time in milliseconds. Response: OK or Error
	Example AT+SLEEPMS=5000 +INIT:DONE,0 *Note* ● Disabled by default in the SDK ● If __SUPPORT_PERI_CMD__ is enabled in the SDK (sys_common_features.h), this command will be enabled	

13 Examples

13.1 Data Transfer Test

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

- IO Ninja: http://ioninja.com/
  - HEXA data and file transmitting function
- Socket Test: http://sockettest.sourceforge.net/
  - Text data transmittable only
- Script Communicator: http://sourceforge.net/projects/scriptcommunicator
  - Socket communication, UART Rx/Tx data color-distinguished output function and HEXA data transmission

The following sections describe test procedures for socket communication between the DA16200 and a PC with IO Ninja. You can run DA16200 AT commands on a serial terminal application on your PC. The terminal must be connected to the UART1 interface of the DA16200.

13.1.1 TCP Server Socket Test

1. DA16200 AT command:
   a. `AT+TRTS=1234` ← Open a TCP server socket of which the port is 1234.
2. PC:
   a. Select **TCP Connection Socket** (1, Figure 8).
   b. Enter the IP address and the port number of DA16200 (2, Figure 8).
   c. Click  to connect the socket (3, Figure 8).
3. DA16200 AT command:
   a. `+TRCTS:0,192.168.0.5,3713` ← A TCP client socket connected, and IP address is 192.168.0.5 and port is 3713.
4. PC:
   a. Send data (4, Figure 8).
5. DA16200 AT command:
   a. `+TRDTS:0,192.168.0.5,3713,10,DIA_AT_TCP` ← Received ten bytes of data “DIA_AT_TCP”.

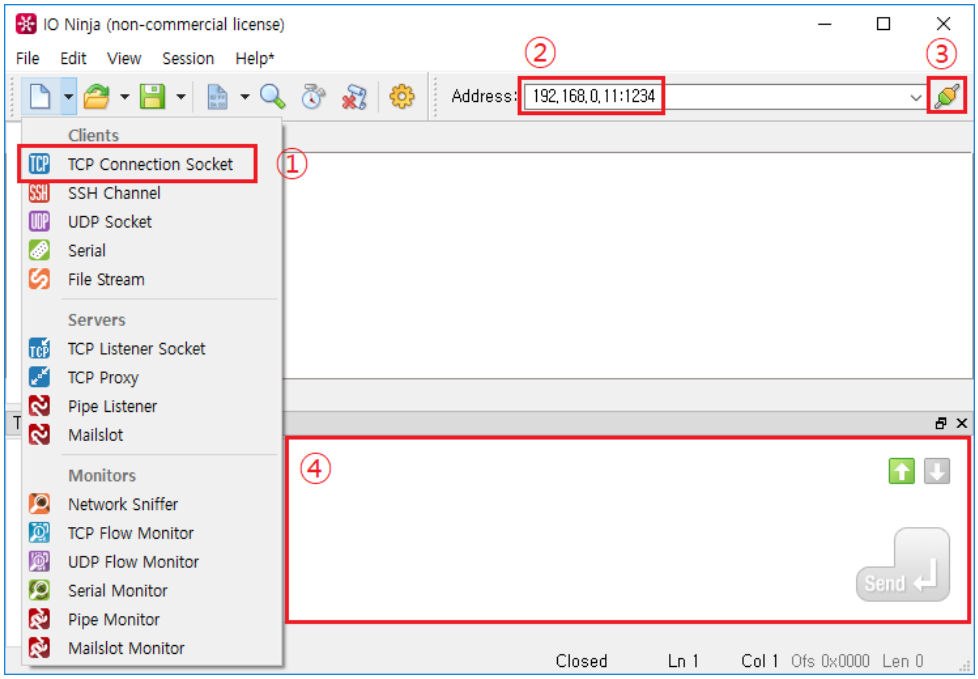


Figure 8: IO Ninja – TCP Client Socket Setting

13.1.2 TCP Client Socket Test

1. PC:
   a. Select **TCP Listener Socket** (1, Figure 9).
   b. Enter the port number to be used (2, Figure 9).
   c. Click  to start to “Listen” (3, Figure 9).
2. DA16200 AT command:
   a. `AT+TRTC=192.168.0.5,1234,2300` ← Open a TCP client socket and set the server IP (192.168.0.5), port (1234), and the local port (2300).
   b. `<ESC>S18,0,0,12345678` ← Send eight bytes of data “12345678”.
3. PC:
   a. Received data.
   b. Send data (4, Figure 9).
4. DA16200 AT command:
   a. `+TRDTC:1,10,DIA_AT_TCP` ← Received ten bytes of data “DIA_AT_TCP”.

