BC660K-GL TCP/IP Application Note

NB-IoT Module Series

Version: 1.1

Date: 2021-07-22

Status: Released

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 200233, China
Tel: +86 21 5108 6236
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.

Disclaimer

While Quectel has made efforts to ensure that the functions and features under development are free from errors, it is possible that these functions and features could contain errors, inaccuracies and omissions. Unless otherwise provided by valid agreement, Quectel makes no warranties of any kind, implied or express, with respect to the use of features and functions under development. To the maximum extent permitted by law, Quectel excludes all liability for any loss or damage suffered in connection with the use of the functions and features under development, regardless of whether such loss or damage may have been foreseeable.

Duty of Confidentiality

The Receiving Party shall keep confidential all documentation and information provided by Quectel, except when the specific permission has been granted by Quectel. The Receiving Party shall not access or use Quectel's documentation and information for any purpose except as expressly provided herein. Furthermore, the Receiving Party shall not disclose any of the Quectel's documentation and information to any third party without the prior written consent by Quectel. For any noncompliance to the above requirements, unauthorized use, or other illegal or malicious use of the documentation and information, Quectel will reserve the right to take legal action.

Copyright

About the Document

Revision History

Version	Date	Author	Description
-	2020-11-18	Quinten SONG	Creation of the document
1.0	2021-01-20	Quinten SONG	First official release
1.1	2021-07-22	Arno DONG	1. Added the socket service type "TCP LISTENER". 2. Updated the range of <contextID>. 3. Updated the description of parameters <remote_port> and <local_port> of AT+QIOPEN (Chapter 2.3.1). 4. Updated the description of parameters <host> and <remote_port> of AT+QISTATE (Chapter 2.3.3). 5. Updated the description of AT+QISEND and related NOTE (Chapter 2.3.4). 6. Added <rai_enable> of AT+QPING (Chapter 2.3.5). 7. Added the URC +QIURC: "incoming" (Chapter 2.4.3). 8. Added an example of TCP server in direct push mode (Chapter 4.3).

Contents

Table Index

1 Introduction

The Quectel BC660K-GL module features an embedded TCP/IP stack, which enables the host to access the Internet directly via AT commands, thus greatly reducing the dependence on PPP and external TCP/IP protocol stacks and lowering costs.

The module provides the following socket services: TCP client, UDP client, TCP listener and UDP service.

1.1. Usage of TCP/IP AT Commands

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

1.2. Description of Data Access Mode

The module supports one data access mode – the direct push mode.

When you open a socket with **AT+QIOPEN**, the value of **<access_mode>**, specified or not, is always 1 (the direct push mode).

In the direct push mode, data can be sent with **AT+QISEND**, and the received data are outputted directly via this URC: **+QIURC: "recv",<connectID>,<current_recv_length>,<data>**.

2 TCP/IP AT Commands

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 to 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: Types of AT Commands

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

2.2. Declaration of AT Command Examples

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

2.3. Description of AT Commands

2.3.1. AT+QIOPEN Open a Socket Service

This command opens a socket service. The service type can be specified by **<service_type>**. The URC **+QIOPEN: <connectID>,<result>** is reported to indicate whether the socket service has been opened successfully.

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",<host>,(**range of supported **<remote_port>**s**),(**range of supported **<local_port>**s**),(**list of supported **<access_mode>**s**)** **OK**
Write Command **AT+QIOPEN=<contextID>,<connectID>,<service_type>,<host>,<remote_port>[,<local_port>[,<access_mode>]]**	Response **OK** **+QIOPEN: <connectID>,<result>** If there is any error: **ERROR**
Maximum Response Time	5 s
Characteristics	The command takes effect immediately. Remain valid after deep-sleep wakeup, but the configurations of **<host>**, **<remote_port>**, **<local_port>**, **<access_mode>** will not be saved to NVRAM.

Parameter

<contextID>	Integer type. Context ID. Range: 0–10. 0 Automatically adapt to the current default PDP context 1–10 Specify a PDP context
<connectID>	Integer type. Socket ID. Range: 0–4.
<service_type>	String type. Socket service type. "TCP" Start a TCP connection as a client "UDP" Start a UDP connection as a client "TCP LISTENER" Start a TCP server to listen for TCP incoming connections "UDP SERVICE" Start a UDP service
<host>	String type. The IP address or domain name of the remote server. The maximum size is 150 bytes.
<remote_port>	Integer type. Port number of the remote server. Range: 1–65535.
<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 (i.e., set to the default 0), indicating the local port is assigned automatically. Otherwise, the local port is assigned as specified. It is recommended to assign a port of more than 5 bits and avoid using the default port(s) of special protocol(s).
<access_mode>	Integer type. The data access mode of a socket service. 1 Direct push mode
<result>	Integer type. The result code. See ***Chapter 3*** for details.

