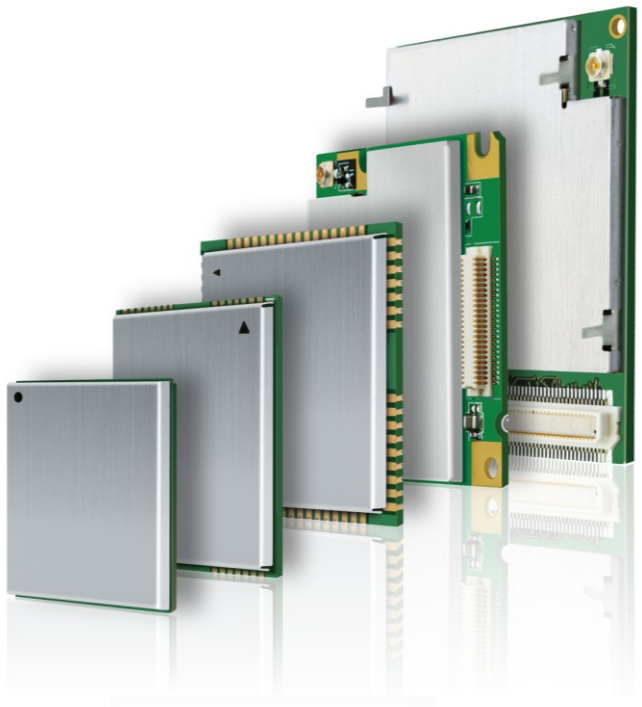
GSM

Quectel Cellular Engine

GSM TCPIP Application Notes

GSM_TCPIP_AN_V1.1



Document Title	GSM TCPIP Application Notes
Version	1.1
Date	2011-09-30
Status	Release
Document Control ID	GSM_TCPIP_AN_V1.1

General Notes

Quectel offers this information as a service to its customers, to support application and engineering efforts that use the products designed by Quectel. The information provided is based upon requirements specifically provided for customers of Quectel. Quectel has not undertaken any independent search for additional information, relevant to any information that may be in the customer's possession. Furthermore, system validation of this product designed by Quectel within a larger electronic system remains the responsibility of the customer or the customer's system integrator. All specifications supplied herein are subject to change.

Contents

0. Revision history

Revision	Date	Author	Description of change
1.00	2009-06-27	Colin HU	Initial
1.01	2009-11-20	Colin HU	Added the introduction on how to maintain and detect TCP connection
1.1	2011-09-07	Jean HU	Reconstruction

1. Introduction

This document introduces how to use the internal TCP/IP stack. Customers can use the powerful and convenient functions to operate the internal TCP/IP stack.

1.1. Reference

Table 1: Reference

SN	Document name	Remark
[1]	Mxx_ATC	AT commands set for Mxx

1.2. Terms and abbreviations

Table 2: Terms and abbreviations

Abbreviation	Description
APN	Access Point Network
CSD	Circuit Switched Data
FGCNT	Foreground Context. The internal TCP/IP stack supports to activate two GPRS PDP contexts at the same time and Foreground context is the context controlled by the UART at present.
GPRS	General Packet Radio Service
MUXIP	The function to visit several servers or listen to multiple clients based on the same GPRS/CSD context
TCP	Transmission Control Protocol
UART	Universal Asynchronous Receiver/Transmitter
UDP	User Datagram Protocol

2. Structure and processes of TCP/IP stack

2.1. Structure of TCP/IP stack

The internal TCP/IP stack is designed to communicate with one or several TCP or UDP ends. The internal TCP/IP stack provides a series of AT commands to open TCP or UDP sessions and communicate with the other TCP or UDP ends.

There are two transmission modes in TCP/IP stack: non-transparent mode (see Chapter 3) and transparent mode (see Chapter 4). In non-transparent mode, the module can establish single connection or multiple connections. When **AT+QIMUX=1** (Multiple Connections), the module supports to establish 6 connections as clients and also supports to listen to multiple clients as a server, up to 5 clients. In transparent mode, the module only can establish single connection. Please refer to ***Figure 1. Structure of TCP/IP stack*.**

