EC200x-CN&EC800x-CN TCP/IP Application Note

LTE Standard Module Series

Version: 1.1

Date: 2023-01-13

Status: Released

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

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

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

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

Legal Notices

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

Use and Disclosure Restrictions

License Agreements

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

Copyright

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

Trademarks

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

Third-Party Rights

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

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

Privacy Policy

To implement module functionality, certain device data are uploaded to Quectel's or third-party's servers, including carriers, chipset suppliers or customer-designated servers. Quectel, strictly abiding by the relevant laws and regulations, shall retain, use, disclose or otherwise process relevant data for the purpose of performing the service only or as permitted by applicable laws. Before data interaction with third parties, please be informed of their privacy and data security policy.

Disclaimer

a) We acknowledge no liability for any injury or damage arising from the reliance upon the information.
b) We shall bear no liability resulting from any inaccuracies or omissions, or from the use of the information contained herein.
c) While we have made every effort to ensure that the functions and features under development are free from errors, it is possible that they could contain errors, inaccuracies, and omissions. Unless otherwise provided by valid agreement, we make no warranties of any kind, either implied or express, and exclude all liability for any loss or damage suffered in connection with the use of features and functions under development, to the maximum extent permitted by law, regardless of whether such loss or damage may have been foreseeable.
d) We are not responsible for the accessibility, safety, accuracy, availability, legality, or completeness of information, advertising, commercial offers, products, services, and materials on third-party websites and third-party resources.

About the Document

Revision History

Version	Date	Author	Description
-	2020-07-24	Larson LI	Creation of the document
1.0	2023-01-11	Larson LI	First official release
1.1	2023-01-13	Larson LI	Added the applicable module: EC800M-CN.

Contents

Table Index

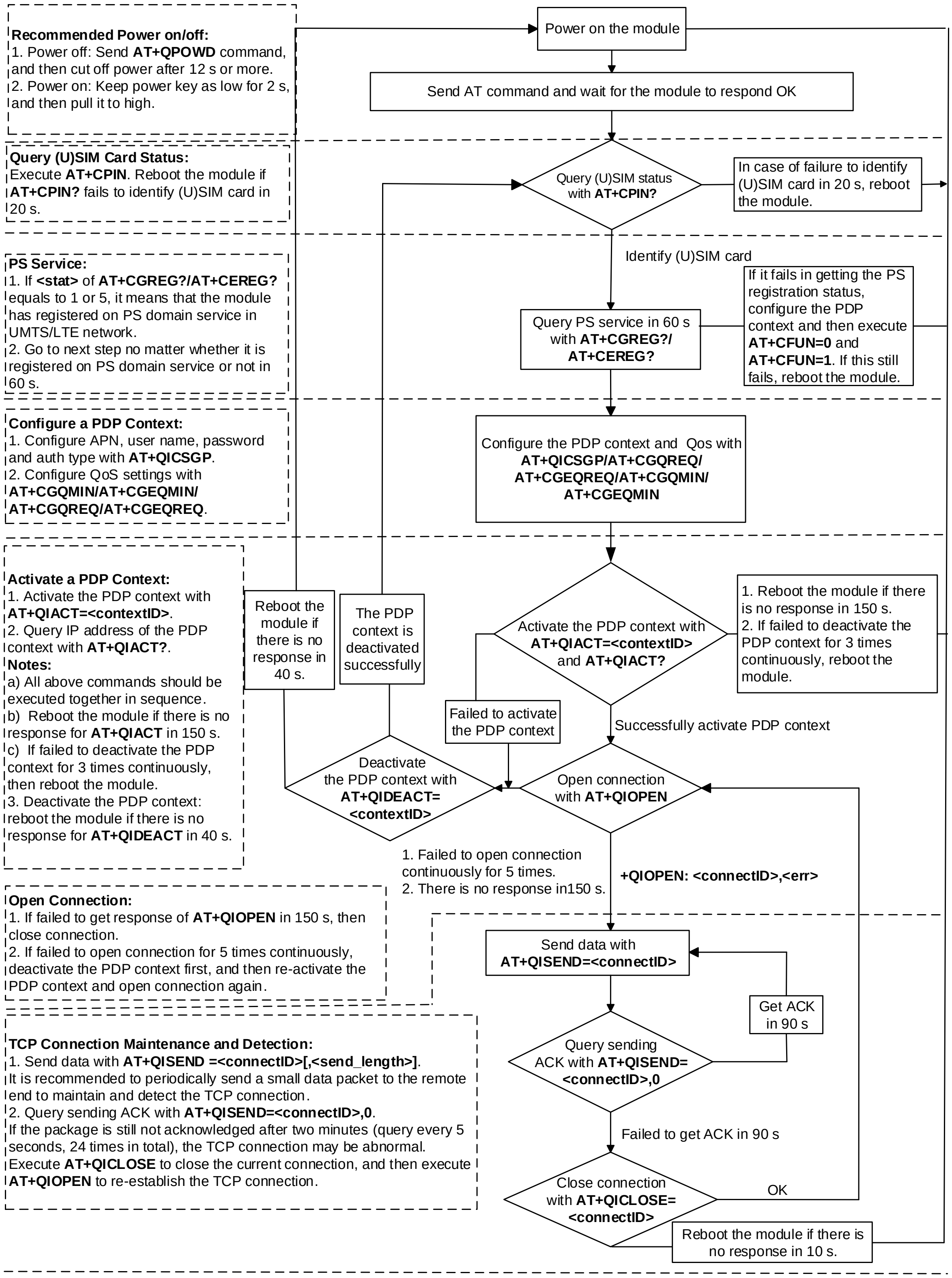
1 Introduction

Quectel LTE Standard EC200M-CN, EC200N-CN, EC800M-CN and EC800N-CN modules feature an embedded TCP/IP stack, which enables the host to access the Internet directly via AT commands. This significantly reduces the dependence on external PPP and TCP/IP protocol stacks and thus minimizes costs.

EC200M-CN, EC200N-CN, EC800M-CN and EC800N-CN modules provide socket services such as TCP client, UDP client, TCP server and UDP server.

1.1. The Process of Using TCP/IP AT Commands

Via TCP/IP AT commands, the host can configure a PDP context, activate/deactivate the PDP context, start/close a socket service and send/receive data via the socket service. The following figure illustrates how to use TCP/IP AT commands.


Power on the module
Send AT command and wait for the module to respond OK
Query (U)SIM status with AT+CPIN?
In case of failure to identify (U)SIM card in 20 s, reboot the module.
Identify (U)SIM card
Query PS service in 60 s with AT+CGREG?/ AT+CEREG?
If it fails in getting the PS registration status, configure the PDP context and then execute AT+CFUN=0 and AT+CFUN=1. If this still fails, reboot the module.
Configure the PDP context and Qos with AT+QICSGP/AT+CGQREQ/ AT+CGEQREQ/AT+CGQMIN/ AT+CGEQMIN
Reboot the module if there is no response in 40 s.
The PDP context is deactivated successfully
Activate the PDP context with AT+QIACT=<contextID> and AT+QIACT?
1. Reboot the module if there is no response in 150 s.
2. If failed to deactivate the PDP context for 3 times continuously, reboot the module.
Failed to activate the PDP context
Successfully activate PDP context
Deactivate the PDP context with AT+QIDEACT= <contextID>
Open connection with AT+QIOPEN
1. Failed to open connection continuously for 5 times.
2. There is no response in150 s.
+QIOPEN: <connectID>,<err>
Send data with AT+QISEND=<connectID>
Get ACK in 90 s
Query sending ACK with AT+QISEND= <connectID>,0
Failed to get ACK in 90 s
Close connection with AT+QICLOSE= <connectID>
OK
Reboot the module if there is no response in 10 s.

Recommended Power on/off:
1. Power off: Send **AT+QPOWD** command, and then cut off power after 12 s or more.
2. Power on: Keep power key as low for 2 s, and then pull it to high.

Query (U)SIM Card Status:
Execute **AT+CPIN**. Reboot the module if **AT+CPIN?** fails to identify (U)SIM card in 20 s.

PS Service:
1. If **<stat>** of **AT+CGREG?/AT+CEREG?** equals to 1 or 5, it means that the module has registered on PS domain service in UMTS/LTE network.
2. Go to next step no matter whether it is registered on PS domain service or not in 60 s.

Configure a PDP Context:
1. Configure APN, user name, password and auth type with **AT+QICSGP**.
2. Configure QoS settings with **AT+CGQMIN/AT+CGEQMIN/ AT+CGQREQ/AT+CGEQREQ**.

Activate a PDP Context:
1. Activate the PDP context with **AT+QIACT=<contextID>**.
2. Query IP address of the PDP context with **AT+QIACT?**.
Notes:
a) All above commands should be executed together in sequence.
b) Reboot the module if there is no response for **AT+QIACT** in 150 s.
c) If failed to deactivate the PDP context for 3 times continuously, then reboot the module.
3. Deactivate the PDP context: reboot the module if there is no response for **AT+QIDEACT** in 40 s.

