FC41D AT Commands Manual

Wi-Fi&Bluetooth Module Series

Version: 1.0.0

Date: 2021-08-25

Status: Preliminary

At Quectel, our aim is to provide timely and comprehensive services to our customers. If you require any assistance, please contact our headquarters:

Quectel Wireless Solutions Co., Ltd.
Building 5, Shanghai Business Park Phase III (Area B), No.1016 Tianlin Road, Minhang District, Shanghai 200233, China
Tel: +86 21 5108 6236
Email: info@quectel.com

Or our local offices. For more information, please visit:
http://www.quectel.com/support/sales.htm.

For technical support, or to report documentation errors, please visit:
http://www.quectel.com/support/technical.htm.
Or email us at: support@quectel.com.

Legal Notices

We offer information as a service to you. The provided information is based on your requirements and we make every effort to ensure its quality. You agree that you are responsible for using independent analysis and evaluation in designing intended products, and we provide reference designs for illustrative purposes only. Before using any hardware, software or service guided by this document, please read this notice carefully. Even though we employ commercially reasonable efforts to provide the best possible experience, you hereby acknowledge and agree that this document and related services hereunder are provided to you on an “as available” basis. We may revise or restate this document from time to time at our sole discretion without any prior notice to you.

Use and Disclosure Restrictions

License Agreements

Documents and information provided by us shall be kept confidential, unless specific permission is granted. They shall not be accessed or used for any purpose except as expressly provided herein.

Copyright

Our and third-party products hereunder may contain copyrighted material. Such copyrighted material shall not be copied, reproduced, distributed, merged, published, translated, or modified without prior written consent. We and the third party have exclusive rights over copyrighted material. No license shall be granted or conveyed under any patents, copyrights, trademarks, or service mark rights. To avoid ambiguities, purchasing in any form cannot be deemed as granting a license other than the normal non-exclusive, royalty-free license to use the material. We reserve the right to take legal action for noncompliance with abovementioned requirements, unauthorized use, or other illegal or malicious use of the material.

Trademarks

Except as otherwise set forth herein, nothing in this document shall be construed as conferring any rights to use any trademark, trade name or name, abbreviation, or counterfeit product thereof owned by Quectel or any third party in advertising, publicity, or other aspects.

Third-Party Rights

This document may refer to hardware, software and/or documentation owned by one or more third parties ("third-party materials"). Use of such third-party materials shall be governed by all restrictions and obligations applicable thereto.

We make no warranty or representation, either express or implied, regarding the third-party materials, including but not limited to any implied or statutory, warranties of merchantability or fitness for a particular purpose, quiet enjoyment, system integration, information accuracy, and non-infringement of any third-party intellectual property rights with regard to the licensed technology or use thereof. Nothing herein constitutes a representation or warranty by us to either develop, enhance, modify, distribute, market, sell, offer for sale, or otherwise maintain production of any our products or any other hardware, software, device, tool, information, or product. We moreover disclaim any and all warranties arising from the course of dealing or usage of trade.

Disclaimer

a) We acknowledge no liability for any injury or damage arising from the reliance upon the information.

b) We shall bear no liability resulting from any inaccuracies or omissions, or from the use of the information contained herein.

c) While we have made every effort to ensure that the functions and features under development are free from errors, it is possible that they could contain errors, inaccuracies, and omissions. Unless otherwise provided by valid agreement, we make no warranties of any kind, either implied or express, and exclude all liability for any loss or damage suffered in connection with the use of features and functions under development, to the maximum extent permitted by law, regardless of whether such loss or damage may have been foreseeable.

d) We are not responsible for the accessibility, safety, accuracy, availability, legality, or completeness of information, advertising, commercial offers, products, services, and materials on third-party websites and third-party resources.

About the Document

Revision History

Version	Date	Author	Description
-	2021-06-18	Brave LIU/ Lidong ZHOU/ Chris PENG	Creation of the document
1.0.0	2021-08-25	Brave LIU/ Lidong ZHOU/ Chris PENG	Preliminary

Contents

Table Index

1 Introduction

This document mainly introduces Wi-Fi, BLE, TCP/UDP, SSL, MQTT and HTTP(s) related AT commands supported by FC41D module.

1.1. Definitions

- **<CR>** Carriage return character.
- **<LF>** Line feed character.
- **<...>** Parameter name. Angle brackets do not appear on the command line.
- **[...]** Optional parameter of a command or an optional part of TA information response. Square brackets do not appear on the command line. When an optional parameter is not given in a command, the new value equals to its previous value or the default settings, unless otherwise specified.
- **Underline** Default setting of a parameter.

1.2. AT Command Syntax

All command lines must start with **AT** or **at** and end with **<CR>**. Information responses and result codes always start and end with a carriage return character and a line feed character: **<CR><LF><response><CR><LF>**. In tables presenting commands and responses throughout this document, only the commands and responses are presented, and **<CR>** and **<LF>** are deliberately omitted.

AT commands implemented by FE41D fall into three categories syntactically: “Basic”, “S Parameter” and “Extended”, as listed below:

- **Basic Command**

These AT commands have the format of **AT<x><n>**, or **AT&<x><n>**, where **<x>** is the command, and **<n>** is/are the argument(s) for that command. An example of this is **ATE<n>**, which tells the DCE (Data Circuit-terminating Equipment) whether received characters should be echoed back to the DTE (Data Terminal Equipment) according to the value of **<n>**. **<n>** is optional and a default will be used if it is omitted.

- **Extended Command**

These commands can be operated in several modes, as shown in the following table:

Table 1: Types of AT Commands

Command Type	Syntax	Description
Test Command	**AT+<cmd>=?**	Test the existence of corresponding Write Command and return information about the type, value, or range of its parameter.
Read Command	**AT+<cmd>?**	Check the current parameter value of a corresponding Write Command.
Write Command	**AT+<cmd>=<p1>[,<p2>[,<p3>[...]]]**	Set user-definable parameter value.
Execution Command	**AT+<cmd>**	Return a specific information parameter or perform a specific action.

Multiple commands can be placed on a single line using a semi-colon (**;**) between commands. In such cases, only the first command should have **AT** prefix. Commands can be in upper or lower case.

Spaces should be ignored when you enter AT commands, except in the following cases:

- Within quoted strings, where spaces are preserved;
- Within an unquoted string or numeric parameter;
- Within an IP address;
- Within the AT command name up to and including a **=**, **?** or **=?**.

On input, at least a carriage return is required. A newline character is ignored so it is permissible to use carriage return/line feed pairs on the input.

If no command is entered after the **AT** token, **OK** will be returned. If an invalid command is entered, **ERROR** will be returned.

Optional parameters, unless explicitly stated, need to be provided up to the last parameter being entered.

1.3. AT Command Responses

When the AT command processor has finished processing a line, it will output **OK**, **ERROR** or **+CME ERROR: <err>** to indicate that it is ready to accept a new command. Solicited information responses are sent before the final **OK**, **ERROR** or **+CME ERROR: <err>**.

Responses will be in the format of:

<CR><LF>+CMD1:<parameters><CR><LF>
<CR><LF>OK<CR><LF>

Or

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

1.4. Declaration of AT Command Examples

The AT command examples in this document are provided to help you learn about how to use the AT commands introduced herein. The examples, however, should not be taken as Quectel's recommendation or suggestions about how you should design a program flow or what status you should set the module into. Sometimes multiple examples may be provided for one AT command. However, this does not mean that there exists a correlation among these examples and that they should be executed in a given sequence.

2 AT Commands Description

2.1. Description of Wi-Fi Related AT Commands

2.1.1. AT+QRST Restart Module

This command restarts the module.

AT+QRST Restart Module	
Execution Command **AT+QRST**	Response **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	/

2.1.2. AT+QVERSION Get Firmware Version

This command gets firmware version of the module.

AT+QVERSION Get Firmware Version	
Execution Command **AT+QVERSION**	Response **+QVERSION: <version>** **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	/

Parameter

<version>	String type. Firmware version number.

2.1.3. AT+QGETIP Get IP Address

This command gets IP address of the module.

AT+QGETIP Get IP Address	
Write Command **AT+QGETIP=<mode>**	Response **+QGETIP: <IP>** **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	/

Parameter

<mode>	String type. Working mode of Wi-Fi. "station" Station mode "ap" Access Point mode
<IP>	String type. IP address of the module.

2.1.4. AT+QSETBAND Configure Baud Rate and Save the Configuration to Flash

This command configures baud rate and save the configuration to the flash.

AT+QSETBAND Configure Baud Rate and Save the Configuration to Flash	
Write Command **AT+QSETBAND=<baud_rate>**	Response **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	This command takes effect immediately. The configuration will be saved automatically.

Parameter

<baud_rate>	Integer type. Serial port baud rate.

2.1.5. AT+QWLANOTA Start OTA Upgrade

This command starts OTA upgrade on the firmware.

AT+QWLANOTA Start OTA Upgrade	
Write Command **AT+QWLANOTA=<URL>**	Response **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	/

Parameter

<URL>	String type. The address where the firmware package is stored on the server.

2.1.6. AT+QDEEPSLEEP Enter Deep Sleep Mode

This command configures the module to enter deep sleep mode.

AT+QDEEPSLEEP Enter Deep Sleep Mode	
Execution Command **AT+QDEEPSLEEP**	Response **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	/

2.1.7. AT+QWLMAC Get MAC Address

This command gets MAC address of the module.

AT+QWLMAC Get MAC Address	
Execution Command **AT+QWLMAC**	Response **+QWLMAC: <MAC>** **OK** Or **ERROR**

Maximum Response Time	300 ms
Characteristics	/

Parameter

<MAC>	Hexadecimal numbers separated by colon symbols. The default MAC address of the module is c8:47:8c:42:00:48.

2.1.8. AT+QAIRKISS Enable/Disable AirKiss Function

This command enables or disables AirKiss function.

AT+QAIRKISS Enable/Disable AirKiss	
Write Command **AT+QAIRKISS=<enable>**	Response **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	/

Parameter

<enable>	Integer type. Enable/Disable AirKiss function. 0 Disable. 1 Enable.

2.1.9. AT+QSTADHCP Enable/Disable Station DHCP

This command enables or disables station DHCP.

AT+QSTADHCP Enable/Disable Station DHCP	
Write Command **AT+QSTADHCP=<enable>**	Response **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	This command takes effect immediately. The configuration will not be saved.

Parameter

<enable>	Integer type. Enable/Disable station DHCP. 0 Disable 1 Enable

2.1.10. AT+QSTADHCPDEF Enable/Disable Station DHCP and Save the Configuration

This command enables or disables station DHCP and saves the configuration.

AT+QSTADHCPDEF Enable/Disable Station DHCP and Save the Configuration	
Write Command **AT+QSTADHCPDEF=<enable>**	Response **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	This command takes effect immediately. The configuration will be saved automatically.

Parameter

<enable>	Integer type. Enable/Disable station DHCP. 0 Disable. 1 Enable

2.1.11. AT+QSOFTAP Enable AP Mode

This command enables AP mode.

AT+QSOFTAP Enable AP Mode	
Write Command **AT+QSOFTAP=<SSID>,<key>**	Response **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	This command takes effect immediately. The configuration will not be saved.

Parameter

<SSID>	String type. AP name.
<key>	String type. AP key.

2.1.12. AT+QAPSTATE Query AP Mode State

This command queries AP mode state.

AT+QAPSTATE Query AP Mode State	
Execution Command **AT+QAPSTATE**	Response **+QAPSTATE: <state>** **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	/

Parameter

<state>	String type. AP mode state.	
	"SOFTAP_DOWN"	Disabled
	"SOFTAP_UP"	Enabled

2.1.13. AT+QAPSTATIC Configure Static IP of AP Mode

This command configures static IP of AP mode.

AT+QAPSTATIC Configure Static IP of AP Mode	
Write Command **AT+QAPSTATIC=<IP>,<mask>,<gate>,<DNS>**	Response **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	This command takes effect immediately. The configuration will not be saved.

Parameter

<IP>	String Type. Static IP address of AP mode.
<mask>	String Type. Subnet Mask.
<gate>	String Type. Gateway.
<DNS>	String Type. Domain name.

2.1.14. AT+QSOFTAPSTOP Disable AP Mode

This command disables AP mode.