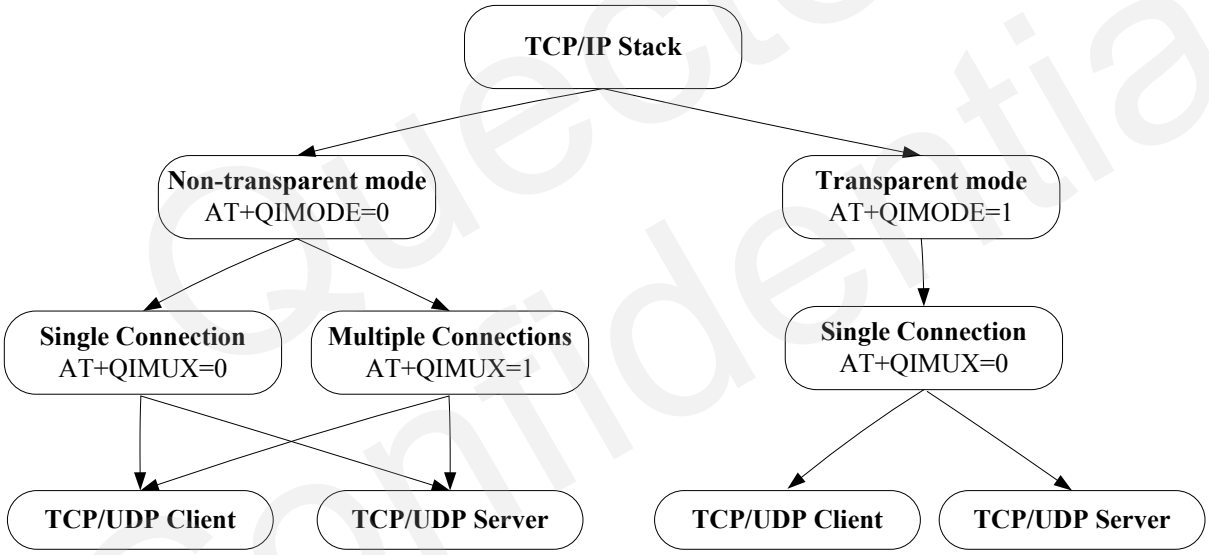


Figure 1: Structure of TCP/IP stack

2.2. Processes of TCP/IP stack

About processes of TCP/IP stack please refer to ***Figure 2. Processes of TCP/IP stack.***

