LTE Standard(A) series

MQTT Application Guide

LTE Standard Module Series

Version: 1.3

Date: 2022-08-05

Status: Controlled file

Shanghai Quectel Communications Technology Co., Ltd. (hereinafter referred to as "Quectel") always aims to provide customers with the most timely and comprehensive services. If you need any help, please feel free to contact our Shanghai headquarters, the contact information is as follows:

Shanghai Quectel Communications Technology Co.,
Ltd. Building 5, Phase 3 (B Zone), Science and Technology Oasis, No. 1016 Tianlin Road, Minhang District, Shanghai
Postcode: 200233 Tel: +86 21 5108 6236 Email: info@quectel.com

Or contact our local office, for details, please log in: http://www.quectel.com/cn/support/sales.htm.

If you need technical support or give feedback on the problems in our technical documents, please feel free to visit the website: http://www.quectel.com/cn/support/technical.htm or send an email to: support@quectel.com.

foreword

Quectel provides this document content to support customers' product design. The customer shall design the product according to the specifications and parameters provided in the document. At the same time, you understand and agree that the reference design provided by Quectel is only an example. You agree to use your independent analysis, evaluation and judgment in designing your intended product. Please read this statement carefully before using any hardware, software or service guided by this document. You hereby acknowledge and agree that although Quectel has taken commercially reasonable efforts to provide the best possible experience, this document and the services it refers to are provided to you on an "as available" basis. Quectel may, at its sole discretion, add, modify or restate this document at any time without prior notice.

Use and Disclosure Restrictions

License

Agreement Unless specifically authorized by Quectel, the recipient of the hardware, software, materials and documents provided by our company must keep the received content confidential and shall not use it for any purpose other than the implementation and development of this project.

Copyright

statement Quectel products and third-party products under this agreement may contain related materials protected by Quectel or third-party materials, hardware, software and documents. Unless you have obtained prior written consent, you shall not obtain, use, or disclose the documents and information provided by our company to a third party, or copy, reprint, plagiarize, publish, display, translate, distribute, Incorporate, modify, or create derivative works thereof. Quectel or third parties have exclusive rights to the copyrighted materials and do not grant or transfer any license to patent, copyright, trademark or service mark rights. For the avoidance of doubt, no purchase of any kind shall be deemed to confer a license other than a normal non-exclusive, royalty-free product use license. Quectel has the right to pursue legal responsibility for any violation of confidentiality obligations, unauthorized use or other illegal forms of malicious use of the documents and information.

Trademarks Unless otherwise specified, nothing in this document grants the right to use any trademarks, trade names and names of Quectel or third parties, or their acronyms, or imitations thereof, in advertising, publicity or otherwise.

Third Party Rights

You understand that this documentation may refer to one or more hardware, software and documentation belonging to third parties ("Third Party Materials"). Your use of such third-party materials shall be subject to all limitations and obligations of this document.

Quectel does not make any express or implied warranties or representations with respect to third-party materials, including but not limited to any implied or statutory merchantability or fitness for a particular purpose, quiet benefit rights, system integration, information accuracy, and licensed technology or the licensee's warranty that it does not infringe any third party's intellectual property rights in relation to the use of the licensed technology. Nothing in this Agreement shall constitute Quectel's development, enhancement, modification, distribution, marketing, sale, offer for sale or otherwise of any Quectel product or any other hardware, software, equipment, tools, information or products. Representation or warranty to maintain production. Further, Quectel disclaims any and all warranties arising out of course of dealing, use or trade.

Privacy statement

In order to realize the functions of Quectel products, specific device data will be uploaded to Quectel or third-party servers (including operators, chip suppliers or servers you designate). Quectel strictly abides by relevant laws and regulations, and only retains, uses, discloses or otherwise processes relevant data for the purpose of realizing product functions or under the circumstances permitted by applicable laws. Before you interact with a third party, please understand its privacy protection and data security policies.

disclaimer

1) Quectel shall not be liable for any damages caused by failure to comply with relevant operating or design specifications.

2) Quectel shall not be liable for any inaccuracies, omissions, or use of information in this document. 3) Quectel tries its best to ensure the completeness, accuracy and timeliness of the functions under development, but does not rule out the possibility of errors or omissions in the above functions. Unless otherwise stipulated in the agreement, Quectel does not make any implied or statutory guarantees for the use of functions under development. To the maximum extent permitted by applicable laws, Quectel shall not be liable for any damages suffered as a result of using functions under development, regardless of whether such damages are foreseeable or not.

4) Quectel is not responsible for the accessibility, security, accuracy, availability, legality and completeness of information, content, advertisements, commercial offers, products, services and materials on third-party websites and third-party resources. responsibility.

document history

revision history

version	date	author	change expression
-	2020-08-12	Luffy LIU/ Larson LI	document creation
1.0	2020-08-12	Luffy LIU/ Larson LI	Controlled version
1.1	2021-04-15	Luffy LIU/ Larson LI	1. Add documentation for EC200N-CN and EC600N-CN modules. 2. Add the setting command AT+QMTCFG="edit/timeout",<client_idx> [,<edit_mode>,<edit_mode>] (Chapter 3.3.1). 1. Add documents
1.2	2022-03-02	Luffy LIU/ Larson LI	applicable to modules EC200A series, EC800N-CN and EG915N-EU. 2. Delete the EG912Y-CN module.
1.3	2022-08-05	Luffy LIU/ Larson LI	1. Add applicable modules EC200M-CN, EC600M-CN, EC800M-CN, EG915N-LA and EG912N-EN. 2. Delete the applicable module EC200T series.

Table of contents

table index

1 Introduction

Quectel LTE Standard(A) series modules support MQTT function. MQTT is an agent-based publish/subscribe mode communication protocol, which is open, simple, lightweight and easy to implement. The biggest advantage of MQTT is that it can provide real-time and reliable message services for remote device connections with very little code and limited network bandwidth.

This document mainly introduces how to use the MQTT function of the following Quectel communication modules through AT commands.

1.1. Applicable modules

Table 1: Applicable modules

Module series	module
LTE Standard(A)	EC200A series
	EC200M-CN
	EC200N-CN
	EC200S series
	EC600M-CN
	EC600N-CN
	EC600S-CN
	EC800M-CN
	EC800N-CN
	EG912N-EN
	EG912Y-EU
	EG915N series

2 MQTT data interaction

This chapter mainly introduces the data exchange mechanism of the MQTT function.