Open Connection:
1. If failed to get response of **AT+QIOPEN** in 150 s, then close connection.
2. If failed to open connection for 5 times continuously, deactivate the PDP context first, and then re-activate the PDP context and open connection again.

TCP Connection Maintenance and Detection:
1. Send data with **AT+QISEND =<connectID>[,<send_length>]**.
It is recommended to periodically send a small data packet to the remote end to maintain and detect the TCP connection.
2. Query sending ACK with **AT+QISEND=<connectID>,0**.
If the package is still not acknowledged after two minutes (query every 5 seconds, 24 times in total), the TCP connection may be abnormal. Execute **AT+QICLOSE** to close the current connection, and then execute **AT+QIOPEN** to re-establish the TCP connection.

Notes:
1. Please note that users need to wait for the final response (for example "**OK", "CME ERROR", "CMS ERROR"**) of the last AT command you entered before you enter the next AT command. You can reboot the module if the module fails to get response in 60 s.
2. Reboot the module if the module has not got response of **AT+QIACT** in 150 s or response of **AT+QICLOSE** in 10 s and in 40 s.
3. It is NOT recommended to frequently reboot the module. When the module has been continuously rebooted for 3 times due to failed AT command execution, it can be rebooted immediately for the first time after that. If it still fails, reboot the module after 10 minutes for the second time, and reboot after 30 minutes for the third time, one hour for the fourth time, etc.

Figure 1: Flow Chart of Using TCP/IP AT Commands

1.2. Description of Data Access Modes

EC200M-CN, EC200N-CN, EC800M-CN and EC800N-CN modules support the following three kinds of data access modes:

- Buffer access mode
- Direct push mode
- Transparent transmission mode

When opening a socket service via **AT+QIOPEN**, the data access mode can be specified with **<access_mode>**. After a socket service is opened, the access mode can be switched with **AT+QISWTMD**.

1. In buffer access mode, data can be sent via **AT+QISEND**. If the module has received the data from the Internet, it will buffer them and report the URC: **+QIURC: "recv",<connectID>**. The host can retrieve the buffered data with **AT+QIRD**.

2. In direct push mode, data can be sent via **AT+QISEND**. If the module has received the data from the Internet, it will output them to COM port directly through the URC:
   +QIURC: "recv",<connectID>,<currectrecvlength><CR><LF><data> or **+QIURC: "recv",<connectID>,<currentrecvlength>,<remoteIP>,<remote_port><CR><LF><data>**.

3. In transparent transmission mode, the corresponding port (such as UART, USB modem port, etc.) is exclusively used for sending/receiving data directly to the Internet. The data received from COM port will be sent to the Internet directly, and the data received from Internet will be outputted via COM port directly.

   - **Exit transparent transmission mode**

   To exit the transparent transmission mode either:

   1) Execute **+++.** Follow the requirements below to prevent **+++** from being misinterpreted as data:
      a) Do not input any character for at least 1 s before and after inputting **+++**.
      b) Input only **+++** within 1 s, and wait until OK is returned. Once OK is retuned, the module switches to buffer access mode.

   OR

   2) Change MAIN_DTR from LOW to HIGH to make the module enter command mode. In this case, set **AT&D1** (see ***document [1]***) before the module enters transparent transmission mode.

   - **Return to transparent access mode**

   To return to transparent transmission mode either:

   1) Execute **AT+QISWTMD**. Before execution specify the **<access_mode>** as 2. Once transparent transmission mode is entered successfully, **CONNECT** is returned.

   OR

2) Execute **ATO**. After a connection exits from transparent transmission mode, executing **ATO** switches the data access mode back to transparent transmission mode. Once transparent transmission mode is entered successfully, **CONNECT** is returned. If no connection has entered transparent transmission mode, **ATO** returns **NO CARRIER**.

NOTE

1. In buffer access mode, if the buffer is not empty, and the module receives data again, it does not report any new URC until all the received data have been retrieved with **AT+QIRD** from the buffer.
2. In transparent transmission mode, AT commands cannot be executed. If the socket connection is closed because of network error or other errors, the module reports **NO CARRIER** and exits the transparent transmission mode. In this case, execute **AT+QICLOSE** to close the socket service.

2 TCP/IP AT Commands

This chapter describes AT commands related to TCP/IP.

2.1. AT Command Introduction

2.1.1. Definitions

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

2.1.2. AT Command Syntax

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

Table 1: Type of AT Commands

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

2.2. Declaration of AT Command Examples

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

2.3. AT Commands

2.3.1. AT+QICFG Configure Optional Parameters

This command configures optional parameters.

AT+QICFG Configure Optional Parameters	
Test Command **AT+QICFG=?**	Response **+QICFG: "transpktsize",(**range of supported **<transpktsize>**s**)** **+QICFG: "transwaittm",(**range of supported **<transwaittm>**s**)** **+QICFG: "dataformat",(**list of supported **<send_data_format>**s**),(**list of supported **<recv_data_format>**s**)** **+QICFG: "viewmode",(**list of supported **<view_mode>**s**)** **+QICFG: "passiveclosed",(**list of supported **<closed>**s**)** **+QICFG: "udp/readmode",(**list of supported **<mode>**s**)** **+QICFG: "udp/sendmode",(**list of supported **<mode>**s**)** **+QICFG: "tcp/keepalive",(**list of supported **<enable>**s**),(**range of supported **<idle_time>**s**),(**range of supported **<interval_time>**s**),(**range of supported **<probe_cnt>**s**)** **+QICFG: "recvind",(**list of supported **<show_length>**s**)** **+QICFG: "tcp/retranscfg",(**range of supported **<retran_times>**s**),(**range of supported **<retran_time>**s**)** **+QICFG: "tcp/accept",(**list of supported **<state>**s**)** **+QICFG: "send/buffersize",(**range of supported **<buffer_size>**s**)** **+QICFG: "send/auto",(**range of supported **<connectID>**s**),(**range of supported **<cycle_time>**s**),(<msg_auto>**data sent**)** **+QICFG: "recv/ignore",(**range of supported **<connectID>**s**),(**Filtered **<msg_ignore>** data**)** **+QICFG: "formatcfg",(**range of supported **<format>**s**)** **+QICFG: "qisend/timeout",(**range of supported **<timeout>**s**)** **+QICFG: "close/mode",(**range of supported **<close_mode>**s**)** **+QICFG: "sendinfo",(**list of supported **<send_view_mode>**s**)**

	+QICFG: "TCP/SendMode",(list of supported **<send_mode>**s**)** **+QICFG: "pdp/retry",(**list of supported **<pdpmode>**s**),(**list of supported **<ratmode>**s**),(**range of supported **<counts>**s**),(**range of supported **<retry_time>**s**)** **OK**
Write Command Query or set the packet size for transparent transmission mode **AT+QICFG="transpktsize"[,<transpktsize>]**	Response If the optional parameter is omitted, query the current setting: **+QICFG: "transpktsize",<transpktsize>** **OK** If the optional parameter is specified, set the packet size for transparent transmission mode: **OK** If there is any error: **ERROR**
Write Command Query or set the waiting time before automatically sending data in transparent transmission mode **AT+QICFG="transwaittm"[,<transwaittm>]**	Response If the optional parameter is omitted, query the current setting: **+QICFG: "transwaittm",<transwaittm>** **OK** If the optional parameter is specified, set the waiting time before automatically sending data in transparent transmission mode: **OK** If there is any error: **ERROR**
Write Command Query or set the format of the data to be sent or received (only for non-transparent mode) **AT+QICFG="dataformat"[,<send_data_format>,<recv_data_format>]**	Response If the optional parameters are omitted, query the current setting: **+QICFG: "dataformat",<send_data_format>,<recv_data_format>** **OK** If the optional parameters are specified, set the format of the data to be sent or received: **Ok** If there is any error: **ERROR**
Write Command Query or set the output format of received data (only for non-	Response If the optional parameter is omitted, query the current setting: **+QICFG: "viewmode",<view_mode>**

transparent mode) **AT+QICFG="viewmode"[,<view_mode>]**	**OK** If the optional parameter is specified, set the output format of the received data: **OK** If there is any error: **ERROR**
Write Command Enable or disable the passive close of TCP connection when the server is closed **AT+QICFG="passiveclosed"[,<closed>]**	Response If the optional parameter is omitted, query the current setting: **+QICFG: "passiveclosed",<closed>** **OK** If the optional parameter is specified, enable or disable the passive close of TCP connection: **OK** If there is any error: **ERROR**
Write Command Query or set the mode when reading UDP data **AT+QICFG="udp/readmode"[,<mode>]**	Response If the optional parameter is omitted, query the current setting: **+QICFG: "udp/readmode",<mode>** **OK** If the optional parameter is specified, set the UDP data read mode: **OK** If there is any error: **ERROR**
Write Command Query or set the mode when sending UDP data **AT+QICFG="udp/sendmode"[,<mode>]**	Response If the optional parameter is omitted, query the current setting: **+QICFG: "udp/sendmode",<mode>** **OK** If the optional parameter is specified, set the UDP data sending mode: **OK** If there is any error: **ERROR**

