LTE Standard(A) series

TCP/IP Application Guide

LTE Standard Module Series

Version: 1.4

Date: 2022-03-02

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 Statement
1.0	2019-10-28	Larson LI	initial release
1.1	2019-10-30	Larson LI	1. Update the meaning of the value of the parameter <authentication> APN authentication method (Chapter 2.1.1) 2. Update the value range of parameter <contextID> (Chapter 2.1.4, Chapter 2.1.6 and Chapter 2.1.12) 1. Add documents applicable to modules
1.2	2020-07-24	Larson LI	EG912Y series and EC200S-CN 2. Update the value range of <idle_time> in AT+QICFG="tcp/keepalive" 3. Add the following AT commands: ÿ AT+QICFG="viewmode" ÿ AT+QICFG="passiveclosed" ÿ AT+QICFG="recvind" ÿ AT+QICFG="tcp/accetp" 1. Added documents applicable to
1.3	2021-04-20	Larson LI	modules EC200N-CN and EC600N-CN 2. Updated the value range of parameter <authentication> (Chapter 2.3.1) 3. Added the following AT commands (Chapter 2.3.15): ÿ AT+QICFG= "tcp/retranscfg" ÿ AT+QICFG="send/buffersize" ÿ AT+QICFG="send/auto" ÿ AT+QICFG="recv/ignore" ÿ AT+QICFG="formatcfg" ÿ AT+QICFG="qisend/timeout" ÿ AT+QICFG="close/mode"
1.4	2022-03-02	Larson LI	1. Add documents applicable to modules EC200A series, EC800N-CN and EG915N-EU. 2. Delete the EG912Y-CN module. 3. Added setting command AT+QICFG="TCP/SendMode" (section 2.3.1 chapter).

Table of contents

table index

1 Introduction

Quectel LTE Standard(A) series modules have a built-in TCP/IP protocol stack, and the Host can directly access the network through AT commands; this greatly reduces the module's dependence on PPP and external TCP/IP protocol stacks, thereby reducing the cost of terminal design. cost.

LTE Standard(A) series modules can provide Socket services such as TCP client, UDP client, TCP server, UDP server, etc. service.

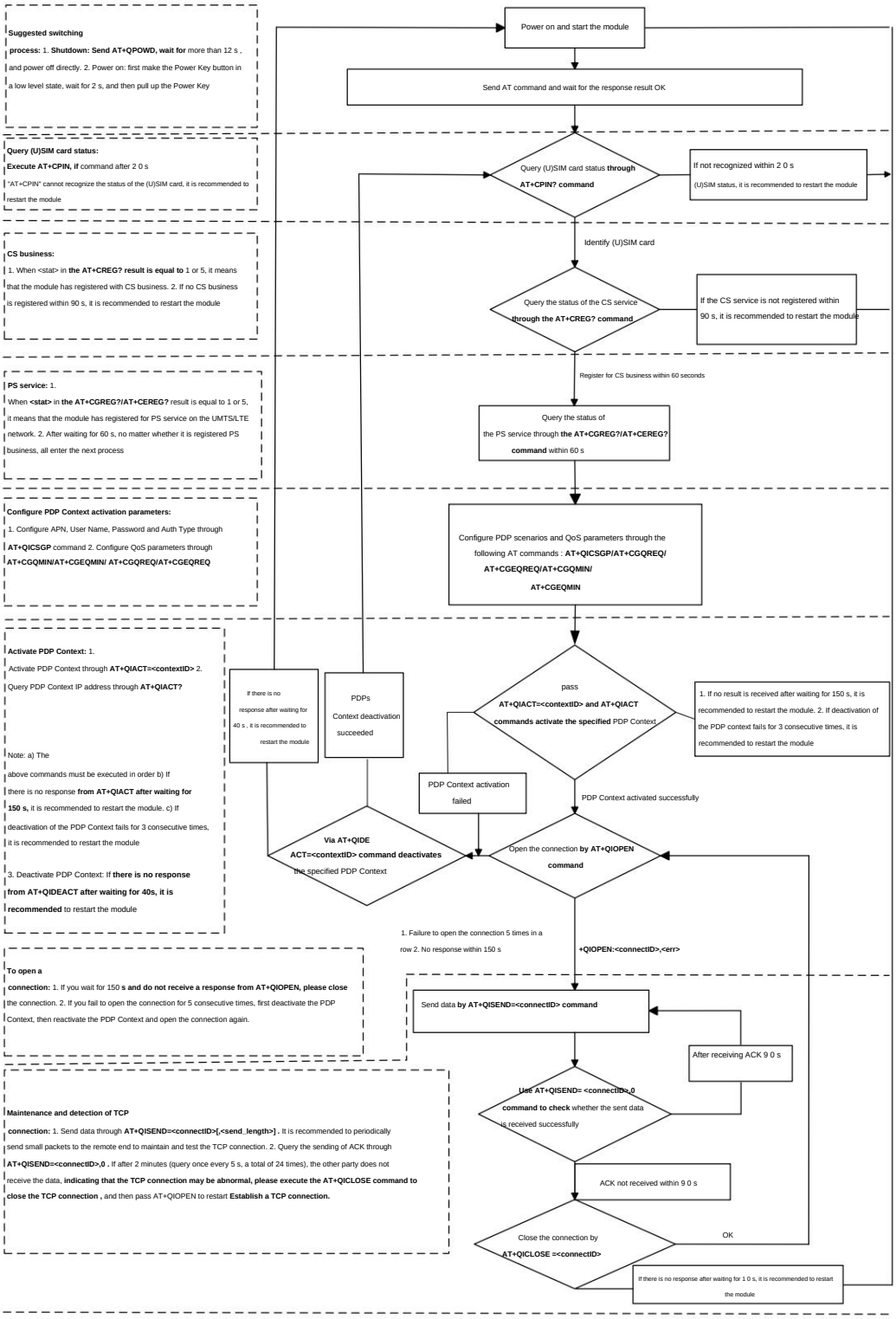
1.1. Applicable modules

Table 1: Applicable modules

Module series	module
LTE Standard(A)	EC200T series
	EC200S series
	EC200A series
	EC200N-CN
	EC600S-CN
	EC600N-CN
	EC800N-CN
	EG912Y-EU
	EG915N-EU

1.2. The process of using TCP/IP AT command

Through TCP/IP AT command, Host can configure PDP scene, activate/deactivate PDP scene, open/close Socket service, and send/receive data through Socket service. The flow of using the TCP/IP AT command is shown in the figure below:



Notes: 1. Please note that the next AT command can only be executed after receiving the execution result of the previous AT command (for example: OK/CME **ERROR/CMS** ERROR). In general, if you do not receive the execution result of the AT command within 60 s, it is recommended to restart the module. 2. If **no AT+QIACT response is received within 150 s, no AT+QICLOSE response is received within** 10 s, and **no AT+QIDEACT response is received within** 40 s, it is recommended to restart the module. 3. It is not recommended to restart the module frequently. If you fail to restart the module for 3 consecutive times due to the execution of AT commands, you can restart immediately for the first time after that, and then wait for 10 minutes before restarting for the second time. If the restart still fails, it is recommended to wait for 30 minutes before restarting for the third time , and so on, gradually increasing the restart interval.

Figure **1:** Flowchart of using TCP/IP AT commands

1.3. Description of data access mode

LTE Standard(A) series modules support the following three data access modes:

ÿ Cache mode ÿ Direct mode ÿ Transparent mode

When using **AT+QIOPEN** to open the Socket **service, you can specify the data access mode of the Socket through the parameter <access_mode> ;** when the Socket service is successfully opened, you can switch the data access mode through **AT+QISWTMD .**

1. In buffer mode, you can send data through **AT+QISEND** command. When receiving data from the network, the module will buffer the received data, And directly report URC **+QIURC: "recv",<connectID>, and then the user can read the cached data through the AT+QIRD** command.