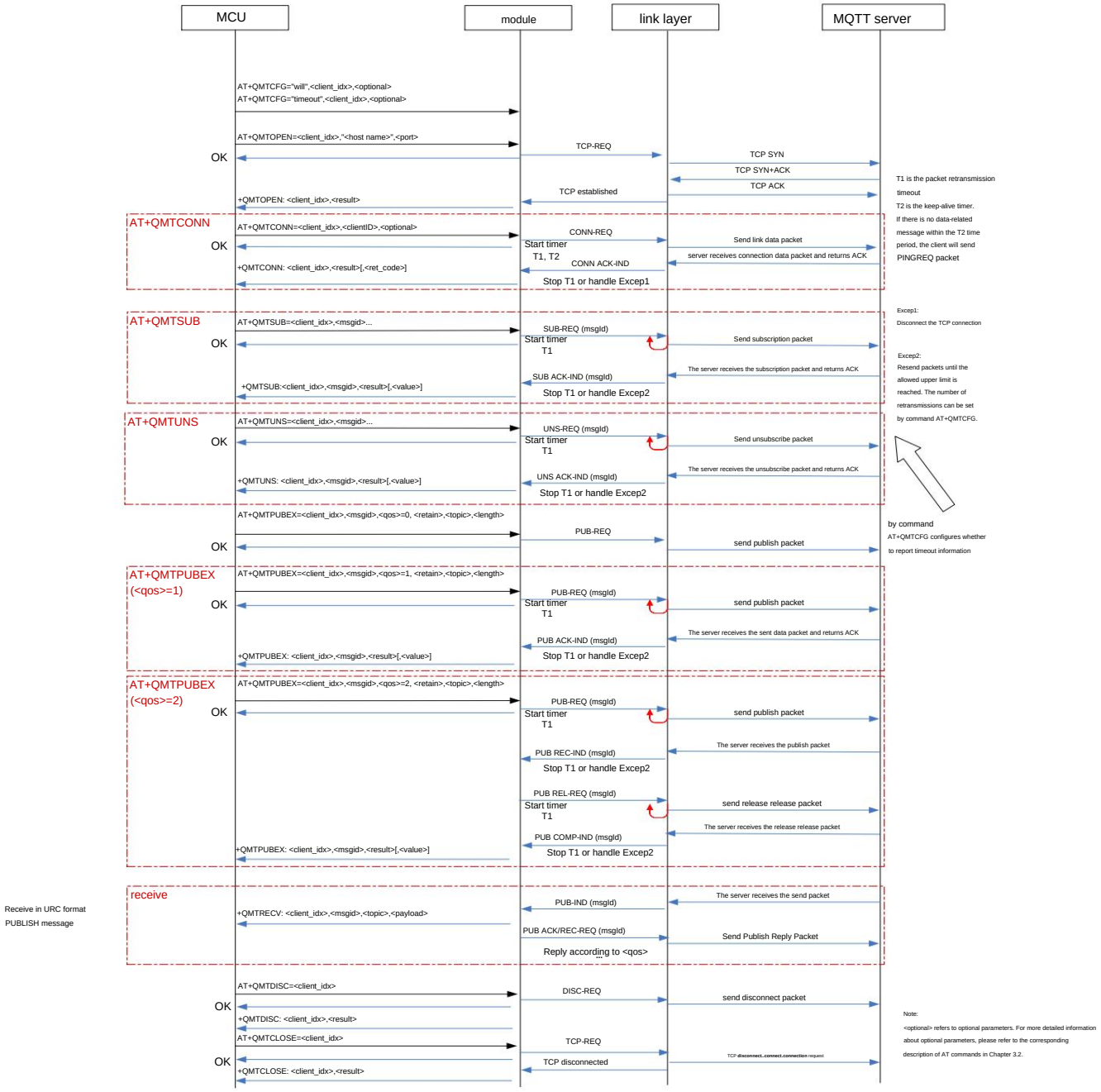

Figure **1: MQTT** data interaction mechanism

3 MQTT related AT commands

This chapter mainly introduces related AT commands for operating MQTT functions.

3.1. AT command description

3.1.1. Definition

ÿ **<CR>** carriage

ÿ **<LF>** return. line

ÿ **<…>** break. parameter name. Angle brackets are not included in the actual

ÿ **[…]** command line. Optional parameter or optional part of the TA info response. The square brackets are not included in the actual command line. Unless otherwise specified, when an optional parameter in a configuration command is omitted, its previously set value or its default value will be used by default.

ÿ Default settings for underlined parameters.

3.1.2. AT command statement

The prefix **AT** or **at** must be added at the beginning of each command line. Typing <CR> will terminate the command line. Usually, the command is followed by the form <CR><LF><response><CR><LF>'s response. In tables representing commands and responses in this document, <CR><LF> are omitted, and only commands and responses are shown.

Table **2: AT** Command Types

AT command type statement	describe
Test command **AT+<cmd>=?**	Tests for the existence of a corresponding command and returns information about the type, value, or range of its arguments.
Query command **AT+<cmd>?**	Query the current parameter value of the corresponding command.
Set command **AT+<cmd>=<p1>[,<p2>[,<p3>[...]]]** to set user-definable parameter value.	
Execute the command **AT+<cmd>**	Return specific parameter information or perform specific operations.

3.2. AT example statement

The examples in this article are only for the convenience of users to understand how to use AT commands, and do not constitute Quectel's suggestions or opinions on the terminal process design, nor does it mean that the module should be set to the state in the corresponding examples. Multiple instances of some AT commands exist without succession or continuity between the instances.

3.3. MQTT related AT command description

3.3.1. AT+QMTCFG configure MQTT optional parameters

This command is used to configure optional parameters of MQTT.