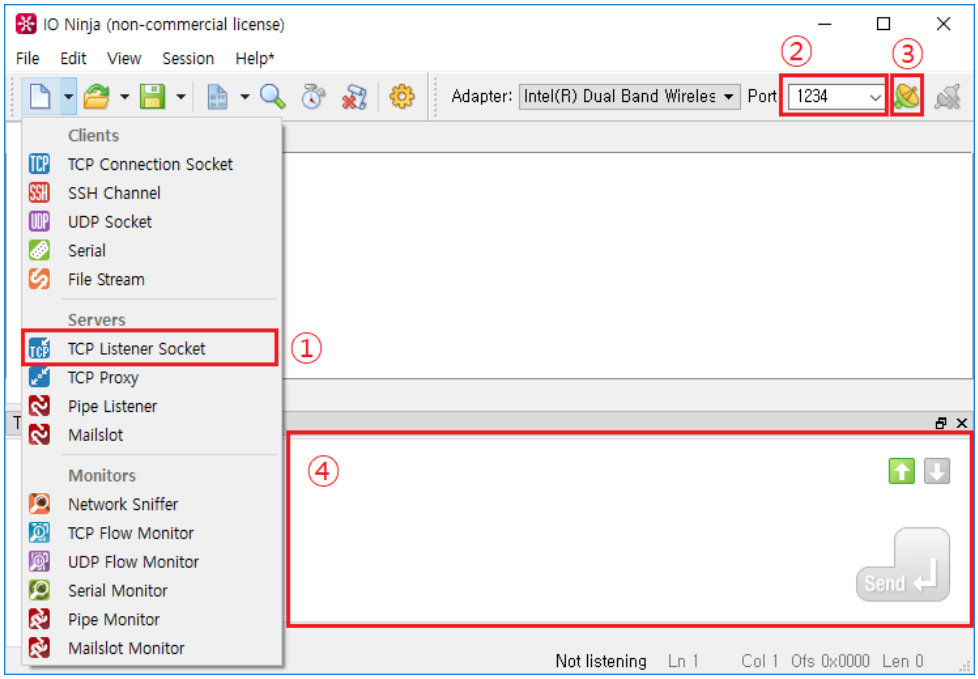


Figure 9: IO Ninja – TCP Server Socket Setting

13.1.3 UDP Socket Test

1. PC:
   a. Select **UDP Socket** (1, Figure 10).
   b. Enter the port number to be used and click  to open the socket (2, Figure 10).
   c. Enter the IP address and port of the counterpart's UDP socket, click  and get ready for data transmission (3, Figure 10).
   d. Enter data and click **Send** to transmit (4, Figure 10).
2. DA16200 AT command:
   a. `AT+TRUSE=4567` ← Open a UDP socket and set the local port (4567).
   b. `AT+TRUR=192.168.0.5,1234` ← Set the remote IP (192.168.0.5) and port (1234).
   c. `<ESC>S210,0,0,1234567890` ← Send ten bytes of data "1234567890".
   d. `+TRDTC:0,10,DIA_AT_UDP` ← Received ten bytes of data "DIA_AT_UDP".

