BC68_CoAP

AT Commands Manual

NB-IoT Module Series

Rev. BC68_CoAP_AT_Commands_Manual_V1.0

Date: 2018-07-18

Status: Preliminary

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

Quectel Wireless Solutions Co., Ltd.
7th Floor, Hongye Building, No.1801 Hongmei Road, Xuhui District, Shanghai 200233, China
Tel: +86 21 5108 6236
Email: info@quectel.com

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

For technical support, or to report documentation errors, please visit:
http://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	2018-07-24	Oven TAO	Initial

Contents

Table Index

1 Introduction

This document gives details of the new AT Command Set supported by Quectel NB-IoT BC68 module.

1.1. Definitions

- <CR>: Carriage return character;
- <LF>: Line feed character;
- <..>: Parameter name. Angle brackets do not appear on command line;
- [..]: Optional parameter. Square brackets do not appear on the command line.

1.2. AT Command Syntax

Table 1: AT Command Syntax

Test Command	**AT+<cmd>=?**	This command returns the list of parameters and value ranges set by the corresponding Write Command or internal processes.
Read Command	**AT+<cmd>?**	This command returns the currently set value of the parameter or parameters.
Write Command	**AT+<cmd>=p1[,p2[,p3[...]]]**	This command sets the user-definable parameter values.
Execution Command	**AT+<cmd>**	This command reads non-variable parameters affected by internal processes in the Module engine.

Multiple commands can be placed on a single line using a semi-colon (";") between commands. Only the first command should have AT prefix. Commands can be in upper or lower case.

When entering AT commands spaces are ignored except in the following cases:

- Within quoted strings, where they are preserved;
- Within an unquoted string or numeric parameter;
- Within an IP address;
- Within the AT command name up to and including a '=', '?' or '=?'.

They can be used to make the input more human-readable. On input, at least a carriage return is required. A newline character is ignored so it is permissible to use carriage return/line feed pairs on the input. For B600, the AT command processor uses carriage return/line feed pairs (\r\n, 0x0D0A) to end lines on its output.
If no command is entered after the AT token, "OK" will be returned. If an invalid command is entered, “ERROR” will be returned.

Optional parameters, unless explicitly stated, need to be provided up to the last parameter being entered.

1.3. AT Command Responses

When the AT Command processor has finished processing a line, it will output either "OK" or "ERROR" indicating that it is ready to accept a new command. Solicited informational responses are sent before the final "OK" or "ERROR". Unsolicited information responses will never occur between a solicited informational response and the final “OK” or “ERROR”.

Responses will be of the format:

<CR><LF>+CMD1:<parameters><CR><LF>
<CR><LF>OK<CR><LF>

Or

<CR><LF><parameters><CR><LF>
<CR><LF>OK<CR><LF>

2 CoAP AT Commands

2.1. AT+QCOAPCREATE Create CoAP Context

The command is used to create the CoAP context. It will give an <err> code and description as an intermediate message if the message cannot be sent.

AT+QCOAPCRETAE Create CoAP Context	
Test Command **AT+QCOAPCREATE=?**	Response **+QCOAPCREATE: <1-65535>** **OK** If there is any error, response: **+CME ERROR:<err>**
Write Command **AT+QCOAPCREATE=<local_port>**	Response **OK** If there is any error, response: **ERROR** Or **+CME ERROR:<err>**
Maximum Response Time	300ms

Parameter

<local_port>	The CoAP Context local port, range value: 1-65535.

Example

```
AT+QCOAPCREATE=?
+QCOAPCREATE: <1-65535>

OK

AT+QCOAPCREATE=56830
OK
```

2.2. AT+QCOAPDEL Delete CoAP Context

The command is used to delete the CoAP context. It will give an <err> code and description as an intermediate message if the message cannot be sent.