AT+QMTCFG Configure MQTT optional parameters	
Test command **AT+QMTCFG=?**	response **+QMTCFG: "version", (supported <client_idx> range), (supported <vsn> list)** **+QMTCFG: "pdpcid", (supported <client_idx>** range), **(supported <cid> range)** **+QMTCFG: "ssl", (supported <client_idx>** range), **(supported** <SSL_enable> list), (supported <SSL_ctx_idx> ranges) **+QMTCFG: "keepalive", (supported <client_idx> range), (supported <keep_alive_time> range)** **+QMTCFG: "session", (supported <client_idx> range), (supported <clean_session> list)** **+QMTCFG: "timeout", (supported <client_idx> range), (supported <pkt_timeout> range), (supported <retry_times> range), (supported <timeout_notice> list)** **+QMTCFG: "will", (supported <client_idx>** range), **(supported <will_fg>** list), **(supported <will_qos>** range), **(supported <will_retain> list), "willtopic","willmessage"** **+QMTCFG: "willex", (supported <client_idx>** range), **(supported <will_fg>** list), **(supported <will_qos>** range), **(supported <will_retain> list), "willtopic", (supported <will_len> range)** **+QMTCFG: "recv/mode", (supported <client_idx> range), (supported** <msg_recv_mode> list), (supported <msg_len_enable> list) **+QMTCFG: "aliauth", (supported <client_idx> range), "product key", "device name", "device secret"** **+QMTCFG: "qmtping", (supported <client_idx> range), (supported <qmtping_interval> range)** **+QMTCFG: "send/mode", (supported <client_idx> range), (supported <send_mode> list)**

	+QMTCFG: "onenet", (supported <client_idx> range), "product id", "access key" **+QMTCFG: "hwauth", (supported <client_idx> range), "product id", "device secret"** **+QMTCFG: "hwprodid", (supported <client_idx> range), "product id", "product secret", "nodeid"** **+QMTCFG: "dataformat", (supported <client_idx> range), (supported** <send_mode> list), (supported <recv_mode> list) **+QMTCFG: "view/mode", (supported <client_idx> range), (supported <view_mode> list)** **+QMTCFG: "edit/timeout", (supported <client_idx> range), (supported** <edit_mode> list), (supported <edit_time> range) **OK**
Set command to configure MQTT protocol version **AT+QMTCFG="version",<client_idx> [,<vsn>]**	If the optional parameter is omitted in the response, the MQTT protocol version is queried: **+QMTCFG: "version",<vsn>** **OK** If an optional parameter is specified and the MQTT connection is not created, configure the MQTT protocol version: **OK** In case of any errors: **ERROR**
Set the command to configure the PDP to be used by the MQTT client **AT+QMTCFG="pdpcid",<client_idx>[,<cid>]**	If the optional parameter is omitted in the response, the PDP used by the current MQTT client is queried: **+QMTCFG: "pdpcid",<cid>** **OK** If an optional parameter is specified and an MQTT connection is not created, configure the PDP to be used by the MQTT client: **OK** In case of any errors: **ERROR**
Set command to configure MQTT SSL mode and SSL context index **AT+QMTCFG="ssl",<client_idx>[,<SSL_enable>[,<SSL_ctx_idx>]]**	If the optional parameter is omitted in response, query the current MQTT SSL mode and SSL context index configuration: **+QMTCFG: "ssl",<SSL_enable>[,<SSL_ctx_idx>]** **OK**

	If optional parameters are specified and the MQTT connection is not created, configure the MQTT SSL mode and SSL context index: **OK** In case of any errors: **ERROR**
Set the command to configure the keep-alive time **AT+QMTCFG="keepalive",<client_idx>[,<keep_alive_time>]**	If the optional parameter is omitted in the response, the current keep-alive time is queried: **+QMTCFG: "keepalive",<keep_alive_time>** **OK** If optional parameters are specified and the MQTT connection is not created, configure the keep-alive time: **OK** In case of any errors: **ERROR**
set command configuration session type **AT+QMTCFG="session",<client_idx>[,<clean_session>]**	If the optional parameter is omitted in the response, the current session type is queried: **+QMTCFG: "session",<clean_session>** **OK** If an optional parameter is specified and an MQTT connection is not created, configure the session type: **OK** In case of any errors: **ERROR**
Set command configuration message transmission timeout **AT+QMTCFG="timeout",<client_idx>[,<pkt_timeout>,<retry_times>,<time out_notice>]**	If the optional parameter is omitted in response, query the currently set message transmission timeout: **+QMTCFG: "timeout",<pkt_timeout>,<retry_times>,<time out_notice>** **OK** If an optional parameter is specified and the MQTT connection is not created, configure the message transmission timeout: **OK** In case of any errors: **ERROR**
Set command to configure Will information **AT+QMTCFG="will",<client_idx>[,<will_fg>[,<will_qos>,<will_retain>,<will_topic>,<will_message>]]**	Response If the optional parameter is omitted, the current Will configuration information will be queried: **+QMTCFG: "will",<will_fg>[,<will_qos>,<will_retain>,<wil**

topic>,<will message>]]	**OK** If optional parameters are specified and the MQTT connection is not created, configure Will information: **OK** In case of any errors: **ERROR**
Set command to configure Will information **AT+QMTCFG="willex",<client_idx>[, <will_fg>[,<will_qos>,<will_retain>,<willtopic>,<will_len>]]**	Response If the optional parameter is omitted, the current Will configuration information will be queried: **+QMTCFG: "willex",<will_fg>[,<will_qos>,<will_retain>,<willtopic>,<will_len>]** **OK** If optional parameters are specified, configure Will information: **>** Input the Will **message; if the data length is greater than <will_len>, send the previous data whose length is equal to <will_len>.** **OK** In case of any errors: **ERROR**
The set command configures the receiving mode of server data **AT+QMTCFG="recv/ mode",<client_idx>[,<msg_recv_mode>[,<msg_len_ enable>]]**	If the optional parameter is omitted in response, the current MQTT data receiving mode is queried: **+QMTCFG: "recv/mode",<msg_recv_mode>,<msg_len_enable>** **OK** If optional parameters are specified and the MQTT connection is not created, configure the receiving mode of server data: **OK** In case of any errors: **ERROR**
Set command to configure Alibaba Cloud device information **AT+QMTCFG="aliauth",<client_idx>[,<product key>,<device name>,<device secret>]**	If the optional parameter is omitted in the response, the device information is queried: **+QMTCFG: "aliauth",<produc key>,<device name>,<dev ice secret>** **OK** If optional parameters are specified and the MQTT connection is not created, configure Alibaba Cloud device information:

	OK In case of any errors: **ERROR**
Set command to configure MQTT heartbeat interval **AT+QMTCFG="qmtping",<client_idx>[,<qmtping_interval>]**	Response If the optional parameter is omitted, query the heartbeat interval of the current MQTT: **+QMTCFG: "qmtping",<qmtping_interval>** If an optional parameter is specified and the MQTT connection is not created, configure the MQTT heartbeat interval: **OK** In case of any errors: **ERROR**
Set the command to configure the sending format of MQTT messages **AT+QMTCFG="send/mode",<client_idx>[,<send_mode>]**	Response If the optional parameter is omitted, the current configuration is queried: **+QMTCFG: "send/mode",<send_mode>** **OK** If an optional parameter is specified and the MQTT connection is not created, configure the sending mode of the MQTT message: **OK** In case of any errors: **ERROR**
Set the command to configure the device information of China Mobile OneNET platform **AT+QMTCFG="onenet",<client_idx>[,<product id>,<access key>]**	Response If the optional parameter is omitted, the current configuration is queried: **+QMTCFG: "onenet",<product id>,<access key>** **OK** If optional parameters are specified and the MQTT connection is not created, configure the device information of China Mobile OneNET platform: **OK** In case of any errors: **ERROR**
Set the command to configure the device information of the Huawei IoT platform **AT+QMTCFG="hwauth",<client_idx> [,<product id>,<device secret>]**	Response If the optional parameter is omitted, the current configuration is queried: **+QMTCFG: "hwauth",<product id>,<device secret>[,<h w_time_enable>]** **OK**

	If optional parameters are specified and the MQTT connection is not created, configure the device information of the Huawei IoT platform: **OK** In case of any errors: **ERROR**
Set the command to configure the device information of the Huawei IoT platform **AT+QMTCFG="hwprodid",<client_id x>[,<product id>,<product secret>,<nodeid>]**	Response If the optional parameter is omitted, the current configuration is queried: **+QMTCFG: "hwprodid",<product id>,<product secret>,<nodeid>[,<hw_time_enable>]** **OK** If optional parameters are specified and the MQTT connection is not created, configure the device information of the Huawei IoT platform: **OK** In case of any errors: **ERROR**
Set command to configure MQTT data format **AT+QMTCFG="dataformat",<client_idx>[,<send_mode>,<recv_mode>]**	Response If the optional parameter is omitted, the current configuration is queried: **+QMTCFG: "dataformat",<send_mode>,<recv_mode>** **OK** If optional parameters are specified and the MQTT connection is not created, configure the MQTT data format: **OK** In case of any errors: **ERROR**
Set the command to configure the echo mode of MQTT data in transparent transmission mode **AT+QMTCFG="view/mode",<client_idx>[,<view_mode>]**	Response If the optional parameter is omitted, the current configuration is queried: **+QMTCFG: "view/mode",<view_mode>** **OK** If optional parameters are specified and the MQTT connection is not created, configure the echo mode of MQTT data in transparent transmission mode: **OK** In case of any errors: **ERROR**
Set command to configure MQTT input data timeout	Response If the optional parameter is omitted, the current configuration is queried:

AT+QMTCFG="edit/timeout",<client_idx>[,<edit_mode>,<edit_time>]	**+QMTCFG: "edit/timeout",<edit_mode>[,<edit_time>]** **OK** If an optional parameter is specified and the MQTT connection is not created, configure the MQTT input data timeout: **OK** In case of any errors: **ERROR**
maximum response time	300 milliseconds
Feature Description	This command takes effect immediately; the parameter configuration is not saved.

parameter

<client_idx>	Integer. MQTT client identifier. Range: 0~5. Integer. MQTT
<vsn>	protocol version. 3 MQTT Protocol v3.1 4 MQTT Protocol v3.1.1 Integer.
<cid>	MQTT client to use PDP. Range: 1~15; Default value: 1. Integer. Configure Will Flag.
<will_fg>	0 No need to configure Will Flag 1 Need to configure Will Flag Integer. The QoS level
<will_qos>	when sending messages. 0 send at most once 1 send at least once 2 only send integer once. The Will Retain flag applies only to PUBLISH messages. 0 When the client publishes a PUBLISH message to the server and the message is successfully sent to the current subscriber, the server does not retain the message 1 When the client publishes a PUBLISH message to the server and the message is successfully sent to the current subscriber, the server retains the message character string type.
<will_retain>	Will topic name. Length range: 1~256 bytes. **<willtopic> <willmessage>** String type. When the client disconnects unexpectedly, this parameter defines the content of the message that needs to be published to the Will Topic.
	Length range: 0~256 bytes. Integer.
<will_len>	The length of the Will message. Range: 0~256. Integer.
<pkt_timeout>	Packet transmission timeout. Range: 1~60; Default value: 5; Unit: second. Integer. The number of
<retry_times>	retransmission times after the data packet transmission times out. Range: 0~10; Default value: 3.
<timeout_notice> integer. Whether to report a timeout message when transmitting data	packets. 0 do not report 1 report

<clean_session> integer. Configure the session type.

0 After the client disconnects, the server must store the client's subscription messages. 1
The server must delete any previously retained client messages. The connection status is "Clean"

<keep_alive_time> **integer .** Keepalive time. Range: 0~3600; Default value: 120; Unit: second. This parameter defines the maximum interval for receiving messages from the client. If the server does not receive messages (interaction data and keep-alive packets) from the client within 1.5 times the set time, the client sends a DISCONNECT message by default, so the server The client connection will be disconnected. 0 means do not disconnect the client connection

<SSL_enable> integer. Configure MQTT SSL mode.

0 use normal TCP connection 1
use SSL TCP secure connection

<SSL_ctx_idx> integer. SSL context index. Range: 0~5.

<msg_recv_mode> integer. Configure the MQTT message receiving mode.

0 The MQTT message received from the server is reported in the form of URC
1 The MQTT message received from the server is not reported in the form of

URC **<msg_len_enable>** integer. Configure whether to include the length of the MQTT message received from the server in the URC.