AT+QSOFTAPSTOP Disable AP Mode	
Execution Command **AT+QSOFTAPSTOP**	Response **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	/

2.1.15. AT+QSTASTATIC Configure Static IP of Station Mode

This command configures static IP of station mode.

AT+QSTASTATIC Configure Static IP of Station Mode	
Write Command **AT+QSTASTATIC=<IP>,<mask>,<gate>,<DNS>**	Response **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	This command takes effect immediately. The configuration will not be saved.

Parameter

<IP>	String Type. Static IP address of station mode.
<mask>	String Type. Subnet Mask.
<gate>	String Type. Gateway.
<DNS>	String Type. Domain name.

2.1.16. AT+QSTAAPINFO Connect Hotspot

This command connects a hotspot.

AT+QSTAAPINFO Connect Hotspot	
Write Command **AT+QSTAAPINFO=<SSID>,<pwd>**	Response **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	This command takes effect immediately. The configuration will not be saved.

Parameter

<SSID>	String type. Name of the hotspot to be connected.
<pwd>	String type. Key of the hotspot to be connected.

2.1.17. AT+QSTAAPINFODEF Connect Hotspot and Save Hotspot Information

This command connects a hotspot and saves the connected hotspot information.

AT+QSTAAPINFODEF Connect Hotspot and Save Hotspot Information	
Write Command **AT+QSTAAPINFO_DEF=<SSID>,<pwd>**	Response **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	This command takes effect immediately. The configuration will be saved automatically.

Parameter

<SSID>	String type. Name of the hotspot to be connected.
<pwd>	String type. Key of the hotspot to be connected.

2.1.18. AT+QSTASTOP Disable Station Mode

This command disables station mode.

QSTASTOP Disable Station Mode	
Execution Command **AT+QSTASTOP**	Response **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	/

2.1.19. AT+QSTAST Query Station Mode State

This command queries station mode state.

AT+QSTAST Query Station Mode State	
Execution Command **AT+QSTAST**	Response **+QSTAST: <state>** **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	/

Parameter

<state>	String type. Station mode state.	
	"STATION_DOWN"	Disabled
	"STATION_UP"	Enabled

2.1.20. AT+QGETWIFISTATE Query the Connected Hotspot

This command queries the connected hotspot when the module is working in station mode.

AT+QGETWIFISTATE Query the Connected Hotspot	
Execution Command **AT+QGETWIFISTATE**	Response **+QGETWIFISTATE: <SSID>,<PSK_type>,<RSSI>,<BSSID>,<channel>**

	OK Or **ERROR**
Maximum Response Time	300 ms
Characteristics	/

Parameter

<SSID>	String type. Name of the connected hotspot
<PSK_type>	String type. Encryption type.
<RSSI>	Integer type. Wi-Fi signal strength.
<BSSID>	String type. MAC address of Wi-Fi network card.
<channel>	Integer type. Channel used by the connected hotspot.

2.1.21. AT+QWSCAN Query the Scanned Wi-Fi Information

This command queries the scanned Wi-Fi information.

AT+QWSCAN Query the Scanned Wi-Fi Information	
Execution Command **AT+QWSCAN**	Response **+QWSCAN: <SSID>,<PSK_type>,<RSSI>,<BSSID>,<channel>** **[…]** **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	/

Parameter

<SSID>	String type. Name of the scanned Wi-Fi.
<PSK_type>	String type. Encryption type.
<RSSI>	Integer type. Wi-Fi signal strength.
<BSSID>	String type. MAC address of Wi-Fi network card.
<channel>	Integer type. Channel used by the scanned Wi-Fi.

2.2. Description of BLE Related AT Commands

2.2.1. AT+QBLEINIT Initialize BLE Service

This command initializes BLE service.

AT+QBLEINIT Initialize BLE Service	
Execution Command **AT+QBLEINIT=<role>**	Response **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	/

Parameter

<role>	Integer type. Module role. 1 Central 2 Peripheral

2.2.2. AT+QBLEADDR Query BLE Device Address

This command queries BLE device address.

AT+QBLEADDR Query BLE Device Address	
Read Command **AT+QBLEADDR?**	Response **+QBLEADDR: <BLE_addr>** **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	/

Parameter

<BLE_addr>	String type. BLE device address. A 48-bit address is represented in a string of hexadecimal numbers, such as 58:D3:91:01:02:03.

2.2.3. AT+QBLENAME Set BLE Name

This command configures a BLE name.

AT+QBLENAME Set BLE Name	
Read Command **AT+QBLENAME?**	Response **+QBLENAME: <BLE_name>** **OK** Or **ERROR**
Write Command **AT+QBLENAME=<BLE_name>**	Response **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	/

Parameter

<BLE_name>	String type. BLE name. The maximum length is 25 bytes.

2.2.4. AT+QBLEADVPARAM Configure BLE Advertising Parameters

This command configures BLE advertising parameters when the module is working as a peripheral.

AT+QBLEADVPARAM Configure BLE Advertising Parameters	
Write Command **AT+QBLEQADVPARAM=<adv_int_min>,<adv_int_max>**	Response **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	This command takes effect immediately. The configuration will not be saved.

Parameter

<adv_int_min>	Integer type. Minimum advertising interval for non-directed advertisings and low duty-ratio directed advertisings. Range: 0x0020–0x4000. Default value: 0x0800 (1.28 s). Unit: timeslot (1

	timeslot = 0.625 ms. So, the corresponding range is 20 ms–10.24 s.)
<adv_int_max>	Integer type. Maximum advertising interval for non-directed advertisings and low duty-ratio directed advertisings. Range: 0x0020–0x4000. Default value: 0x0800 (1.28 s). Unit: timeslot (1 timeslot = 0.625 ms. So, the corresponding range is 20 ms–10.24 s.)

NOTE

To configure advertising parameters, **AT+QBLEADVPARAM** should be executed before starting advertising with **AT+QBLEADVSTART**.

2.2.5. AT+QBLEADVDATA Set BLE Advertising Data

This command sets BLE advertising data when the module is working as a peripheral.

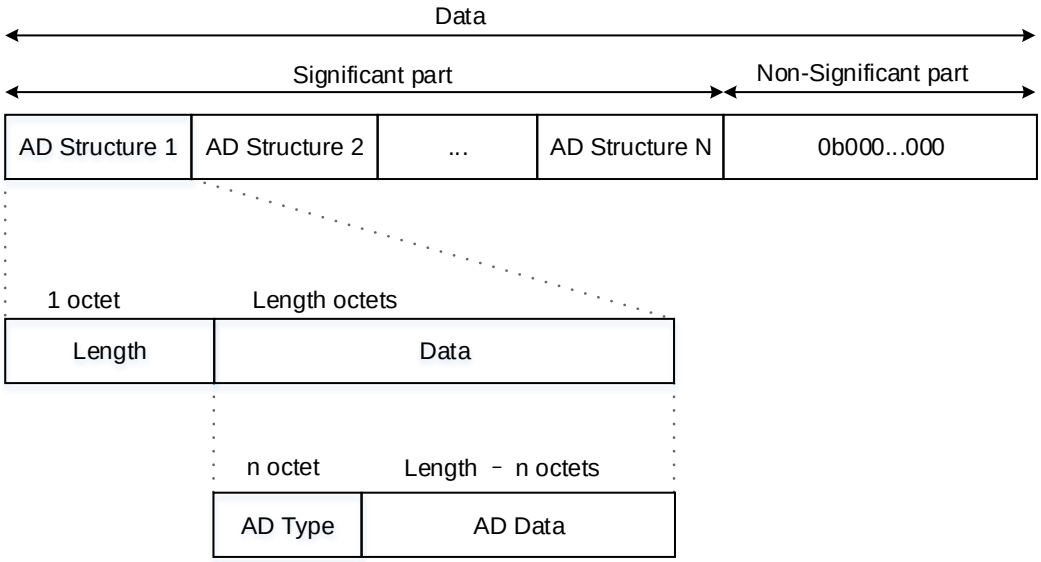

Figure 1: Message Format of BLE Advertising Data

AT+QBLEADVDATA Set BLE Advertising Data	
Write Command **AT+QBLEADVDATA=<adv_data>**	Response **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	This command takes effect immediately. The configuration will not be saved.

Parameter

<adv_data>	String type. Advertising data. Consisting of three fields (i.e, multiple AD Structures). The composition conforms to the message format shown above and the content must be a hexadecimal string. Length: Length of AD structure. The length includes AD type and AD data but not the length of the field length which is 1 byte. The maximum length is 0x1e, that is the maximum length of data field is 30 bytes. AD Type: Advertising data type, such as TX Power Level (0x1A), Local Name (0x09), Le Role (0x1C) and Service UUIDs (0x16). After the peer scans the advertising, the meaning of the advertising data can be determined from the AD Type. AD Data: Advertising data, organized in a big-endian way.

NOTE

For details of the types and meaning of AD Type, please see the official documentation *Core Specification 5.2* (https://www.bluetooth.com/specifications/specs/core-specification/)

2.2.6. AT+QBLEGATTSSRV Establish a BLE Service

This command establishes a BLE service when the module is working as a peripheral.

AT+QBLEGATTSSRV Establish a BLE Service	
Write Command **AT+QBLEGATTSSRV=<srv_UUID>**	Response **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	This command takes effect immediately. The configuration will not be saved.

Parameter

<srv_UUID>	String type. Service UUID. Length: 4 bytes.

NOTE

There is only one BLE service established at a time with this command.

2.2.7. AT+QBLEGATTSCHAR Establish BLE Characteristic

This command establishes BLE characteristics when the module is working as a peripheral.

AT+QBLEGATTSCHAR Establish BLE Characteristic	
Write Command **AT+QBLEGATTSCHAR=<char_UUID>**	Response **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	This command takes effect immediately. The configuration will not be saved.

参数

<char_UUID>	String type. Characteristic UUID. Length: 4 bytes. Up to 5 characteristics are supported.

2.2.8. AT+QBLEADVSTART Start BLE Advertising

This command starts BLE advertising when the module is working as a peripheral.

AT+QBLEADVSTART Start BLE Advertising	
Execution Command **AT+QBLEADVSTART**	Response **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	/

2.2.9. AT+QBLEADVSTOP Stop BLE Advertising

This command stops BLE advertising when the module is working as a peripheral.

AT+QBLEADVSTOP Stop BLE Advertising	
Execution Command **AT+QBLEADVSTOP**	Response **OK** Or **ERROR**
Maximum Response Time	300 ms

Characteristics	/

2.2.10. AT+QBLEGATTSNTFY Send GATT Data

This command sends GATT data when the module is working as a peripheral.

AT+QBLEGATTSNTFY Send GATT Data	
Write Command **AT+QBLEGATTSNTFY=<UUID>,<data>**	Response **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	This command takes effect immediately. The configuration will not be saved.

参数

<UUID>	String type. Characteristic UUID. Length: 4 bytes.
<data>	String type. GATT data.

2.2.11. AT+QBLESCAN Start/Stop BLE Scan

This command starts or stops BLE scan when the module is working as a central.

AT+QBLESCAN Start/Stop BLE Scan	
Write Command **AT+QBLESCAN=<scan>**	Response If **<scan>** is 0: **OK** Or **ERROR** If **<scan>** is 1: **OK** **+QBLESCAN:<name>,<address_type>,<address>** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	/

参数

<scan>	Integer type. Start/stop BLE scan. 0 Stop 1 Start
<name>	String type. BLE device name.
<address_type>	Integer type. BLE device address type. 0 Public address 1 Random address
<address>	String type. BLE device address. Length: 6 bytes.

2.2.12. AT+QBLESCANPARAM Set BLE Scan Parameters

This command sets BLE scan parameters when the module is working as a central.

AT+QBLESCANPARAM Set BLE Scan Parameter	
Write Command **AT+QBLESCANPARAM=<scan_interval>,<scan_window>**	Response **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	This command takes effect immediately. The configuration will not be saved.

Parameter

<scan_interval>	Integer type. This is defined as the time interval from when the Controller started its last LE scan until it begins the subsequent LE scan. Range: 0x0004–0x4000. Default value: 0x0010 (10 ms). Unit: timeslot (1 timeslot = 0.625 ms. So, the corresponding time range is 2.5 ms to 10.24 s.)
<scan_window>	Integer type. The duration of the LE scan. **<scan_window>** shall be less than or equal to **<scan_interval>**. Range: 0x0004–0x4000. Default value: 0x0010 (10 ms). Unit: timeslot (1 timeslot = 0.625 ms. So, the corresponding range is 2.5 ms–10.24 s.)

