QUECTEL
Build a Smarter World

BC25&BC32 TCP/IP

AT Commands Manual

LPWA Module Series

Rev. BC25&BC32_TCP/IP_AT_Commands_Manual_V1.0

Date: 2019-07-19

Status: Released

www.quectel.com

Our aim is to provide customers with timely and comprehensive service. For any assistance, please contact our company headquarters:

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

Or our local office. 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 to: support@quectel.com

GENERAL NOTES

QUECTEL OFFERS THE INFORMATION AS A SERVICE TO ITS CUSTOMERS. THE INFORMATION PROVIDED IS BASED UPON CUSTOMERS’ REQUIREMENTS. QUECTEL MAKES EVERY EFFORT TO ENSURE THE QUALITY OF THE INFORMATION IT MAKES AVAILABLE. QUECTEL DOES NOT MAKE ANY WARRANTY AS TO THE INFORMATION CONTAINED HEREIN, AND DOES NOT ACCEPT ANY LIABILITY FOR ANY INJURY, LOSS OR DAMAGE OF ANY KIND INCURRED BY USE OF OR RELIANCE UPON THE INFORMATION. ALL INFORMATION SUPPLIED HEREIN IS SUBJECT TO CHANGE WITHOUT PRIOR NOTICE.

COPYRIGHT

About the Document

History

Revision	Date	Author	Description
1.0	2019-07-19	Quinten SONG	Initial

Contents

1 Introduction

Quectel BC25 and BC32 modules feature embedded TCP/IP stack, which enables the host to access the Internet directly over AT commands. This greatly reduces the dependence on PPP and TCP/IP protocol stacks and thus minimizes the cost.

BC25 and BC32 provide the following socket services: TCP client and UDP client.

1.1. The Process of Using TCP/IP AT Commands

Through TCP/IP AT commands, the host can open/close socket and send/receive data via socket.

1.2. Description of Data Access Modes

BC25 and BC32 support the following two data access modes:

- Buffer access mode
- Direct push mode

When opening a socket via AT+QIOPEN, the data access mode can be specified with the parameter <access_mode>. After a socket is opened, AT+QISWTMD could be used to change the data access mode. In both modes, data can be sent with AT+QISEND/AT+QISENDEX command.

1. In buffer access mode, the module will buffer the data and report a URC in the format of "+QIURC: "recv",<connectID>" when receiving data. The data can be read with AT+QIRD by the host.

NOTE

In buffer access mode, if the buffer is not empty, the module will not report a new URC until all received data have been read via AT+QIRD from buffer.

2. In direct push mode, the received data will be outputted directly in a URC in the following format: "+QIURC: "recv",<connectID>,<current_recv_length><CR><LF><data>".

2 Description of TCP/IP AT Commands

2.1. Description of AT Commands

2.1.1. AT+QIOPEN Open a Socket

This command is used to open a socket to open the connection. The service type can be specified with <service_type>, and the data access mode can be specified with <access_mode>. The URC "+QIOPEN: <connectID>,<err>" indicates whether the socket has been opened successfully.

AT+QIOPEN Open a Socket	
Test Command **AT+QIOPEN=?**	Response **+QIOPEN: (1-3),(0-5),"TCP/UDP/TCP LISTENER/UDP SERVICE","<IP_address>/<domain_name>",<remote_port>[,<local_port>[,(0-1)][,(0-1)]]]** **OK**
Write Command **AT+QIOPEN=<contextID>,<connectID>,"<service_type>","<IP_address>/<domain_name>",<remote_port>[,<local_port>[,<access_mode>[,<protocol_type>]]]**	Response **OK** **+QIOPEN: <connectID>,<err>** If there is an error related to ME functionality: **ERROR**
Maximum Response Time	300ms

Parameter

<contextID>	Integer type. Context ID. The range is 1-3.	
<connectID>	Integer type. Socket index. The range is 0-5.	
<service_type>	String type. Socket 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 TCP connection
	"UDP SERVICE"	Start a UDP service
<IP_address>	String type. IP address of remote server, such as "220.18.23.22".	

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

<remote_port> The port of the remote server, only valid when <service_type> is "TCP" or "UDP". The range is 1-65535.

<local_port> Local port number.
If <service_type> is "TCP LISTENER" or "UDP SERVICE", this parameter must be specified. The range is 1-65535.
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 assigned as specified. The range is 0-65535.