Write Command Query or set whether to send TCP keep-alive information. **AT+QICFG="tcp/keepalive"[,<enable>[,<idle_time>,<interval_time>,<probe_cnt>]]**	Response If the optional parameters are omitted, query the current configurations **+QICFG: "tcp/keepalive",<enable>[,<idle_time>,<interval_time>,<probe_cnt>]** **OK** If the optional parameters are entered, set whether to send TCP keep-alive information: **OK** If there is any error: **ERROR**
Write Command Enable or disable data length presence in URC reported after data reception in buffer mode **AT+QICFG="recvind"[,<show_length>]**	Response If the optional parameter is omitted, query the current setting: **+QICFG: "recvind",<show_length>** **OK** If the optional parameter is specified, enable or disable data length presence in URC reported: **OK** If there is any error: **ERROR**
Write Command Query or set the number of retransmissions and retransmission interval **AT+QICFG="tcp/retranscfg"[,<retran_times>,<retran_time>]**	Response If the optional parameter is omitted, query the current setting: **+QICFG: "tcp/retranscfg",<retran_times>,<retran_time>** **OK** If the optional parameter is specified, set the number of retransmissions and retransmission interval: **OK** If there is any error: **ERROR**
Write Command Query/enable or disable the automatic reception of the TCP connection from the client **AT+QICFG="tcp/accept"[,<state>]**	Response If the optional parameter is omitted, query the current setting: **+QICFG: "tcp/accept",<state>** **OK**

	If the optional parameter is specified, enable or disable the automatic reception of the TCP connection from the client: **OK** If there is any error: **ERROR**
Write Command Query or set the maximum length of a single transmission data **AT+QICFG="send/buffersize"[,<buffer_size>]**	Response If the optional parameter is omitted, query the current setting: **+QICFG: "send/buffersize",<buffer_size>** **OK** If the optional parameter is specified, set the maximum length of a single transmission data: **OK** If there is any error: **ERROR**
Write Command Query or set to send heartbeat packet data periodically **AT+QICFG="send/auto",<connectID>[,<cycle_time>[,<msg_auto>]]**	Response If the optional parameter is omitted, query the current setting: **+QICFG: "send/auto",<connectID>,<cycle_time>[,<msg_auto>]** **OK** If the optional parameter is specified, set to send heartbeat packet data periodically: **OK** If there is any error: **ERROR**
Write Command Query or set to filter the specified data **AT+QICFG="recv/ignore",<connectID>[,<msg_ignore>]**	Response If the optional parameter is omitted, query the current setting: **+QICFG: "recv/ignore",<connectID>[,<msg_ignore>]** **OK** If the optional parameter is specified, set to filter the specified data: **OK** If there is any error: **ERROR**
Write Command Query or set the format of sending	Response If the optional parameter is omitted, query the current setting:

AT+QISEND to return **>** **AT+QICFG="formatcfg"[,<format>]**	**+QICFG: "formatcfg",<format>** **OK** If the optional parameter is specified, set the output format of **>**: **OK** If there is any error: **ERROR**
Write Command Query or set the maximum response time for sending **AT+QISEND** **AT+QICFG="qisend/timeout"[,<timeout>]**	Response If the optional parameter is omitted, query the current setting: **+QICFG: "qisend/timeout",<timeout>** **OK** If the optional parameter is specified, set the timeout time after output **>**: **OK** If there is any error: **ERROR**
Write Command Query or set up asynchronous disconnection of TCP connection **AT+QICFG="close/mode"[,<close_mode>]**	Response If the optional parameter is omitted, query the current setting: **+QICFG: "close/mode",<close_mode>** **OK** If the optional parameter is specified, set up asynchronous disconnection of TCP connection: **OK** If there is any error: **ERROR**
Write Command Enable or disable displaying the return information of **AT+QISEND** or **AT+QISENDEX** in URC form **AT+QICFG="sendinfo"[,<send_view_mode>]**	Response If the optional parameter is omitted, query the current setting: **+QICFG: "sendinfo",<send_view_mode>** **OK** If the optional parameter is specified, enable or disable displaying the return information of **AT+QISEND** or **AT+QISENDEX** in URC form: **OK**

	If there is any error: **ERROR**
Write Command Set the mode when sending **SEND OK** **AT+QICFG="TCP/SendMode"[,<send_mode>]**	Response If the optional parameter is omitted, query the current setting: **+QICFG: "TCP/SendMode",<send_mode>** **OK** If the optional parameter is specified, set the mode when sending **SEND OK**: **OK** If there is any error: **ERROR**
Write Command Configure retry times and time of PDP activation and deactivation **AT+QICFG="pdp/retry",<pdpmode>,<ratmode>[,<counts>,<retry_time>]**	Response If the optional parameter is omitted, query the current setting: **+QICFG: "pdp/retry",<pdpmode>,<ratmode>,<counts>,<retry_time>** **OK** If the optional parameter is specified, configure retry times and time of PDP activation and deactivation: **OK** If there is any error: **ERROR**
Maximum Response Time	300 ms
Characteristic	This command takes effect immediately. The configurations will not be saved.

Parameter

<transpktsize>	Integer type. In transparent transmission mode, the max length of the data packet to be sent. Range: 1–1460. Default value: 1024. Unit: byte.
<transwaittm>	Integer type. In transparent transmission mode, the waiting time before sending the data automatically, if the length of data to be sent is less than the specified value of **<transpktsize>**. Range: 0–20. Default value: 2. Unit: 100 ms.
<send_data_format>	Integer type. Format of the data to be sent. Suffix "0x" is not needed when the mode is set as Hex mode as the module will automatically form two bytes to one ASCII code. 0 Text mode

	1 Hex mode
<recv_data_format>	Integer type. Format of the data to be received. Suffix "0x" is not needed when the mode is set as Hex mode as the module will automatically form two bytes to one ASCII code. 0 Text mode 1 Hex mode
<view_mode>	Integer type. Output format of received data. 0 data header\r\ndata. 1 data header,data.
<closed>	Integer type. Enable or disable the passive close of TCP connection when the server is closed. 0 Disable 1 Enable
<mode>	Integer type. Mode when sending UDP data. 0 Disable block mode. 1 Enable stream mode.
<enable>	Integer type. Enable or disable sending of TCP keep-alive information. 0 Disable 1 Enable
<idle_time>	Integer type. Indicates the cycle time of triggered keepalive. Range: 1–1800. Unit: s.
<interval_time>	Integer type. Indicates the interval of sending a packet in cycle time. Range: 25–100. Unit: s.
<probe_cnt>	Integer type. Packet transmission count in cycle time. Range: 3–10.
<show_length>	Integer type. Enable or disable data length presence in URC reported after data reception in TCP/IP buffer mode. 0 Disable 1 Enable
<retran_times>	Integer type. Number of reconnections within cycle time. Range: 3–12.
<retran_time>	Integer type. Reconnect interval within cycle time. Range: 5–1000. Unit: ms.
<state>	Integer type. 0 Disable auto accepting incoming TCP connection from the client. 1 Enable auto accepting incoming TCP connection from the client.
<buffer_size>	Integer type. Maximum number of bytes sent at one time. Range: 1460–10240.
<connectID>	Integer type. Socket service index. Range: 0–11.
<cycle_time>	Integer type. Set the interval for sending heartbeat packets. Range: 20–86400. Unit: s.
<msg_auto>	String type. Send the content of the heartbeat packet.
<msg_ignore>	String type. Filter data content.
<format>	Integer type. Control **>** output format. 0 0D0A> 1 0D0A>0D0A
<timeout>	Integer type. Send data timeout time. Range: 0–120. Unit: s.

<close_mode>	Integer type. Enable or disable asynchronous disconnection of TCP connection. 0 Disable 1 Enable
<send_view_mode>	Integer type. Enable or disable displaying the return information of **AT+QISEND** or **AT+QISENDEX** in URC form. 0 Disable 1 Enable
<send_mode>	Integer type. The return mode of **SEND OK**. 0 Return **SEND OK** immediately after sending data 1 Return **SEND OK** after receiving ACK from server 2 Return **<connectID>,SEND OK** after receiving ACK from server 3 Return **+QIURC: SEND OK** after receiving ACK from server 4 Return **+QIURC: <connectID>,SEND OK** after receiving ACK from server
<pdpmode>	Integer type. Set the PDP activation or deactivation mode. 0 Deactivation mode 1 Activation mode
<ratmode>	Integer type. Set the network type. 0 4G mode 1 2G mode
<counts>	Integer type. Number of retry times. Range: 2–4. Default value: 4. The actual number of activation or deactivation is equal to the value of **<counts>** + 1.
<retry_time>	Integer type. The maximum time for a single activation and deactivation. Unit: s.