2. In direct mode, you can send data through **AT+QISEND .** When receiving data from the network, the data will be directly output to the COM port in the following format: **+QIURC: "recv",<connectID>,<currectrecvlength><CR><LF><data>** or **+QIURC: "recv",<connectID >,<currentrecvlength>,<remoteIP>,<remote_port><CR><LF><data>.**

3. In the transparent transmission mode, the corresponding serial port (such as UART port, USB Modem port, etc.) will enter the exclusive mode, the data received through the COM port will be directly sent to the network end, and the data received from the network will be directly transmitted from the COM port . output. +++ can be used to exit the transparent transmission mode: **when the input +++ returns OK ,** the access mode will switch to the cache mode, and if you need to switch back to the transparent transmission mode, you can use **the AT+QISWTMD** command.

ÿ Exit transparent transmission mode

The user can exit the transparent transmission mode through +++ or DTR (need to set AT&D1 first), in order to prevent +++ from being sent as data In actual operation, the following steps must be followed:

1) No other data can be input within 1 second or longer before +++ input; 2) +++ must be input within 1 second, and no other data can be input; **3)** No other data can be input within 1 second after +++ input Input any other data; **4) Make the module exit the transparent transmission mode through +++ or** DTR (set AT&D1) until the module returns OK; at this time, the module exits successfully out of transparent transmission mode.

ÿ Switch to transparent transmission mode

1) By executing **the AT+QISWTMD command: When executing this command, specify <access_mode> as** 2; if it returns **CONNECT,** it means that it has successfully switched to the transparent transmission mode. 2) Through **ATO** mode: After exiting the transparent transmission mode, you can switch back to the transparent transmission mode through **ATO ;** if you return to **CONNECT, it means that** you have successfully switched to the transparent transmission mode. If the module has not entered the transparent transmission mode before, executing **ATO** will return **NO CARRIER.**

Remark

1. In cache mode, if the cache is not empty, the module will not wait until all received data in the cache is read through **AT+QIRD .** Report a new URC.
2. In transparent transmission mode, AT commands cannot be executed. If the Socket connection is disconnected due to network errors or other reasons, the module will report **NO** CARRIER and exit the transparent transmission mode. In this case, you can execute **AT+QICLOSE** to close the Socket service.

2 Detailed Explanation of TCP/IP AT Commands

This chapter mainly describes AT commands related to TCP/IP.

2.1. AT command description

2.1.1. 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.

2.1.2. 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.

2.2. 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.

2.3. AT command description

2.3.1. AT+QICFG configuration optional parameters

This command is used to configure optional parameters.

AT+QICFG configuration optional parameters	
test command **AT+QICFG=?**	response: **+QICFG:** "transpktsize", (supported <transpktsize> range) **+QICFG:** "transwaittm", (supported <transwaittm> range) **+QICFG: "dataformat", (list of supported <send_data_format>), (list of supported <recv_data_format>)** **+QICFG:** "viewmode", (list of supported <viewmode>) **+QICFG: "passiveclosed", (list of supported <closed>s)** **+QICFG: "udp/readmode", (list of supported <mode>)** **+QICFG: "udp/sendmode", (list of supported <mode>)** **+QICFG: "tcp/keepalive", (supported <enable> list), (supported <idle_time> range), (supported <interval_time> range), (supported <probe_cnt> range)** **+QICFG: "recvind", (list of supported <show_length>)** **+QICFG: "tcp/retranscfg", (supported <retran_times> range), (supported <retran_time> range)** **+QICFG: "tcp/accept", (supported <state> list)** **+QICFG:** "send/buffersize", (supported <buffer_size> range) **+QICFG: "send/auto", (supported <contextID> range), (supported <cycle_ time> range), (sent <msg_auto> data)** **+QICFG: "recv/ignore", (supported <contextID> scope), (filtered <msg _ignore> data)** **+QICFG:** "formatcfg", (supported <format> range) **+QICFG: "qisend/timeout", (supported <timeout> range)** **+QICFG:** "close/mode", (supported <close_mode> range) **+QICFG:** "TCP/SendMode", (list of supported <send_mode>) **OK**
set command	response

Set the maximum length of data to be sent **AT+QICFG="transpktsize"[,<transpktsize>]**	If the optional parameter is omitted, the current configuration is queried: **+QICFG: "transpktsize",<transpktsize>** **OK** If an optional parameter is specified, set the maximum length of the data to be sent: **OK** In case of error: **ERROR**
Set the command to set the waiting time before automatically sending data in transparent transmission mode **AT+QICFG="transwaittm"[,<transwaittm>]**	Response If the optional parameter is omitted, the current configuration is queried: **+QICFG: "transwaittm",<transwaittm>** **OK** If optional parameters are specified, set the waiting time before automatically sending data in transparent transmission mode: **OK** In case of error: **ERROR**
Set the command to set the sending and receiving data format (only in non-transparent mode) **AT+QICFG="dataformat"[,<send_data_format>,<recv_data_format>]**	Response If the optional parameter is omitted, the current configuration is queried: **+QICFG: "dataformat",<send_data_format>,<recv_data_format>** **OK** If optional parameters are specified, set the sending and receiving data format: **OK** In case of error: **ERROR**
Set the command to set the output format of the received data (only in non-transparent mode) **AT+QICFG="viewmode"[,<view_mode>]**	Response If the optional parameter is omitted, the current configuration is queried: **+QICFG: "viewmode",<view_mode>** **OK** If optional parameters are specified, set the output format of the received data: **OK** In case of error: **ERROR**

Set the command to set whether to close passively when the server is closed TCP connection **AT+QICFG="passive closed"[,<closed>]**	Response If the optional parameter is omitted, the current configuration is queried: **+QICFG: "passive closed",<closed>** **OK** If an optional parameter is specified, set whether to passively close the TCP connection when the server is closed: **OK** In case of error: **ERROR**
Set command to set the read mode of UDP data **AT+QICFG="udp/readmode"[,<mode>]**	Response If the optional parameter is omitted, the current configuration is queried: **+QICFG: "udp/readmode",<mode>** **OK** Specify optional parameters, then set the read mode of UDP data: **OK** In case of error: **ERROR**
Set command to set the sending mode of UDP data **AT+QICFG="udp/sendmode"[,<mode>]**	Response If the optional parameter is omitted, the current configuration is queried: **+QICFG: "udp/sendmode",<mode>** **OK** Specify optional parameters, then set the sending mode of UDP data: **OK** In case of error: **ERROR**
Set command to set whether to send TCP keep-alive information **AT+QICFG="tcp/** **keepalive"[,<enable>[,<idle_time>,<interval_time>,<probe_cnt>]]**	Response If the optional parameter is omitted, the current configuration is queried: **+QICFG: "tcp/keepalive",<enable>[,<idle_time>,<interval_time>,<probe_cnt>]** **OK** Specify optional parameters to set whether to send TCP keep-alive information: **OK** In case of error:

	ERROR
Set the command to set in the cache mode, whether the URC reported by the module after receiving the data shows the data length **AT+QICFG="recvind"[,<show_length>]**	Response If the optional parameter is omitted, the current configuration is queried: **+QICFG: "recvind",<show_length>** **OK** Specify optional parameters, then set whether to display the data length in the URC reported after receiving the data: **OK** In case of error: **ERROR**
Set the command to set the number of retransmissions and the retransmission interval **AT+QICFG="tcp/retranscfg"[,<retran_times>,<retran_time>]**	Response If the optional parameter is omitted, the current configuration is queried: **+QICFG: "tcp/retranscfg",<retran_times>,<retran_time>** **OK** If optional parameters are specified, set the number of retransmissions and retransmission interval: **OK** In case of error: **ERROR**
The set command enables or disables the automatic receipt of TCP connection **AT+QICFG="tcp/accept"[,<stat e>]**	Response If the optional parameter is omitted, the current configuration is queried: **+QICFG: "tcp/accept",<state>** **OK** Specify optional parameters to set whether to automatically accept TCP connections from clients: **OK** In case of error: **ERROR**
Set the command to set the maximum length of data sent at a time **AT+QICFG="send/buffersize"[,<buffer_size>]**	Response If the optional parameter is omitted, the current configuration is queried: **+QICFG: "send/buffersize",<buffer_size>** **OK** If optional parameters are specified, the maximum length of data sent at a time is set: **OK**