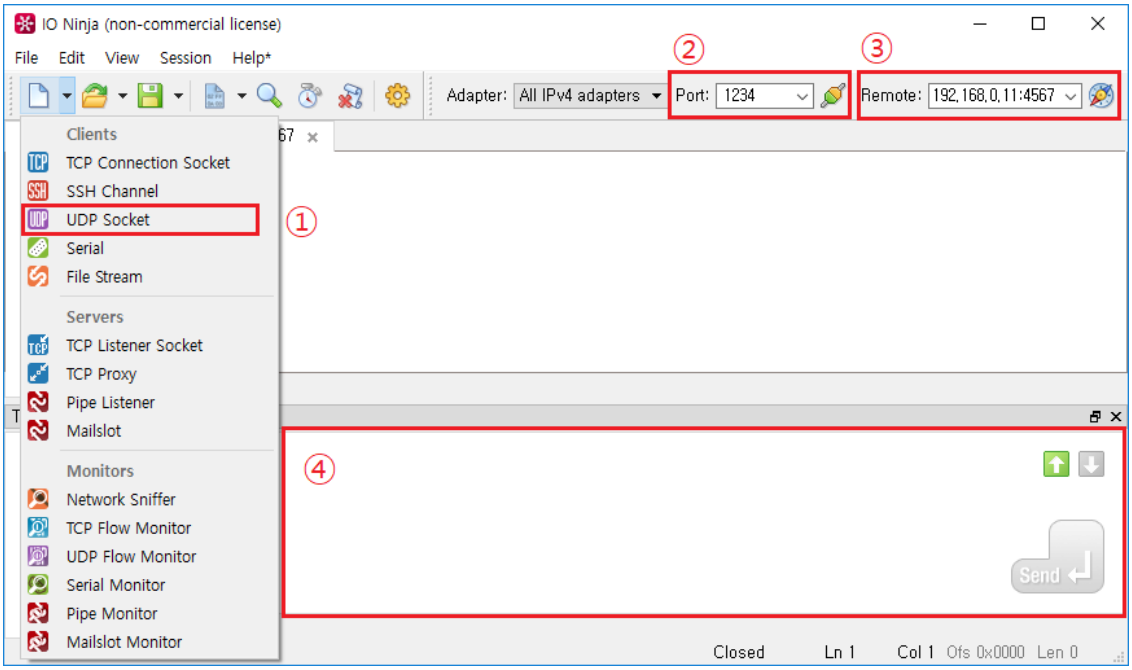


Figure 10: IO Ninja – UDP Socket Setting

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

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

Define	Value	Define	Value
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		

Appendix C User UART Configuration

C.1 How to Run AT-CMD on UART2

AT-CMD is running on UART1 by default.

AT-CMD, depending on user hardware's configuration, can be running on UART2.

Build SDK with the following change (in **bold** font) in config_generic_sdk.h, then UART2 is used for AT-CMD.

```
...
// AT-CMD features
      #define __SUPPORT_ATCMD__                              // Support AT-CMD
        #ifdef __SUPPORT_ATCMD__
              #undef __ATCMD_IF_UART1__              // AT-CMD over UART1
              #define __ATCMD_IF_UART2__             // AT-CMD over UART2

              #undef __USER_UART_CONFIG__ // Support Customer's UART configuration

              #undef __ATCMD_IF_SPI__                // AT-CMD over SPI
              #undef __ATCMD_IF_SDIO__               // AT-CMD over SDIO
        #endif /* __SUPPORT_ATCMD__ */
...
```

C.2 User UART Configuration

There's a feature called "User UART Configuration" that is enabled by __USER_UART_CONFIG__ (available in SDK V2.3.4.0 and later version).

If SDK is built with __USER_UART_CONFIG__, a user can configure AT-CMD's UART Setting programmatically. (ATB will not be available with __USER_UART_CONFIG__ enabled).