2.3.2. AT+QICSGP Configure Parameters of a TCP/IP Context

This command configures the **<APN>**, **<username>**, **<password>** and other parameters of a TCP/IP context. The QoS settings can be configured with **AT+CGQMIN**, **AT+CGEQMIN**, **AT+CGQREQ** and **AT+CGEQREQ**. For more information about the AT commands, see ***document [1]***.

AT+QICSGP Configure Parameters of a TCP/IP Context	
Test Command **AT+QICSGP=?**	Response **+QICSGP: (**range of supported **<contextID>**s**),(**range of supported **<context_type>**s**),<APN>,<username>,<password>,(**range of supported **<authentication>**s**),(**list of supported **<CDMA_pwd>**s**)** **OK**
Write Command Query a specific context configuration **AT+QICSGP=<contextID>**	Response **+QICSGP: <context_type>,<APN>,<username>,<password>,<authentication>**

	OK
Write Command Configure the context **AT+QICSGP=<contextID>[,<context_type>,<APN>[,<username>,<password>)[,<authentication>[,<CDMA_pwd>]]]]**	Response **OK** If there is any error: **ERROR**
Maximum Response Time	/
Characteristic	This command takes effect immediately. The configurations will not be saved.

Parameter

<contextID>	Integer type. Context ID. Range: 1–15.
<context_type>	Integer type. Protocol type. 1 IPv4 2 IPv6 3 IPv4v6
<APN>	String type. Access point name.
<username>	String type. Username. Maximum length: 127 bytes.
<password>	String type. Password. Maximum length: 127 bytes.
<authentication>	Integer type. Authentication methods. 0 None 1 PAP 2 CHAP 3 PAP or CHAP
<CDMA_pwd>	Integer type. To configure whether to enable saving of **<username>** and **<password>** over CDMA network. 0 Disable 1 Enable

Example

```
AT+QICSGP=1                              //Query the configuration of context 1.
+QICSGP: 1,"","","",0

OK
AT+QICSGP=1,1,"UNINET","","",1          //Configure context 1. China Unicom APN: "UNINET".
OK
```

2.3.3. AT+QIACT Activate a PDP Context

Before activating a PDP context with **AT+QIACT**, the context should be configured with **AT+QICSGP**. After activation, the IP address can be queried with **AT+QIACT?**.

Although the range of **<contextID>** is 1–15, the module supports maximum three PDP contexts activated simultaneously. Depending on the network, it may take max.150 s to return **OK** or **ERROR** after executing **AT+QIACT**. Other AT commands can be executed only after the response is returned.

AT+QIACT Activate a PDP Context	
Test Command **AT+QIACT=?**	Response **+QIACT: (**range of supported **<contextID>**s**)** **OK**
Read command **AT+QIACT?**	Response Return the list of the currently activated contexts and their IP addresses: **+QIACT: 1,<context_state>,<context_type>[,<IP_address>]** **[.....** **+QIACT: 15,<context_state>,<context_type>[,<IP_address>]]** **OK**
Write Command Activate a specified PDP context **AT+QIACT=<contextID>**	Response **OK** If there is any error: **ERROR**
Maximum Response Time	150 s, determined by the network.
Characteristic	/

Parameter

<contextID>	Integer type. Context ID. Range: 1–15.
<context_state>	Integer type. Context state. 0 Deactivated 1 Activated
<context_type>	Integer type. Protocol type. 1 IPv4 2 IPv6

	3 IPv4v6
<IP_address>	String type. Local IP address after the context is activated.

2.3.4. AT+QIDEACT Deactivate a PDP Context

This command deactivates a specific context and closes all TCP/IP connections set up in this context. Depending on the network, it may take max. 40 s to return **OK** or **ERROR** after executing **AT+QIDEACT**. Other AT commands can be executed only after the response is returned.

AT+QIDEACT Deactivate a PDP Context	
Test Command **AT+QIDEACT=?**	Response **+QIDEACT: (**range of supported **<contextID>**s**)** **OK**
Write Command **AT+QIDEACT=<contextID>**	Response **OK** If there is any error: **ERROR**
Maximum Response Time	40 s, determined by the network.
Characteristic	/

Parameter

<contextID>	Integer type. Context ID. Range: 1–15.

2.3.5. AT+QIOPEN Open a Socket Service

The command opens a socket service. The service type can be specified by **<service_type>**. The data access mode (buffer access mode, direct push mode and transparent transmission mode) can be specified by **<access_mode>**. The URC **+QIOPEN** indicates if the socket service has been opened successfully.

1. If **<service_type>** is "TCP LISTENER", the module works as TCP server. After accepting a new TCP connection, the module automatically specifies a **<connectID>** and reports the URC **+QIURC: "incoming",<connectID>,<serverID>,<remoteIP>,<remote_port>**. The range of **<connectID>** is 0–11. The type of this new incoming connection is "TCP INCOMING" and the **<access_mode>** of "TCP INCOMING" is the same as that of "TCP LISTENER".

2. If **<service_type>** is "UDP SERVICE", UDP data can be sent to or received from the remote IP via **<local_port>**.

- Send data: execute **AT+QISEND=<connectID>,<send_length>,<remoteIP>,<remote_port>**.
- Receive data in direct push mode: the module reports the URC **+QIURC: "recv",<connectID>,<currentrecvlength>,<remoteIP>,<remote_port><CR><LF><data>**.
- Receive data in buffer access mode: the module reports the URC **+QIURC: "recv",<connectID>**, and then the received data can be retrieved with **AT+QIRD=<connectID>**.

3. It is suggested to wait for 150 s for **+QIOPEN: <connectID>,<err>** to be outputted after executing the Write Command. If the response cannot be received in 150 s, use **AT+QICLOSE** to close the socket.

AT+QIOPEN Open a Socket Service	
Test Command **AT+QIOPEN=?**	Response **+QIOPEN: (**range of supported **<contextID>**s**),(**range of supported **<connectID>**s**),"TCP/UDP/TCP LISTENER/UDP SERVICE","<IP_address>/<domain_name>",(**range of supported **<remote_port>**s**),(**range of supported **<local_port>**s**),(**range of supported **<access_mode>**s**)** **OK**
Write Command **AT+QIOPEN=<contextID>,<connectID>,<service_type>,"<IP_address>/<domain_name>",<remote_port>[,<local_port>[,<access_mode>]]**	Response If the service is in transparent transmission mode (**<access_mode>**=2) and the service is opened successfully: **CONNECT** If there is any error: **ERROR** Error description can be got via **AT+QIGETERROR**. If the service is in buffer access mode (**<access_mode>**=0) or direct push mode (**<access_mode>**=1): **OK** **+QIOPEN: <connectID>,<err>** **<err>** is 0 when the service is opened successfully. In other cases, **<err>** is not 0.
Maximum Response Time	150 s, determined by the network.
Characteristic	/

Parameter

<contextID>	Integer type. Context ID. Range: 1–15.
<connectID>	Integer type. Socket service index. Range: 0–11.
<service_type>	String type. Socket service type.

	"TCP"	Start a TCP connection as a client
	"UDP"	Start a UDP connection as a client
	"TCP LISTENER"	Start a TCP server to listen to incoming TCP connections
	"UDP SERVICE"	Start a UDP service

<IP_address> String type.
If **<service_type>** is "TCP" or "UDP", it indicates the IP address of remote server, such as 220.180.239.212.
If **<service_type>** is "TCP LISTENER" or "UDP SERVICE", enter 127.0.0.1.

<domain_name> String type. Domain name address of the remote server.

<remote_port> Integer type. Port number of the remote server. Range: 0–65535.
If **<service_type>** is "TCP" or "UDP", this parameter must be specified. Range: 1–65535.
If **<service_type>** is "TCP LISTENER" or "UDP SERVICE", specify this parameter as 0.

<local_port> Integer type. Local port number. Range: 0–65535.
If **<service_type>** is "TCP LISTENER" or "UDP SERVICE", this parameter must be specified. Range: 1–65535.
If **<service_type>** is "TCP" or "UDP", this parameter can be omitted and the default value is 0. The local port is assigned automatically if **<local_port>** is 0. Otherwise, the local port is assigned as specified.

<access_mode> Integer type. Data access mode of the socket service.

0	Buffer access mode
1	Direct push mode
2	Transparent transmission mode

<err> Integer type. Error codes of the operation. See ***Chapter 4*** for more information.

2.3.6. AT+QICLOSE Close a Socket Service

This command closes a specified socket service. Depending on the network, it will take max. 10 s (default value, can be modified by **<timeout>**) to return **OK** or **ERROR** after executing **AT+QICLOSE**. Other AT commands can be executed only after the response is returned.