	In case of error: **ERROR**
Set the command to set the periodic sending of heartbeat packet data **AT+QICFG="send/auto",<contextID>[,<cycle_time>[,<msg_auto>]]**	Response If the optional parameter is omitted, the current configuration is queried: **+QICFG: "send/auto",<contextID>,<cycle_time>[,<msg_aut o>]** **OK** Specify optional parameters to set the cycle and data for sending heartbeat packets: **OK** In case of error: **ERROR**
set command set filter specified data **AT+QICFG="recv/ignore",<contextID>[,<msg_ignore>]**	Response If the optional parameter is omitted, the current configuration is queried: **+QICFG: "recv/ignore",<contextID>[,<msg_ignore>]** **OK** Specify optional parameters, then set the data to be filtered: **OK** In case of error: **ERROR**
Set the command to set the format **of sending the AT+QISEND command to return >** **AT+QICFG="formatcfg"[,<format>]**	Response If the optional parameter is omitted, the current configuration is queried: **+QICFG: "formatcfg",<format>** **OK** Specify optional parameters, then Settings > Output Format: **OK** In case of error: **ERROR**
Set the command to set the maximum response time for sending the **AT+QISEND** command **AT+QICFG="qisend/timeout"[,<ti meout>]**	Response If the optional parameter is omitted, the current configuration is queried: **+QICFG: "qisend/timeout",<timeout>** **OK** Specify an optional parameter, then set the timeout after the output >:

	OK In case of error: **ERROR**
Set the command to set the asynchronous disconnection of **the TCP** connection **AT+QICFG="close/mode"[,<close_mode>]**	Response If the optional parameter is omitted, the current configuration is queried: **+QICFG: "close/mode",<close_mode>** **OK** Specify an optional parameter to set the asynchronous disconnection of the TCP connection: **OK** In case of error: **ERROR**
set command set **SEND OK** return mode **AT+QICFG="TCP/SendMode"[,<send_mode>]**	Response If the optional parameter is omitted, the current configuration is queried: **+QICFG: "TCP/SendMode",<send_mode>** **OK** Specify optional parameters, then set **the SEND OK** return mode: **OK** In case of error: **ERROR**
maximum response time	300 milliseconds
Feature Description	This command takes effect immediately; the parameter configuration is not saved.

parameter

<contextID>	Integer. PDP scene ID. Range: 1~15. Integer. In
<transpktsize>	transparent transmission mode, the maximum byte length of the data packet to be sent. Range: 1~1460; Default value: 1024; Unit: byte. **Integer. In transparent transmission mode, when the data to be sent is less than**
<transwaittm>	**<transpktsize>, the waiting time before automatically sending data .** Range: 0~20; Default: 2; Unit: 100 milliseconds. **<send_data_format>** integer. Send data format. When it is set to hexadecimal mode, it does not need to be prefixed with 0x, and the module will automatically combine two bytes into an ASCII code. 0 text mode 1 hexadecimal mode
	–
<recv_data_format> integer. Receive data format. When it is set to hexadecimal mode, it does not need to be prefixed with 0x, and the module will automatically combine two bytes into an ASCII code.	

	0 text mode 1 hexadecimal mode
<view_mode>	integer. 0 Output mode of received data: data header\r\ndata. 1 Output mode of received data: data header, data. Integer. 0 disables block mode 1 enables
<mode>	stream mode integers. 0 disables automatically accepting TCP connections from clients 1 enables automatically accepting TCP connections from clients Integer. Enable or disable sending TCP
<state>	keepalive messages. 0 disable 1 enable integer. In cache mode, enable or disable the module to display the data length in the URC reported after receiving the data. 0 disable 1 enable integer.
<enable>	Trigger keep alive cycle time. Range: 1~1800; unit: second. Integer. Time between sending packets in cycle time. Range: 25~100; unit: second. Integer. Number of times a packet is sent
<show_length>	within the cycle time. Range: 3~10. Integer. Enables or disables automatic disconnection of TCP connections after server shutdown. 0 disable 1 enable integer. The number of reconnections within the cycle time. Range: 3~12. Integer. Reconnect interval during cycle time. Range: 5~1000; unit: millisecond. Integer. The maximum number of bytes sent at one time. Range: 1460~10240. Integer. Set the time period
<idle_time>	for sending heartbeat packets. Range: 20~86400; unit: second. String type. Send heartbeat packet content.
<interval_time>	String type. Filter data content. Integer. Controls > Output Format.
<probe_cnt>	
<closed>	–
<retran_times>	
<retran_time>	
<buffer_size>	
<cycle_time>	
<msg_auto>	
<msg_ignore>	
<format>	0 0D0A> 1 0D0A>0D0A
<timeout>	Integer. Send data timeout. Range 0~120; unit: second. Integer.
<close_mode>	Enables or disables disconnecting TCP connections asynchronously. 0 disable 1 enable **integer. SEND OK** returns to mode. 0 Return **SEND OK** immediately after sending data 1
<send_mode>	Return **SEND OK** after receiving server ACK 2 **Return <connectID>, SEND OK after** receiving server ACK 3 **Report +QIURC: SEND OK after** receiving server ACK 4 **Report+ after** receiving server ACK **QIURC: <connectID>, SEND OK**

2.3.2. AT+QICSGP configures TCP/IP scene parameters

This command is used to configure <APN>, <username>, <password> and other TCP/IP scene parameters. QoS configuration needs to pass **AT+CGQMIN, AT+CGEQMIN, AT+CGQREQ** and **AT+CGEQREQ** commands; for details about the AT commands, please refer to document ***[1].***

AT+QICSGP configures **TCP/IP** scene parameters	
test command **AT+QICSGP=?**	response **+QICSGP: (supported <contextID> scope), (supported <context_type> scope), <APN>, <username>, <password>, (supported <authentication>** scope), (supported <CDMA_pwd> list) **OK**
Set the command to query the specified scene configuration **AT+QICSGP=<contextID>**	response **+QICSGP: <context_type>,<APN>,<username>,<password>,<authentication>** **OK**
Set the command to configure the specified scene **AT+QICSGP=<contextID>[,<context_ type>,<APN>[,<username>,<passwo rd>) [,<authentication>[,<CDMA_pw d>]]]]**	response **OK** In case of error: **ERROR**
maximum response time	/
Feature Description	This command takes effect immediately; the parameter configuration is automatically saved.

parameter

<contextID>	Integer. scene ID. Range: 1~15.
<context_type> integer. agreement type.	1 IPv4 2 IPv6 3 IPv4v6 string
<APN>	type. Access point name. String
<username>	type. username. Maximum length: 127 bytes. String type.
<authentication> integer. AP	password. Maximum length: 127 bytes. **<password>** P authentication method 0 None 1 PAP 2CHAP

	3 PAP or CHAP
<CDMA_pwd> integer. Whether to save <username> and <password> under CDMA **network.**	
	0 do not save 1
	save

example

AT+QICSGP=1	//Query scene 1 configuration.
+QICSGP: 1,"","","",0	
OK	
AT+QICSGP=1,1,"UNINET","","",1	//Configure Scenario 1, APN is configured as "UNINET" (China Unicom).
OK	

2.3.3. AT+QIACT activates PDP scene

Before using AT+QIACT to activate the PDP scene, you need to use AT+QICSGP to configure the scene. After the scene is activated, you can pass AT+QIACT? Query the IP address.

Although the range of <contextID> is 1~15, the module can only activate up to 3 PDP contexts at the same time. Affected by the network status, after executing **AT+QIACT ,** the maximum time to wait for the return result **OK** or **ERROR** is 150 seconds. Before the result is returned, no AT command can be executed.

AT+QIACT activates **PDP** scene	
test command **AT+QIACT=?**	response **+QIACT: (supported <contextID> range)** **OK**
query command **AT+QIACT?**	The response returns all currently activated scenes and IP addresses: **+QIACT: 1,<context_state>,<context_type>,<IP_address>** **.....** **+QIACT: 15,<context_state>,<context_type>,<IP_address>** **OK**
set command **AT+QIACT=<contextID>**	Response activates the specified scene: **OK** In case of error: **ERROR**
maximum response time	Affected by network status, the maximum response time is 150 seconds.