NOTE

1. Currently, only **<contextID>**=0 is supported.
2. It is recommended to wait for 60 seconds for the URC **+QIOPEN: <connectID>,<result>** to return.
3. If the connection fails, **AT+QICLOSE=<connectID>** should be executed to close the socket.
4. This command should be executed after the IP address URC (e.g. **+IP: 10.18.237.42**, indicating the client has successfully registered to the network) is reported.
5. When a UDP session is created, the module can automatically backup the latest UDP configurations, and the MCU can send/receive data directly after being woken up from sleep.
6. If **<local_port>** is set to the specified local port number, after the socket is closed with **AT+QICLOSE**, it is recommended to wait for 120 seconds before reusing **AT+QIOPEN**.
7. Only direct push mode is supported while buffer access mode is not supported by the module.

2.3.2. AT+QICLOSE Close a Socket Service

This command closes a specified socket service.

AT+QICLOSE Close a Socket Service	
Test Command **AT+QICLOSE=?**	Response **+QICLOSE: (**range of supported **<connectID>**s**)** **OK**
Write Command **AT+QICLOSE=<connectID>**	Response If closed successfully: **OK** **CLOSE OK** If there is any error: **ERROR**
Maximum Response Time	5 s
Characteristics	-

Parameter

<connectID>	Integer type. Socket ID. Range: 0–4.

2.3.3. AT+QISTATE Query Socket Service Status

This command queries the socket service status.

AT+QISTATE Query Socket Service Status	
Test Command **AT+QISTATE=?**	Response **+QISTATE: 0,(**range of supported **<contextID>**s**)** **+QISTATE: 1,(**range of supported **<connectID>**s**)** **OK**
Read Command **AT+QISTATE?**	Response Return the status of all existing connections: **[+QISTATE: <connectID>,<service_type>,<host>,<remote_port>,<local_port>,<socket_state>,<contextID>,<access_mode>]** **[…]**

	OK If there is any error: **ERROR**
Write Command When **<query_type>**=0, check the connection status of a specified context: **AT+QISTATE=<query_type>,<contextID>**	Response Return the status of all existing connections under the specified context: **[+QISTATE: <connectID>,<service_type>,<host>,<remote_port>,<local_port>,<socket_state>,<contextID>,<access_mode>]** **[…]** **OK** If there is any error: **ERROR**
Write Command When **<query_type>**=1, check the connection status of a specified socket service: **AT+QISTATE=<query_type>,<connectID>**	Response Return the connection status of a specified socket service: **[+QISTATE: <connectID>,<service_type>,<host>,<remote_port>,<local_port>,<socket_state>,<contextID>,<access_mode>]** **OK** If there is any error: **ERROR**
Maximum Response Time	5 s
Characteristics	-

Parameter

<query_type>	Integer type. Query type.	
	0	Query connection status by **<contextID>**
	1	Query connection status by **<connectID>**
<contextID>	Integer type. Context ID. Range: 0–10.	
	0	The current default PDP context
	1–10	A specified PDP context
<connectID>	Integer type. Socket ID. Range: 0–4.	
<service_type>	String type. Service type.	
	"TCP"	TCP connection as a client
	"UDP"	UDP connection as a client

	"TCP LISTENER"	TCP server to listen for TCP incoming connections
	"TCP INCOMING"	TCP connection accepted by a TCP server
	"UDP SERVICE"	UDP service
<host>	String type. The IP address or domain name of the remote server. The maximum size is 150 bytes. If **<service_type>** is "TCP" or "UDP", it is the IP address or domain name of remote server. If **<service_type>** is "TCP LISTENER" or "UDP SERVICE", it is the local IP address. If **<service_type>** is "TCP INCOMING", it is the IP address of the remote client.	
<remote_port>	Integer type. Port number of the remote server. If **<service_type>** is "TCP" or "UDP", it is the port of the remote server. If **<service_type>** is "TCP LISTENER", the port is invalid, the value is always 0. If **<service_type>** is "TCP INCOMING", it is the port of the remote client.	
<local_port>	Integer type. Local port number assigned.	
<socket_state>	Integer type. Socket service state. 0 "Initial": the client connection has not been established 1 "Connecting": the client is connecting 2 "Connected": the client connection has been established 3 "Closing": the client connection is closing 4 "Remote Closing": the client connection is being closed by a remote server	
<access_mode>	Integer type. Data access mode. 1 Direct push mode	

NOTE

1. Currently, only **<contextID>**=0 is supported.
2. If no list of **+QISTATE:** is displayed in the response, there is no connection.