<access_mode> Integer type. Data access mode of sockets.
0 Buffer access mode
1 Direct push mode

<protocol_type> Integer type. Protocol type.
0 IPv4
1 IPv6

NOTES

1. It is recommended to wait 176 seconds for URC "+QIOPEN: <connectID>,<err>" if the connection has been established with parameter <domain_name>, and to wait 36 seconds for the URC if the connection has been established with parameter <IP_address>.
2. TCP LISTENER is not supported on IPv6.
3. UDP SERVICE is not supported now.
4. When the module wakes up from PSM, the socket needs to be re-opened with AT+QIOPEN.

2.1.2. AT+QICLOSE Close a Socket

The command is used to close the specified socket.

AT+QICLOSE Close a Socket	
Test Command **AT+QICLOSE=?**	Response **+QICLOSE: (0-5)** **OK**
Write Command **AT+QICLOSE=<connectID>**	Response If closed successfully: **OK** **CLOSE OK** If there is an error related to ME functionality: **ERROR**
Maximum Response Time	300ms

Parameter

<connectID>	Integer type. Socket index. The range is 0-5.

2.1.3. AT+QISTATE Query Socket Status

This command is used to query the socket status.

AT+QISTATE Query Socket Status	
Test Command **AT+QISTATE=?**	Response **OK**
Read Command **AT+QISTATE?**	Response Return the status of all existing connections: List of (**+QISTATE: <connectID>,<service_type>,<IP_address>,<remote_port>,<local_port>,<socket_state>,<contextID>,<access_mode>)** **OK** If there is an error related to ME functionality: **ERROR**
Write Command If <query_type> is 0, query connection status of the specified context **AT+QISTATE=<query_type>,<contextID>**	Response **+QISTATE: <connectID>,<service_type>,<IP_address>,<remote_port>,<local_port>,<socket_state>,<contextID>,<access_mode>** **OK** If there is an error related to ME functionality: **ERROR**
Write Command If <query_type> is 1, query connection status of the specified socket **AT+QISTATE=<query_type>,<connectID>**	Response **+QISTATE: <connectID>,<service_type>,<IP_address>,<remote_port>,<local_port>,<socket_state>,<contextID>,<access_mode>** **OK** If there is an error related to ME functionality: **ERROR**
Maximum Response Time	300ms

Parameter

<query_type>	Integer type. Query type. 0 Query connection status by <contextID> 1 Query connection status by <connectID>
<contextID>	Integer type. Context ID. The range is 1-3.
<connectID>	Integer type. Socket index. The range is 0-5.
<service_type>	String type. Service type. “TCP” TCP connection as a client “UDP” UDP connection as a client “TCP LISTENER” Start a TCP server to listen to TCP connection “TCP INCOMING” Start a TCP connection accepted by a TCP server “UDP SERVICE” Start a UDP service
<IP_address>	String type. IP address of remote client. If <service_type>=“TCP” or “UDP”, it is the IP address of remote server. If <service_type>=“TCP LISTENER” or “UDP SERVICE”, it is local IP address If <service_type>=“TCP INCOMING”, it is the IP address of remote client
<remote_port>	Integer type. Port number of remote server. 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, the value is always 0. If <service_type>=“TCP INCOMING”, it is the port of remote client
<local_port>	Integer type. Local port number assigned. If <local_port> is 0, local port will be assigned automatically
<socket_state>	Integer type. Socket state. 0 “idle”: client connection has not been established 1 “connecting”: client is connecting 2 “connected”: client connection has been established 3 “closing”: client connection is closing 4 “remote closing”: server connection is closing 5 “closed”: client connection is closed
<access_mode>	Data access mode. 0 Buffer access mode 1 Direct push mode

2.1.4. AT+QISEND Send Text String Data

The command is used to send socket data in text string format via specified connection.

AT+QISEND Send Text String Data	
Test Command **AT+QISEND=?**	Response **+QISEND: (0-5),(1-512),"<data>"**

	OK
Write Command **AT+QISEND=<connectID>,<send_length>,"<data>"**	Response If data is sent successfully: **OK** **SEND OK** Otherwise: **OK** **SEND FAIL** If there is an error related to ME functionality: **ERROR**
Write Command Send data with changeable length **AT+QISEND=<connectID>** After the response ">", input <data> to be sent. Tap Ctrl+Z to send, and tap Esc to cancel the operation	Response **> <data>** **OK** If connection has been established and sending is successful: **SEND OK** If connection has been established but sending buffer is full or send fail: **SEND FAIL** If connection has not been established, abnormally closed, or parameter is incorrect: **ERROR**
Write Command Send data with fixed length **AT+QISEND=<connectID>,<send_length>** After the response ">", input <data> until the data length is equal to <send_length>	Response **> <data>** If connection has been established and sending is successful: **OK** **SEND OK** If connection has been established but sending buffer is full: **OK** **SEND FAIL** If connection has not been established, abnormally closed,

	or parameter is incorrect: **ERROR**
Maximum Response Time	300ms

Parameter

<connectID>	Integer type. Socket index. The range is 0-5.
<send_length>	Integer type. The length of data to be sent. The max length is 512 bytes.
<data>	The text or hex string data to be sent.