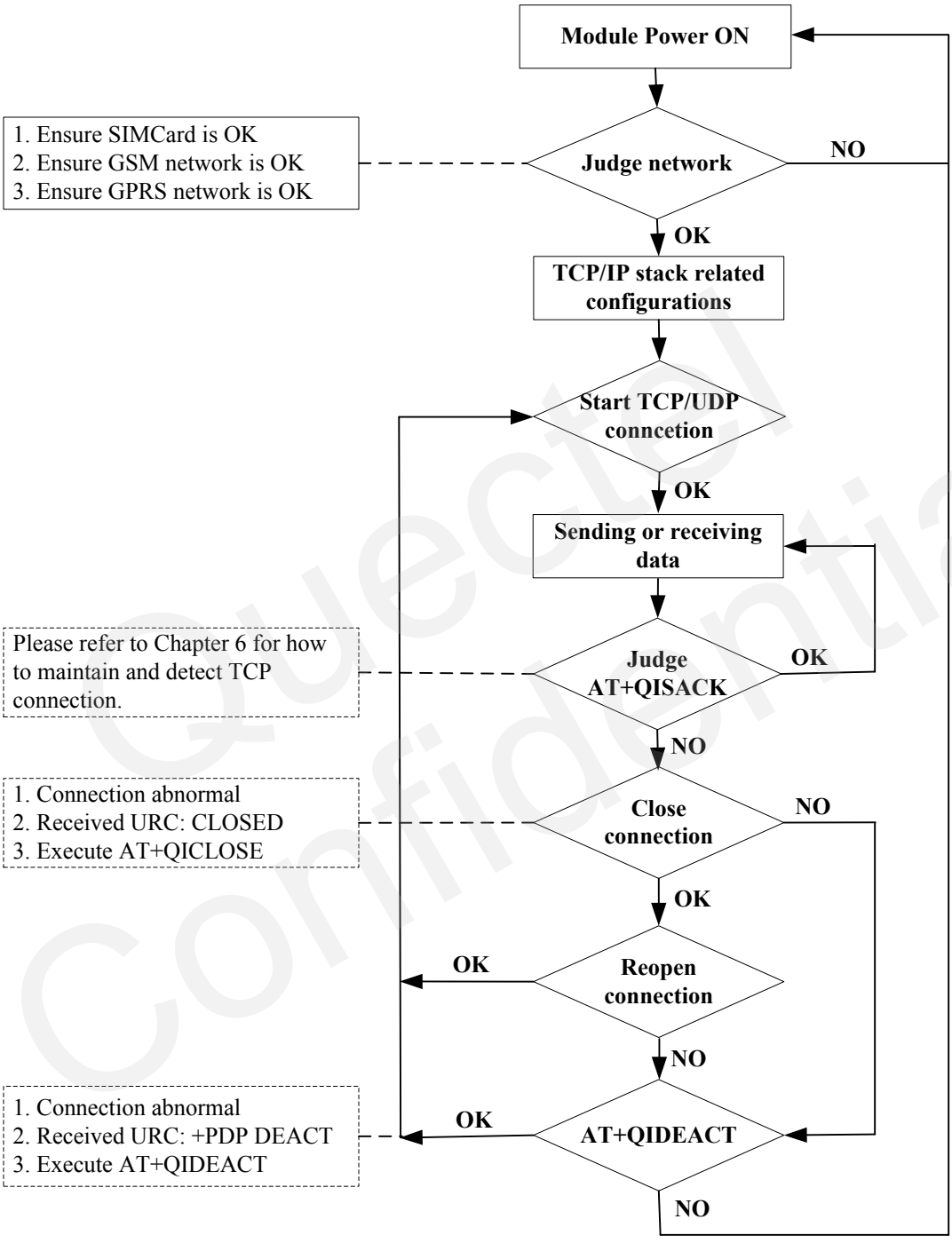


Figure 2: Processes of TCP/IP stack

3. Non-transparent mode

Use **AT+QIMODE** command to select TCPIP stack mode. It is non-transparent mode when **AT+QIMODE=0**. When **AT+QIMUX=0**, the module supports to establish a TCP connection or a UDP connection as a client (see Section 3.1.), and it also supports to listen to a client as a server (see Section 3.2.). When **AT+QIMUX=1** (Multiple Connections), the module supports to establish 6 connections at most as clients (see Section 3.3.) and also supports to listen to multiple clients as a TCP server, up to 5 clients (see Section 3.4.).

Following is the configuration before establishing TCP or UDP connection in non-transparent mode.
First, configure the context and transmission mode.

```
AT+QIFGCNT=0                    // Set the context 0 as FGCNT.
OK
AT+QICSGP=1,“CMNET”             //Set APN.
OK
AT+QIMUX=0                      // 0 disables the function of MUXIP. 1 enables the function.
OK
AT+QIMODE=0                     // Set the session mode as non-transparent.
OK
AT+QIDNSIP=0                    // Use IP address to establish TCP/UDP session.
OK
```

Second, activate the PDP context. (These steps also can be ignored when start TCP or UDP connection by **AT+QIOPEN** or enter listening state by **AT+QISERVER**)

```
AT+QIREGAPP                     // Register the TCP/IP stack.
OK
AT+QIACT                        // Activate FGCNT.
OK
AT+QILOCIP                      //Get Local IP address.
10.180.217.206
```

3.1. Single TCP or UDP connection as a client

Establish TCP or UDP connection by **AT+QIOPEN** (***AT+QIOPEN="Protocol","IP Address","Port"***). **CONNECT OK** will be output after the module established a TCP or UDP connection. The following is an example to use related commands.

Step1: Start TCP or UDP connection and send/receive data

```
AT+QIOPEN=“TCP”, “124.79.167.121”,7007   // Visit the remote TCP server. And the address of
                                            the remote server is an IP address.
OK
```

CONNECT OK	// **CONNECT OK** means the module has been successfully connected to the remote TCP server.

Step2: Send and receive data after establishing connection.

AT+QISEND	// Send data to the remote server.
> Welcome to use Quectel module.	// '>' from the UART indicates the following input data is considered as data to be sent. After receiving '>', input data ('Welcome to use Quectel module.').The maximum length of the data is 1460. The data beyond 1460 will be omitted.
<Ctrl+Z>	// After input data, use **<Ctrl+Z>** to send data. (**<Ctrl+Z>** is a combined AT command which uses to submit operation).
SEND OK	// **SEND OK** means the data has been sent.
AT+QISACK	// Query the total size of the data sent and acknowledged.
+QISACK: 30, 20, 10	// The total size of the data sent is 30, the total size of the data acknowledged is 20 and the length of the data unacknowledged is 10.
OK	
Welcome to use Quectel module.	// Received data from remote server (124.79.167.121:7007)). Besides, it is allowed to add some other information at the beginning of the received data controlled by the commands, please refer to Section 5.3.

Step3: Close the TCP or UDP connection and deactivate GPRS/CSD context.