AT+QCOAPDEL Delete CoAP Context	
Execution Command **AT+QCOAPDEL**	Response **OK** If there is any error, response: **ERROR** Or **+CME ERROR:<err>**
Maximum Response Time	5s

Example

```
AT+QCOAPCREATE=56830          // Create CoAP Context.
OK

AT+QCOAPDEL                   // Delete CoAP Context.
OK
```

2.3. AT+QCOAPADDRES Add CoAP Client resources

The command is used to configure the CoAP option. It will give an <err> code and description as an intermediate message if the message cannot be sent.

AT+QCOAPOPTION Configure CoAP option	
Test Command **AT+QCOAPADDRES=?**	Response **+QCOAPADDRES: <1-50>,"<resource>"** **OK** If there is any error, response: **+CME ERROR:<err>**

Write Command **AT+QCOAPADDRES=<length>,<resource>**	Response **OK** If there is any error, response: **ERROR** Or **+CME ERROR:<err>**
Maximum Response Time	300ms

Parameter

<length>	The CoAP client resources, range: 1-50.
<resource>	The resource name.

Example

```
AT+QCOAPADDRES=4,"/t/d"  // Add the CoAP resource, the value is "/t/d"
OK
```

2.4. AT+QCOAPHEAD Configure CoAP Head and Token values

The command is used to configure the CoAP head and token values. It will give an <err> code and description as an intermediate message if the message cannot be sent.

AT+QCOAPHEAD Configure CoAP Head and Token values	
Test Command **AT+QCOAPHEAD=?**	Response **+QCOAPHEAD: <mode>[,[<msgid>][,<tkl>,<token>]]** **OK** If there is any error, response: **ERROR** Or **+CME ERROR:<err>**
Write Command **AT+QCOAPHEAD=<mode>[,[<msgid>][,<tkl>,<token>]]**	Response **OK** If there is any error, response: **ERROR** Or

	+CME ERROR:<err>
Maximum Response Time	300ms

Parameter

<mode>	The CoAP Head and Token parameter, range value: 1-5. 1 Generate message id and token values randomly. 2 Generate message id, and configure the token values. 3 Only configure message id, not needed token values. 4 Configure message id, and generate the token values randomly. 5 Configure message and token values.
<msgid>	The message id, only needed configure when the <**mode**> value is 3, 4, 5. Range value: 0-65535.
<tkl>	The token values length, only needed configure when the <**mode**> value is 1, 2. Range value: 1-8.
<token>	The token values, hexadecimal format string, only need configure when the <**mode**> value is 1, 2.

NOTE

1. If not set this command, the module will generate message id randomly, and not needed token values.

Example

```
AT+QCOAPHEAD=1              // Generate message id and token values randomly.
OK

AT+QCOAPHEAD=2,8,0102030405060708      // Generate message id, and configure the token values
                                          0102030405060708.
OK

AT+QCOAPHEAD=3,13940        // Configure message id 13940, and not needed token values.
OK

AT+QCOAPHEAD=4,13940        // Configure message id 13940, and generate token valuse.
OK

AT+QCOAPHEAD=5,13940,4,02040608      // Configure message id, and configure token values
                                        02040608.
```

OK

2.5. AT+QCOAPOPTION Configure CoAP option

The command is used to configure the CoAP option. It will give an <err> code and description as an intermediate message if the message cannot be sent.

AT+QCOAPOPTION Configure CoAP option	
Test Command **AT+QCOAPOPTION=?**	Response **+QCOAPOPTION: <opt_cnt,<opt_name>,"<opt_value>"[,...]** **OK** If there is any error, response: **ERROR** Or **+CME ERROR:<err>**
Write Command **AT+QCOAPOPTION=<opt_cnt>,<opt_name>,<opt_value>[,...]**	Response **OK** If there is any error, response: **ERROR** Or **+CME ERROR:<err>**
Maximum Response Time	300ms

Parameter

<opt_cnt>	The option parameter count, range value: 1-12.	
<opt_name>	The option name, refer the RFC 7252.	
	1	If-Match
	3	Uri-Host
	4	ETag
	5	If-None-Match
	6	Observe
	7	Uri-Port
	8	Location-Path
	11	Uri-Path
	12	Content-Format