2.2.13. AT+QBLECONN Connect a Peripheral

This command connects a peripheral when the module is working as a central.

AT+QBLECONN Connect a Peripheral	
Write Command **AT+QBLECONN=<addr_type>,<peer_addr>**	Response **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	This command takes effect immediately. The configuration will not be saved.

参数

<addr_type>	Integer type. Peripheral address type. 0 Public address 1 Random address
<peer_addr>	String type. Peripheral address.

2.2.14. AT+QBLECONNPARAM Configure Connection Parameters

This command configures connection parameters when the module is working as a central.

AT+QBLECONNPARAM Configure Connection Parameters	
Write Command **AT+QBLECONNPARAM=<con_interval>,<timeout>,<latency>**	Response **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	This command takes effect immediately. The configuration will not be saved.

Parameter

<con_interval>	Integer type. Connection interval. Range: 0x0006–0x0C80. Unit: 1.25 ms. So, the corresponding time range is 7.5 ms to 4 s.
<timeout>	Integer type. Supervision timeout for the LE Link. Range: 0x000A to 0x0C80. Unit: 10 ms. So, the corresponding time range is 100 ms to 32 s.
<latency>	Integer type. Slave latency for the connection in a certain number of connection events. Range: 0–499 (0x0000 to 0x01F3).

2.2.15. AT+QBLECFGMTU Configure Maximum Transmission Unit for BLE

This command configures maximum transmission unit for BLE when the module is working as a central.

AT+QBLECFGMTU Configure Maximum Transmission Unit for BLE	
Read Command **AT+QBLECFGMTU?**	Response **+QBLECFGMTU: <MTU_value>** **OK** Or **ERROR**
Write Command **AT+QBLECFGMTU=<MTU_value>**	Response **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	This command takes effect immediately. The configuration will not be saved.

Parameter

<MTU_value>	Integer type. Maximum transmission unit value. Range: 23–512. Unit: byte.

2.2.16. AT+QBLEGATTCWR Send Data

This command sends GATT data when the module is working as a central.

AT+QBLEGATTCWR Send Data	
Write Command **AT+QBLEGATTSNTFY=<UUID>,<data>**	Response **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	This command takes effect immediately. The configuration will not be saved.

Parameter

<UUID>	String type. Characteristic UUID. Length: 4 bytes.
<data>	String type. GATT data.

2.2.17. AT+QBLEGATTCRD Read Data

This command reads GATT data when the module is working as a central.

AT+QBLEGATTCRD Read Data	
Write Command **AT+QBLEGATTCRD=<UUID>**	Response **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	This command takes effect immediately. The configuration will not be saved.

Parameter

<UUID>	String type. Characteristic UUID. Length: 4 bytes.

2.2.18. AT+QBLEDISCONN Disconnect BLE Connection

This command disconnects a BLE connection.

AT+QBLEDISCONN Disconnect BLE Connection	
Execution Command **AT+QBLEDISCONN**	Response **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	/

2.2.19. AT+QBLESTAT Query the State of BLE Device

This command queries the state of BLE device.

AT+QBLESTAT Query the State of BLE Device	
Execution Command **AT+QBLESTAT**	Response **+QBLESTAT: <BLE_state>** **OK** Or **ERROR**

Maximum Response Time	300 ms
Characteristics	/

Parameter

<BLE_state>	String type. The current state of BLE device.	
	"NOINIT"	Uninitialized
	"INIT"	Initialized
	"ADVERTISING"	Advertising
	"NOADVERTISING"	No advertising
	"CONNECTED"	Connected
	"DISCONNECTED"	Disconnected

2.3. Description of TCP/UDP Related AT commands

2.3.1. AT+QIOPEN Open TCP/UDP Socket Service

This command opens TCP/UDP socket service. The maximum response time of establishing TCP connection as a client is determined by the maximum timeout of the TCP three-way handshake. The maximum timeout of the TCP three-way handshake supported by FC41D module is 120 s. The maximum response time of establishing other services is about 300 ms.

AT+QIOPEN Open TCP/UDP Socket Service	
Test Command **AT+QIOPEN=?**	Response **+QIOPEN: (**range of supported **<socketID>**s**),(**list of supported **<service_type>**s**),"ip/dns",(**range of supported **<remote_port>**s**),(**range of supported **<local_port>**s**),(**range of supported **<access_mode>**s**)** **OK**
Write Command **AT+QIOPEN=<socketID>,<service_type>,<remote_IP/domain_name>,<remote_port>,<local_port>,<access_mode>**	Response If **<access_mode>** is 2: **CONNECT** Or **ERROR** If **<access_mode>** is not 2: **OK** **+QIOPEN: <socketID>,<err>**

	Or **ERROR**
Maximum Response Time	/
Characteristics	/

Parameter

<socketID>	Integer type. Socket ID. Range: 0–11.
<service_type>	String type. Socket service type. "TCP" Start a TCP connection as a client "UDP" Start a UDP connection as a client "TCP LISTENER" Start a TCP server to listen for TCP incoming connections "UDP SERVICE" Start a UDP service
<remoteIP>	String type. IP address of the remote server. It is valid only when **<service_type>** is "TCP " or "UDP".
<domain_name>	String type. Domain name of the remote server. It is valid only when **<service_type>** is "TCP" or "UDP".
<remote_port>	Integer type. Port of the remote server. Range: 1–65535. It is valid only when **<service_type>** is "TCP" or "UDP".
<local_port>	Integer type. Local port. Range: 1–65535. When **<service_type>** is "TCP LISTENER"/"UDP SERVICE", local port must be specified.
<access_mode>	Integer type. Data access mode of serial port. 0 Command mode. Socket sends and receives data through AT command. 1 URC Mode. Socket sends data through AT command and receives data through the incoming data indication URC. 2 Transparent transmission mode. The serial port sends the received data through socket directly. The socket pushes the received data to the serial port directly.
<err>	Integer type. Result code. See ***Chapter 5***.

NOTE

When **<service_type>** is "TCP LISTENER"/"UDP SERVICE", **<access_mode>** cannot be set to 2.

2.3.2. AT+QISTATE Query the State of TCP/UDP Socket Service

AT+QISTATE Query the State of TCP/UDP Socket Service	
Write Command **AT+QISTATE=<socketID>**	Response **+QISTATE: <socketID>,<service_type>,<remote_addr>,<**

	remote_port>,<local_port>,<state> **OK**
Read Command **AT+QISTATE?**	Response Return the state of all existing connections: **[+QISTATE: <socketID>,<service_type>,<remote_addr>, <remote_port>,<local_port>,<state>]** **[…]** **OK**
Execution Command **AT+QISTATE**	Response Return the state of all existing connections: **[+QISTATE: <socketID>,<service_type>,<remote_addr>, <remote_port>,<local_port>,<state>]** **[…]** **OK**
Maximum Response Time	300 ms
Characteristics	/

Parameter

<socketID>	Integer type. Socket ID. Range: 0–11.
<service_type>	String type. Socket service type. "TCP" Start a TCP connection as a client "UDP" Start a UDP connection as a client "TCP LISTENER" Start a TCP server to listen for TCP incoming connections "UDP SERVICE" Start a UDP service "TCP INCOMING" Start a TCP connection accepted by the TCP server
<remote_addr>	String type. IP address of the remote server. It is valid only when **<service_type>** is "TCP"/"UDP"/"TCP INCOMING".
<remote_port>	Integer type. Port of the remote server. It is valid only when **<service_type>** is "TCP"/"UDP"/"TCP INCOMING".
<local_port>	Integer type. Local port.
<state>	Integer type. Socket service state. 0 "Initial": client connection has not been established 1 "Opening": client is connecting or server is trying to listen 2 "Connected": client connection has been established 3 "Listening". server is listening. 4 "Closing": client connection is closing

2.3.3. AT+QISEND Send Data Through TCP/UDP Socket Service

AT+QISEND Send Data Through TCP/UDP Socket Service	
Test Command **AT+QISEND=?**	Response **+QISEND: (**range of supported **<socketID>**s**),(**range of supported **<send_len>**s**),"hex_data"[,"remote_ip",(**range of supported **<remote_port>**s**)]** **OK**
Write Command Query details of sending data **AT+QISEND=<socketID>,0**	Response **+QISEND: <total_send_size>,<acked_size>,<unack_size>** **OK**
Write Command **<service_type>** is "UDP SERVICE" **AT+QISEND=<socketID>,<send_len>,<hex_data>,<remote_ip>,<remote_port>**	Response **+QISEND: <actual_send_len>** **OK** Or **ERROR**
Write Command **<service_type>** is "TCP"/"UDP"/"TCP INCOMING" **AT+QISEND=<socketID>,<send_len>,<hex_data>**	Response **+QISEND: <actual_send_len>** **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	/

Parameter

<socketID>	Integer type. Socket ID. Range: 0–11.
<send_len>	Integer type. Length of the data to be sent. Unit: byte.
<hex_data>	Hex string type. The data to be sent.
<remote_ip>	String type. Destination address of the data to be sent to.
<remote_port>	Integer type. Destination port of the data to be sent to. It is valid only when **<service_type>** is "UDP SERVICE". Range: 1–65535.
<actual_send_len>	Integer type. Actual length of the data written to the socket. Unit: byte.
<total_send_size>	Integer type. Total length of the data written to the socket.
<acked_size>	Integer type. Length of the data that has been acknowledged.
<unacked_size>	Integer type. Length of the data that has not been acknowledged.

2.3.4. AT+QIRD Read Received Data from TCP/UDP Socket Service

AT+QIRD Read the Data Received from TCP/UDP Socket Service	
Test Command **AT+QIRD=?**	Response **+QIRD: (**range of supported **<socketID>**s**),(**range of supported **<read_len>**s**)** **OK**
Write Command Query the details of the received data. **AT+QIRD=<socketID>,0**	Response **+QISEND: <total_recv_size>,<read_size>,<unread_size>** **OK**
Write Command **<service_type>** is "UDP SERVICE" **AT+QIRD=<socketID>,<read_len>**	Response **+QIRD: <actual_read_len>,<remote_addr>,<remote_port><CR><LF>data** **OK** Or **ERROR**
Write Command **<service_type>** is "TCP"/"UDP"/"TCP INCOMING" **AT+QIRD=<socketID>,<read_len>**	Response **+QISEND: <actual_read_len><CR><LF>data** **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	/

Parameter

<socketID>	Integer type. Socket ID. Range: 0–11.
<read_len>	Integer type. Length of data to be read. Unit: byte.
<remote_addr>	String type. Source address of the received data.
<remote_port>	Integer type. Source port of received data. It is valid only when **<service_type>** is "UDP SERVICE".
<actual_read_len>	Integer type. Length of the actually read data. Unit: byte.
<total_recv_size>	Integer type. Total length of the data received by the socket.
<read_size>	Integer type. Length of the data that has been read.
<unread_size>	Integer type. Length of the data that has not been read.

2.3.5. AT+QIACCEPT Accept/Reject Remote Incoming Connections from TCP/UDP Socket Service

AT+QIACCEPT Accept/Reject Remote Incoming Connections from TCP/UDP Socket Service	
Test Command **AT+QIACCEPT=?**	Response **+QIACCEPT: (**range of supported **<listener_socketID>**s**),(**list of supported **<accept>**s**),(**range of supported **<incoming_socketID>**s**)** **OK**
Write Command Accept/Reject incoming connection **AT+QIACCEPT=<listener_socketID>,<accept>[,<incoming_socketID>]**	Response **[+QIACCEPT: <incoming_socketID>,<remote_addr>,<remote_port>]** **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	/

Parameter

<listener_socketID>	Integer type. Socket ID of TCP server. Range: 0–11.
<accept>	Integer type. Accept/Reject remote incoming request of TCP/UDP Socket service. 0 Reject 1 Accept
<incoming_socketID>	Integer type. Socket ID for incoming connection. It is valid only when **<accept>** is 1. Range: 0–11.
<remote_addr>	String type. Source address of the incoming connection.
<remote_port>	Integer type. Source port of the incoming connection.