AT+QICLOSE Close a Socket Service	
Test Command **AT+QICLOSE=?**	Response **+QICLOSE: (**range of supported **<connectID>**s**),(**range of supported **<timeout>**s**)** **OK**
Write Command **AT+QICLOSE=<connectID>[,<timeout>]**	Response If the socket service is closed successfully: **OK** If the command failed to close the socket service:

	ERROR
Maximum Response Time	10 s by default, determined by the setting of **<timeout>**.
Characteristic	/

Parameter

<connectID>	Integer type. Socket service index. Range: 0–11.
<timeout>	Integer type. Timeout value for the response to be outputted. If the **FIN ACK** of the other peer is not received within **<timeout>**, the module will be forced to close the socket. Range: 0–65535. Default value: 10. Unit: s.

2.3.7. AT+QISTATE Query Socket Service Status

The command queries the socket service status. If the **<query_type>** is 0, it returns the statuses of all existing socket services in the specified context. If the **<query_type>** is 1, it will return the status of a specified socket service.

AT+QISTATE Query Socket Service Status	
Test Command **AT+QISTATE=?**	Response **OK**
Read/Execution Command **AT+QISTATE?** or **AT+QISTATE**	Response Return the status of all existing connections: **+QISTATE: <connectID>,<service_type>,<IP_address>,<remote_port>,<local_port>,<socket_state>,<contextID>,<serverID>,<access_mode>,<AT_port>** **[…]** **OK**
Write Command If **<query_type>** is 0, query the connection status of a specified context **AT+QISTATE=<query_type>,<contextID>**	Response Return the status of all existing connections in a specified context: **+QISTATE: <connectID>,<service_type>,<IP_address>,<remote_port>,<local_port>,<socket_state>,<contextID>,<serverID>,<access_mode>,<AT_port>** **[…]** **OK**
Write Command If **<query_type>** is 1, query the connection status of a specified socket	Response **+QISTATE: <connectID>,<service_type>,<IP_address>,<remote_port>,<local_port>,<socket_state>,<contextID>,<**

service **AT+QISTATE=<query_type>, <connectID>**	**serverID>,<access_mode>,<AT_port>** **OK**
Maximum Response Time	300 ms
Characteristic	/

Parameter

<query_type> Integer type. Query type.
0 Query connection status of all socket services in a specified context
1 Query connection status of a specified socket service

<contextID> Integer type. Context ID. Range: 1–15.

<connectID> Integer type. Socket service index. Range: 0–11.

<service_type> String type. Socket service type.

"TCP"	Start a TCP connection as a client
"UDP"	Start a UDP connection as a client
"TCP LISTENER"	Start a TCP server to listen to incoming TCP connections
"TCP INCOMING"	Start a TCP connection accepted by a TCP server
"UDP SERVICE"	Start a UDP service

<IP_address> String type. IP address.
If **<service_type>**="TCP" or "UDP", it is the IP address of remote server.
If **<service_type>**="TCP LISTENER" or "UDP SERVICE", it is the local IP address.
If **<service_type>**="TCP INCOMING", it is the IP address of a remote client.

<remote_port> Integer type. Remote port number.
If **<service_type>**="TCP" or "UDP", it is the port of remote server.
If **<service_type>**="TCP LISTENER" or "UDP SERVICE", the port is invalid.
If **<service_type>**="TCP INCOMING", it is the port of a remote client.

<local_port> Integer type. Local port number.
If **<local_port>**=0, then the local port is assigned automatically.

<socket_state> Integer type. Socket service status.
0 "Initial": connection has not been established
1 "Opening": client is connecting or server is trying to listen
2 "Connected": client/incoming connection has been established
3 "Listening": server is listening
4 "Closing": connection is closing

<serverID> Integer type. Only valid when **<service_type>** is "TCP INCOMING". **<serverID>** represents the ID of the server that accepts the incoming TCP connection, and the value is the same as **<connectID>** of the server's "TCP LISTENER".

<access_mode> Integer type. Data access mode.
0 Buffer access mode
1 Direct push mode
2 Transparent transmission mode

<AT_port>	String type. COM port of socket service.	
	"usbmodem"	USB modem port
	"usbat"	USB AT port
	"uart1"	UART port1
	"cmux1"	MUX port 1
	"cmux2"	MUX port 2
	"cmux3"	MUX port 3
	"cmux4"	MUX port 4

2.3.8. AT+QISEND Send Data

In buffer access mode (**<access_mode>**=0) or direct push mode (**<access_mode>**=1), the data can be sent with **AT+QISEND**. If the data have been sent to the module successfully, **SEND OK** is returned, otherwise **SEND FAIL** or **ERROR** is returned.

- **SEND FAIL** indicates the sending buffer is full. In this case, resending of the data can be tried.
- **ERROR** indicates an error in the sending data process. In this case, wait for some time before resending the data. Maximum length of data to be sent: 1460 bytes.
- **SEND OK** means that the data have been sent to the peer, but it does not mean they have reached the server successfully. You can query whether the data have reached the server with **AT+QISEND=<connectID>,0**.

AT+QISEND Send Data	
Test Command **AT+QISEND=?**	Response **+QISEND: (**range of supported **<connectID>**s**),(**range of supported **<send_length>**s**)** **OK**
Write Command Send variable-length data when **<service_type>** is "TCP", "UDP" or "TCP INCOMING" **AT+QISEND=<connectID>**	Response **>** After the response **>**, input the data to be sent. Tap **Ctrl + Z** to send data, and tap **Esc** to cancel the operation. (1) If **<send_view_mode>**=0: If the connection has been established and the data is sent successfully: **SEND OK** If the connection has been established but the sending buffer is full: **SEND FAIL** If the connection has not been established, abnormally closed,

	or any parameter is incorrect: **ERROR** (2) If **<send_view_mode>**=1: If the connection has been established and the data is sent successfully: **+QISEND: <connectID>,<status>,<Freesize>** **OK** If the connection has not been established, abnormally closed, or any parameter is incorrect: **ERROR**
Write Command Send fixed-length data when <**service_type>** is "TCP", "UDP" or "TCP INCOMING" **AT+QISEND=<connectID>,<send_length>**	Response **>** After the response **>**, input the data until the data length equals to **<send_length>**. (1) If **<send_view_mode>**=0: If the connection has been established and the data is sent successfully: **SEND OK** If the connection has been established but the sending buffer is full: **SEND FAIL** If the connection has not been established, abnormally closed, or any parameter is incorrect: **ERROR** (2) If **<send_view_mode>**=1: If the connection has been established: **+QISEND: <connectID>,<status>,<Freesize>** **OK** If the connection has not been established, abnormally closed, or any parameter is incorrect: **ERROR**
Write Command If **<service_type>** is "UDP SERVICE" **AT+QISEND=<connectID>,<send_len**	Response Send fixed length data to a specified remote IP address and remote port. The **<service_type>** must be "UDP SERVICE".

gth>,<remoteIP>,<remote_port>	**>** After the response **>**, input the data until the data length equals **<send_length>** If the connection is established and the data are sent successfully: **SEND OK** If the connection is established but the sending buffer is full: **SEND FAIL** If the connection is not established, abnormally closed, or any parameter is incorrect: **ERROR**
Write Command When **<send_length>** is 0, query the sent data **AT+QISEND=<connectID>,0**	Response If the specified connection exists: **+QISEND: <total_send_length>,<ackedbytes>,<unacked bytes>** **OK** If the specified **<connectID>** does not exist, or there is any other error: **ERROR**
Maximum Response Time	/
Characteristic	/

Parameter

<connectID>	Integer type. Socket service index. Range: 0–11.
<send_view_mode>	Integer type. Enable or disable display **AT+QISEND** command execution information in URC form. 0 Disable 1 Enable
<status>	Integer type. 0 Send data to the socket buffer successfully 1 The receiving buffer is full, send failed
<Freesize>	Integer type. Free space in the current buffer. Range: 0–10240. Unit: byte.
<send_length>	Integer type. Length of data to be sent. Range: 0–1460. Unit: byte.
<remoteIP>	String type. Remote IP address (must be dot format). Valid only when **<service_type>** is "UDP SERVICE".
<remote_port>	Integer type. Remote port. Valid only when **<service_type>** is "UDP SEVERVICE".

<total_send_length>	Integer type. Total length of sent data. Unit: byte.
<ackedbytes>	Integer type. Total length of acknowledged data. Unit: byte.
<unackedbytes>	Integer type. Total length of unacknowledged data. Unit: byte.

2.3.9. AT+QIRD Read the Received TCP/IP Data

In buffer access mode, after receiving data, the module will buffer it and report **+QIURC: "recv",<connectID>**, and then the data can be read by **AT+QIRD**.

Please note that if the buffer is not empty, and the module receives data again, it will not report a new URC until all the received data has been read via **AT+QIRD** from buffer.