	14	Max-Age
	15	Uri-Query
	17	Accept
	20	Location-Query
	23	Block2
	27	Block1
	28	SIZE
	35	Proxy-Uri
	39	Proxy-Scheme
	60	Size1
<opt_value>	The option value, string type. The length of value string: 1-180. If the **<opt_name>** is 12 or 17, the **<opt_value>** should be the below value: “0” text-plain “40” application/link-format “41” application/xml “42” application/octet-stream “47” application/exi “50” application/json	

Example

```
AT+QCOAPOPTION=1,11,”rd”      // Configure the CoAP option 11(Uri-Path), the value is “rd”
OK

AT+QCOAPOPTION=2,11,”rd”,15,”ep=86370303”   // Configure the CoAP option 11(Uri-Path),
                              the value is “rd”, and option 15(Uri-Query) is “ep=86370303”
OK

AT+QCOAPOPTION=2,11,”.well-know”,11,”core”   // Configure the CoAP option 11(Uri-Path),
                              the value is “.well-know”, and option 11(Uri-Path) is “core”
OK
```

2.6. AT+QCOAPSEND Send CoAP Request

This command is used to send CON or NON data to the IOT platform. After sending CON data, the sending result will be automatically notified to the terminal. Terminal can also use the command AT+QCOAPDATASTATUS? to query the status of the CON data that has been sent.

AT+QCOAPSEND	**Send data with mode**
Test Command **AT+QCOAPSEND=?**	Response **+QCOAPSEND:**

	<type>,<method/rspcode>,<ip_addr>,<port> **OK** If there is any error, response: **ERROR** Or **+CME ERROR:<err>**
Write Command **AT+QCOAPSEND=<type>,<method/rspcode>,<ip_addr>,<port>** After **">"** is responded, input the data to be sent. Tap **"CTRL+Z"** to send, and tap **"ESC"** to cancel the operation.	Response **OK** If there is any error, response: **ERROR** Or **+CME ERROR:<err>**
Maximum Response Time	300ms

Parameter

<type>	The CoAP Protocol of message type, range: 0-3, refer the RFC 7252. 0: CON 1: NON 2: ACK 3: RST
<method>	The CoAP Protocol of method, refer the RFC 7252. 1: GET 2: POST 3: PUT 4: DELETE
<rspcode>	The CoAP Protocol of response code, refer the RFC 7252. 0 Empty Message 201 2.01, Created 202 2.02, Deleted 203 2.03, Valid 204 2.04, Changed 205 2.05, Content 400 4.00, Bad Request 401 4.01, Unauthorized 402 4.02, Bad Option 403 4.03, Forbidden 404 4.04, Not Found 405 4.05, Method Not Allowed

	406	4.06, Not Acceptable
	412	4.12, Precondition Failed
	413	4.13, Request Entity Too Large
	415	4.15, Unsupported Content-Format
	500	5.00, Internal Server Error
	501	5.01, Not Implemented
	502	5.02, Bad Gateway
	503	5.03, Service Unavailable
	504	5.04, Gateway Timeout
	505	5.05 Proxying Not Supported
<ip_addr>	The CoAP Server.	
<port>	The CoAP Server Port.	

Example

```
AT+QCOAPSEND=1,1,139.196.41.136,5683    //Send Get request with NON type to the CoAP Server.
                                        //After receiving '>', input data and then send it. The maximum length of the data is 1024 bytes and the data that beyond 1024 bytes will be omitted. After inputting data,tap "Ctrl+Z" to send
OK

AT+QCOAPSEND=0,2,139.196.41.136,5683    //Send Post request with CON type to the CoAP server.
>0102
OK
//Received data with message id 61440, and the data length is 25, code is 2.05.
+QCOAPURC: "rsp" 2.05,61440,25,4E692048616F2066726F6D20436F41502E4E45542052464320
```

Quectel Implementation

- There is a maximum data length of 512 bytes.
  If sending CON data, it must acquire the state (fail/timeout/success/got reset message) of sending CON data before sending the next CON or NON data.