NOTES

1. "SEND OK" only indicates that the data has arrived the protocol stack.
2. <data> can be sent successfully only when the <service_type> of socket checked out with AT+QISTATE command is "TCP", "UDP" or "TCP INCOMING", if it is "TCP LISTENER" or "UDP SERVICE", ERROR will be returned.
3. In data mode after ">" is responded, if the sent data is empty, "SEND FAIL" will be returned immediately after tapping CTRL+Z.

2.1.5. AT+QIRD Retrieve the Received TCP/IP Data

This command is used to read the received socket data from a specified connection.

In buffer access mode, after receiving data, the module will buffer it and then report URC "+QIURC: "recv",<connectID>[,<current_recv_length>]" to external MCU.

AT+QIRD Retrieve the Received TCP/IP Data	
Test Command **AT+QIRD=?**	Response **+QIRD: (0-5),(1-512)** **OK**
Write Command When <service_type> is "TCP", "UDP" or "TCP INCOMING" **AT+QIRD=<connectID>,<read_length>**	Response **+QIRD: <actual_read_length>[,<remaining_length>]** **<data>** **OK** If there is no data: **+QIRD: 0** **OK**

	If there is an error related to ME functionality: **ERROR**
Maximum Response Time	300ms

Parameter

<connectID>	Integer type. Socket index. The range is 0-5.
<read_length>	Integer type. The maximum length of data to be retrieved. The range is 1-512. Unit: byte.
<actual_read_length>	Integer type. The actual length of received data. Unit: byte.
<remaining_length>	The remaining length of last received data. Unit: byte.
<data>	The retrieved data.

NOTES

1. If the module receives data when the receive buffer is not empty, new URC will not be reported until all the received data has been retrieved from the buffer.
2. If "AT+QICFG="showlength",1" is configured, then <current_recv_length> and <remaining_length> will be prompted, please refer to ***Chapter 2.1.11***.
3. The remaining length is not the total received bytes in the buffer but only indicates the current remaining data stored in one node.

2.1.6. AT+QISENDEX Send Hex String Data

This command can be used to send socket data in hex string format via specified connection.

AT+QISENDEX Send Hex String Data	
Test Command **AT+QISENDEX=?**	Response **+QISENDEX: (0-5),(1-512),"<hex_string>"** **OK**
Write Command **AT+QISENDEX=<connectID>,<send_length>,"<hex_string>"**	Response If the hex string data is sent successfully: **OK** **SEND OK** Otherwise: **OK** **SEND FAIL**

	If there is an error related to ME functionality: **ERROR**
Write Command Send data with changeable length **AT+QISENDEX=<connectID>** After response “>”, input <hex_string> to be sent. Tap Ctrl+Z to send, and tap Esc to cancel the operation	Response **> <hex_string>** **OK** If connection has been established and sending is successful: **SEND OK** If connection has been established but sending buffer is full or sending fails: **SEND FAIL** If connection has not been established, abnormally closed, or parameter is incorrect: **ERROR**
Write Command Send data with fixed length **AT+QISENDEX=<connectID>,<send_length>** After response “>”, input <hex_string> until the data length is equal to <send_length>	Response **> <hex_string>** If connection has been established and sending is successful: **OK** **SEND OK** If connection has been established but sending buffer is full: **OK** **SEND FAIL** If connection has not been established, abnormally closed, or parameter is incorrect: **ERROR**
Maximum Response Time	300ms

Parameter

<connectID>	Integer type. Socket index. The range is 0-5.
<send_length>	Integer type. The length of data to be sent, and the max length is 512 bytes.
<hex_string>	The hex string data to be sent.

NOTES

1. "SEND OK" only indicates that the data arrives the protocol stack.
2. <hex_string> can be sent successfully only when the <service_type> of socket checked out with AT+QISTATE command is "TCP", "UDP" or "TCP INCOMING", if it is "TCP LISTENER" or "UDP SERVICE", ERROR will be returned.
3. In data mode after ">" is responded, if the sent data is empty, "SEND FAIL" will be returned immediately after tapping CTRL+Z.