2.3.6. AT+QISWTMD Switch Data Access Mode

AT+QISWTMD Switch Data Access Mode	
Test Command **AT+QISWTMD=?**	Response **+QISWTMD: (**range of supported **<socketID>**s**),(**range of supported **<access_mode>**s**)**

	OK
Write Command **AT+QISWTMD=<socketID>,<access_mode>**	Response **OK** Or **CONNECT** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	/

Parameter

<socketID>	Integer type. Socket ID. Range: 0–11.
<access_mode>	Integer type. Data access mode of serial port. 0 Command mode. Socket sends and receives data through AT command 1 URC mode. Socket sends data through AT command and reports URC indicating incoming data. 2 Transparent transmission mode. The serial port sends the received data through socket directly. The socket pushes the received data to the serial port directly.

2.3.7. AT+QICLOSE Close TCP/UDP Socket Service

AT+QICLOSE Close TCP/UDP Socket Service	
Test Command **AT+QICLOSE=?**	Response **+QICLOSE: (**range of supported **<socketID>**s**),(**range of supported **<close_timeout>**s**)** **OK**
Write Command **AT+QICLOSE=<socketID>,<close_timeout>**	Response **OK** **+QIURC: "closed",<socketID>** Or **ERROR**
Maximum Response Time	/
Characteristics	/

Parameter

<socketID>	Integer type. Socket ID. Range: 0–11.
<close_timeout>	Integer type. The timeout for closing. Range: 0–60. Unit: second.

2.3.8. AT+QICFG Configure Optional Parameters for TCP/UDP Socket Service

AT+QICFG Configure Optional Parameters for TCP/UDP Socket Service	
Test Command **AT+QICFG=?**	Response **+QICFG: "transpktsize",(**range of supported **<transpktsize>**s**)** **+QICFG: "transwaittm",(**range of supported **<transwaittm>**s**)** **+QICFG: "accept/mode",(**list of supported **<accept_mode>**s**)** **OK**
Write Command Configure/query subcontracting criteria for data transmission in transparent transmission mode **AT+QICFG="transpktsize"[,<transpktsize>]**	Response If the optional parameter is omitted, query the current configuration: **+QICFG: "transpktsize",<transpktsize>** **OK** If the optional parameter is specified, configure subcontracting criteria for data transmission in transparent transmission mode: **OK** Or **ERROR**
Write Command Configure/query the waiting time for the serial port to receive user data in transparent transmission mode. **AT+QICFG="transwaittm"[,<transwaittm>]**	Response If the optional parameter is omitted, query the current configuration: **+QICFG: "transwaittm",<transwaittm>** **OK** If the optional parameter is specified, configure the waiting time for the serial port to receive user data in transparent transmission mode: **OK** Or **ERROR**
Write Command Configure/query the acceptance mode of "TCP incoming"	Response If the optional parameter is omitted, query the current configuration:

AT+QICFG="accept/mode"[,<accept_mode>]	**+QICFG: "accept/mode",<accept_mode>** **OK** If the optional parameter is specified, configure the acceptance mode of "TCP incoming": **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	/

Parameter

<transpktsize>	Integer type. Maximum length of the packets to be sent in bytes. Range: 1–1460. Default value: 1024. Unit: byte.
<transwaittm>	Integer type. The waiting time for receiving data if the received data is less then **<transpktsize>** in transparent transmission mode. If **<transwaittm>** is 0, the data is sent as soon as it is received. Range: 0–20. Default value: 2. Unit: 100 ms.
<accept_mode>	Integer type. Acceptance mode of TCP incoming connection. 0 Accept automatically 1 Accept manually by **AT+QIACCEPT**

2.3.9. AT+QIGETERROR Query the Result Code of TCP/UDP Socket Service

AT+QIGETERROR Query the Result Code of TCP/UDP Socket Service	
Execution Command **AT+QIGETERROR**	Response **+QIGETERROR: <err>,<description>** **OK**
Maximum Response Time	300 ms
Characteristics	/

Parameter

<err>	Integer type. Result code. See ***Chapter 5***.
<description>	String type. Description for error.

2.3.10. +++ Exit Transparent Transmission Mode

+++ Exit Transparent Transmission Mode	
Execution Command **+++**	Response **OK**
Maximum Response Time	300 ms
Characteristics	/

NOTE

After exiting transparent Transmission mode with **+++**, if the socket connection is connected, the connection will enter transparent transmission mode again with **ATO**.

2.3.11. ATO Enter Transparent Transmission Mode

ATO Enter Transparent Transmission Mode	
Execution Command **ATO**	Response **CONNECT** Or **NO CARRIER**
Maximum Response Time	300 ms
Characteristic	/

NOTE

If there is no connection in transparent transmission mode before, **ATO** returns **NO CARRIER**.

2.4. Description of SSL Related AT Commands

2.4.1. AT+QSSLCERT Upload/Download/Delete SSL Certificate

AT+QSSLCERT Upload/Download/Delete SSL Certificate	
Test Command	Response

AT+QSSLCERT=?	**+QSSLCERT: (**list of supported **<cert_type>**s**),(**range of supported **<operation_mode>**s**)[,<length>]** **OK**
Write Command (Upload SSL certificate) **AT+QSSLCERT=<cert_type>,2,<length>**	Response **CONNECT** //After **CONNECT** is returned, input file data. The inputted data will be written into flash automatically. When the data length reaches **<length>**, the module exits data mode.. **OK** Or **ERROR**
Write Command (Download SSL certificate) **AT+QSSLCERT=<cert_type>,1**	Response **CONNECT** //Output file data **OK** Or **ERROR**
Write Command (Delete SSL certificate) **AT+QSSLCERT=<cert_type>,0**	Response **OK** Or **ERROR**
Read Command **AT+QSSLCERT?**	Response **[+QSSLCERT: <cert_type>,<exist_flag>]** **[…]** **OK**
Maximum Response Time	Determined by the speed at which the user inputs data
Characteristics	This command takes effect immediately. The configuration will be saved automatically.

Parameter

<cert_type>	String type. Certificate type. "CA" CA certificate "User Cert" Client certificate "User Key" Client Key document
<operation_mode>	Integer type. Operation mode. 0 Delete SSL certificate 1 Download SSL certificate 2 Upload SSL certificate
<length>	Integer type. Length of the certificate content. It is valid only when

	<operation_mode> is 2.
<exist_flag>	Integer type. Whether the certificate exists. 0 Not exist 1 Exist

2.4.2. AT+QSSLCFG Configure/Query SSL Context Parameters

AT+QSSLCFG Configure/Query SSL Context Parameters	
Test Command **AT+QSSLCFG=?**	Response **+QSSLCFG: "version",(**range of supported **<SSL_ctxID>**s**),(**range of supported **<SSL_version>**s**)** **+QSSLCFG: "verify",(**range of supported **<SSL_ctxID>**s**),(**range of supported **<verify_level>**s**)** **+QSSLCFG: "ciphersuite",(**range of supported **<SSL_ctxID>**s**),<cs_id>** **+QSSLCFG: "negotiatetimeout",(**range of supported **<SSL_ctxID>**s**),(**range of supported **<negotiate_time>**s**)** **+QSSLCFG: "sni",(**range of supported **<SSL_ctxID>**s**),(**range of supported **<SNI_value>**s**)** **OK**
Write Command Configure/Query the SSL version **AT+QSSLCFG="version",<SSL_ctxID>[,<SSL_version>]**	Response If the optional parameter is omitted, query the current configuration: **+QSSLCFG: "version",<SSL_ctxID>,<SSL_version>** **OK** If the optional parameter is specified, configure the SSL version: **OK** Or **ERROR**
Write Command Configure/Query the SSL verification level **AT+QSSLCFG="verify",<SSL_ctxID>[,<verify_level>]**	Response If the optional parameter is omitted, query the current configuration: **+QSSLCFG: "verify",<SSL_ctxID>,<verify_level>** **OK** If the optional parameter is specified, configure the SSL verification level:

	OK Or **ERROR**
Write Command Configure/Query the SSL cipher suite **AT+QSSLCFG="ciphersuite",<SSL_ctxID>[,[<cs_id>[,<cs_id>[,…]]]**	Response If the optional parameters are omitted, query the current configuration: **+QSSLCFG: "ciphersuite",<SSL_ctxID>[,<cs_id>[,<cs_id>[,…]]]** **OK** If any of the optional parameters are specified, configure the SSL cipher suite: **OK** Or **ERROR**
Write Command Configure/Query the timeout of SSL negotiation **AT+QSSLCFG="negotiatetimeout",<SSL_ctxID>[,<negotiate_time>]**	Response If the optional parameter is omitted, query the current configuration: **+QSSLCFG: "negotiatetimeout",<SSL_ctxID>,<negotiate_time>** **OK** If the optional parameter is specified, configure the timeout of SSL negotiation: **OK** Or **ERROR**
Write Command Configure/Query TLS Server Name Indication **AT+QSSLCFG="sni",<SSL_ctxID>[,<SNI_value>]**	Response If the optional parameter is omitted, query the current configuration: **+QSSLCFG: "sni",<SSL_ctxID>,<SNI_value>** **OK** If the optional parameter is specified, enable/disable TLS Server Name Indication: **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	This command takes effect immediately.

	The configuration will not be saved.

Parameter

<SSL_ctxID>	Integer type. SSL context ID. Range: 0–5.
<SSL_version>	Integer type. SSL version. 0 SSL3.0. 1 TLS1.0 2 TLS1.1 3 TLS1.2 4 All
<verify_level>	Integer type. SSL verification level. 0 No verification 1 One-way verification (The client verifies the legality of the server). 2 Two-way verification (The client and the server verify the legality of each other).
<cs_id>	Hex integer type. Start with 0x. 0xFFFF supports all cipher suites. 0x0004 TLS_RSA_WITH_RC4_128_MD5 0x0005 TLS_RSA_WITH_RC4_128_SHA 0x002F TLS_RSA_WITH_AES_128_CBC_SHA 0x0035 TLS_RSA_WITH_AES_256_CBC_SHA 0x003C TLS_RSA_WITH_AES_128_CBC_SHA256 0x003D TLS_RSA_WITH_AES_256_CBC_SHA256 0xc027 TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 0xc02F TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
<negotiate_time>	Integer type. Timeout of negotiation. Range: 120–300. Default value: 120. Unit: second.
<SNI_value>	Integer type. Enable/Disable TSL Server Name Indication. 0 Disable 1 Enable

2.4.3. AT+QSSLOPEN Open SSL Client

This command opens an SSL client and establishes an SSL connection. Establishing an SSL connection includes TCP three-way handshake and SSL handshake. Hence the maximum timeout is the sum of TCP three-way handshake timeout (120 s) and **<negotiate_time>**.

AT+QSSLOPEN Open SSL Client	
Test Command **AT+QSSLOPEN=?**	Response **+QSSLOPEN: (**range of supported **<SSL_ctxID>**s**),(**range of supported **<socketID>**s**),"ip/dns",(**range of **<remote_port>**s**),(**range of supported **<local_port>**s**),(**range

	of supported **<access_mode>**s**)** **OK**
Write Command **AT+QSSLOPEN=<SSL_ctxID>,<socketID>,<remote_IP/domain_name>,<remote_port>,<local_port>,<access_mode>**	Response If **<access_mode>** is 2: **CONNECT** Or **ERROR** If **<access_mode>** is not 2: **OK** **+QIOPEN: <socketID>,<err>** Or **ERROR**
Maximum Response Time	/
Characteristics	/

Parameter

<SSL_ctxID>	Integer type. SSL context ID. Range: 0–5.
<socketID>	Integer type. Socket ID. Range: 0–11.
<remoteIP>	String type. IP address of the remote server.
<domain_name>	String type. Domain name of the remote server.
<remote_port>	Integer type. Port of the remote server. Range: 1–65535. It is valid only when **<service_type>** is "TCP" or "UDP".
<local_port>	Integer type. Local port. Range: 1–65535.
<access_mode>	Integer type. Data access mode of serial port. 0 Command mode. Socket sends and receives data through AT command. 1 URC Mode. Socket sends data through AT command and receives data through the incoming data indication URC. 2 Transparent transmission mode. The serial port sends the received data through socket directly. The socket pushes the received data to the serial port directly.
<err>	Integer type. Result code. See ***Chapter 5***.