2.3.4. AT+QISEND Send Hex/Text String Data

This command sends socket data in hex/text string format via a specified connection.

AT+QISEND Send Hex/Text String Data	
Test Command **AT+QISEND=?**	Response **+QISEND: (**range of supported **<connectID>**s**),<remote_ip>,<remote_port>,(**range of supported **<send_length>**s**), <data>,(**range of supported **<rai_mode>**s**)** **OK**
Write Command Send data in non-data mode when	Response If the data is sent successfully:

<service_type> is "TCP" or "UDP" or "TCP INCOMING" **AT+QISEND=<connectID>,<send_length>,<data>[,<rai_mode>]**	**OK** **SEND OK** If the command is executed successfully but the data sending is failed: **OK** **SEND FAIL** If there is any other error: **ERROR**
Write Command Send data of variable lengths in data mode when **<service_type>** is "TCP" or "UDP" or "TCP INCOMING" **AT+QISEND=<connectID>**	Response **>** Upon receiving the response **>**, the module enters data mode. When the input data length reaches the maximum length (1024 bytes) or when you tap "**Ctrl**" **+** "**Z**", the data will be sent out; If you tap "**Esc**", the sending will be canceled. If the data is sent successfully: **OK** **SEND OK** If the command is executed successfully but the data sending is failed: **OK** **SEND FAIL** If there is any other error: **ERROR**
Write Command Send data of a fixed-length in data mode when **<service_type>** is "TCP" or "UDP" or "TCP INCOMING" **AT+QISEND=<connectID>,<send_length>**	Response **>** Upon receiving the response **>**, the module enters data mode. Then, type the data to be sent until the data length reaches the value of **<send_length>**. If the data is sent successfully: **OK** **SEND OK** If the command is executed successfully but the data sending

	is failed: **OK** **SEND FAIL** If there is any other error: **ERROR**
Write Command Send data in non-data mode when **<service_type>** is "UDP SERVICE" **AT+QISEND=<connectID>,<remote_ip>,<remote_port>,<send_length>,<data>[,<rai_mode>]**	Response If the data is sent successfully: **OK** **SEND OK** If the command is executed successfully but the data sending is failed: **OK** **SEND FAIL** If there is any other error: **ERROR**
Write Command Send data of variable-lengths in data mode when **<service_type>** is "UDP SERVICE" **AT+QISEND=<connectID>,<remote_ip>,<remote_port>**	Response **>** After **>** is responded, the module enters data mode. When the input data length reaches the maximum length (1024 bytes) or when you tap "**Ctrl**" **+** "**Z**", the data will be sent out; if you tap "**Esc**", the sending will be canceled. If the data is sent successfully: **OK** **SEND OK** If the command is executed successfully but the data sending is failed: **OK** **SEND FAIL** If there is any other error: **ERROR**
Write Command Send data of a fixed-length in data mode when **<service_type>** is "UDP	Response **>** After **>** is returned, the module enters data mode. After that,

SERVICE" **AT+QISEND=<connectID>,<remote_ip>,<remote_port>,<send_length>**	type the data to be sent until the data length reaches the value of **<send_length>**. If the data is sent successfully: **OK** **SEND OK** If the command is executed successfully but the data sending is failed: **OK** **SEND FAIL** If there is any other error: **ERROR**
Write Command Check the total lengths of data sent, acknowledged and not acknowledged **AT+QISEND=<connectID>,0**	Response **+QISEND: <sent>,<acked>,<nAcked>** **OK** If there is any error: **ERROR**
Maximum Response Time	5 s
Characteristics	-

Parameter

<connectID>	Integer type. Socket ID. Range: 0–4.
<remote_ip>	String type. The remote IP address. It is valid only when **<service_type>** is "UDP SERVICE".
<remote_port>	Integer type. The remote port is only valid when **<service_type>** is "UDP SERVICE".
<send_length>	Integer type. The length of data to be sent. Unit: bytes. The maximum length is 1024 bytes in text mode and 512 bytes in hex mode.
<data>	String type. The hex/text string data to be sent.
<rai_mode>	Integer type. Release assistance indication. Range: 0–2. 0 Do not use release assistance indication. 1 Request the core network to release RRC connection immediately after receiving an uplink data packet. 2 Request the core network to release RRC connection immediately after sending a downlink data packet.
<sent>	Integer type. A numeric indicating the total length of data sent in a session. Unit: byte.