2.1.7. AT+QISWTMD Switch Data Access Modes

This command can be used to switch between two data access modes: buffer access mode and direct push mode. When opening a new socket, the data access mode can be specified with the parameter <access_mode> of AT+QIOPEN.

AT+QISWTMD Switch Data Access Modes	
Test Command **AT+QISWTMD=?**	Response **+QISWTMD: (0-5),(0-1)** **OK**
Read Command **AT+QISWTMD?**	Response **OK**
Write Command **AT+QISWTMD=<connectID>,<access_mode>**	Response **OK** If there is an error related to ME functionality: **ERROR**
Maximum Response Time	300ms

Parameter

<connectID>	Integer type. Socket index. The range is 0-5.	
<access_mode>	Integer type. The data access modes of the socket.	
	0	Buffer access mode
	1	Direct push mode

NOTES

1. The switch of data access mode will take effect immediately.
2. The configuration of <access_mode> will be saved to NVRAM automatically.

2.1.8. AT+QPING Ping a Remote Server

This command is used to test the Internet protocol reachability of a host device.

AT+QPING Ping a Remote Server	
Test Command **AT+QPING=?**	Response **+QPING: (1-3),"<host>"[,(1-255)[,(1-10)[,(32-200)]]]** **OK**
Write Command **AT+QPING=<contextID>,"<host>"[,<time_out>[,<ping_num>[,<ping_size>]]]**	Response If a remote server is pinged successfully: **OK** **+QPING: <result>[,<IP_address>,<bytes>,<time>,<ttl>]** **[…]** **+QPING: <finresult>[,<sent>,<rcvd>,<lost>,<min>,<max>,<avg>]** If there is an error related to ME functionality: **ERROR**
Maximum Response Time	300ms

Parameter

<contextID> Integer type. Context ID. The range is 1-3.

<host> The host address in string type. The format is a domain name or a dotted decimal IP address

<time_out> Integer type. The maximum time to wait for the response of each ping request. Unit: second. Range: 1-255. Default: 4.

<ping_num> Integer type. The maximum time of ping request. Range: 1-10. Default: 4.

<ping_size> Integer type. The ping size. Range: 32-200. Default: 32.

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

0	Recv response of remote server. Display: <IP_address>,<bytes>,<time>,<ttl>
1	Ping timeout.
Others	Refer to ***Chapter 3*** for specified error codes.

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

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

<time> Integer type. The time consuming of the ping request. Unit: ms.

<ttl> Integer type. The time to live value of the ping request.

<finresult>	Integer type. The final result of the ping operation. 2 Pinged successful Others Refer to ***Chapter 3*** for specified error codes.
<sent>	Integer type. The total number of bytes sent by the ping requests.
<rcvd>	Integer type. The total number of bytes received in the ping response.
<lost>	Integer type. The total number of bytes lost in the ping requests.
<min>	Min response time. Unit: ms.
<max>	Max response time. Unit: ms.
<avg>	Average response time. Unit: ms..

2.1.9. AT+QNTP Synchronize Local Time with NTP Server

This command is used to synchronize the local time with the Coordinated Universal Time (UTC) via the NTP server.

Please refer to ***Chapter 3*** for specific <err> codes.

AT+QNTP Synchronize Local Time with NTP Server	
Test Command **AT+QNTP=?**	Response **+QNTP: (1-3),"<server>"[,<port>[,(0-1)]]** **OK**
Write Command **AT+QNTP=<contextID>,"<server>"[,<port>[,<autosettime>]]**	Response If successfully synchronized: **OK** **+QNTP: <err>,<time>** If there is an error related to ME functionality: **ERROR**
Maximum Response Time	300ms

Parameter

<contextID>	Integer type. Context ID. The range is 1-3.
<server>	String type. The address of NTP server.
<port>	Integer type. The port of NTP server.
<autosettime>	Integer type. Whether to automatically synchronize local time with UTC. 0 Not synchronize 1 Synchronize
<time>	String type. The time synchronized from NTP server. The format is "YYYY/MM/DD,hh:mm:ss±zz".

	The range of "zz" is -48 ~ 56.

NOTE

When <autosettime> is set to 1, RTC will be updated to the synchronized time automatically.

2.1.10. AT+QIDNSGIP Get IP Address by Domain Name

This command is used to covert the specified domain name to IP address format.

Please refer to ***Chapter 3*** for specific <err> codes.