Feature Description	/

parameter

<contextID>	Integer. scene ID. Range: 1~15.
<context_state> integer. scene state. 0 deactivate 1 activate	
<context_type> integer. agreement type.	
1 IPv4	
2 IPv6	
3 IPv4v6	
<IP_address> String type. The local IP address after the scene is activated.	

2.3.4. AT+QIDEACT deactivate PDP scene

This command is used to deactivate a specific scene and disconnect all TCP/IP connections established in this scene. Affected by the network status, after executing **AT+QIDEACT ,** the maximum time to wait for the return result **OK** or **ERROR** is 40 seconds. No AT command can be executed until the result is returned.

AT+QIDEACT to activate **PDP** scene	
test command **AT+QIDEACT=?**	response **+QIDEACT: (supported <contextID> range)** **OK**
The set command activates the specified PDP scene **AT+QIDEACT=<contextID>**	response **OK** In case of error: **ERROR**
maximum response time	Affected by network status, the maximum response time is 40 seconds.
Feature Description	/

parameter

<contextID>	Integer. scene ID. Range: 1~15.

2.3.5. AT+QIOPEN to open Socket service

This command is used to open the Socket **service. The service type can be specified by <service_type>, and the data access mode (cache mode, straight-through mode and transparent mode) can be configured through <access_mode>, URC +QIOPEN** will indicate whether the Socket service is opened successfully.

1. **If <service_type> is "TCP** LISTENER", then the module works as a TCP server. After accepting a new TCP connection, **the module will automatically specify a <connectID> and report** URC **+QIURC: "incoming", <connectID>, <serverID>, <remoteIP>, <remote_port>; where the range of <connectID> is** 0 ~11; the request type of the new connection is "TCP INCOMING **", and its <access_mode> value** is the same as the parameter value of "TCP LISTENER".

2. **If <service_type> is "UDP SERVICE", you can** send and receive UDP data to remote IP via <local_port>.

   ÿ Send data: execute **AT+QISEND=<connectID>,<send_length>,<remoteIP>,<remote_port>.** ÿ Receive data in direct mode: the module will report URC **+QIURC: "recv",<connectID>,<currentrecvlength>, <remoteIP>,<remote_port><CR><LF><data>.**

   ÿ Receive data in cache mode: the module will report URC **+QIURC: "recv",<connectID>, and then the user can pass AT+QIRD=<connectID> read data.**

3. **Wait for the output of +QIOPEN: <connectID>,<err>, it is recommended to wait for** 150 seconds; if the URC is not received within 150 seconds, you need to call Close the Socket by **AT+QICLOSE .**

AT+QIOPEN to open the **Socket** service	
test command **AT+QIOPEN=?**	response: **+QIOPEN: (supported <contextID> range),(supported <connectID> range), "TCP/UDP/TCP LISTENER/UDP SERVICE", "<IP_ad dress>/<domain_name>",<remote_port>,<local_port >, (supported** <access_mode> range) **OK**
set command **AT+QIOPEN=<contextID>,<connectID>,<service_type>,"<IP_address>/<domain_name>",<remote_port>[,<local_port>[,<access_mode>]]**	Response: **If the data access mode is transparent mode (<access_mode>=2) and the service is opened successfully :** **CONNECT** In case of error: **ERROR** You can get the error description through **AT+QIGETERROR .** **If the data access mode is caching mode (<access_mode>=0) or straight-through mode (<access_mode>=1):** **OK** **+QIOPEN: <connectID>,<err>**

	<err> is 0 when the service is opened successfully , **otherwise <err> is not** 0.
maximum response time	Affected by network status, the maximum response time is 150 seconds.
Feature Description	/

parameter

<contextID>	Integer. scene ID. Range: 1~15. Integer.
<connectID>	Socket connection number. Range: 0~11.
<service_type> String type. Socket service type.	
	"TCP" Client establishes TCP connection
	"UDP" Client establishes UDP connection
	"TCP LISTENER" establishes a TCP server to listen for TCP connections
	"UDP SERVICE" establish UDP service
<IP_address> String type.	
	If <service_type> is "TCP" or "UDP", it is the IP address of the remote server, such as 220.180.239.212.
	If <service_type> is "TCP LISTENER" or "UDP SERVICE", please enter 127.0.0.1.
<domain_name> string type. The domain name address of the remote server.	
<remote_port>	Integer. Remote server port. Range: 0~65535. Only valid when <service_type> is "TCP" or "UDP". Integer. local port. Range: 0~65535. **If <service_type> is "TCP** LISTENER" or "UDPSERVICE", this parameter must be specified.
<local_port>	If <service_type> is "TCP" or "UDP", and <local_port> is 0, then the local port will be assigned automatically; otherwise, the local port will be specified.
<access_mode> integer. The data access mode of the Socket service. 0 buffer	mode 1 direct mode 2 transparent mode error code. Please refer to
<err>	No. 4 chapter

2.3.6. AT+QICLOSE closes the Socket service

This command is used to close a specific Socket service. Affected by the network status, after executing **AT+QICLOSE ,** the maximum time to wait for the return result **OK** or **ERROR** is 10 **seconds (this is the default value and can be modified by <timeout>).** No AT command **can be executed until the result is returned .**

AT+QICLOSE closes the **Socket** service	
test command **AT+QICLOSE=?**	response **+QICLOSE:** (supported <connectID> range), (supported <timeout> **range)**

	OK
set command **AT+QICLOSE=<connectID>[,<timeout>]**	The response closes successfully: **OK** Shutdown failed: **ERROR**
maximum response time	The default is 10 **seconds, depending on the value of <timeout>**
Feature Description	/

parameter

<connectID> integer. Socket connection number. Range: 0~11. **<timeout>** **integer. Output the timeout value of the response result. If** the FIN ACK **from other clients is not received within the <timeout> time range ,** the module will forcibly close the Socket. Range: 0~65535; Default value: 10; Unit: second.

2.3.7. AT+QISTATE query Socket service status

This command is used to query **the status of the Socket service. If <query_type> is** 0, return all existing Socket services in a specific scenario **status; if <query_type> is** 1, it returns the status of a specific Socket service.

AT+QISTATE query **Socket** service status	
Test command **AT+QISTATE=?**	response **OK**
Query/execute command **AT+QISTATE? or AT+QISTATE**	The response returns the status of all existing connections: **+QISTATE:** **<connectID>,<service_type>,<IP_address>,<remote_port>,<local_port>,<socket_state>,<contextID>,<s** **…** **OK**
Set the command **if <query_type> is** 0, query the connection status in a specific scenario **AT+QISTATE=<query_type>,<contex tID>**	The response returns the status of all existing connections for a particular scenario: **+QISTATE:** **<connectID>,<service_type>,<IP_address>,<remote_port>,<local_port>,<socket_state>,<contextID>,<s** **…**

	OK
Set command **If <query_type> is** 1, query the connection status of a specific Socket service **AT+QISTATE=<query_type>,<connect ID>**	response **+QISTATE:** **<connectID>,<service_type>,<IP_address>,<remote_port>,<local_port>,<socket_state>,<contextID>,** **OK**
maximum response time	300 milliseconds
Feature Description	/

parameter

<query_type> integer. query type. 0 Query the connection status of all Socket services in a specific scenario 1 Query the connection status of a specific Socket service **<contextID>** integer. scene ID. Range: 1~15. **<connectID>** integer. Socket connection number. Range: 0~11.

<service_type> String type. Socket service type.

"TCP"	Client establishes TCP connection
"UDP"	Client establishes UDP connection

"TCP LISTENER" establishes a TCP server to listen for TCP connections

"TCP INCOMING" establishes a TCP connection accepted by the TCP server

"UDP SERVICE" establish UDP service

<IP_address> String type. IP address.

If <service_type>="TCP" or "UDP", it refers to the remote server IP address. **If <service_type>="TCP** LISTENER" or "UDP SERVICE", it refers to the local IP address. **If <service_type>="TCP** INCOMING", then Refers to the remote client IP address **<remote_port>** integer. The remote port number. If <service_type>="TCP" or "UDP", it refers to the remote server port. **If <service_type>="TCP** LISTENER" or "UDP SERVICE", the port is invalid. **If <service_type>="TCP** INCOMING", it refers to the remote client port port **<local_port>** integer. The local port number. **If <local_port> is** 0, the local port is assigned automatically.