AT+QIRD Read the Received TCP/IP Data	
Test Command **AT+QIRD=?**	Response **+QIRD: (**range of supported **<connectID>**s**),(**range of supported **<read_length>**s**)** **OK**
Write Command When **<service_type>** is "TCP", "UDP", "TCP INCOMING" **AT+QIRD=<connectID>[,<read_length>]**	Response If the specified connection has received the data, response: **+QIRD: <read_actual_length><CR><LF><data>** **OK** If there are no data: **+QIRD: 0** **OK** If the connection does not exist: **ERROR**
Write Command When **<service_type>** is "UDP SERVICE" **AT+QIRD=<connectID>**	Response If data exist: **+QIRD: <read_actual_length>,<remoteIP>,<remote_port><CR><LF><data>** **OK** If there are no data: **+QIRD: 0** **OK**

	If the connection does not exist: **ERROR**
Write Command When **<read_length>** is 0, query the retrieved data length **AT+QIRD=<connectID>,0**	Response If the specified connection exists: **+QIRD: <total_receive_length>,<have_read_length>,<unread_length>** **OK** If there is any error: **ERROR**
Maximum Response Time	/
Characteristic	/

Parameter

<connectID>	Integer type. Socket service index. Range: 0–11.
<read_length>	Integer type. Maximum length of data to be read. Range: 0–1500. Unit: byte.
<read_actual_length>	Integer type. Length of actually retrieved data. Unit: byte.
<data>	String type. Retrieved data. Unit: byte.
<remoteIP>	String type. Remote IP address. Valid only when **<service_type>** is "UDP SERVICE".
<remote_port>	Integer type. Remote port number. Valid only when **<service_type>** is "UDP SERVICE".
<total_receive_length>	Integer type. Total length of the received data. Unit: byte.
<have_read_length>	Integer type. Length of data that have been retrieved. Unit: byte.
<unread_length>	Integer type. Length of data that has not been retrieved. Unit: byte.

2.3.10. AT+QISENDEX Send Hex String Data

This command sends hex string data and cannot be applied to "UDP SERVICE" and "TCP LISTENER" sockets.

AT+QISENDEX Send Hex String Data	
Test Command **AT+QISENDEX=?**	Response **+QISENDEX: (**range of supported **<connectID>**s**),<hex_string>** **OK**
Write Command **AT+QISENDEX=<connectID>,<hex_string>**	Response (1) If **<send_view_mode>**=0: If the hex string is sent successfully:

	SEND OK If the sending buffer is full: **SEND FAIL** If the connection does not exist: **ERROR** (2) If **<send_view_mode>**=1: If the connection has been established: **+QISENDEX: <connectID>,<status>,<Freesize>** **OK** If the connection has not been established, abnormally closed, or any parameter is incorrect: **ERROR**
Maximum Response Time	/
Characteristic	/

Parameter

<connectID>	Integer type. Socket service index. Range: 0–11.
<hex_string>	String type. Hex string data. Max. length: 512 bytes.
<send_view_mode>	Integer type. Enable or disable display **AT+QISEND** command execution information in URC form. 0 Disable 1 Enable
<status>	Integer type. 0 Send data to the socket buffer successfully 1 The receiving buffer is full, send failed
<Freesize>	Integer type. Free space in the current buffer. Range: 0–10240. Unit: byte.

2.3.11. AT+QISWTMD Switch Data Access Mode

This command switches the data access modes between buffer access mode, direct push mode and transparent transmission mode. When a socket service is started, the data access mode can be specified via the **<access_mode>** of **AT+QIOPEN**. After opening a socket, the data access mode can be changed with **AT+QISWTMD**.

AT+QISWTMD Switch Data Access Mode	
Test Command **AT+QISWTMD=?**	Response **+QISWTMD: (**range of supported **<connectID>**s)**,(**range of supported **<access_mode>**s**)** **OK**
Write Command **AT+QISWTMD=<connectID>,<access_mode>**	Response If data access mode is switched successfully and **<access_mode>** is 0 or 1: **OK** If data access mode is switched successfully and **<access_mode>** is 2, the module enters the intended data mode: **CONNECT** If there is any error: **ERROR**
Maximum Response Time	/
Characteristic	This command takes effect immediately. The configuration will not be saved.

Parameter

<connectID>	Integer type. Socket service index. Range: 0–11.
<access_mode>	Integer type. Data access modes of the connection. 0 Buffer access mode 1 Direct push mode 2 Transparent transmission mode

2.3.12. AT+QPING Ping a Remote Server

This command tests the reachability of a host on an Internet protocol network. Before using the ping utility, the host should activate the context of the corresponding **<contextID>** with **AT+QIACT**. The command returns the result within **<timeout>** and the default value of **<timeout>** is 4 s.

AT+QPING Ping a Remote Server	
Test Command **AT+QPING=?**	Response **+QPING: (**range of supported **<contextID>**s)**,<host>,(**range of supported **<timeout>**s)**,(**range of supported **<pingnum>**s)

	OK
Write Command **AT+QPING=<contextID>,<host>[,<timeout>[,<pingnum>]]**	Response If a remote host is pinged successfully: **OK** **+QPING: <result>[,<IP_address>,<bytes>,<time>,<ttl>]** **[…]** **+QPING: <finresult>[,<sent>,<rcvd>,<lost>,<min>,<max>,<avg>]** If there is any error: **ERROR**
Maximum Response Time	/
Characteristic	/

Parameter

<contextID> Integer type. Context ID. Range: 1–15.

<host> String type. Host address in string type. It is a domain name or a dotted decimal IP address.

<timeout> Integer type. Maximum time to wait for the response of each ping request. Range: 1–255. Default value: 4. Unit: s.

<pingnum> Integer type. Maximum number of ping requests. Range: 1–10. Default value: 4.

<result> Integer type. Result of each ping request.

0 Received the ping response from the host. In this case, it is followed by **,<IP_address>,<bytes>,<time>,<ttl>**.

Others See ***Chapter 4*** for more information.

<IP_address> String type. IP address of the remote server formatted as a dotted decimal IP.

<bytes> Integer type. The length of each sent ping request. Unit: byte.

<time> Integer type. The time wait for the response of the ping request. Unit: ms.

<ttl> Integer type. Time to live value of the response packet for the ping request.

<finresult> Integer type. The final result of the command.

0 It is finished normally. It is successful to activate the context and find the host. In this case, it is followed by **,<sent>,<rcvd>,<lost>,<min>,<max>,<avg>**.

Others Result codes. See ***Chapter 4*** for more information.

<sent> Integer type. Total number of sent ping requests.

<rcvd> Integer type. Total number of the ping requests that have received responses.

<lost> Integer type. Total number of timed out ping requests.

<min> Integer type. Minimum response time. Unit: ms.

<max> Integer type. Maximum response time. Unit: ms.

<avg>	Integer type. Average response time. Unit: ms.

2.3.13. AT+QNTP Synchronize Local Time with NTP Server

This command synchronizes the local time with Universal Time Coordinated (UTC) through the NTP server. Before time synchronization, the host should activate the context corresponding to **<contextID>** with **AT+QIACT**. Depending on the network, it will take max. 125 s to return the result.

AT+QNTP Synchronize Local Time with NTP Server	
Test command **AT+QNTP=?**	Response **+QNTP: (**range of supported **<contextID>**s**),<server>,(**range of supported **<port>**s**),(**list of supported **<autosettime>**s**)** **OK**
Read command **AT+QNTP?**	Response If in the process of local time synchronization: **+QNTP: <server>,<port>** **OK**
Write command **AT+QNTP=<contextID>,<server>[,<port>[,<autosettime>]]**	Response If the local time is synchronized with NTP server successfully: **OK** **+QNTP: <err>,<time>** If there is any error: **ERROR**
Maximum Response Time	125 s, determined by the network.
Characteristic	This command takes effect immediately. The configuration will not be saved.

Parameter

<contextID>	Integer type. Context ID. Range: 1–15.
<server>	String type. NTP server address.
<port>	Integer type. NTP server port number. Range: 1–65535. Default value: 123.
<autosettime>	Integer type. Whether to automatically set synchronized time as local time. 0 Not set 1 Set
<err>	Integer type. Error codes of the operation. See ***Chapter 4*** for more information.
<time>	String type. Time synchronized with NTP server.

	The format is "YYYY/MM/DD,hh:mm:ss±zz". Range of zz: -48–+56.

2.3.14. AT+QIDNSCFG Configure Address of DNS Server

Before setting the DNS server address, the host must activate the context of corresponding **<contextID>** with **AT+QIACT**.

AT+QIDNSCFG Configure Address of DNS Server	
Test command **AT+QIDNSCFG=?**	Response **+QIDNSCFG:** (range of supported **<contextID>**s),**<pridnsaddr>,<secdnsaddr>** **OK**
Write Command **AT+QIDNSCFG=<contextID>[,<pridnsaddr>[,<secdnsaddr>]]**	Response If the optional parameters are omitted, query the current DNS server addresses of a specified PDP context: **+QIDNSCFG: <contextID>,<pridnsaddr>,<secdnsaddr>** **OK** If any of the optional parameters is specified, configure the primary and secondary DNS server addresses of a specified PDP context: **OK** If there is any error: **ERROR**
Maximum Response Time	/
Characteristic	This command takes effect immediately. The configurations are not saved.