For example, if a user wants to run AT-CMD on UART2 w/ static baud rate as 230400, SDK should be configured as below (see necessary changes highlighted in **bold** font) before build.

```
// config_generic_sdk.h
...
// AT-CMD features
      #define __SUPPORT_ATCMD__                              // Support AT-CMD
        #ifdef __SUPPORT_ATCMD__
              #undef __ATCMD_IF_UART1__     // AT-CMD over UART1
              #define __ATCMD_IF_UART2__    // AT-CMD over UART2

              #define __USER_UART_CONFIG__ // Support Customer's UART configuration
              ...
        #endif /* __SUPPORT_ATCMD__ */
...

// user_atcmd.h
...
#if defined ( __USER_UART_CONFIG__ )
/*
 * Customer configuration for AT-CMD 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 boots, AT-CMD 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: AT-CMD is initialized in 115200
  - MCU: ATB=230400
  - MCU: Now it should change its UART baud rate to 230400 to communicate with DA16200

// __USER_UART_CONFIG__ **enabled**

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

Appendix D DA16200 Cipher Suites

No.	Cipher Suite Supported by DA16200	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

No.	Cipher Suite Supported by DA16200	Hex Code
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
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

Revision History

Revision	Date	Description
2.5	07-Sep-2021	Updated table format of ATCMD (Prerequisite, example, note added) Added Section 9.1 Zeroconf Commands Added Section 10.2 Secure Socket Commands
2.4	17-June-2021	Updated MQTT Section 9.1.3, 9.1.4, and 9.1.5 AT+NWDHDNS deleted (not needed as WAN Port not available in Soft-AP mode) Added AT+NWMQALPN, AT+NWMQSNI, AT+NWMQCSUIT, AT+SETDPMSLP1EXT, AT+DPMABNWFCCNT Updated AT+WFJAP, AT+WFJAPA : Optional parameter <hidden> added Updated Section 9.1.1, 9.1.2
2.3	01-Apr-2021	Added OTA Update command Added support for SDK V3.x.x.x
2.2	15-Mar-2021	Added Appendix B HTTP API Return Values Added Appendix C Added AT+NWMQAUTO and ATB
2.1	13-Jan-2021	New section added: Section 8.1 Wi-Fi Function Commands for WPA3 minor update (typo, or minor change done)
2.0	08-Dec-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: 7.1.4 MQTT Example: Changing Subscription Topic while running 7.1.5 MQTT Example: Reading Subscription Topic while running
1.9	11-Nov-2020	AT+NWCCRT, <ESC>C updated AT+NWSNS updated AT+NWHTS updated AT+NWHTSS updated
1.8	18-Aug-2020	Added SNTP command to Section 5 Network Function Commands Added Http-client command to Section 7.2 HTTP Commands Added MCU FW update command using OTA to Section 7.3 OTA Commands paragraph
1.7	30-June-2020	Added 2.4 Configuration for MCU Wake-up Correct typos and wordings
1.6	29-Apr-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	03-Apr-2020	Added AT+BIDX for changing boot index in Chapter 4. Added example code of MQTT command in Section 7.1.1 ~ 7.1.3 Updated RF Test function commands in Chapter 9. Updated GPIO commands in Section 10.5
1.4	21-Oct-2019	Updated 2.2 Serial Port configuration steps. Removed draft status
1.3	15-Oct-2019	Error correction Added explanation to serial program at page 6

Revision	Date	Description
1.2	07-Oct-2019	Editorial review and add code: UM-B-111
1.1	25-Jul-2019	Added OTP Memory Address for writing MAC address in page 27
1.0	03-Jul-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.

Disclaimer

Unless otherwise agreed in writing, the Dialog Semiconductor products (and any associated software) referred to in this document are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of a Dialog Semiconductor product (or associated software) can reasonably be expected to result in personal injury, death or severe property or environmental damage. Dialog Semiconductor and its suppliers accept no liability for inclusion and/or use of Dialog Semiconductor products (and any associated software) in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Information in this document is believed to be accurate and reliable. However, Dialog Semiconductor does not give any representations or warranties, express or implied, as to the accuracy or completeness of such information. Dialog Semiconductor furthermore takes no responsibility whatsoever for the content in this document if provided by any information source outside of Dialog Semiconductor.

Dialog Semiconductor reserves the right to change without notice the information published in this document, including, without limitation, the specification and the design of the related semiconductor products, software and applications. Notwithstanding the foregoing, for any automotive grade version of the device, Dialog Semiconductor reserves the right to change the information published in this document, including, without limitation, the specification and the design of the related semiconductor products, software and applications, in accordance with its standard automotive change notification process.

Applications, software, and semiconductor products described in this document are for illustrative purposes only. Dialog Semiconductor makes no representation or warranty that such applications, software and semiconductor products will be suitable for the specified use without further testing or modification. Unless otherwise agreed in writing, such testing or modification is the sole responsibility of the customer and Dialog Semiconductor excludes all liability in this respect.

Nothing in this document may be construed as a license for customer to use the Dialog Semiconductor products, software and applications referred to in this document. Such license must be separately sought by customer with Dialog Semiconductor.

All use of Dialog Semiconductor products, software and applications referred to in this document is subject to Dialog Semiconductor's Standard Terms and Conditions of Sale, available on the company website (www.dialog-semiconductor.com) unless otherwise stated.

Dialog, Dialog Semiconductor and the Dialog logo are trademarks of Dialog Semiconductor Plc or its subsidiaries. All other product or service names and marks are the property of their respective owners.

RoHS Compliance

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

Contact Dialog Semiconductor

General Enquiry:

Enquiry Form

Local Offices:

https://www.dialog-semiconductor.com/contact/sales-offices