<socket_state> integer. Socket service status.

0 "Initial": The connection has not been established 1 "Opening": The client is connecting or the server is trying to listen 2 "Connected": The client connection has been established 3 "Listening": The server is listening 4 "Closing": The connection is disconnected **Integer . Only valid when <service_type> is "TCP** INCOMING". <serverID> represents which server accepted the TCP connection, this parameter value and the **<connectID> value of the server "TCP LISTENER"**

<serverID>

	Same.	
<access_mode> integer. Data access mode.		
	0 cache mode 1	
	passthrough mode	
	2 transparent	
<AT_port>	transmission mode string type. COM port of Socket	
	service. "usbmodem" USB modem port "usbat"	
		USB AT port
	"uart1"	UART port 1 MUX
	"cmux1"	port 1 MUX port 2
	"cmux2"	MUX port 3 MUX
	"cmux3"	port 4
	"cmux4"	

2.3.8. AT+QISEND to send data

If the data access mode of a specific Socket service is cache mode (<access_mode>=0) or straight-through mode **(<access_mode>=1), data can be sent via AT+QISEND .** If the data is successfully sent to the module, it will return **SEND OK,** otherwise it will return **SEND FAIL** or ERROR. SEND **FAIL** means that the sending buffer is full, and you can try to resend the data; ERROR means that an error was encountered during sending data, please send the data again after a delay for a period of time, the maximum length of the sent data is 1460 **bytes; SEND OK** does not mean that the data It has been successfully sent to the server, you can check whether the data has been sent to the server by **AT+QISEND=<connectID>,0 .**

AT+QISEND to send data	
Test command **AT+QISEND=?**	response **+QISEND: (supported <connectID> range), (supported <send_length** h> range) **OK**
Set the command if <service_type>="TCP", "UDP" or " TCP INCOMING", send variable length data **AT+QISEND=<connectID>**	response **>** After responding **> ,** enter the data to be sent; press **Ctrl + Z** to send the data, and press **Esc** to cancel sending. If the connection has been established and the sending is successful: **SEND OK** If the connection has been established, but the sending buffer is full: **SEND FAIL** If the connection is not established, disconnected abnormally or the parameters are incorrect: **ERROR**

Set the command **if <service_type>="TCP",** "UDP" or "T CP INCOMING", send fixed-length data **AT+QISEND=<connectID>,<send_len gth>**	response **>** After the response **> , enter data with a length equal to <send_length>.** If the connection has been established and the sending is successful: **SEND OK** If the connection has been established, but the sending buffer is full: **SEND FAIL** If the connection is not established, disconnected abnormally or the parameters are incorrect: **ERROR**
Set the command **if <service_type>="UDP** SERVICE" **AT+QISEND=<connectID>,<send_len** **gth>,<remoteIP>,<remote_port>**	Response to this command can be used to send fixed byte length data to a specific remote IP address and **remote port, <service_type>** must be "UDP SERVICE". **>** After the response **> , enter data with a length equal to <send_length>.** If the connection has been established and the sending is successful: **SEND OK** If the connection has been established, but the sending buffer is full: **SEND FAIL** If the connection is not established, disconnected abnormally or the parameters are incorrect: **ERROR**
Set the command **when <send_length> is** 0, you can query the sent data **AT+QISEND=<connectID>,0**	Response If the specified connection exists: **+QISEND: <total_send_length>,<ackedbytes>,<unackedbytes>** **OK** In case of error: **ERROR**
maximum response time	/
Feature Description	/

parameter

<connectID>	Integer. Socket connection number. Range:

0~11. **<send_length>** integer. Send data byte length. Range: 0~1460; unit: byte.

<remoteIP>	String type. Remote IP address (must be in dotted decimal notation). Only when <service_type> is "UDP SERVICE" is valid.
<remote_port>	**Integer. remote port.** Only valid **when <service_type> is "UDP** SERVICE".

<total_send_length> integer. Total length of sent data. Unit: byte.

<ackedbytes> integer. The total length of received data. Unit: byte.

<unackedbytes> integer. Total length of data not received. Unit: byte.

2.3.9. AT+QIRD read received TCP/IP data

In cache mode, after receiving the data, the module will first cache the data and report +QIURC: **"recv",<connectID>, and then pass AT+QIRD** reads the data.

When there is still data in the cache, if the module receives data again, it will not report the URC again; it will not report a new URC until all the data in the cache is read.

AT+QIRD read received **TCP/IP** data	
test command **AT+QIRD=?**	response **+QIRD: (supported <connectID> range), (supported <read_length> range)** **OK**
Set the command when <service_type> is "TCP"/"UDP"/"TCP INCOMING" **AT+QIRD=<connectID>[,<read_length h>]**	Response if data is received for a particular connection: **+QIRD: <read_actual_length><CR><LF><data>** **OK** If no data: **+QIRD: 0** **OK** If the connection does not exist: **ERROR**
Set the command **when <service_type> is "UDP** SERVICE" **AT+QIRD=<connectID>**	Response if data exists: **+QIRD: <read_actual_length>,<remoteIP>,<remote_port><CR><LF><data>**

	OK If no data: **+QIRD: 0** **OK** If the connection does not exist: **ERROR**
Set the command to query the read data **when <read_length> is 0** **AT+QIRD=<connectID>,0**	Response If the specified connection exists: **+QIRD: <total_receive_length>,<have_read_length>,<unread_length>** **OK** In case of error: **ERROR**
maximum response time	/
Feature Description	/

parameter

<connectID>	Integer. Socket connection number. Range:
<read_length>	0~11. Integer. The maximum length of data to be read. Range: 0~1500; unit: byte.
<read_actual_length> integer. The length of the actual received data. Unit: byte.	
<remoteIP>	String type. Remote IP **address. Only valid when <service_type> is "UDP** SERVICE". **Integer.**
<remote_port>	**remote port. Only valid if <service_type> is "UDP** SERVICE". Integer. read data. Unit: byte.
<data>	**<total_receive_length>** integer. Total length of received data. Unit: byte. **<have_read_length>** integer. The length of the read data. Unit: byte. Integer. The length of the unread data. Unit: byte. **<unread_length>**

2.3.10. AT+QISENDEX send hex string data

This command is used to send hexadecimal string data, not applicable to "UDP SERVICE" and "TCP LISTENER" two Socket service types.

AT+QISENDEX send hexadecimal string data	
Test command **AT+QISENDEX=?**	response **+QISENDEX:** (supported range of <connectID>), <hex_string> **OK**
set command **AT+QISENDEX=<connectID>,<hex_string>**	Response If the data is sent successfully: **SEND OK** If the sending buffer is full: **SEND FAIL** If the connection does not exist: **ERROR**
maximum response time	/
Feature Description	/

parameter

<connectID> integer. Socket connection number. Range: 0~11. **<hex_string>** String type. Hex string data. Maximum length: 512 bytes.

2.3.11. AT+QISWTMD switch data access mode

This command is used to switch the data access mode: cache mode, direct mode and transparent mode. When establishing the Socket service, you can specify the data access mode through <access_mode> of **AT+QIOPEN ,** after the Socket service is opened, you can switch the mode through **AT+QISWTMD .**

AT+QISWTMD switch data access mode	
test command **AT+QISWTMD=?**	response **+QISWTMD:** (supported <connectID> range), (supported <access_mode> **range)** **OK**
set command	response

AT+QISWTMD=<connectID>,<access_mode>	**If <access_mode>=0** or 1 and the data access mode is switched successfully: **OK** **If <access_mode>=2** and the data access mode is switched successfully, the module will enter the data mode: **CONNECT** In case of error: **ERROR**
maximum response time	/
Feature Description	This command takes effect immediately; the parameter configuration is not saved.

parameter

<connectID>	Integer. Socket connection number. Range: 0~11.
<access_mode>	integer. Connection data access mode. 0 cache mode 1 direct mode 2 transparent mode

2.3.12. AT+QPING for remote server Ping operation

This command is used to check the reachability of Host network protocol. Before using the Ping tool, the Host should first be activated by **AT+QIACT**

For the scene corresponding to <contextID>, the result will be returned within the period of <timeout>, and the default value of <timeout> is 4 seconds.