0 does not
contain 1
contains the length

Alibaba Cloud. **<product key> <device name>** string type. The product key issued by
released by Alibaba Cloud. **<device secret>** string type. The device verification
certificate issued by Alibaba Cloud/Huawei platform. **<qmtping_interval> <send_mode>**

Integer. Heartbeat interval. Range: 5~60; Default value: 5; Unit: second. Integer.
MQTT message sending format. 0 string 1 hexadecimal string type. Product ID
released by China Mobile OneNET platform/Huawei IoT platform. String type. The
product key issued by China Mobile OneNET platform/Huawei IoT platform.

<product id> **<access key> <hw_time_enable>** integer. Whether to enable timestamp verification when the device is connected to the Huawei IoT platform.

0 disable 1
enable

<product secret> string type. Product verification certificate issued by Huawei platform.

<nodeid> String type. Device identification code, which uniquely identifies a physical

<recv_mode> device. Integer. MQTT message reception format. 0 string 1 hexadecimal integer. The echo mode of MQTT data in transparent transmission mode. 0 does not echo 1 echo integer. Configure whether to exit when the MQTT input

<view_mode> data times out. 0 disable 1 enable integer. MQTT input data timeout. Range 1~120; unit: second.

<edit_mode>

<edit_time>

Remark

1. **If <will_fg>=1, then <will_qos>, <will_retain>, <willtopic> and <willmessage> must be specified; if <will_fg> is not** 1, the above parameters will be omitted.
2. <clean_session>=0 is valid only when the server supports the operation of storing session information . 3. If the MQTT connection is configured in SSL **mode, <SSL_ctx_idx> must be specified. In addition, customers need to use AT+QSSLCFG** to configure the SSL version, cipher suite, security level, CA certificate, client certificate and client key, and ignore the RTC time, which will be used in the MQTT SSL handshake process.
4. Please ensure that the data cannot time out during transmission.
5. **AT+QMTCFG="aliauth" is only used for Aliyun, if it has been configured, you can omit <username> in AT+QMTCONN and <password>.**

3.3.2. AT+QMTOPEN open MQTT client network

This command is used to open the network for MQTT clients.

AT+QMTOPEN Open the **MQTT** client network test	
command **AT+QMTOPEN=?**	response **+QMTOPEN:** (supported <client_idx> range), "hostname", (supported **<port> range)** **OK**
query command **AT+QMTOPEN?**	response **[+QMTOPEN: <client_idx>,<host_name>,<port>]** **OK** In case of any errors: **ERROR**
set command **AT+QMTOPEN=<client_idx>,<host_name>,<port>**	response **OK** **+QMTOPEN: <client_idx>,<result>** In case of any errors: **ERROR**
maximum response time	120 seconds, subject to network status
Feature Description	/

parameter

<client_idx> integer. MQTT client identifier. Range: 0~5. **<host_name>** String type. Server address, which can be an IP address or a domain name. Maximum length: 100 bytes. Integer. server port. Range: 1~65535. Integer.	
<port>	Command execution result. -1 Failed to open the network 0 Successfully opened the network 1 Parameter error
<result>	2 MQTT identifier is occupied 3 Failed to activate PDP 4 Failed to resolve the domain name 5 Network disconnection caused an error

3.3.3. AT+QMTCLOSE close MQTT client network

This command is used to close the MQTT client network.

AT+QMTCLOSE Close **the MQTT** client network	
Test command **AT+QMTCLOSE=?**	response **+QMTCLOSE: (supported <client_idx> range)** **OK**
set command **AT+QMTCLOSE=<client_idx>**	response **OK** **+QMTCLOSE: <client_idx>,<result>** In case of any errors: **ERROR**
maximum response time	30 seconds
Feature Description	/

parameter

<client_idx>	Integer. MQTT client identifier. Range: 0~5. Integer. Command
<result>	execution result. -1 Failed to close the network 0 Closed the network successfully

3.3.4. AT+QMTCONN connects the client to the MQTT server

This command is used when the client requests to connect to the MQTT server. When the client establishes a TCP/IP Socket connection with the server, A protocol layer session must be created using the CONNECT stream.

AT+QMTCONN connects the client to the MQTT server	
test command **AT+QMTCONN=?**	response **+QMTCONN: (supported <client_idx> range), "clientid","userna me","password"** **OK**
Query command **AT+QMTCONN?**	response **[+QMTCONN: <client_idx>,<state>]** **OK** In case of any errors: **ERROR**
set command **AT+QMTCONN=<client_idx>,<clientid>[,<username>,<password>]**	response **OK** **+QMTCONN: <client_idx>,<result>[,<ret_code>]** In case of any errors: **ERROR**
maximum response time	**<pkt_timeout> value (default** 5 seconds), affected by network status
Feature Description	/

parameter

<client_idx>	Integer. MQTT client identifier. Range: 0~5. String type. Client
<clientid>	identifier. String type. Client username, which can be used
<username>	for authorization. String type. The password corresponding to
<password>	the client username can be used for authorization. Integer. Command execution
<result>	result. 0 Data packet sent successfully and received ACK from the server 1 Data packet retransmission 2 Data packet sending failure Integer. MQTT connection state.
<state>	1 MQTT initialization 2 MQTT is connecting 3 MQTT has been successfully connected

	4 MQTT Disconnecting Integer.
<ret_code>	Connection status return code. 0 Connection accepted 1 Connection refused: Protocol version not accepted 2 Connection refused: Identifier rejected 3 Connection refused: Server unavailable 4 Connection refused: Bad username or password 5 Connection refused: Unauthorized integer. Packet transmission timeout.
<pkt_timeout>	Range: 1~60; Default value: 5; Unit: second. The timeout period can be configured by **AT+QMTCFG="timeout",<client_idx>[,<pkt_timeout>,<retry_times>,<timeout_notice>].**

Remark

If a certain client ID has already connected to the server, and another client uses the same ID to reconnect, the server must disconnect the original client connection, and then the CONNECT flow of the new client can be performed.

3.3.5. **AT+QMTDISC** client disconnects from **MQTT** server

Use this command when the client requests to disconnect from the MQTT server. Client sends DISCONNECT message to server table indicates that the TCP/IP connection with the server is about to be disconnected.