2.7. AT+QCOAPDATASTATUS Query CON Messages Sent Status

This command queries the status of the sending CON data to CoAP Server.
Please refer to ***Chapter 5*** for possible <err> values.

AT+QCOAPDATASTATUS Query CON Messages Sent Status	
Read Command	Response

AT+QCOAPDATASTATUS?	**+QCOAPDATASTATUS: <status>** **OK** If there is any error, response: **ERROR** Or **+CME ERROR:<err>**
s	300ms

Parameter

<status>	Status of the CON data has been sent 0 Have not sent 1 Sent, waiting response of IoT platform 2 Sent failed 3 Timeout 4 Success 5 Got reset message

Example

```
AT+QCOAPSEND=0,1,139.196.41.136,5683    //Send GET request with CON type to CoAP server.
>                                       //After receiving '>', input data and then send it.
OK

AT+QCOAPDATASTATUS?
+QCOAPDATASTATUS:1                      //Sent, waiting response of CoAP server.

OK

+QCOAPURC: "rsp",2.05,61440,25, 4E692048616F2066726F6D20436F41502E4E455420524643
//Received data with message id 61440, and the data length is 29, code is 2.05.


AT+QCOAPDATASTATUS?
+QLWDATASTATUS:4                        //Success.

OK
```

Quectel Implementation

- This command only queries the status of the CON data that has been sent.

2.8. AT+QCOAPCFG CoAP Configuration Command

The command is used to config the CoAP function parameter. It will give an <err> code and description as an intermediate message if the message cannot be sent.

AT+QCOAPCFG CoAP Configuration Command	
Test Command **AT+QCOAPCFG=?**	Response **OK**
Read Command **AT+QCOAPCFG?**	Response **OK**
Write Command **AT+QCOAPCFG="Showra"[,<Showra>]**	Response **OK** If error is related to ME functionality: **ERROR** Or **+CME ERROR:<err>** If the second parameter is omitted, query the "Showra" value. **+QCOAPCFG: "Showra",<Showra>** **OK**
Write Command **AT+QCOAPCFG="Showrspopt"[,<Showrspopt>]**	Response **OK** If error is related to ME functionality: **ERROR** Or **+CME ERROR:<err>** If the second parameter is omitted, query the "Showrspopt" value. **+QCOAPCFG: "Showrspopt",<Showrspopt>** **OK**
Maximum Response Time	300ms
Write Command **AT+QCOAPCFG="Dtls"[,<Dtlsenable>,<nat_type>,<pskid>,<psk>]**	Response **OK** If error is related to ME functionality: **ERROR** Or **+CME ERROR:<err>** If the second parameter is omitted, query the "Dtls" value.

	+QCOAPCFG: "Dtls",<Dtlsenable><nat_type>,<pskid> **OK**

Parameter

<Showra>	Set whether or not to display the address of sender. 0 Do not show the address. Default. 1 Show the address; the format to show the address is like: **+QCOAPURC: "rsp",[<ip_addr>,<port>],<type>,<rspcode>,<msgid>[,<length>,<data>]**
<Showrspopt>	Set whether or not to display the coap option of sender. 0 Do not show the coap option. Default. 1 Show the coap option; the format to show the address is like: **+QCOAPURC: "rsp",<type>,<rspcode>,<msgid>[,<opt_cnt>,<opt_name>,"<opt_value>"[,...]][,<length>,<data>]**.
<Dtlsenable>	Set whether or not enable coap with dtls 0 disable dtls 1 enable dtls
<nat_type>	Standard DTLS NAT network. 0 The DTLS link is rebuilt every 30s when the data is sent 1 Every time when the module is powered on or when the module IP address changes, the DTLS encryption mode will be triggered
<pskid>	Indicates the PSK index. The fixed length is 15 decimal digits, In addition, this parameter must also be set to the same value on the server.
<psk>	Indicates the PSK. This parameter must be set to a 16-digit hexadecimal number. In addition, it must also be set to the same value on the server.