2.4.4. AT+QSSLSEND Send data Through SSL Client

AT+QSSLSEND Send data Through SSL Client	
Test Command **AT+QSSLSEND=?**	Response **+QSSLSEND: (**range of supported **<socketID>**s**),(**range of supported **<send_len>**s**),"hex_data"** **OK**
Write Command **AT+QSSLSEND=<socketID>,<send_len>,<hex_data>**	Response **+QSSLSEND: <actual_send_len>** **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	/

Parameter

<socketID>	Integer type. Socket ID. Range: 0–11.
<send_len>	Integer type. Length of the data to be sent. Unit: byte.
<hex_data>	Hex string type. The data to be sent.
<actual_send_len>	Integer type. Actual length of data written to the socket. Unit: byte.

2.4.5. AT+QSSLRECV Read Received data from SSL Client

AT+QSSLRECV Read Received data from SSL Client	
Test Command **AT+QSSLRECV=?**	Response **+QSSLRECV: (**range of supported **<socketID>**s**),(**range of supported **<read_len>**s**)** **OK**
Write Command **AT+QSSLRECV=<socketID>,<read_len>**	Response **+QSSLRECV: <actual_read_len><CR><LF><data>** **OK** Or **ERROR**

Maximum Response Time	300 ms
Characteristics	/

Parameter

<socketID>	Integer type. Socket ID. Range: 0–11.
<read_len>	Integer type. Length of data to be read. Unit type.
<actual_read_len>	Integer type. Length of the actual read data. Unit: byte.
<data>	String type. Actual read data.

2.4.6. AT+QSSLSTATE Query SSL Client State

AT+QSSLSTATE Query SSL Client State	
Write Command **AT+QSSLSTATE=<socketID>**	Response **+QSSLSTATE: <socketID>,"SSL CLIENT",<remote_addr>,<remote_port>,<local_port>,<state>** **OK**
Read Command **AT+QSSLSTATE?**	Response Return the state of all existing connections: **[+QSSLSTATE: <socketID>,"SSL CLIENT",<remote_addr>,<remote_port>,<local_port>,<state>]** **[…]** **OK**
Execute Command **AT+QSSLSTATE**	Response Return the state of all existing connections: **[+QSSLSTATE: <socketID>,"SSL CLIENT",<remote_addr>,<remote_port>,<local_port>,<state>]** **[…]** **OK**
Maximum Response Time	300 ms
Characteristics	/

Parameter

<socketID>	Integer type. Socket ID. Range: 0–11.

<remote_addr>	String type. IP address of the remote server.
<remote_port>	Integer type. Port of the remote server.
<local_port>	Integer type. Local port.
<state>	Integer type. Socket service state 0 "Initial": client connection has not been established 1 "Opening": client is connecting or server is trying to listen 2 "Connected": client connection has been established 3 "Listening". server is listening. 4 "Closing": client connection is closing

2.4.7. AT+QSSLCLOSE Close SSL Client

AT+QSSLCLOSE Close SSL Client	
Test Command **AT+QICLOSE=?**	Response **+QSSLCLOSE: (**range of supported **<socketID>**s**),(**range of supported **<close_timeout>**s**)** **OK**
Write Command **AT+QSSLCLOSE=<socketID>[,<close_timeout>]**	Response **OK** **+QSSLURC: "closed",<socketID>** Or **ERROR**
Maximum Response Time	/
Characteristics	/

Parameter

<socketID>	Integer type. Socket ID. Range: 0–11.
<close_timeout>	Integer type. The timeout for closing SSL client. Range: 0–60. Default value: 10. Unit: second.

2.5. Description of MQTT Related AT Commands

2.5.1. AT+QMTCFG Configure Optional Parameters of MQTT Client

This command configures the optional parameters of MQTT client.

AT+QMTCFG Configure Optional Parameters of MQTT Client	
Test Command **AT+QMTCFG=?**	Response **+QMTCFG: "version",(**range of supported **<clientID>**s**),(**list of supported **<vsn>**s**)** **+QMTCFG: "SSL",(**range of supported **<clientID>**s**),(**list of supported **<SSL_enable>**s**),(**range of supported **<SSL_ctxID>**s**)** **+QMTCFG: "keepalive",(**range of supported **<clientID>**s**),(**range of supported **<kalive_tm>**s**)** **+QMTCFG: "session",(**range of supported **<clientID>**s**),(**list of supported **<clean_session>**s**)** **+QMTCFG: "timeout",(**range of supported **<clientID>**s**),(**range of supported **<delivery_tm>**s**),(**range of supported **<delivery_cnt>**s**),(**list of supported **<timeout_repor**t**>**s**)** **+QMTCFG: "will",(**range of supported **<clientID>**s**),(**list of supported **<will_flag>**s**),(**range of supported **<will_QoS>**s**),(**list of supported **<will_retain>**s**),"willtopic","willmessage"** **+QMTCFG: "recv/mode",(**range of supported **<clientID>**s**),(**list of supported **<recvmode>**s**)** **OK**
Write Command Query/Configure the MQTT protocol version. **AT+QMTCFG="version",<clientID>[,<vsn>]**	Response If the optional parameter is omitted, query the current configuration: **+QMTCFG: "version",<clientID>,<vsn>** **OK** If the optional parameter is specified, configure the MQTT protocol version: **OK** Or **ERROR**
Write Command Query/Configure SSL enabling status **AT+QMTCFG="SSL",<clientID>[,<SS**	Response If the optional parameters are omitted, query the current

L_enable>[,<SSL_ctxID>]]	configuration: **+QMTCFG: "SSL",<clientID>,<SSL_enable>[,<SSL_ctxID>]** **OK** If any of the optional parameter is specified, enable or disable SSL connection: **OK** Or **ERROR**
Write Command Query/Configure the keep-alive time of MQTT protocol. **AT+QMTCFG="keepalive",<clientID>[,<kalive_tm>]**	Response If the optional parameter is omitted, query the current configuration: **+QMTCFG: "keepalive",<clientID>,<kalive_tm>** **OK** If the optional parameter is specified, configure the keep-alive time of MQTT protocol: **OK** Or **ERROR**
Write Command Query/configure the session type of MQTT protocol. **AT+QMTCFG="session",<clientID>[,<clean_session>]**	Response If the optional parameter is omitted, query the current configuration: **+QMTCFG: "session",<clientID>,<clean_session>** **OK** If the optional parameter is specified, configure the session type of MQTT protocol: **OK** Or **ERROR**
Write Command Query/Configure the timeout for awaiting a response. **AT+QMTCFG="timeout",<clientID>[,<delivery_tm>,<delivery_cnt>,<timeout_report>]**	Response If the optional parameters are omitted, query the current configuration: **+QMTCFG: "timeout",<clientID>,<delivery_tm>,<delivery_cnt>,<timeout_report>** **OK** If the optional parameters are specified, configure the timeout

	for awaiting a response. **OK** Or **ERROR**
Write Command Query/Configure Will information of MQTT protocol. **AT+QMTCFG="will",<clientID>[,<will_flag>,<will_QoS>,<will_retain>,<will_topic>,<will_message>]**	Response If the optional parameters are omitted, query the current configuration: **+QMTCFG: "will",<clientID>,<will_flag>,<willQoS>,<will_retain>,<will_topic>,<will_message>** **OK** If the optional parameters are specified, configure Will information of MQTT protocol. **OK** Or **ERROR**
Write Command Query/Configure how to read messages for MQTT client. **AT+QMTCFG="recv/mode",<clientID>[,<recvmode>]**	Response If the optional parameter is omitted, query the current configuration: **+QMTCFG: "recv/mode",<clientID>,<recvmode>** **OK** If the optional parameter is specified, configure how to read messages for MQTT client: **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	This command takes effect immediately. The configuration will not be saved.

Parameter

<clientID>	Integer type. MQTT client number. Range: 0–5.
<vsn>	Integer type. MQTT protocol version. 3 MQTT protocol v3 4 MQTT protocol v4
<SSL_enable>	Integer type. Enable or disable SSL. 0 Disable. Use Raw TCP connection for MQTT 1 Enable. Use SSL TCP secure connection for MQTT
<SSL_ctxID>	Integer type. SSL context ID. Range: 0–5.

<kalive_tm>	Integer type. Keep-alive time. The maximum idle time allowed for no data interaction between the MQTT client and server. Between the MQTT client and the server, the PingReq and PingResp messages are used to keep alive. Range: 1–3600. Default value: 120. Unit: second.
<clean_session>	Integer type. The value of the field corresponding to the session type in the MQTT CONNECT messages. 0 The connection is considered persistent. After the client disconnects, any subscribed topics and information with QoS set to 1 or 2 are saved until the client reconnects to the server. 1 After the client disconnects, all subscribed topics will be removed.
<delivery_tm>	Integer type. The maximum time that MQTT client waits for a response from the server after sending an MQTT message. Range: 1–60. Default value: 5. Unit: second.
<delivery_cnt>	Integer type. The maximum retransmission counts of MQTT message. Range: 1–10. Default value: 3.
<timeout_report>	Integer type. Whether to report a URC when MQTT message is retransmitted. 0 Not report 1 Report (See **AT+QMTSUB**, **AT+QMTPUB** and **AT+QMTUNS**.)
<will_flag>	Integer type. The value of the field corresponding to the **<will_flag>** in the MQTT CONNECT messages. 0 Ignore the configurations of **<will_QoS>**,**<will_retain>**,**<will_topic>** and **<will_message>**. 1 Send the configurations of **<will_QoS>**,**<will_retain>**,**<will_topic>** and **<will_message>** to server through MQTT CONNECT message.
<will_QoS>	Integer type. Quality of service corresponding to **<will_message>**. 0 At most once 1 At least once 2 Exactly once
<will_retain>	Integer type. Whether the server permanently saves the published **<will_message>** after the MQTT client is disconnected unexpectedly. 0 Not save 1 Save
<will_topic>	String type. Will topic name.
<will_message>	String type. Message published to the Will topic after the client is disconnected unexpectedly.
<recvmode>	Integer type. How to receive messages published by the server. 0 Direct push mode. 1 Buffer mode. Reading with **AT+QMTRECV**.

NOTE

Before connecting a client to MQTT server, the optional parameters need to be configured.

2.5.2. AT+QMTOPEN Open a Session Between MQTT Client and Server

This command opens a session between MQTT client and server.

AT+QIOPEN Open a Session Between MQTT Client and Server	
Test Command **AT+QMTOPEN=?**	Response **+QMTOPEN: (**range of supported **<clientID>**s**),"hostname",(**range of supported **<port>**s**)** **OK**
Write Command **AT+QMTOPEN=<clientID>,<hostname>,<port>**	Response **OK** **+QMTOPEN: <clientID>,<result>** Or **ERROR**
Read Command **AT+QMTOPEN?**	Response Returns the server information of all existing connections: **[+QMTOPEN: <clientID>,<hostname>,<port>]** **[…]** **OK**
Maximum Response Time	300 ms
Characteristics	/

Parameter

<clientID>	Integer type. MQTT client number. Range: 0–5.
<hostname>	String type. MQTT server address.
<port>	Integer type. MQTT server port.
<result>	Integer type. Results of the command execution. -1 Failed to connect the socket 0 MQTT session is opened successfully 1 Wrong parameter 2 **<clientID>** is occupied 3 Failed to activate PDP 4 Failed to parse domain name 5 Socket connection is closed abnormally.

2.5.3. AT+QMTCLOSE Close a Session Between MQTT Client and Server

This command closes a session between MQTT client and server

AT+QMTOPEN Close a Session Between MQTT Client and Server	
Test Command **AT+QMTOPEN=?**	Response **+QMTOPEN: (**range of supported **<clientID>**s**)** **OK**
Write Command **AT+QMTOPEN=<clientID>**	Response **OK** **+QMTOPEN: <clientID>,<result>** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	/

Parameter

<clientID>	Integer type. MQTT client number. Range: 0–5.
<result>	Integer type. Results of the command execution. -1 The execution is failed. 0 The execution is successful.

2.5.4. AT+QMTCONN Connect a Client to MQTT Server

This command connects a client to MQTT server.