AT+QMTDISC client disconnects from **MQTT** server	
test command **AT+QMTDISC=?**	response **+QMTDISC: (supported <client_idx> ranges)** **OK**
set command **AT+QMTDISC=<client_idx>**	response **OK** **+QMTDISC: <client_idx>,<result>** In case of any errors: **ERROR**
maximum response time	30 seconds
Feature Description	/

parameter

<client_idx>	Integer. MQTT client identifier. Range: 0~5. Integer.
<result>	Command execution result. -1 disconnection failed 0 disconnected successfully

3.3.6. AT+QMTSUB subscribe topic

This command is used to subscribe to one or more topics. The client sends a SUBSCRIBE message to subscribe to one or more topics to the server. When subscribing When publishing messages to a topic, the server transmits these messages to the client as PUBLISH messages.

AT+QMTSUB subscribe topic test	
command **AT+QMTSUB=?**	response **+QMTSUB: (supported <client_idx>** range), **<msgid>, list of ["topic", qos]** **OK**
Set command **AT+QMTSUB=<client_idx>,<msgid> ,<topic1>,<qos1>[,<topic2>,<qos2> ...]**	response **OK** **+QMTSUB: <client_idx>,<msgid>,<result>[,<value>]** In case of any errors: **ERROR**
maximum response time	**<pkt_timeout> × <retry_times> (default** 15 seconds), affected by network status
Feature Description	/

parameter

<client_idx> integer. MQTT client identifier. Range: 0~5. **<msgid> <topic>**	Integer. Packet identifier. Range: 1~65535. String type. Topics to which clients subscribe or unsubscribe.
<qos>	Integer. The QoS level of the message published by the client. 0 send at most once 1 send at least once 2 only send integer once. Command execution result. 0 Packet sent successfully and received ACK from server 1 Packet
<result>	retransmitted 2 Packet sent failed

<value>	Integer. If <result>=0, then <value> will be the vector of the confirmed QoS class; if <result>=1, it will indicate the number of data packet retransmissions; **if <result>=2, it will not be displayed.**

<pkt_timeout> integer. Packet transmission timeout. Range: 1~60; Default value: 5; Unit: second. The timeout period can be configured by **AT+QMTCFG="timeout",<client_idx>[,<pkt_timeout>,<retry_times>,<timeout_notice>] . <retry_times>** integer. The number of retries after packet transmission failures. Range: 0~10; Default value: 3.

Remark

The <msgid> is only displayed in messages whose QoS bits in the fixed header indicate that the QoS class is 1 or 2. The <msgid> must be unique within a group of in-flight messages **in a particular communication direction .** Generally speaking, the value of this parameter increases one by one according to the number of messages, but there is no such requirement in actual situations.

3.3.7. AT+QMTUNS unsubscribe topic

This command is used to unsubscribe from one or more topics. The client sends an UNSUBSCRIBE message to the server to unsubscribe from the identified topic.

AT+QMTUNS unsubscribe topic	
Test command **AT+QMTUNS=?**	response **+QMTUNS: (supported <client_idx>** range), **<msgid>, list of ["topic"]** **OK**
Set command **AT+QMTUNS=<client_idx>,<msgid>,<topic1>[,<topic2>...]**	response **OK** **+QMTUNS: <client_idx>,<msgid>,<result>[,<value>]** In case of any errors: **ERROR**
maximum response time	**<pkt_timeout> × <retry_times> (default** 15 seconds), affected by network status
Feature Description	/

parameter

<client_idx> integer. MQTT client identifier. Range: 0~5. **<msgid> <topic>**	Integer. Packet identifier. Range: 1~65535. String type. The topic that the client wants to subscribe or unsubscribe from.
<result>	Integer. Command execution result.

<value>	0 Data packet sent successfully and received ACK from the server 1 Data packet retransmission 2 Data packet sending failure Integer. If <result>=0, then <value> will be the vector of the confirmed QoS class; if <result>=1, it will indicate the number of data packet retransmissions; **if <result>=2, it will not be displayed.**

<pkt_timeout> Packet transmission timeout. Range: 1~60; Default value: 5; Unit: second. The timeout period can be configured by **AT +QMTCFG="timeout",<tcpconnectID>[,<pkt_timeout>,<retry_times>,<timeout_notice>] . <retry_times>** integer. The number of retries after packet transmission failures. Range: 0~10; Default value: 3.

3.3.8. AT+QMTPUBEX publish message

Through this command, the client can publish a fixed-length message to the server, and then the server distributes it to interested subscribers. Each PUBLISH message is associated with a topic name. If a client subscribes to one or more topics, when the subscribed topic publishes messages, the server will transmit these messages to the client as PUBLISH messages.

AT+QMTPUBEX release news	
test command **AT+QMTPUBEX=?**	response **+QMTPUBEX:** (supported <client_idx> range), <msgid>, (supported **<qos> range), (supported <retain> list), "topic","length"** **OK**
set command **AT+QMTPUBEX=<client_idx>,<msgid>,<qos>,<retain>,<topic>,<length>**	response **>** After the response>, enter the data to be sent, if the actual length of the data is greater than <length>, the extra bytes will be deleted **OK** **+QMTPUBEX: <client_idx>,<msgid>,<result>[,<value>]** In case of any errors: **ERROR**
maximum response time	**<pkt_timeout>** × **<retry_times> (default** 15 seconds), affected by network status
Feature Description	/

parameter

<client_idx> integer. MQTT client identifier. Range: 0~5. **<msgid>**

Integer. Packet message identifier. Range: 0~65535. Only when <qos>=0 , this parameter value is 0. Integer. The QoS

<qos> class at which the client wants to publish messages. 0 send at most once 1 send at least once 2 only send integer once.

Whether the server saves the message after it is sent to the current subscriber. 0 does not save 1 saves string type. Topic

to be published. Integer. The data length of the message to be published. Integer. Command execution result. 0 The data

packet is sent successfully and the server's **ACK is received (when <qos>=0 ,** the data is released, no ACK is required)

<retain> 1 The data packet is retransmitted 2 The integer of the data packet sending failure. If <result>=1, it indicates the number

of data packet retransmissions; **if <result>=0** or 2, this parameter is not displayed.

<topic>

<length>

<result>

<value>

<pkt_timeout> integer. Data transfer timeout. Range: 1~60; Default value: 5; Unit: second. The timeout period can be configured by **AT +QMTCFG="timeout",<client_idx>[,<pkt_timeout>,<retry_times>,<timeout_notice>] . <retry_times>** integer. The number of retries after packet transmission failures. Range: 0~10; Default value: 3.