NOTES

1. Effect immediately.
2. This command cannot be saved.

Example

```
AT+QCOAPCFG?
+QCOAPCFG: "Showra",0

OK

AT+QCOAPCFG="Showra",1
OK
```

2.9. AT+QCOAPALISIGN Calculate CoAP sign value of the AliCloud

The command is used to config the CoAP function parameter. It will give an <err> code and description as an intermediate message if the message cannot be sent.

AT+QCOAPALISIGN Calculate CoAP sign value of the AliCloud	
Test Command **AT+QCOAPALISIGN=?**	Response **+QCOAPALISIGN: "<dev_id>","<dev_name>","<dev_secret>","<product_key>"** **OK**
Write Command **AT+QCOAPALISIGN="<dev_id>","<dev_name>","<dev_secret>","<product_key>"**	Response **+QCOAPALISIGN: "<sign>"** **OK** If error is related to ME functionality: **ERROR** Or **+CME ERROR:<err>**
Maximum Response Time	300ms

Parameter

<dev_id>	Device ID issued by AliCloud.
<dev_name>	Device name issused by AliCloud.
<dev_secret>	Device secret key issused by AliCloud.
<product_key>	Product key issued by AliClode.
<sign>	The calculated sign value, string type.

Example

```
AT+QCOAPALISIGN=?
+QCOAPALISIGN: "<dev_id>","<dev_name>","<dev_secret>","<product_key>"

OK

AT+QCOAPALISIGN=
"IoTxCoAPTestDev.1","NBIOT","GKjaM1cyHfDA6q2ioXLSpuscoTrl8vuY","a1FiGo98MSd"
+QCOAPALISIGN: "62f3a734a991b9e82b162a10ffa6d874"
```

```
OK
```

3 CoAP Related Notifications

This chapter gives CoAP related notifications and their descriptions.

Table 2: CoAP Related Notifications

Index	Notification Display	Description
[1]	**+QCOAPURC: ”rsp”[<ip_addr>,<port>,]<type>,<rspcode>,<msgid>[,<opt_cnt>,<opt_name>,”<opt_value>”[,…]][,<length>,<data>]**	When send the CON message, the CoAP server will response ACK or RST, if the module receive this response, will report the URC.
[2]	**+QCOAPURC: ”req”,[<ip_addr>,<port>,]<type>,<method>,<msgid>,<mode>[,<tkl>,<token>][,<opt_name>,”<opt_value>”[,...]][,<length>,<data>]**	When the CoAP server send a request, if the module received this request, will report the URC.

3.1. Description of Response URC

The module reports the CoAP response event to the device.

Notify the device that the response data from the Server	
+QCOAPURC: ”rsp”,[<ip_addr>,<port>,]<type>,<rspcode>,<msgid>[,<opt_cnt>,<opt_name>,"<opt_value>"[,...]][,<length>,<data>]	Notify the response from CoAP server or IoT platform.

Parameter

<ip_addr>	The CoAP server ip address, it will show when set **AT+QCOAPCFG=”Showra”,1**.
<port>	The CoAP server port, it will show when set **AT+QCOAPCFG=”Showra”,1**.
<type>	The CoAP Protocol of message type, range: 0-3, refer the RFC 7252.

	0: CON 1: NON 2: ACK 3: RST
<rspcode>	Please refer the 3.4 section.
<msgid>	The CoAP message id.
<opt_cnt>	The count of option, it will show when set **AT+QCOAPCFG="Showrspopt",1**
<opt_name>	The option name, it will show when set **AT+QCOAPCFG="Showrspopt",1**.
<opt_value>	The option value, it will show when set **AT+QCOAPCFG="Showrspopt",1**.
<length>	The data length. The max length is 512 bytes.
<data>	If the **<opt_name>** is 12, and the **<opt_value>** is 0 "text-plain", 41 "application/xml" or 50 "application/json", the data format is text string, else hexadecimal format string.