Parameter

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

2.3.15. AT+QIDNSGIP Get IP Address by Domain Name

This command gets an IP address by domain name. Before querying the DNS, the host should activate the context of the corresponding **<contextID>** with **AT+QIACT**. Depending on the network, it will return a result in max. 60 s.

AT+QIDNSGIP Get IP Address by Domain Name	
Test Command **AT+QIDNSGIP=?**	Response **+QIDNSGIP: (**range of supported **<contextID>**s)**,<hostname>** **OK**
Write Command **AT+QIDNSGIP=<contextID>,<hostname>**	Response **OK** If there is any error: **ERROR** The result will be returned as URC. **+QIURC: "dnsgip",<err>,<IP_count>,<DNS_ttl>** **[.....** **+QIURC: "dnsgip",<hostIPaddr>]**
Maximum Response Time	60 s, determined by the network.
Characteristic	/

Parameter

<contextID>	Integer type. Context ID. Range: 1–15.
<hostname>	String type. Domain name.
<err>	Integer type. Result code. See ***Chapter 4*** for more information.
<IP_count>	Integer type. Number of IP addresses corresponding to the **<hostname>**.
<DNS_ttl>	Integer type. Time to live of the DNS. Unit: s.
<hostIPaddr>	String type. IP address of **<hostname>**.

2.3.16. AT+QISDE Control Whether to Echo the Data for AT+QISEND

This command controls whether to echo the data for **AT+QISEND**.

AT+QISDE Control Whether to Echo the Data for AT+QISEND	
Test command **AT+QISDE=?**	Response **+QISDE: (**list of supported **<echo>**s**)**

	OK
Read command **AT+QISDE?**	Response **+QISDE: <echo>** **OK**
Write Command **AT+QISDE=<echo>**	Response **OK** If there is any error: **ERROR**
Maximum Response Time	300 ms
Characteristic	This command takes effect immediately. The configuration is not saved.

Parameter

<echo>	Numeric type. Whether to echo the data for **AT+QISEND**. 0 Do not echo the data 1 Echo the data

2.3.17. AT+QIGETERROR Query the Error Code of the Last AT Command

If **ERROR** is returned after TCP/IP commands are executed, the detailed information about a result code can be queried with **AT+QIGETERROR**. Please note that **AT+QIGETERROR** only returns the result code of the last TCP/IP AT command.

AT+QIGETERROR Query the Error Code of the Last AT Command	
Test command **AT+QIGETERROR=?**	Response **OK**
Execution Command **AT+QIGETERROR**	Response **+QIGETERROR: <err>,<errcode_description>** **OK**
Maximum Response Time	300 ms
Characteristic	/

Parameter

<err>	Integer type. Error codes of the operation. See ***Chapter 4*** for more information.
<errcode_description>	A string parameter indicates the details of error information. See ***Chapter 4*** for more information.

2.4. Description of URCs

+QIURC: is used at the beginning of the URC of TCP/IP AT commands to be reported to the host. The URC contains the reports about incoming data, closed connection and incoming connection and etc. Actually, there is **<CR><LF>** both before and after URC, but **<CR><LF>** is intentionally not presented.

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

When TCP socket service is closed by remote peer or due to a network error, the URC is outputted, and the status of socket service is "closing" (**<socket_state>**=4). **AT+QICLOSE=<connectID>** can be used to change the **<socket_state>** to initial.

+QIURC: "closed" URC Indicating Connection Closed	
+QIURC: "closed",<connectID>	Socket service connection is closed.

Parameter

<connectID>	Integer type. Socket service index. Range: 0–11.

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

In buffer access mode or direct push mode, after receiving data, the module reports a URC to the host.

In buffer access mode, after receiving data, the module reports **+QIURC: "recv",<connectID>** to notify the host. Then host can retrieve data with **AT+QIRD**.

In direct push mode, the received data are outputted to COM port directly.

+QIURC: "recv" URC Indicating Incoming Data	
+QIURC: "recv",<connectID>	Indicates incoming data in buffer access mode. The host can retrieve data via **AT+QIRD.**
+QIURC: "recv",<connectID>,<curre	Indicates incoming data in direct push mode when the

ntrecvlength><CR><LF><data>	**<service_type>** is "TCP", "UDP" or "TCP INCOMING".
+QIURC: "recv",<connectID>,<currentrecvlength>,<remoteIP>,<remote_port><CR><LF><data>	Indicates incoming data in direct push mode when **<service_type>** is "UDP SERVICE".

Parameter

<connectID>	Integer type. Socket service index. Range: 0–11.
<currentrecvlength>	Integer type. Length of actually received data.
<remoteIP>	String type. Remote IP address.
<remote_port>	Integer type. Remote port number.
<data>	Integer type. Received data. Unit: byte.

NOTE

If the buffer is not empty, and the module receives data again, it does not report a new URC until all the received data have been retrieved with **AT+QIRD** from the buffer.

2.4.3. +QIURC: "incoming full" Indicate Incoming Connection Reaches the Limit

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

+QIURC: "incoming full" Indicate Incoming Connection Reaches the Limit	
+QIURC: "incoming full"	Indicates that the number of incoming connections has reached the limit.

2.4.4. +QIURC: "incoming" Indicate Incoming Connection

If the **<service_type>** is "TCP LISTENER", when a remote client connects to this server, the host will automatically assign a free **<connectID>** for the new connection. The range of **<connectID>** is 0–11. In such a case, the module will report the URC. The **<service_type>** of the new connection will be "TCP INCOMING", and the **<access_mode>** will be buffer access mode.

+QIURC: "incoming" Indicate Incoming Connection	
+QIURC: "incoming",<connectID>,<serverID>,<remoteIP>,<remote_port>	When the new incoming connection is accepted by **<serverID>**, the allocated **<connectID>**, **<remoteIP>** and **<remote_port>** will be informed by this URC.

Parameter

<connectID>	Integer type. Assign this socket service for the incoming connection, which is automatically specified by the module. Range: 0–11.
<serverID>	Integer type. The incoming **<connectID>** accepted by the server whose **<service_type>** is "TCP LISTENER" and listening socket ID is **<serverID>**.
<remoteIP>	String type. Remote IP address of the incoming **<connectID>**.
<remote_port>	Integer type. Remote port of the incoming **<connectID>**.

2.4.5. +QIURC: "pdpdeact" Indicate PDP Deactivation

PDP context may be deactivated by the network. The module will report the URC to the host about PDP deactivation. In such a case, the host must execute **AT+QIDEACT** to deactivate the context and reset all connections.

+QIURC: "pdpdeact" Indicate PDP Deactivation	
+QIURC: "pdpdeact",<contextID>	**<contextID>** context is deactivated.

Parameter

<contextID>	Integer type. Context ID. Range: 1–15.

3 Examples

3.1. Configure and Activate a Context

3.1.1. Configure a Context

```
AT+QICSGP=1,1,"UNINET","","",1          //Configure context 1. China Unicom APN: "UNINET".
OK
```

3.1.2. Activate a Context

```
AT+QIACT=1                              //Activate context 1. Depending on the network, the maximum
                                          response time is 150 s.
OK                                      //Activated the context successfully.
AT+QIACT?                               //Query the context state.
+QIACT: 1,1,1,"10.7.157.1"

OK
```

3.1.3. Deactivate a Context

```
AT+QIDEACT=1                            //Deactivate context 1.

OK                                      //Deactivated the context successfully. Depending on the
                                          network, the maximum response time is 40 s.
```

3.2. TCP Client Works in Buffer Access Mode

3.2.1. Set up a TCP Client Connection and Enter Buffer Access Mode

```
AT+QIOPEN=1,0,"TCP","220.180.239.212",8009,0,0    //Context is 1 and <connectID> is 0. Before
                                                  using AT+QIOPEN, the host should activate
                                                  the context with AT+QIACT first.
```

```
OK

+QIOPEN: 0,0                                  //TCP client connected successfully. It is
                                              suggested to wait for 150 s for the URC
                                              +QIOPEN: <connectID>,<err>. If the URC
                                              cannot be received in 150 s, the host could
                                              use AT+QICLOSE to close the socket.
AT+QISTATE=1,0                                //Query the connection status of socket
                                              service 1.
+QISTATE: 0,"TCP","220.180.239.212",8009,65514,2,1,0,0,"usbmodem"

OK
```