Remark

1. If the command is executed successfully and the response is OK, the client can continue to publish new data packets. The maximum number of transmitted packets cannot be greater than slip The size of the sliding window (the size of the sliding window is 5), otherwise it will return **ERROR.**
2. After the command is executed, the client can send data, that is, payload information. The maximum length of each input data is 1500 bytes. 3. The publisher can issue PUBLISH messages to the server, and the server can also issue PUBLISH messages to subscribers. When the server publishes a message to the subscriber, it will return URC to notify the Host to read **the data sent by the MQTT server: +QMTRECV: <client_idx>,<msgid>,<topic>[,<payload_length>],<payload>, about** URC description For details, please refer to

3.3.9. AT+QMTRECV reads messages from cache

This command is used to read messages from the storage cache, which will be stored in this cache when the server reports messages.

AT+QMTRECV reads messages from cache	
Test command **AT+QMTRECV=?** Query	response **OK**
command **AT+QMTRECV?**	response **+QMTRECV:** **<client_idx>,<store_status_0>,<store_status_1>,<store_status_2>,<store_status_3>,<store_status_4>** **OK** If there is no MQTT connection: **OK**
set command **AT+QMTRECV=<client_idx>[,<recv_id>]**	response **[+QMTRECV: <client_idx>,<msgid>,<topic>[,<payload_len>],<payload>** **[…]]** **OK** If there is no MQTT connection: **ERROR**
maximum response time	/
Feature Description	/

parameter

<client_idx> integer. MQTT client identifier. Range: 0~5. **<store_status>** integer. Indicates whether

there are messages in the cache, and a maximum of 5 pieces of information can be stored in the cache, so a maximum of 5 pieces of information can be reported at the same time

URC.

0 There is no data in the cache

1 There is data in the cache

<recv_id> integer. Indicates the serial number of each received data. Range: 0~4. **<msgid>**

<topic> <payload_len> integer. The length of the payload information. Range: 0~65535. **<payload>** string type. qos>=0 , this parameter can be 0. String type. Topics

Payload information related to topic name. to be published.

4 MQTT -related URCs

This chapter mainly describes the URC related to MQTT.

Table 3: URCs related to MQTT

Serial number URC format	describe
[1] +QMTSTAT: <client_idx>,<err_code>	When the state of the MQTT link layer changes, the client will disconnect MQTT connection and report URC.
[2] +QMTRECV: <client_idx>,<msgid>,<topic>[,<payload_len>],<payload>	When the client receives the data packet from the MQTT server, it will report the URC. Report the URC when the message received
[3] +QMTRECV: <client_idx>,<recv_id>	from the MQTT server is stored in the cache. When the state of the MQTT chain layer changes, the client will close the MQTT connection
[4] +QMTPING: <client_idx>,<result>	and report this URC.

4.1. +QMTSTAT URC of MQTT link layer state change

This URC starts with +QMTSTAT:, and this URC will be reported when the status of the MQTT link layer changes.

+QMTSTAT URC of MQTT link layer state change	
+QMTSTAT: <client_idx>,<err_code>	When the state of the MQTT link layer changes, the client will disconnect the MQTT connection and report URC.

parameter

<client_idx>	Integer. MQTT client identifier. Range: 0~5. error code.
<err_code>	Please refer to the table below for details.

Table **4:** Error codes in URC

<err_code> description		Solution
1	The connection was disconnected or reset by the server	Execute **AT+QMTOPEN** to rebuild the MQTT connection.
2	Sending PINGREQ packet timed out or failed	Deactivate the PDP first, then activate the PDP and rebuild MQTT connection.
3	Sending CONNECT packet timed out or failed	1. Check whether the entered user name and password are correct. 2. Make sure the Client ID is not already in use. 3. Re-establish the MQTT connection and try to send again CONNECT packet to the server. 1.
4	Timeout or failure to receive CONNACK packet	Check whether the entered user name and password are correct. 2. Make sure the Client ID is not already in use. 3. Re-establish the MQTT connection and try to send again CONNECT packet to the server.
5	The client sends a DISCONNECT packet to the server, but the server actively disconnects the MQTT connection	Normal Process.
6	Because sending data packets always fails, the client actively disconnects MQTT connection	1. Make sure the data is correct. 2. Maybe due to network congestion or other errors, try to re-establish the MQTT connection.
7	The link is down or the server is unavailable	Make sure the current link or server is available.
8	The client actively disconnects the MQTT connection	Try to re-establish the connection.
9~255	reserved for future use	

4.2. +QMTRECV notifies the Host to read the URC of the MQTT packet

This URC **starts with +QMTRECV: and is mainly used to notify** the Host to read the data packets sent by the MQTT server.

+QMTRECV notifies **the Host** to read **the URC** of the **MQTT** packet	
+QMTRECV: <client_idx>,<msgid>,<topic>[,<payload_len>],<payload>	Notify the Host to read the data packets sent by the MQTT server.
+QMTRECV: <client_idx>,<recv_id>[,<payload_len>]	Report the URC when the message received from the MQTT server is stored in the cache.

parameter

<client_idx>	Integer. MQTT client identifier. Range: 0~5. Integer. The message
<msgid>	identifier for the packet.

<topic>	String type. Topics received from the MQTT server. Integer. The
<payload_len>	length of the payload information. String type. The payload
<payload> <recv_id>	associated with the subject name. Integer. Indicates the serial number of each received data. Range: 0~4.

4.3. +QMTPING notifies the URC of the Ping state within the MQTT keepalive time

This URC **starts with +QMTPING:, if the server does not** receive the message from the client within 1.5 times the keep-alive time, it means that the client sends a DISCONNECT message, the server will disconnect from the client, and report the URC at this time.

+QMTPING notifies the URC of the Ping status within the MQTT keepalive time	
+QMTPING: <client_idx>,<result>	When the state of the MQTT chain layer changes, the client will close the MQTT connection and report this URC.

parameter

<client_idx>	Integer. MQTT client identifier. Range: 0~5. Integer. The
<result>	result of the ping status. 1 failed

5 example

This chapter mainly provides examples to explain how to use MQTT related commands.