AT+QPING for remote server **Ping** operation	
Test command **AT+QPING=?**	response **+QPING: (supported <contextID> range), <host>, (supported <timeout> range), (supported <pingnum> range)** **OK**
set command **AT+QPING=<contextID>,<host>[,<ti meout>[,<pingnum>]]**	Response If the remote server Ping operation is successful: **OK** **+QPING: <result>,<IP_address>,<bytes>,<time>,<ttl>** **…** **+QPING: <finresult>,<sent>,<rcvd>,<lost>,<min>,<max>,** **<avg>**

	In case of error: **ERROR**
maximum response time	/
Feature Description	/

parameter

<contextID> integer. scene ID. Range: 1~15. String type.

<host> <timeout> integer. Host address, in the format of domain name or dotted decimal IP address. Set the maximum time to wait for each ping request response. Range: 1~255; Default value: 4; Unit: second. **<pingnum>** integer. Set the maximum number of Ping requests. Range: 1~10; Default value: 4. Integer. Results of each ping request. **<result>** Ping response received from server , **followed by** server IP address. The format is dotted decimal. Integer. Send Ping request byte length. Unit: byte. **<IP_address>** **<bytes>** **<time>** **<ttl>** 0 For other values, please refer to **<IP_address>** string type. Wait for response packet time after sending a ping request. Unit: milliseconds. Integer. Remote response packet TTL value. **<ttl>** **<finresult>** integer. The final result after the **<time>** integer. Ping request is executed. No. 4 chapter

	0	It ends normally, the scene is successfully activated and **the host is found. In this case, <sent>, <rcvd>, <lost>, <min>, <max>, <avg>** other value error codes follow, please refer to the integer type. The number of times a ping request is sent.
<sent>	Integer. The number of ping requests that got a response. No. 4 chapter	
<rcvd>	Integer. The number of ping requests that timed out.	
<lost>	Integer. Minimum response time. Unit: milliseconds.	
<min>	Integer. Maximum response time. Unit: milliseconds.	
<max>	Integer. average response time. Unit: milliseconds.	
<avg>		

2.3.13. AT+QNTP use NTP server to synchronize local time

This command is used to synchronize the local time to UTC time through the NTP server. Before the synchronization, the Host needs to be activated **by AT+QIACT** The scene corresponding to <contextID>. Affected by the network status, it takes up to 125 seconds to wait for the returned result.

AT+QNTP use **NTP** server to synchronize local time	
test command **AT+QNTP=?**	response **+QNTP: (supported <contextID> range), <server>, (supported <port> range),** (supported <autosettime> list) **OK**
query command	response

AT+QNTP?	If during synchronization time: **+QNTP: <server>,<port>** **OK**
set command **AT+QNTP=<contextID>,<server>** **[,<port>[,<autosettime>]]**	Response sync successfully: **OK** **+QNTP: <err>,<time>** In case of error: **ERROR**
maximum response time	Affected by network status, the maximum response time is 125 seconds.
Feature Description	This command takes effect immediately; the parameter configuration is not saved.

parameter

<contextID> integer. scene ID. Range: 1~15.

<server> String type. NTP server address. Integer.

NTP server port. Range: 1~65535. **<port>** integer. Whether to automatically set the synchronization time to the local time. 0 not set 1 set error code. Please refer to the string type. The time synchronized from the NTP server. The format is YYYY/MM/DD,hh:mm:ssÿzz, the range of zz: -48 ~ +56.

<err> No. *4* chapter

<time>

2.3.14. **AT+QIDNSCFG** configure **DNS** server address

This command is used to configure DNS server address. Before configuring DNS address, Host needs **to activate <contextID>** through **AT+QIACT** corresponding scene.

AT+QIDNSCFG Configure **DNS** server address	
test command **AT+QIDNSCFG=?**	response: **+QIDNSCFG: (supported <contextID> scope),<pridnsaddr>,<secdns addr>** **OK**
set command **AT+QIDNSCFG=<contextID>[,<pridnsaddr>[,<secdnsaddr>]]**	Response: If the optional parameter is omitted, query the current DNS server address of the specified PDP scenario: **+QIDNSCFG: <contextID>,<pridnsaddr>,<secdnsaddr>**

	OK If any optional parameter is specified, set the primary DNS server address and secondary DNS server address of the specified PDP scenario: **OK** In case of error: **ERROR**
maximum response time	/
Feature Description	This command takes effect immediately; the parameter configuration is not saved.

parameter

<contextID>	Integer. PDP scene ID. Range: 1~15.
<pridnsaddr> String type. Primary DNS server address. **<secdnsaddr>** String type. Secondary DNS server address.	

2.3.15. AT+QIDNSGIP Use domain name to obtain IP address

Before querying DNS, Host needs to activate the context corresponding to <contextID> through **AT+QIACT .** Affected by the network status, wait for the return

It may take up to 60 seconds for the response to be returned.

.AT+QIDNSGIP Use the domain name to obtain **the IP** address	
Test command **AT+QIDNSGIP=?**	response: **+QIDNSGIP: (supported range of <contextID>),<hostname>** **OK**
set command **AT+QIDNSGIP=<contextID>,<hostname>**	response: **OK** In case of error: **ERROR** Return result in URC format **+QIURC: "dnsgip",<err>,<IP_count>,<DNS_ttl> [...]** **+QIURC: "dnsgip",<hostIPaddr>]**
maximum response time	Affected by network status, the maximum response time is 60 seconds.
Feature Description	/

parameter

<contextID> integer. PDP scene ID. Range: 1~15. **<hostname>** string type.

domain name. error code. Please refer to

<err> No. 4 chapter.

<IP_count> integer. The number of IP addresses **corresponding to <hostname> .**

<DNS_ttl> integer. DNS TTL value. Unit: seconds. **<hostIPaddr> string type.**

<hostname>IP address.

2.3.16. AT+QISDE controls whether to echo the data to be sent by AT+QISEND

This command is used to control whether to echo the data to be sent by **AT+QISEND ,** that is, whether to echo the input data to be sent.

AT+QISDE controls whether to echo the data to be sent by **AT+QISEND**	
test command **AT+QISDE=?**	response: **+QISDE: (list of supported <echo>s)** **OK**
query command **AT+QISDE?**	response: **+QISDE: <echo>** **OK**
set command **AT+QISDE=<echo>**	response: **OK** In case of error: **ERROR**
maximum response time	300 milliseconds
Feature Description	This command takes effect immediately; the parameter configuration is not saved.

parameter

<echo> Integer. Whether to echo the data to be sent by **AT+QISEND .**

0 do not echo 1

echo

2.3.17. AT+QIGETERROR query last AT command error code

If **ERROR** is returned after executing the TCP/IP command , the error details can be queried through **AT+QIGETERROR .** Note that **AT+QIGETERROR** only returns the last TCP/IP AT command error code.

AT+QIGETERROR query last **AT** command error code	
Test command **AT+QIGETERROR=?**	response: **OK**
Excuting an order **AT+QIGETERROR**	response: **+QIGETERROR: <err>,<errcode_description>** **OK**
maximum response time	300 milliseconds
Feature Description	/

parameter

<err>	error code. Please refer to **<errcode_description>** string type. Indicates detailed error information. Please refer to No. 4 chapter. No. 4 chapter.

2.4. URC description

The URC of the TCP/IP AT command is uniformly reported to the Host **in the form of +QIURC:,** involving data reception, connection disconnection and access, etc. at the URC

There are <CR><LF> before and after , which are not displayed in this article.

2.4.1. +QIURC: "closed" connection disconnection notification

When the TCP Socket service connection is disconnected by the remote client or disconnected due to network abnormality, the module will report the URC, and the Socket **will be in the "Closing" state (<socket_state>=4), call AT+QICLOSE=<connectID> Afterwards,** the Socket service can be restored to the "Initial" state.

+QIURC: "closed" connection disconnection notification	
+QIURC: "closed",<connectID>	Socket service connection is disconnected

parameter

<connectID>	Integer. Socket connection number. Range: 0~11.