3.2. Description of Request URC

The module reports the CoAP request event to the device.

Notify the device that the request data from the Server	
+QCOAPURC: "req",[<ip_addr>,<port>,]<type>,<method>,<msgid>,<mode>[,<tkl>,<token>][,<opt_name>,"<opt_value>"[,...]][,<length>,<data>]	Notify the request from CoAP server or IoT platform.

Parameter

<ip_addr>	The CoAP server ip address, it will show when set **AT+QCOAPCFG="Showra",1**..
<port>	The CoAP server port, it will show when set **AT+QCOAPCFG="Showra",1**..
<type>	The CoAP Protocol of message type, range: 0-3, refer the RFC 7252. 0: CON 1: NON 2: ACK 3: RST
<method>	The CoAP Protocol of method, refer the RFC 7252. 1: GET 2: POST 3: PUT

	4: DELETE
<msgid>	The CoAP message id.
<mode>	Indicates the existence of TOKEN, OPTION, and DATA, Hexadecimal format. Bit 0: The TOKEN exists. Bit 1-6: The count of OPTION. Bit 7: The DATA exists.
<tkl>	The token values length.
<token>	The token values, hexadecimal format string.
<opt_name>	The option name.
<opt_value>	The option value.
<length>	The data length. The max length is 1024 bytes.
<data>	If the **<opt_name>** is 12, and the **<opt_value>** is 0 “text-plain”, 41 “application/xml” or 50 “application/json”, the data format is text string, else hexadecimal format string.

4 Examples

4.1. Send CoAP Request to the CoAP Server

```
AT+CGATT?                                   // Query the PS service attach status.
+CGATT: 1                                   // Attached to the PS service.
OK

AT+QCOAPCREATE=56830        // Create the CoAP context.
OK

AT+QCOAPSEND=0,1,5.39.83.206,5683    // Send Get request with CON type to the server.
OK

AT+QCOAPDATASTATUS?      // Query the data sent status.
+QCOAPDATASTATUS: 1      // Sent, waiting response of IoT platform.

OK

// Received data from the Server, the code is 2.05, message id is 6685, and the data length is 25, the data is Hexadecimal format string.
+QCOAPURC: "req",2,2.05,6685,25,4E692048616F2066726F6D20436F41502E4E455420524643

AT+QCOAPDATASTATUS?      // Query the data sent status.
+QCOAPDATASTATUS: 4      // Success.

OK

AT+QCOAPHEAD=3,5566     // Configure the CoAP head message is 5566, and not needed the token.
OK

AT+QCOAPSEND=0,1,139.196.187.107,5683   // Send Get request with CON type to the server.
>                                       //type Ctrl+Z payload is none
OK

// Received data from the Server, the code is 2.05, message id is 5566 (message id is the same with the set id), and the data length is 25, the data is Hexadecimal format string.
+QCOAPURC: "req",2,2.05,5566,29,4E692048616F2066726F6D20436F41502E4E455420524643
```

AT+QCOAPOPTION=1,11,”rd” // Configure the CoAP option, the Uri-Path is “/rd”.
OK

AT+QCOAPSEND=0,1,139.196.187.107,5683 // Send Get request with CON type to the server.
> //type **Ctrl+Z** payload is none
OK

// Received data from the Server, the code is 2.05, message id is 61441, and the data length is 17, the data is Hexadecimal format string.
+QCOAPURC:"req",2,2.05,5566,17,323031382F312F392031313A31313A3139

AT+QCOAPHEAD=5,6677,4,30323436 // Configure the CoAP head and Token values.
OK

AT+QCOAPOPTION=1,11,”hello” // Configure the CoAP option, the Uri-Path is “/hello”.
OK