AT+QMTCONN Connect a Client to MQTT Server	
Test Command **AT+QMTCONN=?**	Response **+QMTCONN: (**range of supported **<clientID>**s**),"client_identity","username","password"** **OK**
Write Command **AT+QMTCONN=<clientID>,<client_identity>,<username>,<password>**	Response **OK** **+QMTCONN: <clientID>,<result>[,<response_code>]** Or **ERROR**

Read Command **AT+QMTCONN?**	Response Return the state of all existing connections: **[+QMTCONN: <clientID>,<state>]** **[…]** **OK**
Maximum Response Time	300 ms
Characteristic	/

Parameter

<clientID>	Integer type. MQTT client number. Range: 0–5.
<client_identity>	String type. MQTT client identity.
<username>	String type. User name of the client.
<password>	String type. Password corresponding to the user's name of the client.
<state>	Integer type. MQTT connection state. 1 MQTT is initialized 2 MQTT is connecting 3 MQTT is connected 4 MQTT is disconnecting
<result>	Integer type. Result of the command execution. 0 CONNECT message is sent successfully and CONNECTACK message is received. 1 CONNECT message is sent successfully but CONNECTACK message is not received in delivery time 2 Failed to send CONNECT message
<response_code>	Integer type. Response code in the CONNECTACK message. 0 Connection Accepted 1 Connection Rejected: Unacceptable Protocol Version 2 Connection Rejected: Identifier Rejected 3 Connection Rejected: Server Unavailable 4 Connection Rejected: Wrong User Name or Password 5 Connection Rejected: Unauthorized

2.5.5. AT+QMTDISC Disconnect a Client from MQTT Server

This command disconnects a client from MQTT server.

AT+QMTDISC Disconnect a Client from MQTT Server	
Test Command **AT+QMTDISC=?**	Response **+QMTDISC: (**range of supported **<clientID>**s**)**

	OK
Write Command **AT+QMTDISC=<clientID>**	Response **OK** **+QMTDISC: <clientID>,<result>** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	/

Parameter

<clientID>	Integer type. MQTT client number. Range: 0–5.
<result>	Integer type. Results of the command execution. -1 The execution is failed. 0 The execution is successful.

2.5.6. AT+QMTSUB Subscribe to Topics

This command subscribes to one or more topics published by MQTT server.

AT+QMTSUB Subscribe to Topics	
Test Command **AT+QMTSUB=?**	Response **+QMTSUB: (**range of supported **<clientID>**s**),(**range of supported **<msgID>**s**),list of["topic",<qos>]** **OK**
Write Command **AT+QMTSUB=<clientID>,<msgID>,<topic1>,<qos1>[,<topic2>,<qos2>[…]]**	Response **OK** **+QMTSUB: <clientID>,<msgID>,<result>[,<value>]** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	/

Parameter

<clientID>	Integer type. MQTT client number. Range: 0–5.

<msgID>	Integer type. SUBSCRIBE message identifier. Range: 0–65535.
<QoS>	Integer type. Quality of service for **<topic>**. 0 At most once 1 At least once 2 Exactly once
<topic>	String type. Topic to be subscribed to.
<result>	Integer type. Result of the command execution. 0 SUBSCRIBE message is sent successfully and SUBACK message is received. 1 SUBSCRIBE message is sent successfully but SUBACK message is not received in delivery time. Retransmission is executed. 2 Failed to send SUBSCRIBE message
<value>	Integer type. When **<result>** is 0, it is a vector of granted QoS levels of SUBACK message. When **<result>** is 1, it indicates the times of SUBSCRIBE message retransmission. When **<result>** is 2, it is not presented.

NOTE

It currently supports subscribing up to 5 topics at a time.

2.5.7. AT+QMTUNS Unsubscribe from Topics

This command unsubscribes from one or more topics. An UNSUBSCRIBE message is sent by the client to the server to unsubscribe from named topics.

AT+QMTUNS Unsubscribe from Topics	
Test Command **AT+QMTUNS=?**	Response **+QMTUNS: (**range of supported **<clientID>**s**),(**range of supported **<msgID>**s**),list of["topic"]** **OK**
Write Command **AT+QMTUNS=<clientID>,<msgID>,<topic1>[,<topic2>[…]]**	Response **OK** **+QMTUNS: <clientID>,<msgID>,<result>** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	/

Parameter

<clientID>	Integer type. MQTT client number. Range: 0–5.
<msgID>	Integer type. UNSUBSCRIBE message identifier. Range: 0–65535.
<topic>	String type. Topic to be unsubscribed from.
<result>	Integer type. Result of the command execution. 0 UNSUBSCRIBE message is sent successfully and UNSUBACK message is received 1 UNSUBSCRIBE message is sent successfully but UNSUBACK message is not received in delivery time. Retransmission is executed. 2 Failed to send UNSUBSCRIBE message

NOTE

It currently supports unsubscribing up to 5 topics at a time.

2.5.8. AT+QMTPUB Publish Message Through MQTT Server

This command publishes messages through MQTT server.

AT+QMTPUB Publish Messages Through MQTT Server	
Test Command **AT+QMTPUB=?**	Response **+QMTPUBEX: (**range of supported **<clientID>**s**),** **(**range of supported **<msgID>**s**),(**range of supported **<QoS>**s**),(**list of supported **<retain>**s**),"topic",(**range of supported **<payload_length>**s**),"payload"** **OK**
Write Command **AT+QMTPUB=<clientID>,<msgID>,<QoS>,<retain>,<topic>,<payload_length>,<payload>**	Response **OK** **+QMTPUB: <clientID>,<result>[,<value>]** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	/

Parameter

<clientID>	Integer type. MQTT client number. Range: 0–5.
<msgID>	Integer type. PUBLISH message identifier. Range: 0–65535.

	When QoS is 0, **<msgID>** must be equal to 0. When QoS is greater than 0, **<msgID>** must be greater than 0.
<QoS>	Integer type. Quality of service for publishing the messages. 0 At most once 1 At least once 2 Exactly once
<retain>	Integer type. After the MQTT client is unexpectedly disconnected, whether the published message is to be saved in the server forever or not. 0 Eliminate 1 Save permanently
<topic>	String type. Topic.
<payload_len>	Integer type. Length of message to be published. Range: 1–1500. Unit: byte.
<payload>	Hexadecimal string type. Message to be published.
<result>	Integer type. Result of the command execution. 0 PUBLISH message is sent successfully and ACK message is received. 1 PUBLISH message is sent successfully but ACK message is not received in delivery time. Retransmission is executed. 2 Failed to send PUBLISH message
<value>	Integer type. When **<result>** is 1, it indicates the times of PUBLISH message retransmission. When **<result>** is 0 or 2, it is not presented.

2.5.9. AT+QMTRECV Read Messages Published by MQTT Server

This command reads messages published by MQTT Server.

AT+QMTRECV Read Messages Published by MQTT Server	
Test Command **AT+QMTRECV=?**	Response **OK**
Read Command **AT+QMTRECV?**	Response **[+QMTRECV: <clientID>,[<store_status>]]** **[…]** **OK**
Write Command **AT+QMTRECV=<clientID>[,<storeID>]**	Response If the optional parameter is omitted, read all buffer messages of the specified client: **[+QMTRECV: <clientID>,<msgID>,<topic>,<payload_len>],<payload>]** **[…]** **OK**

	If the optional parameter is specified, read the messages specified by **<storeID>** of the specified client: **+QMTRECV: <clientID>,<msgID>,<topic>,<payload_len>,<payload>** **OK**
Maximum Response Time	300 ms
Characteristics	/

Parameter

<clientID>	Integer type. MQTT client number. Range: 0–5.
<storeID>	Integer type. The ID of messages stored in the buffer. Range: 0–4.
<store_status>	Integer type. Indicating whether there is a message stored in the buffer. 0 No message in the buffer 1 One or more messages stored in the buffer
<msgID>	Integer type. PUBLISH message identifier. Range: 0–65535. When QoS is 0, **<msgID>** must be equal to 0. When QoS is greater than 0, **<msgID>** must be greater than 0.
<topic>	String type. The topic received from MQTT server.
<payload_len>	Integer type. Length of the received message.
<payload>	String type. Received message.

2.6. Description of HTTP(s) Related AT Commands

HTTP requests include establishing a HTTP session and interacting of HTTP body; hence the maximum response time is determined by the time of establishing a HTTP session and interacting of HTTP body. The maximum timeout of the TCP three-way handshake supported by FC41D module is 120 s. The maximum timeout of the SSL handshake is determined by **<SSL_negotiate_time>** (Default value: 300 s). After sending the HTTP request, the response time is **<wait_response_time>** (Default value: 60 s).

2.6.1. AT+QHTTPCFG Configure/Query Parameters for HTTP(s) Client

AT+QHTTPCFG Configure/Query Parameters for HTTP(s) Client	
Test Command **AT+QHTTPCFG=?**	Response **+QHTTPCFG: "url",<url_string>** **+QHTTPCFG: "header",<hname>,<hvalue>** **+QHTTPCFG: "auth",<username>,<password>**

	+QHTTPCFG: "response/output",(list of supported **<output_mode>**s**)** **+QHTTPCFG: "response/header",(**list of supported **<save_header>**s**)** **+QHTTPCFG: "sslctxid",(**range of supported **<SSL_ctxID>**s**)** **OK**
Write Command Configure/Query the URL to be accessed **AT+QHTTPCFG="url"[,<url_string>]**	Response If the optional parameter is omitted, query the current URL: **+QHTTPCFG: "url",<url_string>** **OK** If the optional parameter is specified, configure the URL to be accessed: **OK** Or **ERROR**
Write Command Configure/Delete/Query the customized Header **AT+QHTTPCFG="header"[,<hname>[,<hvalue>]]**	Response If the optional parameters are omitted, query the current configuration: **[+QHTTPCFG: "header",<hname>,<hvalue>]** **[…]** **OK** If **<hname>** is specified and **<hvalue>** is omitted, delete the corresponding Header: **OK** Or **ERROR** If any of the optional parameters are specified, configure the customized Header: **OK** Or **ERROR**
Write Command Configure/Query the Basic Auth parameters **AT+QHTTPCFG="auth"[,<username>,<password>]**	Response If the optional parameters are omitted, query the current configuration: **+QHTTPCFG: "auth",<username>,<password>** **OK**

	If any of the optional parameters are specified, configure the Basic Auth parameters: **OK** Or **ERROR**
Write Command Configure/Query the output mode of response data. **AT+QHTTPCFG="response/output"[,<output_mode>]**	Response If the optional parameter is omitted, query the current configuration: **+QHTTPCFG: "response/output",<output_mode>** **OK** If the optional parameter is specified, configure the output mode of response data: **OK** Or **ERROR**
Write Command Configure/Query whether to save response header. **AT+QHTTPCFG="response/header"[,<save_header>]**	Response If the optional parameter is omitted, query the current configuration: **+QHTTPCFG: "response/header",<save_header>** **OK** If the optional parameter is specified, configure whether to save response header: **OK** Or **ERROR**
Write Command Configure/Query SSL context ID for HTTP(s) session. **AT+QHTTPCFG="sslctxid"[,<SSL_ctxID>]**	Response If the optional parameter is omitted, query the current configuration: **+QHTTPCFG: "sslctxid",<SSL_ctxID>** **OK** If the optional parameter is specified, configure the SSL context ID for HTTP(s) session: **OK** Or **ERROR**
Maximum Response Time	300 ms

Characteristics	This command takes effect immediately. The configuration will be saved automatically.

Parameter

<url_string>	String type. The URL to be accessed.
<hname>	String type. HTTP Header name.
<hvale>	String type. HTTP Header value.
<username>	String type. The user's name of HTTP Basic Auth.
<password>	String type. The user's password of HTTP Basic Auth.
<save_header>	Integer type. Whether to save the response header. 0 Not save 1 Save
<output_mode>	Integer type. The output mode of response data. 0 Read the response data with **AT+QHTTPREAD** 1 Push the response data directly with URC (See ***Chapter 3.6.5***.)
<SSL_ctxID>	Integer type. SSL context ID. Range: 0–5.

2.6.2. AT+QHTTPGET Send GET Request to HTTP(s) Server

This command sends GET request to HTTP(s) server. If the GET request is sent successfully, the result of GET request is reported to MCU with **+QHTTPGET: <result>[,<status_code>[,<content_length>]]**.