2.4.2. +QIURC: "recv" data receiving notification

In cache mode or direct mode, the module will report URC to notify Host after receiving data.

ÿ In cache mode: URC **format is +QIURC: "recv",<connectID>;** after URC is reported, Host can read data through **AT+QIRD .** Please note that if the cache is not empty and the module receives data again, the module will report a new URC only after the Host reads all the received data through **AT+QIRD .** ÿ In the direct output mode: the received data will be directly output from the COM port.

+QIURC: "recv" data receiving notification	
+QIURC: "recv",<connectID>	In the cache mode, the module will report the URC when it receives the data, and the Host can obtain the data through **AT+QIRD .**
+QIURC: "recv",<connectID>,<currentrecvlength><CR><LF><data>	**In direct mode, when <service_type>="TCP", "UDP" or "TCP** INCOMING", the module will report the URC when it receives data.
+QIURC: "recv",<connectID>,<currentrecvlength>,<remoteIP>,<remote_port><CR><LF><data>	**In direct mode, when <service_type>="UDP** SERVICE", the module will report the URC when it receives data.

parameter

<connectID>	Integer. Socket connection number. Range: 0~11.
<currentrecvlength>	integer. The length of the actual received data. String type.
<remoteIP>	Remote IP address. Integer. remote port.
<remote_port>	Integer. read data. Unit: byte.
<data>	

2.4.3. +QIURC: "incoming full" client connection full notification

If the client connection has reached the limit, or there are no Socket system resources available for allocation, the module will report **URC: +QIURC: "incoming full"** when there is a new client connection request .

+QIURC: "incoming full" client connection full notification	
+QIURC: "incoming full"	Client connection request is full

2.4.4. +QIURC: "incoming" client connection notification

If <service_type> is "TCP LISTENER", when a remote client connects to this server, Host will **automatically assign an idle <connectID> to the new connection, where <connectID> ranges from** 0 to 11. At this time, the module will report the URC. The <service_type> of the new connection **is "TCP INCOMING", and the <access_mode> is the caching mode.**

+QIURC: "incoming" client connection notification	
+QIURC: "incoming",<connectID>,<serverID>,<remoteIP>,<remote_port>	When <serverID> accepts a new connection request, the <connectID> assigned to the new connection **and the <remoteIP> and <remote_port> of the new connection will be** reported through this URC.

parameter

<connectID> integer. The module automatically specifies the assigned Socket service for client connections. Range: 0~11.

<serverID> Integer. The server that accepts client connections to <connectID>. The <service_type> of the server is "TCP LISTENER", and the listening Socket ID **is <serverID>. <remoteIP> string type.** The remote IP address **of the client connection <connectID> . <remote_port> integer. The remote port where the client connects to <connectID>.**

2.4.5. +QIURC: "pdpdeact" PDP deactivation notification

PDPs can be deactivated by the network. After the PDP is deactivated, the module will report the URC to notify the Host, and the Host needs to execute AT+QIDEACT Deactivate the scene and reset all connections.

+QIURC: "pdpdeact" PDP deactivation notification
+QIURC: "pdpdeact",<contextID> indicates that the <contextID> scene is deactivated

parameter

<contextID> integer. scene ID. Range: 1~15.

3 examples

3.1. Scene configuration and activation

3.1.1. Scene configuration

```
AT+QICSGP=1,1,"UNINET","","",1          //Configure Scenario 1, APN is configured as "UNINET" (China Unicom).
OK
```

3.1.2. Scene activation

```
AT+QIACT=1                              //Activate scene 1, affected by the network status, the maximum response time is 150 seconds.

OK                                      //The scene is activated
AT+QIACT?                               successfully. //Query the state of the scene.
+QIACT: 1,1,1,"10.7.157.1"

OK
```

3.1.3. Scene deactivation

```
AT+QIDEACT=1                            //Deactivate scene 1. //
OK                                      Scene deactivation is successful, affected by the network status, the maximum response time is 40 seconds.
```

3.2. TCP client works in cache mode

3.2.1. Create a TCP client connection and enter cache mode

```
AT+QIOPEN=1,0,"TCP","220.180.239.212",8009,0,0      // The scene is 1, <connectID> is 0. Before executing AT+QIOPEN, Host needs to use AT+QIACT to activate the scene.

OK
```

```
+QIOPEN: 0,0                                  //TCP client connection is successful, wait for URC: +QIOPE N: <connectID>, <err> response, it is recommended to wait for 150 seconds. If no URC response is received within 150 seconds, the Host can use AT+QICLOSE to disconnect the Socket. //Check if the connection status of <connectID> is 0.
AT+QISTATE=1,0
+QISTATE: 0,"TCP","220.180.239.212",8009,65514,2,1,0,0,"usbmodem"

OK
```

3.2.2. Send data in **Buffer** mode

```
AT+QISEND=0                                   //Send variable length data.
> test1<ctrl+Z>
SEND OK                                       //SEND OK does not mean that the data has been successfully sent to the server. Host can communicate Use AT+QISEND=0,0 to check whether the data has reached the server.
AT+QISEND=0,4                                 //Send fixed-length data, the byte length is 4 bytes.
> test
SEND OK
AT+QISEND=0,0                                 //Query the length of the sent data.
+QISEND: 9,9,0

OK
AT+QISENDEX=0,"3132333435"                    //Send hexadecimal string data.
SEND OK
AT+QISEND=0,0                                 //Query the sent data length, confirmed data length and unconfirmed data length.
+QISEND: 14,14,0

OK
```

3.2.3. Receive data from remote server in Buffer mode

```
+QIURC: "recv",0                              // <connectID> is 0 to receive data. //Read
AT+QIRD=0,1500                                data, the byte length is 1500 bytes. //The actual
+QIRD: 14                                     received data length is 14 bytes.
test1

OK
AT+QIRD=0,1500
+QIRD: 0                                      // There is no data in the cache.

OK
AT+QIRD=0,0                                   //Query the total length of received data, including read and unread data.
```

```
+QIRD: 14,14,0

OK
```

3.2.4. Disconnect

```
AT+QICLOSE=0                    //Disconnect the connection whose <connectID> is 0. Affected by network status, the maximum response time is 10 seconds.
OK
```

3.3. The TCP client works in transparent transmission mode

3.3.1. Create a TCP client connection and enter the transparent transmission mode

```
AT+QIOPEN=1,0,"TCP","220.180.239.212",8009,0,2 //Scene is 1, <connectID> is 0. Before executing AT+QIOPEN , Host needs to use AT+QIACT
                                               to activate the scene. //The connection is successful, wait for the
CONNECT                                        URC response result CONNECT is recommended to wait for 150
                                               seconds. If there is no response from URC within 150 seconds, the
                                               Host can disconnect the Socket through AT+QICLOSE .
```

3.3.2. Send data in transparent transmission mode

```
<All data got from COM port will be sent to internet directly>
```

3.3.3. Transparent transmission mode to receive remote data

```
Test 1                          //All data received from the network will be output directly through the COM port.
```

3.3.4. Disconnect the TCP client connection

```
AT+QICLOSE=0                    // After using +++ to exit the transparent transmission mode, the Host can disconnect
                                the TCP connection through AT+QICLOSE . Affected by network status, the maximum
                                response time is 10 seconds.
OK
```

3.4. TCP client works in direct mode

3.4.1. Create a TCP client connection and enter the straight-through mode

```
AT+QIOPEN=1,0,"TCP","220.180.239.212",8009,0,1 //Scene is 1, <connectID> is 0. Before executing AT+QIOPEN , Host needs to use
                                                AT+QIACT to activate the scene.
OK

+QIOPEN: 0,0                                    //The connection is successful, wait for URC: +QIOPEN: <connectID>,<err> It
                                                is recommended to wait for 150 seconds for the response, if there is no
                                                response within 150 seconds, the Host can disconnect the Socket through
AT+QISTATE=1,0                                  AT+QICLOSE . //Query whether the connection status of <connectID> is 0.
+QISTATE: 0,"TCP","220.180.239.212",8009,65344,2,1,0,1,"usbmodem"

OK
```