AT+QCOAPSEND=0,1,139.196.187.107,5683 // Send Get request with CON type to the server.
> //type Ctrl+Z payload is none
OK

// Received data from the Server, the code is 2.05, message id is 6677, and the data length is 22, the data is Hexadecimal format string.
+QCOAPURC:"req",2,2.05,6677,22,68656C6C6F20776F72642066726F6D20736572766572

// Set the “Showra” to 1,when received the data from the peer, will show the ip address and port.
AT+QCOAPCFG=”Showra”,1
OK

AT+QCOAPADDRES=4,”/t/d”
OK

AT+QCOAPOPTION=3,11,"t",11,"r",15,"ep=863703030822519"
OK

AT+QCOAPSEND=0,2, 139.196.187.107,5683
>
OK
// Received response data from the Server 180.101.147.115:5683, the type is ACK, the code is 2.04, message id is 6677.
+QCOAPURC:"rsp",180.101.147.115,5683,2,2.04,2802

// Received request data from the Server 180.101.147.115:5683, the type is CON, the code is GET,

message id is 50670, and the <MODE> is 09, the TOKEN and OPTION exists, and the OPTION count is 4.

+QCOAPURC:"req",180.101.147.115,5683,0,1,50670,09,8,1A84BFE989C01C08,6,"0",7,"54321",11,"t",11,"d"

//When received the request data, the client should response it.

AT+QCOAPHEAD=5,50670,8,1A84BFE989C01C08

OK

AT+QCOAPSEND=1,205,139.196.187.107,5683

> //type **Ctrl+Z** payload is none

OK

AT+QCOAPDEL // Delete the CoAP context.

OK

4.2. Register to China Telecom IOT Platform

AT+QCOAPCREATE=56830 // Create the CoAP context

OK

AT+QCOAPHEAD=1 // Generate message id and token values randomly.

OK

// Configure the CoAP option 11(Uri-Path), the value is “rd”, and option 12 the value is “42” and option 15(Uri-Query) the value is “lwm2m=1.0” and option 15(Uri-Query) is “ep=867725030002525;460041850403693” …..

AT+QCOAPOPTION=6,11,"rd",12,"42",15,"lwm2m=1.0",15,"ep=867725030002525;460041850403693",15,"b=U",15,"lt=86400"

OK

AT+QCOAPADDRES=6,"/4/0/2" // Add the CoAP resource, the value is “/4/0/2”

OK

AT+QCOAPADDRES=6,"/4/0/3" // Add the CoAP resource, the value is “/4/0/3”

OK

AT+QCOAPADDRES=6,"/4/0/8" // Add the CoAP resource, the value is “/4/0/8”

OK

AT+QCOAPADDRES=6,"/3/0/9" // Add the CoAP resource, the value is “/3/0/9”

OK

AT+QCOAPADDRES=6,"/3/0/7" // Add the CoAP resource, the value is “/3/0/7”

OK

```
AT+QCOAPADDRES=7,"/19/0/0"      // Add the CoAP resource, the value is “/19/0/0”
OK

AT+QCOAPADDRES=7,"/19/1/0"      // Add the CoAP resource, the value is “/19/1/0”
OK

//Send Register Message to China Telecom IOT Platform
AT+QCOAPSEND=0,2,180.101.147.115,5683
>3c2f3e3b72743d226f6d612e6c776d326d222c3c2f312f303e2c3c2f332f303e2c3c2f342f303e2c3c2f352f303e2c3c2f32302f303e2c3c2f31392f303e
OK
// Received response data from the Server, the type is ACK, the code is 2.01, message id is 55018.
+QCOAPURC: "rsp",2,2.01,55018
// Received request data from the Server, the type is CON, the code is GET, message id is 62084, and the <MODE> is 09, the TOKEN and OPTION exists, and the OPTION count is 4.

+QCOAPURC: "req",0,1,62084,09,8,1A98EB114F501C07,6,"0",11,"19",11,"0",11,"0"

//When received the request data, the client should response it.
//Config the coap token same as the request message
AT+QCOAPHEAD=5,62084,8,1A98EB114F501C07
OK

AT+QCOAPSEND=2,205,180.101.147.115, 5683
>                                                    //type Ctrl+Z payload is none
OK

AT+QCOAPDEL                     // Delete the CoAP context.
OK
```