AT+QIDNSGIP Get IP Address by Domain Name	
Test Command **AT+QIDNSGIP=?**	Response **+QIDNSGIP: (1-3),"<hostname>"** **OK**
Write Command **AT+QIDNSGIP=<contextID>,"<hostname>"**	Response **OK** **+QIURC: "dnsgip",<err>,<IP_count>,<DNS_ttl>** **[+QIURC: "dnsgip",<hostIPaddr>]** **[…]** If there is an error related to ME functionality: **ERROR**
Maximum Response Time	300ms

Parameter

<contextID>	Integer type. Context ID. The range is 1-3.
<hostname>	String type. Domain name.
<IP_count>	Integer type. The number of the IP addresses corresponding to the <hostname>.
<DNS_ttl>	Integer type. The time to live of the DNS.
<hostIPaddr>	String type. The IP address of <hostname>.

2.1.11. AT+QICFG Configure Optional Parameters

The command is used to configure optional parameters for TCP/IP functionalities.

AT+QICFG Configure Optional Parameters	
Test Command **AT+QICFG=?**	Response **+QICFG: "dataformat",(0,1),(0,1)** **+QICFG: "viewmode",(0,1)** **+QICFG: "showlength",(0,1)** **OK**
Write Command Set the data format for sending and receiving **AT+QICFG="dataformat"[,<send_data_format>,<recv_data_format>]**	Response **+QICFG: "dataformat",<send_data_format>,<recv_data_format>** **OK** If there is an error related to ME functionality: **ERROR**
Write Command Set the received data output format **AT+QICFG="viewmode"[,<view_mode>]**	Response **+QICFG: "viewmode",<view_mode>** **OK** If there is an error related to ME functionality: **ERROR**
Write Command Set the optional length viewable in buff access mode **AT+QICFG="showlength"[,<show_length_mode>]**	Response **+QICFG: "showlength",<show_length_mode>** **OK** If there is an error related to ME functionality: **ERROR**
Maximum Response Time	300ms

Parameter

<send_data_format>	Integer type. Sending data format. 0 Text mode 1 Hex mode
<recv_data_format>	Integer type. Receiving data format. 0 Text mode 1 Hex mode

<view_mode>	Integer type. Received data output format. 0 Received data output format: data header\r\ndata 1 Received data output format: data header,data
<show_length_mode>	Integer type. Whether to prompt the optional length parameters in buffer access mode. 0 Do not show the optional length in buffer access mode 1 Show the optional length in buffer access mode

NOTES

1. The configuration of the parameters will take effect immediately.
2. The configuration of the parameters <send_data_format>, <recv_data_format>, <view_mode> as well as <show_length_mode> will be saved to NVRAM automatically.
3. Currently <send_data_format> can only be configured in Text mode, for Hex mode, it can be configured with AT+QISENDEX.
4. Optional length parameters of <show_length_mode> refer to <current_recv_length> of URC indicating incoming data and <remaining_length> of AT+QIRD command
5. Please refer to AT+QIRD command for <show_length_mode> configuration.

2.1.12. AT+QIGETERROR Query the Last Error Code

This command is used to query the <err> code and specific description of the <err> code returned by the last TCP/IP command.

AT+QIGETERROR Query the Last Error Code	
Test Command **AT+QIGETERROR=?**	Response **OK**
Execution Command **AT+QIGETERROR**	Response **+QIGETERROR: <err>,<errcode_description>** **OK** If there is an error related to ME functionality: **ERROR**
Maximum Response Time	300ms

Parameter

<errcode_description>	A string parameter indicates the details of error information. Please refer to ***Chapter 3*** for details of <err> codes and corresponding description.

2.1.13. AT+QIKALIVE Configure TCP Socket Keepalive Function

This command is used to config whether to enable TCP socket keepalive function and the parameters

AT+QIKALIVE Configure TCP Socket Keepalive Function	
Test Command **AT+QIKALIVE=?**	Response **+QIKALIVE: <keepalive_on>[,<keepalive_idle_time>[,<keepalive_interval_time>[,<keepalive_count>,["<keepalive_data>"]]]]** **OK**
Read Command **AT+QIKALIVE?**	Response **+QIKALIVE: <keepalive_on>,<keepalive_idle_time>,<keepalive_interval_time>,<keepalive_count>** **OK** If there is an error related to ME functionality: **ERROR**
Execution Command **AT+QIKALIVE=<keepalive_on>[,<keepalive_idle_time>[,<keepalive_interval_time>[,<keepalive_count>,["<keepalive_data>"]]]]**	Response **OK** If there is an error related to ME functionality: **ERROR**