AT+QHTTPCFG Send GET Request to HTTP(s) Server	
Test Command **AT+QHTTPGET=?**	Response **+QHTTPGET: (**range of supported **<wait_response_time>**s**)** **OK**
Write/Execution Command **AT+QHTTPGET[=<wait_response_time>]**	Response **OK** **+QHTTPGET: <result>[,<status_code>[,<content_length>]]** Or **ERROR**
Maximum Response Time	/
Characteristics	/

Parameter

<wait_response_time>	Integer type. The time of awaiting response from server after the GET request

	is sent. Range: 60–65535. Default value: 60. Unit: second.
<result>	Integer type. Result of sending GET request. 0 Success Other Failure
<status_code>	Integer type. HTTP status code.
<content_length>	Integer type. Length of the GET request body.

2.6.3. AT+QHTTPPOST Send POST Request to HTTP(s) Server

This command sends POST request to HTTP(s) server. If the POST request is sent successfully, the result of POST request is reported to MCU with **+QHTTPPOST: <result>[,<status_code>[,<content_length>]]**.

AT+QHTTPPOST Send POST Request to HTTP(s) Server	
Test Command **AT+QHTTPPOST=?**	Response **+QHTTPPOST: (**range of supported **<body_length>**s**),(**range of supported **<body_wait_interval>**s**),(**range of supported **<wait_response_time>**s**)[,<name>[,<file_name>[,<content_type>]]]** **OK**
Execution Command (Non form-data POST) **AT+QHTTPPOST=<body_length>[,<body_wait_interval>[,<wait_response_time>]]**	Response **CONNECT** //Input body. When the length of inputted data reaches **<body_length>**, the module exits data mode. **OK** **+QHTTPPOST: <result>[,<status_code>[,<content_length>]]** Or **ERROR**
Execution Command (Form-data POST) **AT+QHTTPPOST=<body_length>,<body_wait_interval>,<wait_response_time>,<name>[,<file_name>[,<content_type>]]**	Response **CONNECT** //Input body. When the length of inputted data reaches **<body_length>**, the module exits data mode. **OK** **+QHTTPPOST: <result>[,<status_code>[,<content_length>]]** Or **ERROR**
Maximum Response Time	/

Characteristics	/

Parameter

<body_length>	Integer type. Length of POST data. Range: 1–102400. Unit: byte.
<body_wait_interval>	Integer type. Maximum waiting time for inputting body in data mode. Range: 1–65535. Unit: second.
<wait_response_time>	Integer type. The time of awaiting response from server after the POST request is sent. Range: 60–65535. Unit: second.
<name>	String type. Form-data name.
<file_name>	String type. Name of the file stored in the HTTP(s) server after the data is uploaded.
<content_type>	String type. The file content types.
<result>	Integer type. Result of sending POST request. 0 Success Other Failure
<status_code>	Integer type. HTTP status mode.
<content_length>	Integer type. Length of the POST request body.

2.6.4. AT+QHTTPPUT Send PUT Request to HTTP(s) Server

This command sends PUT request to HTTP(s) server. If the PUT request is sent successfully, the result of PUT request is reported to MCU with **+QHTTPPUT: <result>[,<status_code>[,<content_length>]]**.

AT+QHTTPPUT Send PUT Request to HTTP(s) Server	
Test Command **AT+QHTTPPUT=?**	Response **+QHTTPPUT: (**range of supported **<body_length>**s**),(**range of supported **<body_wait_interval>**s**),(**range of supported **<wait_response_time>**s**)** **OK**
Execution Command **AT+QHTTPPUT=<body_length>[,<body_wait_interval>[,<wait_response_time>]]**	Response **CONNECT** //Input body. When the length of inputted data reaches **<body_length>**, the module exits data mode. **OK** **+QHTTPPUT: <result>[,<status_code>[,<content_length>]]** Or **ERROR**

Maximum Response Time	/
Characteristics	/

Parameter

<body_length>	Integer type. Length of PUT data. Range: 1–102400. Unit: byte.
<body_wait_interval>	Integer type. Maximum waiting time for inputting body in data mode. Range: 1–65535. Unit: second.
<wait_response_time>	Integer type. The time of awaiting response from server after the PUT request is sent. Range: 60–65535. Unit: second.
<result>	Integer type. Result of sending PUT request. 0 Success Other Failure
<status_code>	Integer type. HTTP status mode.
<content_length>	Integer type. Length of the PUT request body.

2.6.5. AT+QHTTPREAD Read Response Data of HTTP(s) Request

This command reads the response data of HTTP(s) request. If the HTTP request is responded successfully and the output mode of response data is configured with **AT+QHTTPCFG="response/output",0**, you can read the response via this command.

AT+QHTTPREAD Read Response Data of HTTP(s) Request	
Test Command **AT+QHTTPREAD=?**	Response **+QHTTPREAD: (**range of supported **<wait_response_interval>**s**)** **OK**
Write/Execution Command **AT+QHTTPREAD[=<wait_response_interval>]**	Response **CONNECT** //Output response data **OK** **+QHTTPREAD: <result>** Or **ERROR**
Maximum Response Data	/
Characteristics	/

Parameter

<wait_response_time>	Integer type. Maximum time of module serial port to awaiting response data. Range: 60–65535. Unit: second.
<result>	Integer type. Result of reading response data. 0 Success Other Failure

3 Description of URCs

The following URCs include Wi-Fi related URCs, BLE related URCs, TCP/UDP related URCs, SSL related URCs, MQTT related URCs and HTTP(s) related URCs.

3.1. Wi-Fi Related URCs

3.1.1. +QSTASTAT URC Indicating Station State Changing

+QSTASTAT URC Indicating Station State Changing	
+QSTASTAT: <event>	This URC indicates the state changes of station.

Parameter

<event>	Integer type. The event that is reported when station state changes	
	"WLAN DISCONNECTED"	Disconnected
	"WLAN_CONNECTED"	Connected
	"GOT_IP"	Got IP
	"SCAN_NO_AP"	Scanned no AP

3.2. BLE Related URCs

3.2.1. +QBLESTAT URC Indicating BLE State Changing

+QBLESTAT URC for BLE State Changing	
+QBLESTAT: <event>	This URC indicates the state changes of BLE.

Parameter

<event>	Integer type. The event that is reported when BLE state changes.	
	"NOINIT"	Uninitialized event

	"INIT"	Initial event
	"ADVERTISING"	Advertising event
	"NOADVERTISING"	NO advertising event
	"CONNECTED"	Connected event
	"DISCONNECTED"	Disconnected event

3.2.2. +QBLEMTU URC Indicating BLE MTU Changing

+QBLEMTU URC Indicating BLE MTU Changing	
+QBLEMTU: <value>	This URC indicates the MTU changes of BLE (**AT+QBLECFGMTU**).

Parameter

<value>	Integer type. Maximum transmission unit value. Range: 23–512. Unit: byte.

3.3. TCP/UDP Related URCs

3.3.1. +QIOPEN URC Indicating Opening Result

After opening socket service with **AT+QIOPEN**, the URC is reported to indicate the opening result of socket service.

+QIOPEN URC Indicating Opening Result	
+QBLESTAT: <event>	The URC indicates the result of TCP/UDP socket service is opening.

Parameter

<socketID>	Integer type. Socket ID. Range: 0–11.
<err>	Integer type. Result code. See ***Chapter 5.***

3.3.2. +QIURC: "recv" URC Indicating Incoming Data

The URC is reported when TCP/UDP socket service receives data.

+QIURC: "recv" URC Indicating Data Receiving	
+QIURC: "recv",<socketID>	The URC indicates the incoming data in command mode.

+QIURC: "recv",<socketID>,<data_len><CR><LF><data>	The URC indicates the incoming data in URC mode when the **<service_type>** is not "UDP SERVICE".
+QIURC: "recv",<socketID>,<data_len>,<remote_addr>,<remote_port><CR><LF><data>	The URC indicates the incoming data in URC mode when the **<service_type>** is "UDP SERVICE".

Parameter

<socketID>	Integer type. Socket ID. Range: 0–11.
<data_len>	Integer type. Data length. Range: 1–1500.
<remote_addr>	String type. Source address of the data.
<remote_port>	Integer type. Source port of the data.

3.3.3. +QIURC: "accept" URC Indicating Incoming Connection

If the acception mode of incoming connection is set to manual with **AT+QICFG="accept/mode"**, the URC is reported as an incoming connection is received.

+QIURC: "accept" URC Indicating Incoming Connection	
+QIURC: "accept",<socketID>	The URC indicates that an incoming connection is received If the acceptance mode of incoming connection is set to manual.

Parameter

<socketID>	Integer type. Socket ID. Range: 0–11.

3.3.4. +QIURC: "closed" URC Indicating Connection Closed

When TCP/UDP socket service is closed, the URC will be reported, and the state of socket service will be "closing".

+QIURC: "closed" URC Indicating Connection Closed	
+QIURC: "closed",<socketID>	The URC indicates that TCP/UDP socket service is closed.

Parameter

<socketID>	Integer type. Socket ID. Range: 0–11.

3.3.5. +QIURC: "incoming" URC Indicating Incoming Connection

If the acceptance mode of incoming connection is set to automatic with **AT+QICFG="accept/mode"**, the URC is reported as an incoming connection is received.

+QIURC: "incoming" URC Indicating Incoming Connection	
+QIURC: "incoming",<incomg_socketID>,<listener_socketID><remote_addr>,<remote_port>	The URC indicates that an incoming connection is received If the acceptance mode of incoming connection is set to automatic

Parameter

<incoming_socketID>	Integer type. Socket ID of TCP server. Range: 0–11.
<listener_socketID>	Integer type. Socket ID of TCP server. Range: 0–11.
<remote_addr>	String type. Source address of the incoming connection.
<remote_port>	Integer type. Source port of the incoming connection.

3.3.6. +QIURC: "incoming full" URC Indicating Incoming Connection Full

If the incoming connection reaches the limit, or no socket system resources can be allocated, then the module will report the URC as **+QIURC: "incoming full"** for the new incoming connection request.

+QIURC: "incoming full" URC Indicating Incoming Connection Full	
+QIURC: "incoming full"	The URC indicates that the incoming connection is full.

3.3.7. NO CARRIER URC Indicating Abnormal Disconnection in Transparent Transmission Mode

NO CARRIER URC Indicating Abnormal Disconnection in Transparent Transmission Mode	
NO CARRIER	The URC indicates that there is an abnormal disconnection in transparent transmission mode.

3.4. SSL Related URCs

3.4.1. +QSSLOPEN URC Indicating SSL Client Opening Result

After opening an SSL client with **AT+QSSLOPEN**, the URC is reported to indicate the opening result of the SSL client.

+QSSLOPEN URC Indicating SSL Client Opening Result	
+QSSLOPEN: <socketID>,<err>	The URC indicates the opening result of the SSL client.

Parameter

<socketID>	Integer type. Socket ID. Range: 0–11.
<err>	Integer type. Result code. See ***Chapter 5.***

3.4.2. +QSSLURC: "recv" URC Indicating Incoming Data

The URC is reported when the SSL client receives data.

+QSSLURC: "recv" URC Indicating Incoming Data	
+QSSLURC: "recv",<socketID>	The URC indicates the incoming data in command mode.
+QSSLURC: "recv",<socketID>,<data_len><CR><LF><data>	The URC indicates the incoming data in URC mode.

Parameter

<socketID>	Integer type. Socket ID. Range: 0–11.
<data_len>	Integer type. Length of the received data. Range: 1–1500. Unit: byte.
<data>	String type. Received data.

3.4.3. +QSSLURC: "closed" URC Indicating SSL Client Closed

When the SSL client is closed, the URC will be reported, and the state of SSL client will be "closing".

+QSSLURC: "closed" URC Indicating SSL Client Closed	
+QSSLURC: "closed",<socketID>	The URC indicates that the SSL client is closed.

Parameter

<socketID>	Integer type. Socket ID. Range: 0–11.

3.5. MQTT Related URCs

3.5.1. +QMTREC URC Indicating Incoming Message

The URC is reported when MQTT client receives new messages.