4.3. Example of using the configure commands

```
AT+QCOAPCREATE=56830          // Create the CoAP context
OK

AT+QCOAPCFG=”Showrspopt”,1          // Show the response coap option
OK

AT+QCOAPOPTION=5,11,"rd",12,"40",15,"ep=I@#@G7pjZBAQUoVav0yXQcQ6v4Ad1nFGXcKhnTWYJt3AE0TpZSj3xDrYMOm6NmohixF@#@M100000089",15,"b=U",15,"lt=2700"
OK

AT+QCOAPSEND=0,2,220.180.239.212,8063
```

```
>3c2f3e3b72743d226f6d612e6c776d326d222c3c2f30
OK

AT+QCOAPDATASTATUS?
+QCOAPDATASTATUS: 4

OK
+QCOAPURC: "rsp",2,2.01,13357,2,8,"rd",8,"BEHJvP29Lb"
// Received URC of response, and Show the options of 8(location-path) “rd” and 8(location-path)
“BEHJvP29Lb”

// Set the “Showra” to 1, when received the data from the peer, will show the ip address and port.
AT+QCOAPCFG=”Showra”,1
OK

AT+QCOAPOPTION=5,11,"rd",12,"40",15,"ep=I@#@G7pjZBAQUoVav0yXQcQ6v4Ad1nFGXcKhnT
WYJt3AE0TpZSj3xDrYMOm6NmohixF@#@M100000089",15,"b=U",15,"lt=2700"
OK

AT+QCOAPSEND=0,2,220.180.239.212,8063
>3c2f3e3b72743d226f6d612e6c776d326d222c3c2f30
OK

+QCOAPURC: "rsp",220.180.239.212,8063,2,2.01,13358,2,8,"rd",8,"dTuzhqieJe"
// Received URC of response, and Show the options of 8(location-path) “rd” and 8(location-path)
“”, and also show the remote server ip address and port.

AT+QCOAPCFG=”Showrspopt”,0
OK

AT+QCOAPCFG=”Showra”,0
OK

AT+QCOAPOPTION=2,11,"hello",15,"name=aaaa"
OK

AT+QCOAPSEND=0,1,220.180.239.212,8098
>                                          //type Ctrl+Z payload is none
OK

// The Content-Format is “0”, and the data format is text string.
+QCOAPURC: "rsp",2,2.05,54750,1,12,"0",41, hi name=aaaa,this is quectel iot platform

AT+QCOAPDEL                    // Delete the CoAP context.
```

```
OK
```

4.4. Example of using the dtls

```
// Open DTLS Settings PSK and PSKID
AT+QCOAPCFG="Dtls",1,0,867726030003414,d42b837b1b6590536cd0739825a9cea8
OK

// Create a coap instance
AT+QCOAPCREATE=56830
OK

AT+QCOAPHEAD=1
OK

//Set up registration information and options
AT+QCOAPOPTION=6,11,"rd",12,"42",15,"lwm2m=1.0",15,"ep=867726030003414",15,"b=U",15,"lt=
                                                                                    86400"
OK
// Add resource /19/0/0
AT+QCOAPADDRES=7,/19/0/0
OK

// Add resource /19/1/0
AT+QCOAPADDRES=7,/19/1/0
OK

//Send registration information and payload
AT+QCOAPSEND=0,2,220.180.239.212,8065
></>;rt="oma.lwm2m",</1/0>,</3/0>,</4/0>,</5/0>,</20/0>,</19/0>
OK

+QDTLSSTAT:0

+QCOAPURC: "rsp",2,2.01,12134        //Received a successful reply
```