Parameters

<keepalive_on>	Integer type. Configure whether to enable the keepalive function 0 not enable 1 enable
<keepalive_idle_time>	Integer type. Indicates how long is the interval between the two keepalive message. The range is 1-36000. Default value: 60. Unit: s.
<keepalive_interval_time>	Integer type. Indicates that when enabling keepalive function, if the module does not receive the ack from the service within <keepalive_interval_time> after sending the keepalive message, the module will resend the keepalive message. The range is 10-20. Default value: 10. Unit: s.
<keepalive_count>	Integer type. Total times to resend the keepalive package when the module cannot receive the ack. Default value: 9.
<keepalive_data>	Hexadecimal type. Indicates the data carried in keepalive packages. The maximum length is 256.

NOTE

Parameter <keepalive_data> is not supported now.

2.1.14. AT+QIDNSCFG Configure DNS Server Function

This command is used to configure DNS server functions and parameters.

AT+QIDNSCFG Configure DNS Server Function	
Test Command **AT+QIDNSCFG=?**	Response **+QIDNSCFG: <PrimaryDns>[,<SecondaryDns>]** **OK**
Read Command **AT+QIDNSCFG?**	Response **+QIDNSCFG: <PrimaryDns>,<SecondaryDns>** **OK** If there is an error related to ME functionality: **ERROR**
Write Command **AT+QIDNSCFG=<PrimaryDns>[,<SecondaryDns>]**	Response **OK** If there is an error related to ME functionality: **ERROR**
Maximum Response Time	300ms

Parameters

<PrimaryDns>	String type. The IP address of primary DNS server, such as “220.18.23.22”.
<SecondaryDns>	String type. The IP address of alternate DNS server, such as “220.18.23.22”.

NOTE

The DNS server cannot be configured until the module is successfully registered on network.

2.2. Description of URC

The URC of TCP/IP AT commands will be reported in the format "<CR><LF>+QIURC:<type>[…]<CR><LF>". For convenience purpose, <CR><LF> before/after the URC is omitted intentionally.

2.2.1. URC Indicating Connection Closed

When TCP socket is closed by remote peer or due to network error, the URC "+QIURC: "closed",<connectID>" will be outputted, and the <socket_state> (indicating the status of the socket) will be changed to "closing". The host must execute "AT+QICLOSE=<connectID>" to change the <socket_state> to "initial".

In buffer access mode, the host can also execute "AT+QIRD=<connextID>,<read_length>" to read the buffer data.

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

Parameter

<connectID>	Integer type. The socket index. The range is 0-5.

2.2.2. URC Indicating Incoming Data

In buffer access mode or the direct push mode, the module will report URC to the host when data is received from the server.

- In buffer access mode, the URC format is:
  +QIURC: "recv",<connectID>[,<current_recv_length>]

- In direct push mode, the URC format is:
  +QIURC: "recv",<connectID>,<current_recv_length><CR><LF><data>

URC of Incoming Data	
+QIURC: "recv",<connectID>,[<current_recv_length>]	The URC indicating incoming data in buffer access mode.
+QIURC: "recv",<connectID>,<current_recv_length><CR><LF><data>	The URC indicating incoming data in direct push mode.

Parameter

<connectID>	Integer type. The socket index. The range is 0-5.

<current_recv_length>	Integer type. The length of actual received data.
<data>	The received data.

2.2.3. URC Indicating Incoming Data Buffer Full

In buffer access mode, if there is no resource that can be allocated for incoming data, then the module will report the following URC.

URC of Incoming Data Buffer Full	
+QIURC: "recv",<connectID>,"buff full"	The URC of incoming data is full.

Parameter

<connectID>	Integer type. The socket index. The range is 0-5.

2.2.4. URC of Incoming Connection Full

If the incoming connection reaches the limit, or no socket system resource can be allocated, then the module will report the URC "+QIURC: "incoming full"" when a new incoming connection is requested.

URC of Incoming Connection Full	
+QIURC: "incoming full"	Incoming connection is full.

2.2.5. URC of Incoming Connection

If the <service_type> is "TCP LISTENER", when a remote client connects to this server, the module will automatically assign a free <connectID> for the new connection. The range of <connectID> is 0-5. 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.

URC of 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 for the incoming connection, which is automatically specified by the module. The range is 0-5.
<serverID>	The incoming <connectID> accepted by the server whose <service_type> is "TCP LISTENER" and listening socket ID is <serverID>.