5.1. Example of MQTT operation without SSL

```
// Configure receive mode.
AT+QMTCFG="recv/mode",0,0,1 OK

//Configure Alibaba Cloud device information.
AT+QMTCFG="aliauth",0,"oyjtmPl5a5j","MQTT_TEST","wN9Y6pZSIly7Exa5qVzcmigEGO4kAazZ"
OK
AT+QMTOPEN=?
+QMTOPEN: (0-5), "hostname", (1-65535)

OK

//MQTT client opens the network.
AT+QMTOPEN=0,"iot-as-mqtt.cn-shanghai.aliyuncs.com",1883 OK


+QMTOPEN: 0,0AT                                   //MQTT client successfully opened the network.
+QMTOPEN?
+QMTOPEN: 0,"iot-as-mqtt.cn-shanghai.aliyuncs.com",1883

OK
AT+QMTCONN=?
+QMTCONN: (0-5),"clientid","username","password"

OK

//The client connects to the MQTT server. //
If you have connected to Alibaba Cloud, you can use AT+QMTCFG="aliauth" to configure the device information in advance, and then you can omit <username>
and <password>.
AT+QMTCONN=0,"clientExample"
OK
```

+QMTCONN: 0,0,0AT //The client successfully connects to the MQTT server.

+QMTSUB=?

+QMTSUB: (0-5),<msgid>,list of ["topic",qos]

OK

//Subscribe to the topic.

AT+QMTSUB=0,1,"topic/example",2 OK

+QMTSUB: 0,1,0,2

AT+QMTSUB=0,1,"topic/pub",0

OK

+QMTSUB: 0,1,0,0

//If the client subscribes to a topic and other devices publish the same topic to the server, the module will report the following information:

+QMTRECV: 0,0,"topic/example",36,"This is the payload related to topic"

// Unsubscribe from the topic.

AT+QMTUNS=0,2,"topic/example"

OK

+QMTUNS: 0,2,0AT

+QMTPUBEX=?

+QMTPUBEX: (0-5),<msgid>,(0-2),(0,1),"topic","length"

OK

//After the response>, enter and send **This is test data, hello** MQTT. The maximum length of the data is 1500 bytes, and the extra part will be ignored.

AT+QMTPUBEX=0,0,0,0,"topic/pub",30 >This is test

data, hello MQTT.

OK

+QMTPUBEX: 0,0,0

//If the client subscribes to the topic named "topic/pub", and other devices publish the same topic to the server, the module will report the following information: + **QMTRECV: 0,0,"topic/pub",30,This is test data, hello MQTT.**

//Disconnect the client from the MQTT server.

AT+QMTDISC=0

OK

+QMTDISC: 0,0 //The connection is disconnected successfully.

5.2. Example of MQTT operation with SSL

For details about SSL-related commands, please refer todocument *[1].*

```
// Configure receive mode.
AT+QMTCFG="recv/mode",0,0,1 OK

//Configure the MQTT session in SSL mode.
AT+QMTCFG="SSL",0,1,2
OK

//If the SSL authorization method is server authentication, store the CA certificate in UFS.
AT+QFUPL="UFS:cacert.pem",1758,100 CONNECT

<Input the cacert.pem data, the size is 1758 bytes> +QFUPL:
1758,384a

OK

//If the SSL authorization method is server authentication, store the CC certificate to UFS.
AT+QFUPL="UFS:client.pem",1220,100 CONNECT

<Input the client.pem data, the size is 1220 bytes> +QFUPL:
1220,2d53

OK

//If the SSL authorization method is server authentication, store the CK certificate to UFS.
AT+QFUPL="UFS:user_key.pem",1679,100 CONNECT

<Input the client.pem data, the size is 1679 bytes> +QFUPL:
1679,335f

OK

// Configure the CA certificate.
AT+QSSLCFG="cacert",2,"UFS:cacert.pem"
OK

//Configure CC certificate.
AT+QSSLCFG="clientcert",2,"UFS:client.pem"
OK
```

```
//Configure CK certificate.
AT+QSSLCFG="clientkey",2,"UFS:user_key.pem"
OK

//Configure SSL parameters.
AT+QSSLCFG="seclevel",2,2 OK                    //SSL authorization method: server authentication.

AT+QSSLCFG="sslversion",2,4 OK                  //SSL authorization version.
AT+QSSLCFG="ciphersuite",2,0xFFFF //
cipher suite.
OK
AT+QSSLCFG="ignorelocaltime",2,1 //Ignore authorization time.
OK

//Start MQTT SSL connection.
AT+QMTOPEN=0,"a1zgnxur10j8ux.iot.us-east-1.amazonaws.com",8883 OK


+QMTOPEN: 0,0

//Connect to the MQTT server.
AT+QMTCONN=0,"M26_0206"
OK

+QMTCONN: 0,0,0

//Subscribe to the topic.
AT+QMTSUB=0,1,"$aws/things/M26_0206/shadow/update/accepted",1 OK


+QMTSUB: 0,1,0,1

//make an announcement.
AT+QMTPUBEX=0,1,1,0,"$aws/things/M26_0206/shadow/update/accepted",32 >This is publish data from
client OK


+QMTPUBEX: 0,1,0

//If the client subscribes to the topic named "$aws/things/M26_0206/shadow/update/accepted" and other devices publish messages of the same
topic to the server, the module will report the following information:
+QMTRECV: 0,1,"$aws/things/M26_0206/shadow/update/accepted",32,"This is publish data from client"
```

```
//The client disconnects the MQTT server.
AT+QMTDISC=0
OK

+QMTDISC: 0,0
```

6 Appendix Reference Documents and Terminology Abbreviations

Table 5: Reference Documents

file name
[1] Quectel_LTE_Standard(A) Series_SSL_Application Guide

Table 6: Terminology Abbreviations

abbreviation	English full name	Chinese full name
ACK	acknowledgment	confirmation message
CA	Certificate Authority	Certificate Authority
IP	Internet Protocol	internet protocol
MQTT	Message Queuing Telemetry Transport Message Queuing Telemetry Transport	
PDPs	Packet Data Protocol	packet data protocol
QoS	Quality of Service	service quality
RTC	Real-Time Clock	Real Time Clock
SSL	Secure Sockets Layer	Secure Sockets Layer
TCP	Transmission Control Protocol	transmission control protocol
UFS	User File System	user file system
URC	Unsolicited Result Code	unsolicited result code