<acked>	Integer type. A numeric indicating the total length of data acknowledged by the remote server, only applicable to TCP sessions.
<nAcked>	Integer type. A numeric indicating the total length of data sent but not acknowledged by the remote server, only applicable to TCP sessions.

NOTE

1. **SEND OK** only indicates that the data has arrived at the protocol stack.
2. Note that **<send_length>** has to equal the length of **<data>**. Specifically, the value of **<send_length>** has to be the actual length of a text **<data>** and half the actual length of a hex **<data>**.
3. Enclose **<data>** in double quotation marks if special characters such as JSON are included.
4. The MCU should wait for the **SEND OK/SEND FAIL** message before issuing the next data sending operation.

2.3.5. AT+QPING Ping a Remote Server

This command tests the Internet protocol reachability of a remoter server.

AT+QPING Ping a Remote Server	
Test Command **AT+QPING=?**	Response **+QPING: (**range of supported **<contextID>**s**),<host>,(**range of supported **<time_out>**s**),(**range of supported **<ping_num>**s**),(**range of supported **<ping_size>**s**),(**list of supported **<rai_enable>**s**)** **OK**
Write Command **AT+QPING=<contextID>,<host>[,<time_out>[,<ping_num>[,<ping_size>[,<rai_enable>]]]]**	Response If the remote server pinged is reachable: **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	5 s
Characteristics	-

Parameter

<contextID>	Integer type. Context ID. Range: 0–10 0 Automatically adapt to the current default PDP context 1–10 Specify a PDP context
<host>	The server address in strings. The format is a domain name or a dotted decimal IP address. The maximum size is 150 bytes.
<time_out>	Integer type. The maximum time to wait for the response of each ping request. Range: 1–255. Default: 4. Unit: second.
<ping_num>	Integer type. The maximum number of ping requests. Range: 1–10. Default: 4.
<ping_size>	Integer type. The ping size. Range: 32–1500. Default: 32.
<rai_enable>	Integer type. Whether RAI identification is enabled. 0 Do not use release assistance indication. 1 Enable RAI flag (carry RAI flag 2 in the last packet of pings).
<result>	Integer type. The result of each ping request. 0 Received the ping response from the server. Others See ***Chapter 3*** for the explanation of specific result codes.
<IP_address>	String type. The server IP address in dotted decimal notation.
<bytes>	Integer type. The length of each sent ping request. Unit: byte.
<time>	Integer type. The time consumed for the round trip of a 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. 0 Ping successful Others See ***Chapter 3*** for the explanation of specific result codes.
<sent>	Integer type. The total number of bytes sent in the ping requests.
<rcvd>	Integer type. The total number of bytes received in the ping responses.
<lost>	Integer type. The total number of bytes lost in the ping requests.
<min>	Integer type. The minimum response time. Unit: ms.
<max>	Integer type. The maximum response time. Unit: ms.
<avg>	Integer type. The average response time. Unit: ms.

NOTE

1. Currently, only **<contextID>**=0 is supported.
2. If **<host>** is an IP address, **AT+QPING** can be used directly to ping the remote server. While if **<host>** is a domain name, make sure a DNS server address is configured before executing the command. See ***Chapter 2.3.8*** on how to configure DNS server address.

2.3.6. AT+QNTP Synchronize Local Time through NTP Server

This command synchronizes the local time with the Universal Time Coordinated (UTC) via the NTP server.

AT+QNTP Synchronize Local Time through NTP Server	
Test Command **AT+QNTP=?**	Response **+QNTP: (**range of supported **<contextID>**s**),<server>,(**range of supported **<port>**s**),(**list of supported **<auto_set_time>**s**)** **OK**
Write Command **AT+QNTP=<contextID>,<server>[,<port>[,<auto_set_time>]]**	Response If successfully synchronized: **OK** **+QNTP: <result>,<time>** If there is any error: **ERROR**
Maximum Response Time	5 s
Characteristics	-

Parameter

<contextID>	Integer type. Context ID. Range: 0–10. 0 Automatically adapt to the current default PDP context 1–10 Specify a PDP context
<server>	String type. Address of the NTP server. The format is a domain name or a dotted decimal IP address. Maximum length: 150 bytes.
<port>	Integer type. Port number of the NTP server. Range: 1–65535. Default: 123.
<auto_set_time>	Integer type. Whether to automatically synchronize the local time with UTC 0 Not synchronize automatically 1 Synchronize automatically
<time>	String type. The time synchronized from NTP server. The format is "YY/MM/DD,hh:mm:ss". The letters represent in turn year (YY), month (MM), day (DD), hour (hh), minute (mm), and second (ss).
<result>	Integer type. The result code. See ***Chapter 3*** for details.