+QMTREC URC Indicating Incoming Message	
+QMTRECV: <clientID>,<storeID>	The URC is reported when the client has received the messages in buffer mode (**<recvmode>**=1).
+QMTRECV: <clientID>,<msgID>,<topic>,<payload_len>,<payload>	The URC is reported when the client has received the messages in direct push mode (**<recvmode>**=0).

Parameter

<clientID>	Integer type. MQTT client number. Range: 0–5.
<storeID>	Integer type. The ID of messages stored in the buffer. Range: 0–4.
<msgID>	integer type. PUBLISH message identifier. Range: 0–65535.
<topic>	String type. Topic to be published by the server.
<payload_len>	Integer type. Length of the received message.
<payload>	String type. Received message.

3.5.2. +QMTSTAT URC Indicating Abnormal Disconnection from MQTT Client

The URC is reported when MQTT client is disconnected abnormally.

+QMTSTAT URC Indicating Abnormal Disconnection from MQTT Client	
+QMTSTAT: <clientID>,<stat>	The URC indicates that the MQTT client is disconnected abnormally.

Parameter

<clientID>	Integer type. MQTT client number. Range: 0–5.
<stat>	Integer type. Reason for abnormal disconnection of a MQTT session. 1 MQTT session is closed passively. 2 MQTT session is closed because the PingReq messages is timeout. 3 MQTT session is closed because the Connect messages is timeout. 4 MQTT session is closed because there is a prompt of connection failure in ConnACK.

3.6. HTTP(s) Related URCs

3.6.1. +QHTTPGET Indicating Sending Result of GET Request

After sending a GET request with **AT+QHTTPGET**, the URC is reported to indicate the sending result of the GET request.

+QHTTPGET Indicating Sending Result of GET Request	
+QHTTPGET: <result>[,<status_code>[,<content_length>]]	The URC is reported to indicate the sending result of the GET request.

Parameter

<result>	Integer type. Result code. See ***Chapter 5.***
<status_code>	Integer type. HTTP status code.
<content_length>	Integer type. Length of GET request body.

3.6.2. +QHTTPPOST Indicating Sending Result of POST Request

After sending a POST request with **AT+QHTTPPOST**, the URC is reported to indicate the sending result of the POST request.

+QHTTPPOST Indicating Sending Result of POST Request	
+QHTTPPOST: <result>[,<status_code>[,<content_length>]]	The URC is reported to indicate the sending result of the POST request.

Parameter

<result>	Integer type. Result code. See ***Chapter 5.***
<status_code>	Integer type. HTTP status code.
<content_length>	Integer type. Length of the POST request body.

3.6.3. +QHTTPPUT Indicating Sending Result of PUT Request

After sending a PUT request with **AT+QHTTPPUT**, the URC is reported to indicate the sending result of the PUT request.

+QHTTPPUT Indicating Sending Result of PUT Request	
+QHTTPPOST: <result>[,<status_code>[,<content_length>]]	The URC is reported to indicate the sending result of the PUT request.

Parameter

<result>	Integer type. Result code. See ***Chapter 5.***
<status_code>	Integer type. HTTP status code.
<content_length>	Integer type. Length of the PUT request body.

3.6.4. +QHTTPREAD Indicating Reading Result of Response Data

After reading the response data with **AT+QHTTPREAD**, the URC is reported to indicate the reading result.

+QHTTPREAD Indicating Reading Result of Response Data	
+QHTTPREAD: <result>	The URC is reported to indicate the reading result of the response data.

Parameter

<result>	Integer type. Result code. See ***Chapter 5.***

3.6.5. +QHTTPURC: "recv" URC Indicating Incoming Response Data

The URC is reported when the HTTP(s) client receives response data.

+QHTTPURC: "recv" URC Indicating Incoming Response Data	
+QHTTPURC: "recv",<length><CR><LF><data>	The URC indicates that the incoming response data.

Parameter

<length>	Integer type. Length of the received data.
<data>	String type. Response data.

4 Example

4.1. TCP/UDP Function

4.1.1. Transparent Transmission Mode

The following example shows how to open or close transparent Transmission mode for TCP/UDP client.

```
AT+QIOPEN=0,"TCP","220.180.239.201",8252,2020,2        //Open a TCP client
CONNECT
//Input data
+++                                                    //Exit transparent Transmission mode
OK
ATO                                                    //Enter transparent Transmission mode
again
CONNECT
//input data
+++                                                    //Exit transparent Transmission mode
OK
AT+QICLOSE=0                                           //Close the TCP client
OK

+QIURC: "closed",0
```

The following example shows how the TCP/UDP client in transparent Transmission mode behaves when disconnected abnormally.

```
AT+QIOPEN=0,"TCP","220.180.239.201",8252,2020,2        //Open a TCP client
CONNECT
//Input data
NO CARRIER                                             //Disconnect abnormally.
```

4.1.2. Non-transparent Transmission Mode

The following example shows how to read TCP/UDP data with AT commands.

```
AT+QIOPEN=0,"TCP","220.180.239.201",8252,2020,0          //Open a TCP client and set <access_mode> to command mode.
OK

+QIOPEN: 0,0
AT+QISEND=0,10,"30313233343536373839"
+QISEND: 10

OK

+QIURC: "recv",0                                         //Receive new data.
AT+QIRD=0,10                                             //Read new data.
+QIRD:10

0123456789

OK
AT+QICLOSE=0                                             //Close the TCP client.
OK

+QIURC: "closed",0
AT+QIOPEN=1,"UDP SERVICE","220.180.239.201",8252,2020,0  //Open a UDP client and set <access_mode> to command mode.
+QIOPEN: 1,0
AT+QISEND=1,10,"30313233343536373839","220.180.239.201",8252
+QISEND: 10

OK

+QIURC: "recv",1                                         //Receive new data.
AT+QIRD=1,10                                             //Read new data.
+QIRD:10,"220.180.239.201",8252

0123456789

OK
AT+QICLOSE=1                                             //Close the UDP service.
OK
```

```
+QIURC: "closed",1
```

The following example shows how to report new data with URC.

```
AT+QIOPEN=0,"TCP","220.180.239.201",8252,2020,0          //Open a TCP client and set
                                                           <access_mode> to URC mode.
OK

+QIOPEN: 0,0
AT+QISEND=0,10,"30313233343536373839"
+QISEND: 10

OK

+QIURC: "recv",0,10                                      //Receive new data.

0123456789
AT+QICLOSE=0                                             //Close the TCP client
OK

+QIURC: "closed",0
AT+QIOPEN=1,"UDP SERVICE","220.180.239.201",8252,2020,0 //Open a UDP client and set
                                                           <access_mode> to URC mode..
OK

+QIOPEN: 1,0
AT+QISEND=1,10,"30313233343536373839","220.180.239.201",8252
+QISEND: 10

OK

+QIURC: "recv",1,10, "220.180.239.201",8252               //Receive new data.

0123456789
AT+QICLOSE=1                                             //Close the UDP service.
OK

+QIURC: "closed",1
```

4.2. MQTT Function

```
AT+QMTCFG="version",1,4                  //Configure MQTT protocol version to v4.
OK
AT+QMTOPEN=1,"220.180.239.212",8306      //Open a MQTT session for MQTT server.
OK

+QMTOPEN: 1,0
AT+QMTCONN=1,"client1","test","test"      //Connect a client to MQTT server.
OK

+QMTCONN: 1,0,0
AT+QMTSUB=1,1,"quectel",1                //Subscribe to topic named "quectel".
OK

+QMTSUB: 1,1,0,1
//Direct push mode
AT+QMTPUB=1,1,1,0,"quectel",3,"333435"   //Publish a message to the topic named "quectel".
OK

+QMTRECV: 1,1,"quectel",3,"345"           //Receive a message from the topic maned "quectel".
//Buffer mode
AT+QMTCFG="recv/mode",0,1
OK
AT+QMTPUB=1,1,1,0,"quectel",2,"3132"     //Publish a message to the topic named "quectel".
OK

+QMTPUB: 1,1,0

+QMTRECV: 0,0
AT+QMTRECV=0,0
+QMTRECV: 1,1,"quectel",2,"12"

OK
AT+QMTUNS=1,1,"quectel"                  //Unsubscribe from topic named "quectel".
OK

+QMTUNS: 1,1,0
AT+QMTDISC=1                             //Disconnect the lient from MQTT server.
OK

+QMTDISC: 1,0
```

4.3. HTTP(s) Function

```
//HTTP(s) GET Example
//Step 1: configure URL
AT+QHTTPCFG="url","http://www.baidu.com"
OK
//Step 2: send GET request
AT+QHTTPGET=120                    //Open Baidu web.
OK

+QHTTPGET: 0,200
//Step 3: read the response data
AT+QHTTPREAD=60                    //Read the response data.
CONNECT
<html>
<head>
    <script>
        location.replace(location.href.replace("https://","http://"));
    </script>
</head>
<body>
    <noscript><meta http-equiv="refresh" content="0;url=http://www.baidu.com/">
.....................
OK

+QHTTPREAD: 0
```

```
//HTTP(s) POST Example
//Step 1：  configure URL
AT+QHTTPCFG="url","http://220.180.239.212:8252/study_log/"        //The URL is only a reference.
OK
//Step 2: send POST request
AT+QHTTPPOST=1024,120,120,"file","test.txt","text/plain"          //Upload file in POST mode.
CONNECT
//Input body. When the length of inputted data reaches <body_length>, the module exits data mode.
OK

+QHTTPPOST: 0,200,1538
//Step 3: read the response data
AT+QHTTPREAD=60                                                   //Read the response data.
CONNECT
<html>
<head>
```

```
.....
OK

+QHTTPREAD: 0
```

```
//HTTP(s) PUT Example
//Step 1: configure URL
AT+QHTTPCFG="url","http://220.180.239.212:8252/uploads/test.txt"      //The URL is only a reference.
OK
//Step 2: send PUT request
AT+QHTTPPUT=1024,120,120                                              //Upload file in PUT mode.
CONNECT
//Input body. When the length of inputted data reaches <body_length>, the module exits data mode.
OK

+QHTTPPUT: 0,200,1538
//Step 3: read the response data
AT+QHTTPREAD=60                                                       //Read the response data.
CONNECT
<html>
<head>
.....
OK

+QHTTPREAD: 0
```

```
//Customized Header Example
//Customizing the parameters Header: Range: bytes=x-x, to realize the function of breakpoint download
AT+QHTTPCFG="header","Range","bytes=0-511"
OK
AT+QHTTPCFG="url","http://116.247.104.27:6023/1M.txt"    //The URC is only a reference.
OK
AT+QHTTPGET=60
OK

+QHTTPGET: 0,200,512
AT+QHTTPREAD=60                                          //Read the response data.
CONNECT
//Data
OK

+QHTTPREAD: 0
```

5 Summary of Result Codes

Table 2: TCP/UDP/SSL Result Codes

Result Code	Description
0	Success
550	Invalid parameter
551	Unknown error
552	Out of memory
553	Socket ID has been used
554	Socket ID does not exist
555	Socket allocate failed
556	Operation is not allowed
557	Operation is not supported

Table 3: HTTP Result Codes

Result Code	Description
0	Success
1	Invalid parameter
2	Unknown error
3	Out of memory
4	Socket failure
5	Operation is not supported

6	Operation is not allowed
7	No network
8	Miss SSL Cert
9	Response timeout
10	Body wait timeout

6 Appendix A Terms and Abbreviations

Table 4: Terms and Abbreviations

Abbreviation	Description
ACK	Acknowledgement
AP	Access Point
BLE	Bluetooth Low Energy
BSSID	Basic Service Set Identifier
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name Server
GATT	Generic Attribute Profile
MAC	Medium Access Control
MQTT	Message Queuing Telemetry Transport
MTU	Maximum Transmission Unit
ID	Mostly refers to Identifier in terms of software
IP	Internet Protocol
OTA	Over-the-Air Technology
PSK	Pre-Shared Key
QoS	Quality of Service
SNI	Server Name Indication
SSID	Service Set Identifier
SSL	Service Set Identifier
TA	Terminal Adapter

TCP	Transmission Control Protocol
TLS	Transport Layer Security
TX	Transmit
UDP	User Datagram Protocol
URC	Unsolicited Result Code
UUID	Universally Unique Identifier
HTTP	Hyper Text Transfer Protocol