3.2.2. Send Data in Buffer Access Mode

```
AT+QISEND=0                        //Send variable-length data.
> test1<ctrl+Z>
SEND OK                            //SEND OK does not mean the data has been sent to the server
                                   successfully. The host can query whether the data has reached the
                                   server via AT+QISEND=0,0.
AT+QISEND=0,4                      //Send fixed-length data and the data length is 4 bytes.
> test
SEND OK
AT+QISEND=0,0                      //Query the length of sent data.
+QISEND: 9,9,0

OK
AT+QISENDEX=0,"3132333435"         //Send Hex string data.
SEND OK
AT+QISEND=0,0                      //Query the length of sent data, acknowledged data and
                                   unacknowledged data.
+QISEND: 14,14,0

OK
```

3.2.3. Retrieve Data from Remote Server in Buffer Access Mode

```
+QIURC: "recv",0               //The received data when <connectID>=0.
AT+QIRD=0,1500                 //Read data, the maximum length of data to be retrieved is 1500 bytes.
+QIRD: 14                      //The length of actually received data is 14 bytes.
test1
```

```
OK
AT+QIRD=0,1500
+QIRD: 0                        //No data in buffer.

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. Close a Connection

```
AT+QICLOSE=0                    //Close a connection whose <connectID> is 0. Depending on the
                                 network, the maximum response time is 10 s.
OK
```

3.3. TCP Client Works in Transparent Access Mode

3.3.1. Set up a TCP Client Connection and Enter Transparent Access Mode

```
AT+QIOPEN=1,0,"TCP","220.180.239.212",8009,0,2 //Context is 1 and <connectID> is 0. Before using
                                                 AT+QIOPEN, the host should activate the
                                                 context with AT+QIACT.
CONNECT                                          //TCP client connected successfully. It is
                                                 suggested to wait for 150 s for the URC
                                                 CONNECT. If the URC cannot be received in
                                                 150 s, the host could use AT+QICLOSE to close
                                                 the socket.
```

3.3.2. Send Data in Transparent Access Mode

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

3.3.3. Retrieve Data from Remote Server in Transparent Access Mode

```
Test 1                                    //All data received from the Internet are outputted
                                          directly via COM port.
```

3.3.4. Close a TCP Client

```
AT+QICLOSE=0                          //After using +++ to exit transparent transmission mode,
                                        the host could use AT+QICLOSE to close the TCP link.
                                        Depending on the network, the maximum response
                                        time is 10 s.
OK
```

3.4. TCP Client Works in Direct Push Mode

3.4.1. Set up a TCP Client Connection and Enter Direct Push Mode

```
AT+QIOPEN=1,0,"TCP","220.180.239.212",8009,0,1 //Context is 1 and <connectID> is 0. Before using
                                               AT+QIOPEN, the host should activate the context
                                               with AT+QIACT.
OK

+QIOPEN: 0,0                                    //TCP client is connected successfully. It is
                                                suggested to wait for 150 s for the URC
                                                +QIOPEN: <connectID>,<err>. If the URC
                                                cannot be received in 150 s, the host could use
                                                AT+QICLOSE to close the socket.
AT+QISTATE=1,0                                  //Query if the connection state 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 Push Mode

```
AT+QISEND=0                           //Send variable-length data.
> test1<ctrl+Z>
SEND OK                               //SEND OK does not mean the data have been received
                                        by the server successfully. Host can query whether the
                                        data have reached the server with AT+QISEND=0,0.
AT+QISEND=0,5                         //Send fixed length data and the data length is 5 bytes.
> test2
SEND OK
AT+QISEND=0,0                         //Query the lengths of sent, acknowledged  and
                                        unacknowledged data.
```

```
+QISEND: 10,10,0                          //A total of 10 bytes of data have been sent, and all 10
                                          bytes have been acknowledged.

OK
```

3.4.3. Retrieve Data from Remote Server in Direct Push Mode

```
+QIURC: "recv",0,4                        //Retrieve data from remote server.
test
```

3.4.4. Close a TCP Client

```
AT+QICLOSE=0                              //Close the connection whose <connectID> is 0.
                                          Depending on the network, the maximum response time
                                          is 10 s.
OK
```

3.5. TCP Server Works in Buffer Access Mode

3.5.1. Start a TCP Server

```
AT+QIOPEN=1,1,"TCP LISTENER","127.0.0.1",0,2020,0 //Context is 1 and <connectID> is 1. Before
                                                  using AT+QIOPEN, the host should activate
                                                  the context with AT+QIACT.
OK

+QIOPEN: 1,0                                      //TCP server is opened 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 TCP Incoming Connection

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

3.5.3. Retrieve Data from Incoming Connection

```
+QIURC: "recv",11            //Received data from remote incoming connection.
AT+QIRD=11,1500              //Retrieve data received from incoming connection.
+QIRD: 4                     //Length of actually retrieved data is 4 bytes.
test

OK
AT+QIRD=11,1500
+QIRD: 0                     //No data in buffer.

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. Close a TCP Server

```
AT+QICLOSE=11                //Close the incoming connection. Depending on the network, the
                             maximum response time is 10 s by default.
OK
AT+QICLOSE=1                 //Close TCP server listening.
OK
```

3.6. UDP Service

3.6.1. Start a UDP Service

```
AT+QIOPEN=1,2,"UDP SERVICE","127.0.0.1",0,3030,0   //Start a UDP service whose <connectID> is
                                                   2 and <contextID> is 1. Before using
                                                   AT+QIOPEN, the host should activate the
                                                   context with AT+QIACT.
OK

+QIOPEN: 2,0                                      //UDP service is opened successfully.
AT+QISTATE=0,1                                    //Query if 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 Remote Client

```
AT+QISEND=2,10,"10.7.89.10",6969               //Send 10-byte long data to a remote client
                                               whose IP is 10.7.89.10 and the remote port is
                                               6969.
>1234567890
SEND OK
```

3.6.3. Receive Data from Remote Client

```
+QIURC: "recv",2                      //Receive data from remote client.
AT+QIRD=2                             //Retrieve UDP data. One whole UDP packet will be outputted.
                                      There is no need to specify the read length.
+QIRD: 4,"10.7.76.34",7687            //The retrieved data length is 4. The remote IP address is
                                      10.7.76.34 and remote port is 7687.
AAAA

OK
AT+QIRD=2                             //Retrieve data.
+QIRD: 0                              //No data in buffer.

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

3.6.4. Close a UDP Service

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

3.7. PING

```
AT+QPING=1,"www.baidu.com"            //Ping www.baidu.com in context 1. Before pinging the
                                      destination IP address, the host should activate the context with
```

```
                                             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     //Synchronize local time with NTP server "202.112.10.36:123".
                                   Before synchronizing the time, the host should activate the
                                   context with AT+QIACT.
OK

+QNTP: 0,"2019/07/21,06:10:59+00"
AT+CCLK?
+CCLK: "19/07/21,06:11:05+00"

OK
```

3.9. Get Last Error Code

```
AT+QIOPEN=1,"TCP","220.180.239.212",8009,0,1   //Send AT+QIOPEN with missing <connectID>.
ERROR
AT+QIGETERROR
+QIGETERROR: 552,invalid parameters

OK
```

4 Summary of Error Codes

If **ERROR** is returned after executing TCP/IP AT commands, the detailed information about errors can be queried with **AT+QIGETERROR**. Please note that **AT+QIGETERROR** just returns the result code of the last TCP/IP AT command.

Table 2: Summary of Error Codes

<err>	<errcode_description>
0	Operate successfully
550	Unknown error
551	Operation blocked
552	Invalid parameters
553	Memory allocation failed
554	Socket creation failed
555	Operation not supported
556	Socket bind failed
557	Socket listen failed
558	Socket write failed
559	Socket read failed
560	Socket accept failed
561	Activate PDP context failed
562	Deactivate PDP context failed
563	Socket identity has been used
564	DNS busy

565	DNS parse failed
566	Socket connect failed
567	Connection reset
568	System busy
569	Operation timeout
570	PDP context deactivated
571	Cancel sending
572	Operation not allowed
573	APN not configured
574	Port busy

5 Appendix References

Table 3: Related Document

Document Name
[1] Quectel_LTE_Standard(A)_Series_AT_Commands_Manual

Table 4: Terms and Abbreviations

Abbreviation	Description
3GPP	3rd Generation Partnership Project
ACK	Acknowledgement
APN	Access Point Name
ASCII	American Standard Code for Information Interchange
CHAP	Challenge Handshake Authentication Protocol
CS	Circuit Switching
DNS	Domain Name System
FIN	Finish
ID	Mostly refers to Identifier in terms of software
IP	Internet Protocol
NTP	Network Time Protocol
NV	Non-Volatile
PAP	Password Authentication Protocol l
PDP	Packet Data Protocol
PPP	Point-to-Point Protocol

PS	Packet Switching
QoS	Quality of Service
TCP	Transmission Control Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
UART	Universal Asynchronous Receiver& Transmitter
UDP	User Datagram Protocol
URC	Unsolicited Result Code
USB	Universal Serial Bus
(U)SIM	(Universal) Subscriber Identity Module
UTC	Coordinated Universal Time