NOTE

1. Currently, only **<contextID>**=0 is supported.
2. When **<auto_set_time>** is set to 1, the module will automatically synchronize its RTC with the UTC

after successfully registered to the network. **AT+CCLK?** can be used to check the updated time.

3. If **<server>** is an IP address, **AT+QNTP** can be used directly to synchronize local time. While if **<server>** is a domain name, make sure a DNS server address is configured before executing the command. See ***Chapter 2.3.8*** on how to configure DNS server address.

2.3.7. AT+QIDNSGIP Get IP Address by Domain Name

This command resolves a specified domain name into its IP address.

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** **+QIDNSGIP: <result>,<IP_count>,<DNS_ttl>** **[+QIDNSGIP: <hostIPaddr>]** If there is any error: **ERROR**
Maximum Response Time	5 s
Characteristics	-

Parameter

<contextID>	Integer type. Context ID. Range: 0–10 0 Automatically adapt to the current default PDP context. 1–10 Specify a PDP context.
<hostname>	String type. Domain name. Maximum length: 150 bytes.
<IP_count>	Integer type. The number of the IP addresses corresponding to the **<hostname>**.
<DNS_ttl>	Integer type. The time to live value of the IP address returned by DNS.
<hostIPaddr>	String type. An IP address of **<hostname>**.
<result>	Integer type. The result code. See ***Chapter 3*** for details.

NOTE

1. Before executing this command, please make sure that a DNS server address has been configured. See ***Chapter 2.3.8*** on how to configure DNS server address.

2. Currently, only **<contextID>**=0 is supported.
3. Currently, only the first IP address returned by the DNS server is displayed.

2.3.8. AT+QIDNSCFG Configure DNS Server Address

This command configures the primary and secondary DNS server addresses.

AT+QIDNSCFG Configure DNS Server Address	
Test Command **AT+QIDNSCFG=?**	Response **+QIDNSCFG: (**range of supported **<contextID>**s**),<pridnsaddr>,<secdnsaddr>** **OK**
Write Command Configure the primary and secondary DNS server addresses **AT+QIDNSCFG=<contextID>,<pridnsaddr>[,<secdnsaddr>]**	Response **OK** If there is any error: **ERROR**
Write Command Check the primary and secondary DNS server addresses after successful configuration **AT+QIDNSCFG=<contextID>**	Response **+QIDNSCFG: <contextID>,<pridnsaddr_ipv4>,<secdnsaddr_ipv4>,<pridnsaddr_ipv6>,<secdnsaddr_ipv6>** **OK** If there is any error: **ERROR**
Maximum Response Time	5 s
Characteristics	The command takes effect immediately. Remain valid after deep-sleep wakeup. The configurations will not be saved to NVRAM.

Parameter

<contextID>	Integer type. Context ID. Range: 0–10. 0 Automatically adapt to the current default PDP context 1–10 Specify a PDP context
<pridnsaddr>	String type. Primary DNS server address in IP format. The maximum size is 64 bytes.
<secdnsaddr>	String type. Secondary DNS server address in IP format. The maximum size is 64 bytes.
<pridnsaddr_ipv4>	String type. IPv4 primary DNS server address in IP format.

<secdnsaddr_ipv4>	String type. IPv4 secondary DNS server address in IP format.
<pridnsaddr_ipv6>	String type. IPv6 primary DNS server address in IP format.
<secdnsaddr_ipv6>	String type. IPv6 secondary DNS server address in IP format.

NOTE

1. Currently, only **<contextID>**=0 is supported.
2. In IPv4 networks, only IPv4 DNS addresses can be set.
3. In IPv6 networks, only IPv6 DNS addresses can be set.
4. The DNS server address should be configured after the module has successfully registered to the network, namely after the IP address URC (e.g. **+IP: 10.18.237.42**) is reported.
5. Since there is no default DNS server, before initiating services related to domain names, please check whether there is a DNS server issued by the network; if not, you need to manually configure a DNS server before initiating the services.

2.3.9. AT+QICFG Configure Optional Parameters

This command configures optional parameters for TCP/IP functionalities.