<remoteIP>	Remote IP address of the incoming <connectID>.
<remote_port>	Remote port of the incoming <connectID>.

3 Summary of <err> Codes

If <err> is returned after executing TCP/IP AT commands, details of the error can be queried via AT+QIGETERROR. Please note that only <err> of the last TCP/IP AT command will be returned by executing AT+QIGETERROR.

Table 1: Summary of Error Codes

<err> Code	Description of Error Code
0	Operation successful
550	Unknown error
551	Operation blocked
552	Invalid parameters
553	Memory not enough
554	Create socket 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	Open PDP context failed
562	Close PDP context failed
563	Socket identity has been used
564	DNS busy

565	DNS parse failed
566	Socket connect failed
567	Socket has been closed
568	Operation busy
569	Operation timeout
570	PDP context broken down
571	Cancel send
572	Operation not allowed
573	APN not configured
574	Port busy

Table 2: Summary of ping result

<err> Code	Description of Error Code
0	PING_ECHO
1	PING_TIMEOUT
2	PING_SUCCESS
3	PING_TCPIP_ERROR
4	PING_ADDRESS_NOT_FOUND
5	PING_PDP_ACT_FAIL

4 Examples

4.1. TCP Client Works in Buffer Access Mode

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

```
AT+QIOPEN=1,0,"TCP","220.18.39.22",8062,1234,0      //Context ID is 1 and the socket index is 0.
OK

+QIOPEN: 0,0                              //Connected successfully. It is suggested to wait for 36
                                            seconds for the URC to be reported.

AT+QISTATE=1,0                            //Query the connection status of socket index 0.
+QISTATE: 0,"TCP","220.18.39.22",8062,1234,2,1,0

OK
```

4.1.2. Send Data in Buffer Access Mode

```
AT+QISEND=0,10,”1234567890”               //Send data, and the data length is 10 bytes.
OK

SEND OK

AT+QISENDEX=0,5,”3031323334”              //Send hex string data.
OK

SEND OK

AT+QISEND=0,10,”1234567890”               //Send data, and the data length is 10 bytes.
OK

AT+QISEND=0,10,”1234567890”               //“SEND OK” of the previous command has not been
                                            returned, so when new data is sent, "ERROR" is returned.
ERROR

SEND OK
```

4.1.3. Receive Data from Remote Server in Buffer Access Mode

```
+QIURC: "recv",0                //The <connectID> 0 received data.

AT+QIRD=0,512                   //Retrieve the data in the buffer, and the data length is 512 bytes.
+QIRD: 10
1234567890

OK

AT+QIRD=0,512                   //Retrieve the data in the buffer, and the data length is 512 bytes.
+QIRD: 0                        //No data in buffer.

OK

AT+QICFG="showlength",1         //Enable to show optional length viewable in buff access mode.
+QICFG: "showlength",1

OK

+QIURC: "recv",0,12             //The <connectID> 0 received data, and the data length is 12 bytes.

AT+QIRD=0,10                    //Retrieve the data, and the data length is 10 bytes.
+QIRD: 10,2                     //Read 10 bytes, and remains 2 bytes.
1234567890

OK

+QIURC: "recv",0,"buff full"    //The <connectID> 0 reports that the buffer is full, and the host has to
                                  use AT+QIRD to read the buffer data.

AT+QICFG="viewmode",1           //Received data output format: data header,data
+QICFG: "viewmode",1

OK

AT+QISEND=0,12,"012345678901"
OK

SEND OK

+QIURC: "recv",0,12

AT+QIRD=0,10
+QIRD: 10,2,0123456789
```

```
OK
```

4.1.4. Close a Connection

```
AT+QICLOSE=0                    //Close a connection whose <connectID> is 0.
OK

CLOSE OK
```

4.2. TCP Client Works in Direct Push Mode

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

```
AT+QIOPEN=1,0,"TCP","220.18.39.22",8062,0,1          //Context ID is 1 and the socket index is 0.
OK

+QIOPEN: 0,0                    //Connected successfully. It is suggested to wait for 36 seconds for the
                                 URC to be reported.

AT+QISTATE=1,0                  //Query the connection status of socket index 0.
+QISTATE: 0,"TCP","220.18.39.22",8062,0,2,1,1

OK
```

4.2.2. Send Data in Direct Push Mode

```
AT+QISEND=0,10,"1234567890"             //Send data, and the data length is 10 bytes.
OK

SEND OK

AT+QISENDEX=0,5,"3031323334"            //Send hex string data.
OK

SEND OK
```