3.4.2. Send data in direct mode

```
AT+QISEND=0                                     //Send variable length data.
> test1<ctrl+Z>
SEND OK                                         //SEND OK does not mean that the data has been successfully sent to the server,
                                                the Host can check whether the data has reached the server through
AT+QISEND=0,5 >                                 AT+QISEND=0,0 . //Send fixed-length data, the length is 5 bytes.
test2
SEND OK
AT+QISEND=0,0                                   //Query the sent data length, confirmed data length and unconfirmed data length.
+QISEND: 10,10,0

OK
```

3.4.3. Receive remote server-side data in direct mode

```
+QIURC: "recv",0,4                              //Receive remote server-side data.
test
```

3.4.4. Disconnect the TCP client connection

```
AT+QICLOSE=0                                    //Disconnect the connection whose <connectID> is 0. Affected by network status, the
                                                maximum response time is 10 seconds.
OK
```

3.5. TCP server works in cache mode

3.5.1. Start TCP server

```
AT+QIOPEN=1,1,"TCP LISTENER","127.0.0.1",0,2020,0 // The scene is 1 Before , <connectID> is 1. Executing Host
                                                  AT+QIOPEN , AT+QIACT is needs to pass
                                                  used to activate the scene.
OK

+QIOPEN: 1,0                                      //TCP server started successfully. //
AT+QISTATE=0,1                                    Query whether the connection status of <contextID> is 1.
+QISTATE: 1,"TCP LISTENER","10.7.157.1",0,2020,3,1,1,0,"usbmodem"

OK
```

3.5.2. Accept the connection request from the client

```
+QIURC: "incoming",11,1,"172.31.242.222",54091 //Accept a TCP connection, <service_type> is "TCP
                                                incoming", <connectID> is 11.
```

3.5.3. Received client data

```
+QIURC: "recv",11                                 // Data received from the remote client
AT+QIRD=11,1500                                   connection. //Read the data received by the client
+QIRD: 4                                          connection. //The actual data length is 4 bytes.
test

OK
AT+QIRD=11,1500
+QIRD: 0                                          // There is no data in the cache.

OK
AT+QIRD=11,0                                      //Query the total length of received data, including read and unread data.
+QIRD: 4,4,0

OK
```

3.5.4. Disconnect TCP server connection

```
AT+QICLOSE=11                                   //Disconnect the client connection, the maximum response time is 10 seconds affected by the network status.
OK
AT+QICLOSE=1                                    //Close the TCP server listening.
OK
```

3.6. UDP service example

3.6.1. Open UDP service

```
AT+QIOPEN=1,2,"UDP SERVICE","127.0.0.1",0,3030,0 //Open UDP service , <connectID> is 2, <contextID> is 1, before executing
                                                 AT+QIOPEN , Host needs Activate the scene by
                                                 AT+QIACT .
OK

+QIOPEN: 2,0                                     //Successfully opened the UDP
AT+QISTATE=0,1                                   service. //Query whether the connection status of <contextID> is 1.
+QISTATE: 2,"UDP SERVICE","10.7.157.1",0,3030,2,1,2,0,"usbmodem"

OK
```

3.6.2. Send UDP data to the server

```
AT+QISEND=2,10,"10.7.89.10",6969                 //Send data with a byte length of 10 bytes to the remote end, the
                                                 remote end IP address is 10.7.89.10, and the port is 6969.
>1234567890
SEND OK
```

3.6.3. Receive remote data

```
+QIURC: "recv",2                                 //Receive remote data. //
AT+QIRD=2                                        Read UDP data. Can output a whole UDP packet, no
                                                 The read length needs to be specified.
+QIRD: 4,"10.7.76.34",7687                       //The data length is 4, the remote IP address is 10.7.76.34, and the
                                                 remote port is 7687.
AAAA

OK
```

```
AT+QIRD=2                                   // read data. // There
+QIRD: 0                                    is no data in the cache.

OK
AT+QISEND=2,10,"10.7.76.34",7687            //Send data to the remote end, the remote end IP address is 10.7.76.34,
                                            and the remote port is 7687.
>1234567890
SEND OK
```

3.6.4. Close **UDP** service

```
AT+QICLOSE=2                                //Close the service.
OK
```

3.7. PING

```
AT+QPING=1,"www.baidu.com"                  //Ping www.baidu.com in scene 1, before pinging the target IP
                                            address, the Host needs to activate the scene through AT+QIACT .
OK

+QPING: 0,"61.135.169.125",32,192,255

+QPING: 0,"61.135.169.125",32,240,255

+QPING: 0,"61.135.169.125",32,241,255

+QPING: 0,"61.135.169.125",32,479,255

+QPING: 0,4,4,0,192,479,288
```

3.8. Synchronize local time

```
AT+QNTP=1,"202.112.10.36",123          //Use the NTP server "202.112.10.36:123" to synchronize the time. Before
                                       synchronizing the time, the Host needs to activate the scene through AT+QIACT .
OK

+QNTP: 0,"2019/09/09,01:32:42+32"
AT+CCLK?
+CCLK: "19/09/09, 01:32:52+32"

OK
```

3.9. Get last error code example

```
AT+QIOPEN=1,"TCP","220.180.239.212",8009,0,1 //Open Socket service, omit <connectID>.
ERROR
AT+QIGETERROR
+QIGETERROR: 552, invalid parameters

OK
```

4 error codes

When the AT command of TCP/IP returns **ERROR ,** the specific error type can be queried by **AT+QIGETERROR .** Note **AT+QIGETERROR** queries the error code of the last TCP/IP AT command.

Table 3: List of Error Codes

<err>	<errcode_description>	Chinese meaning
0	Operation success	Successful operation
550	Unknown error	unknown mistake
551	Operation blocked	operation blocked
552	Invalid parameters	invalid parameter
553	Memory not enough	Not enough storage
554	Socket creation failed	Failed to create socket
555	Operation not supported	operation not supported
556	Socket bind failed	Socket binding failed
557	Socket listen failed	Socket monitoring failed
558	Socket write failed	Socket write failed
559	Socket read failed	Socket read failed
560	Socket accept failed	Socket accept failure
561	PDP Context opening failed	Failed to open PDP scene
562	PDP context closure failed	Failed to close PDP scene
563	Socket Identity has been used	Socket ID is occupied
564	DNS busy	DNS busy
565	DNS parse failed	DNS resolution failed

566	Socket connect failed	Socket connection failed
567	Socket has been closed	Socket is closed
568	Operation busy	busy operation
569	Operation timeout	operation timed out
570	PDP context broken down	Faulty in PDP scene
571	Cancel sending	cancel send
572	Operation not allowed	operation not allowed
573	APN not configured	APN not configured
574	port busy	port busy

5 Appendix Reference Documents and Terminology Abbreviations

Table 4: Reference Documents

file name
[1] Quectel_LTE_Standard(A) Series_AT Command Manual

Table 5: Terminology Abbreviations

Abbreviation	English full name	Chinese full name
3GPP	3rd Generation Partnership Project	3rd Generation Partnership Project
ACK	acknowledgment	confirmation character
APNs	Access Point Name	access point name
ASCII	American Standard Code for Information Interchange American Standard Code for Information Interchange	
CHAP	Challenge Handshake Authentication Protocol	Challenge Handshake Authentication Protocol
CS	Circuit Switching	circuit switching
DNS	Domain Name System	domain name system
FIN	finish	Finish
ID	Identifier	identifier
IP	Internet Protocol	internet protocol
NTP	Network Time Protocol	Network Time Protocol
NV	Non-Volatile	non-volatile
PAP	Password Authentication Protocol	password authentication protocol
PDPs	Packet Data Protocol	packet data protocol
PPP	Point-to-Point Protocol	peer-to-peer protocol

P.S.	Packet Switching	packet switching
QoS	Quality of Service	service quality
TCP	Transmission Control Protocol	transmission control protocol
TTL	Time to Live	survival time
UART	Universal Asynchronous Receiver& Transmitter Universal Asynchronous Receiver & Transmitter	
UDP	User Datagram Protocol	UDP
URC	Unsolicited Result Code	unsolicited result code
USB	Universal Serial Bus	Universal Serial Bus
(U)SIM	(Universal) Subscriber Identity Module	(Generic) User Identity Module
UTC	Coordinated Universal Time	UTC