AT+QICFG Configure Optional Parameters	
Test Command **AT+QICFG=?**	Response **+QICFG: "dataformat",(**list of supported **<send_data_format>**s**),(**list of supported **<recv_data_format>**s**)** **+QICFG: "showRA",(**list of supported **<showRA_mode>)** **OK**
Write Command Set the data format for sending and receiving **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 any of the optional parameters is specified, set the data format for sending or receiving: **OK** If there is any error: **ERROR**
Write Command Configure whether or not to display the address of sender	Response If the optional parameter is omitted, query the current configuration:

AT+QICFG="showRA"[,<showRA_mode>]	**+QICFG: "showRA",<showRA_mode>** **OK** If the optional parameter is specified, configure whether to display the address of sender: **OK** If there is any error: **ERROR**
Maximum Response Time	5 s
Characteristics	These commands take effect immediately. The configurations will be saved to NVRAM automatically and remain valid after deep-sleep wakeup.

Parameter

<send_data_format>	Integer type. Format of data to be sent. 0 Text mode 1 Hex mode
<recv_data_format>	Integer type. Format of data received. 0 Text mode 1 Hex mode
<showRA_mode>	Integer type. Indicates whether to display the address of the remote end, including its IP address in dotted decimal notation, while displaying the received data. 0 Do not display the address. 1 Display the address.

2.4. Description of URCs

The TCP/IP URCs are reported in this format: **<CR><LF>+QIURC: <type>[…]<CR><LF>**. In this document, **<CR><LF>** at the beginning and end of each URC are omitted for brevity.

NOTE

1. When the module is in PSM, URCs will not be reported.
2. When the module is in DRX/eDRX mode, there will be a delay in URC reporting and the time delay depends on the paging cycle.
3. When the module is in the connected mode, URCs will be reported promptly.

4. The maximum length of each URC is 1400 bytes.
5. The maximum length for downlink data is 1024 bytes; When a packet exceeds the length limit, it will be divided into multiple pieces.

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

When a TCP socket service is closed by a 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 service) will change to "closing".

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

Parameter

<connectID>	Integer type. The socket ID. Range: 0–4.

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

In the direct push mode, the module reports an URC to the host after receiving data from the server. In this mode, the URC format is: **+QIURC: "recv",<connectID>,<current_recv_length>,<data>**.

+QIURC: "recv" URC Indicating Data Incoming	
+QIURC: "recv",<connectID>,<current_recv_length>,<data>	Indicating incoming data in the direct push mode when **<service_type>** is “TCP” or “UDP”.
+QIURC: "recv",<connectID>[,<current_recv_length>],<recv_ip>,<recv_port>,<data>	Indicating incoming data in direct push mode when **<service_type>** is "UDP SERVICE" or the value of **<showRA_mode>** is 1.

Parameter

<connectID>	Integer type. The socket ID. Range: 0-4.
<current_recv_length>	Integer type. The length of data actually received.
<recv_ip>	The sender’s IP address.
<recv_port>	The number of the port on which the packet is sent.
<data>	String type. The received data.
<showRA_mode>	Integer type. Indicates whether to display along with the received data the

	address of the remote end, including its IP address in dotted decimal notation. 0 Do not display the address. 1 Display the address.

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

If the **<service_type>** is "TCP LISTENER", when a remote client connects to this server, the module automatically assigns a free **<connectID>** for the new connection and reports this URC. The **<service_type>** of the new connection is "TCP INCOMING".

+QIURC: "incoming" URC Indicating Incoming Connection	
+QIURC: "incoming",<connectID>,<serverID>,<remoteIP>,<remote_port>	Indicating incoming connection when the new incoming connection is accepted by **<serverID>**. The allocated **<connectID>**, **<remoteIP>** and **<remote_port>** are informed by this URC.

Parameter

<connectID>	Integer type. Index of the socket service assigned for the incoming connection, which is automatically specified by the module. Range: 0–4.
<serverID>	Integer type. ID of the listening socket whose **<service_type>** is "TCP LISTENER" and which accepts the incoming **<connectID>**.
<remoteIP>	String type. Remote IP address of the incoming **<connectID>**.
<remote_port>	Integer type. Remote port of the incoming **<connectID>**.

3 Summary of Result Codes

Table 2: Summary of Result Codes

Result Code	Description
0	Operation successful
550	Unknown error
551	Operation blocked
552	Parameters invalid
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 connection failed
567	Socket closed

568	Operation busy
569	Operation timeout
570	PDP context broke down
571	Send Canceled
572	Operation not allowed
573	APN not configured
574	Port busy

4 Examples

As the operations in the TCP client service in the direct push mode is basically the same as those in the UDP client service in the mode, this chapter only gives examples for the operations in TCP client service and UDP service.