4.2.3. Receive Data from Remote Server in Direct Push Mode

```
+QIURC: "recv",0,5                      //Receive data from remote server.
12345

AT+QICFG="viewmode",1                   //Received data output format: data header,data
+QICFG: "viewmode",1
```

```
OK

AT+QISEND=0,12,"012345678901"
OK

SEND OK

+QIURC: "recv",0,12,012345678901
```

4.2.4. Close TCP Client

```
AT+QICLOSE=0                              //Close a connection whose <connectID> is 0.
OK

CLOSE OK
```

4.3. TCP Server Works in Buffer Access Mode

4.3.1. Start a TCP Server

```
AT+QIOPEN=1,0,"TCP LISTENER","192.168.2.6",0,2020,0   //Context ID is 1 and the socket index is 0.
                                                        Before using AT+QIOPEN, the host
                                                        should activate the context with
                                                        AT+CGACT first.
OK

+QIOPEN: 0,0                                          //TCP server is opened successfully.
AT+QISTATE?                                           //Query connection status of context 0
+QISTATE: 0,"TCP LISTENER","192.168.2.6",0,2020,2,1,0

OK
```

4.3.2. Accept TCP Incoming Connection

```
+QIURC: "incoming",2,0,"192.168.2.2",36566    //Accept a TCP connection, <service_type> is "TCP
                                                incoming", and <connectID> is 2.
```

4.3.3. Receive Data from Incoming Connection

```
+QIURC: "recv",0                              //Received data from remote incoming connection.
AT+QIRD=0,512                                 //Read data received from the incoming connection.
+QIRD: 4                                      //Actual data length is 4 bytes.
test                                          //The data
```

```
OK
AT+QIRD=0,512
+QIRD: 0                                  //No data in buffer.

OK
```

4.3.4. Close a TCP Server

```
AT+QICLOSE=2                           //Close incoming connection. Depending on the network,
                                         the maximum response time is 10s.
OK

CLOSE OK
AT+QICLOSE=0                             //Close TCP server listening.
OK

CLOSE OK
```

4.4. Example of UDP Service

4.4.1. Start a UDP Service

```
AT+QIOPEN=1,0,"UDP SERVICE","192.168.2.6",0,2020,0   //Start a UDP service, Context ID is 1 and
                                                      the socket index is 0. Before using
                                                      AT+QIOPEN, the host should activate
                                                      the context with AT+CGACT first.
OK

+QIOPEN: 0,0                                          //Start UDP service successfully.
AT+QISTATE?                                           //Query connection status of context 0
+QISTATE: 0,"UDP SERVICE","192.168.2.6",0,2020,2,1,0

OK
```

4.4.2. Receive Data from Remote

```
+QIURC: “recv”,2                        //Receive data from remote incoming connection.
AT+QIRD=2,4                             //Read UDP data. One whole UDP packet will be
                                          outputted. There is a need to specify the read
                                          length.
+QIRD: 4                                  //Data length is 4
AAAA
```

```
OK
```

4.4.3. Close a UDP Service

```
AT+QICLOSE=0                                 //Close the service.
OK

CLOSE OK
```

4.5. Ping a Remote Server

```
AT+QPING=1,"sh.quectel.com"                  //Ping sh.quectel.com in the context 1.
OK

+QPING: 0,220.18.29.21,32,192,255

+QPING: 0,220.18.23.21,32,240,255

+QPING: 0,220.18.23.21,32,241,255

+QPING: 0,220.18.23.21,32,479,255

+QPING: 2,4,4,0,192,479,287
```

4.6. Synchronize Local Time

```
AT+QNTP=1,"ntp5.aliyun.com"          //Synchronize local time with NTP server ntp5.aliyun.com.
OK

+QNTP: 0,"18/04/20,11:08:20+32"
```

4.7. Getting the Last Error Code

```
AT+QIOPEN=1,"UDP","220.18.39.22",8063,0,1        //Start socket and omit <connectID>.
ERROR

AT+QIGETERROR
+QIGETERROR: 552,invalid parameters
```

```

OK
```

5 Appendix A Reference

Table 1: Terms and Abbreviations

Abbreviation	Description
DNS	Domain Name System
IP	Internet Protocol
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
ME	Mobile Equipment
NTP	Network Time Protocol
NVRAM	Non-Volatile Radon Access Memory
PPP	Point to Point Protocol
PSM	Power Saving Mode
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
URC	Unsolicited Result Code
UTC	Universal Time Coordinated