4.1. TCP Client Service in Direct Push Mode

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

```
//Open a socket service of which both the context ID and the socket ID are 0.
AT+QIOPEN=0,0,"TCP","220.180.239.212",8062,0,1
OK

+QIOPEN: 0,0                    //Connected successfully. It is recommended to wait for 60 s for the
                                  URC to be reported.
AT+QISTATE=1,0                  //Query the connection status of socket service 0.
+QISTATE: 0,"TCP","220.180.239.212",8062,0,2,0,1

OK
```

4.1.2. Send Data in Direct Push Mode

```
AT+QICFG="dataformat",0,0       //Configure to send data in Text mode.
OK
AT+QICFG="dataformat"           //Query the format in which data is sent.
+QICFG: "dataformat",0,0

OK
AT+QISEND=0,5,"12345"           //Send data of which the length is 5 bytes in non-data mode.
OK

SEND OK
AT+QISEND=0                     //Send data of variable lengths in data mode.
>
12345                           //After ">" is responded, input the data and tap "Ctrl" + "Z" to send it.
```

```
OK

SEND OK
AT+QISEND=0,5                    //Send data in a fixed-length of 5 bytes in data mode.
>
12345                            //After ">" is responded, input the data until the length reaches 5 bytes.
OK

SEND OK
AT+QICFG="dataformat",1,0        //Configure to send data in Hex mode.
OK
AT+QICFG="dataformat"            //Query the format in which data is sent.
+QICFG: "dataformat",1,0

OK
AT+QISEND=0,5,"3132333435"       //Send data of which the length is 5 bytes.
OK

SEND OK
AT+QISEND=0                      //Send data of variable lengths.
>
3132333435                       //After ">" is responded, input the data and tap "Ctrl" + "Z" to send it.
OK

SEND OK
AT+QISEND=0,5                    //Send data in a fixed-length of 5 bytes in data mode.
>
3132333435                       //After ">" is responded, input the data until the length reaches 5 bytes.
OK

SEND OK
```

4.1.3. Receive Data from Remote Server in Direct Push Mode

```
AT+QICFG="showRA",1
OK
AT+QICFG="showRA"
+QICFG: "showRA",1

OK
+QIURC: "recv",0,5,"220.180.239.212",8062,"12345"        //Received data from a remote server.

AT+QISEND=0,12,"012345678901"                            //Send data to a remote server.
```

```
OK

SEND OK

+QIURC: "recv",0,12, "220.180.239.212",8062,"012345678901"      //Received from the remote server
                                                                  the data you just sent to it.
```

4.1.4. Close a Connection

```
AT+QICLOSE=0                          //Close a connection whose socket ID is 0.
OK

CLOSE OK
```

4.2. UDP SERVICE in Direct Push Mode

4.2.1. Set up a UDP SERVICE Connection and Enter Direct Push Mode

```
//Open a socket service of which both the context ID and the socket ID are 0.
AT+QIOPEN=0,0,"UDP SERVICE","127.0.0.1",1,1234,1
OK

+QIOPEN: 0,0                  //Connected successfully. It is recommended to wait for 60 seconds for
                              the URC to be reported.

AT+QISTATE=1,0                //Query the connection status of a UDP service whose socket ID is 0.
+QISTATE: 0,"UDP SERVICE","127.0.0.1",1,1234,2,0,1

OK
```

4.2.2. Send Data in Direct Push Mode

```
AT+QISEND=0,"220.180.239.212",8196,10,"1234567890"      //Send data of 10 bytes in non-data mode.
OK

SEND OK
AT+QISEND=0,"220.180.239.212",8196          //Send data of variable lengths in data mode.
>
1234567890                        //After ">" is responded, input the data and tap "Ctrl" + "Z" to send it.
OK
```

```
SEND OK
AT+QISEND=0,"220.180.239.212",8196,10        //Send data in a fixed-length of 10 bytes in data mode.
>
1234567890                                   //After ">" is responded, input the data until the length reaches 10 bytes.
OK

SEND OK
```

4.2.3. Receive Data from UDP Client in Direct Push Mode

```
+QIURC: "recv",0,5,"220.180.239.212",8196,"12345"                    //Received data from a UDP client.

AT+QISEND=0, "220.180.239.212",8196,12,"012345678901"                //Send data to a UDP client.
OK

SEND OK

+QIURC: "recv",0,12, "220.180.239.212",8196,"012345678901"           //Received from the UDP client the data you just sent to it.
```

4.2.4. Close a Connection

```
AT+QICLOSE=0        //Close a connection whose socket ID is 0.
OK

CLOSE OK
```

4.3. TCP Server in Direct Push Mode

4.3.1. Set up a TCP Server Connection and Enter Direct Push Mode

```
AT+QIOPEN=0,0,"TCP LISTENER","192.168.4.26",0,5000,1        //Context is 0 and <connectID> is 0.
OK

+QIOPEN: 0,0                    //Connected successfully. It is recommended to wait for 60 seconds for the URC to be reported.
AT+QISTATE=0,0                  //Query the connection status of socket service 0.
+QISTATE: 0,"TCP LISTENER","192.168.4.26",0,5000,2,0,1

OK
```

```
+QIURC: "incoming",1,0,"192.168.4.30",34835                    //Connected successful.
AT+QISTATE?                    //Query the connection status of all Connect ID.
+QISTATE: 0,"TCP LISTENER","192.168.4.26",0,5000,2,0,1
+QISTATE: 1,"TCP INCOMING","192.168.4.30",34835,5000,2,0,1

OK
```

4.3.2. Send Data in Direct Push Mode

```
AT+QISEND=1,10,"1234567890"   //Send data, and the data length is 10 bytes.
OK

SEND OK
AT+QISEND=1                    //Send data of variable lengths in data mode.
>
1234567890                     //After ">" is responded, input the data and tap "Ctrl" + "Z" to send it.
OK

SEND OK
AT+QISEND=1,10                 //Send data in a fixed-length of 10 bytes in data mode.
>
1234567890                     //After ">" is responded, input the data until the length reaches 10 bytes.
OK

SEND OK
```

4.3.3. Receive Data from Remote Client in Direct Push Mode

```
+QIURC: "recv",1,10,"1234567890"                    //Receive data from the remote client.
```

4.3.4. Close TCP Server

```
AT+QICLOSE=0                   //Close a connection whose socket service index is 0.
OK

CLOSE OK
AT+QISTATE?                    //Query the connection status of all Connect ID.
OK                             //If no list of +QISTATEs is returned, all <ConnectID> have been closed.
```

4.4. Ping a Remote Server

```
AT+QIDNSCFG=0,"218.2.2.2","8.8.8.8"          //Configure DNS server addresses.
OK
AT+QIDNSCFG=0
+QIDNSCFG: 0,"218.2.2.2","8.8.8.8"

OK
AT+QPING=0,"iot.quectel.com"                 //Ping the remote server iot.quectel.com.
OK

+QPING: 0,"47.100.63.174",32,560,88

+QPING: 0,"47.100.63.174",32,220,88

+QPING: 0,"47.100.63.174",32,230,88

+QPING: 0,"47.100.63.174",32,280,88

+QPING: 0,4,4,0,220,560,322
```

4.5. Synchronize Local Time Through NTP Server

```
AT+QIDNSCFG=0,"218.2.2.2","8.8.8.8"          //Configure DNS server addresses.
OK
AT+QIDNSCFG=0
+QIDNSCFG: 0,"218.2.2.2","8.8.8.8"

OK
AT+QNTP=0,"ntp5.aliyun.com"                  //Synchronize local time with the time on the NTP server
                                               ntp5.aliyun.com.
OK

+QNTP: 0,"2019/06/11,11:08:20"
```

4.6. Configure DNS Server Address

```
AT+QIDNSCFG=0,"218.2.2.2","8.8.8.8"
OK
AT+QIDNSCFG=0
+QIDNSCFG: 0,"218.2.2.2","8.8.8.8"

OK
```

4.7. Get IP Address by Domain Name

```
AT+QIDNSCFG=0,"218.2.2.2","8.8.8.8"
OK
AT+QIDNSCFG=0
+QIDNSCFG: 0,"218.2.2.2","8.8.8.8"

OK
AT+QIDNSGIP=0,"www.baidu.com"
OK

+QIDNSGIP: 0,1,0
+QIDNSGIP: 14.215.177.39
```

5 Appendix A Reference

Table 3: Terms and Abbreviations

Abbreviation	Description
APN	Access Point Name
DNS	Domain Name System
DRX	Discontinuous Reception
eDRX	extended Discontinuous Reception
HEX	Hexadecimal
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
PDP	Packet Data Protocol
PPP	Point to Point Protocol
PSM	Power Saving Mode
RAI	Release Assistance Indication
RRC	Radio Resource Control
TA	Terminal Adaptor
TCP	Transmission Control Protocol
UDP	User Datagram Protocol

URC	Unsolicited Result Code
UTC	Coordinated Universal Time