AT+QICLOSE	// Close the session with the remote server.
CLOSE OK	
AT+QIDEACT	// Deactivate GPRS/CSD context.
DEACT OK	

3.2. Single TCP or UDP connection as a server

AT+QISERVER is used to enter listening mode.

First, take module A as a server.

Second, use module B to establish TCP or UDP connection with the server by **AT+QIOPEN** (***AT+QIOPEN="Protocol","IP Address","Port"***). **CONNECT OK** will be output after the module established a TCP or UDP connection. **REMOTE IP: IP Address** will be indicated in the module A after connection is established successfully.

Note:

The server module (Module A) also can establish a TCP or UDP connection with remote server as a client.

Step1: Enter listening state.

```
// Take module A as a server.
AT+QISERVER                    // Enter listening state to listen to a TCP client and it is also
                                  supported to implement AT+QISERVER=1 to listen to a UDP
                                  client.
OK

SERVER OK                      // Enter listening state successfully.
AT+QILOCIP                     // Query the local IP address.
10.79.142.227
AT+QILPORT?                    // Query the local TCP and UDP port.
TCP: 2020
UDP: 3030

OK
```

Step2: Accept a remote client connection and start a connection to a remote server.

```
// Use module B to establish TCP or UDP connection with the module server A (IP address is the first
module's IP address, the TCP or UDP port is the first module's port.) (About how to establish TCP or
UDP connection in non-transparent mode, please refer to Section3.1 or Section 3.3)
REMOTE IP: 10.79.141.26        // The TCP client whose address is '10.79.141.26' tries to visit the
                                  module. After the module listens to the visit and accepts the visit
                                  successfully, it will output the following string through UART. It
                                  doesn't output the following string until receiving the first packet
                                  from the UDP client '10.79.141.26'.

// The module also supports to establish connection with remote server as a client after entering listening
mode. Following is server module used to start a TCP connection as a client.
AT+QIOPEN="TCP", "124.79.167.121",7007
                               // Visit the remote TCP server. And the address of the remote
                                  server is an IP address.
OK

CONNECT    OK                  // CONNECT OK means the module has been successfully
                                  connected to the remote TCP server.
```

Step3: Select to send data to remote client or remote server.

```
AT+QISRVC=2                    // Choose the connection in which MS acts as a server.
OK
AT+QISEND                      // Send data to the remote client.
> Welcome to use Quectel module.  // '>' from the UART indicates the following input data is
                                  considered as data to be sent. After receiving '>', input data
                                  ('Welcome to use Quectel module.'). The maximum length of
                                  the data is 1460. The data beyond 1460 will be omitted.
```

<Ctrl+Z>	// After input data, use **<Ctrl+Z>** to send data.
SEND OK	// **SEND OK** means the data has been sent.
AT+QISACK	// Query the total size of the data sent and acknowledged.
+QISACK: 30, 20, 10	// The total size of the data sent is 30, the total size of the data acknowledged is 20 and the length of the data unacknowledged is 10.
OK	
AT+QISRVC=1	// Choose the connection in which MS serves as a client.
OK	
AT+QISEND	// Send data to the remote server.
> Welcome to use Quectel module.	// '>' from the UART indicates the following input data is considered as data to be sent. After receiving '>', input data ('Welcome to use Quectel module.').The maximum length of the data is 1460. The data beyond 1460 will be omitted.
<Ctrl+Z>	// After input data, use **<Ctrl+Z>** to send data.
SEND OK	// **SEND OK** means the data has been sent.
AT+QISACK	// Query the total size of the data sent and acknowledged.
+QISACK: 30, 20, 10	// The total size of the data sent is 30, the total size of the data acknowledged is 20 and the length of the data unacknowledged is 10.
OK	
Welcome to use Quectel module.	// Received data from Module B (The module A is limited to receive data from remote server (124.79.167.121:7007)). Besides, it is allowed to add some other information at the beginning of the received data controlled by the commands, such as data length, protocol type, IP address and port, please refer to Section 5.3 for details.

Step4: Close remote client connection or remote server connection. Deactivate GPRS/CSD context.

AT+QISRVC=2	// Choose the connection in which MS act as a server.
OK	
AT+QICLOSE	// The first **AT+QICLOSE**, close the session with the remote client.
CLOSE OK	
AT+QICLOSE	// The second **AT+QICLOSE**, close the server listening. (If there is no connection, the first **AT+QICLOSE** will close the server listening).
CLOSE OK	
AT+QISRVC=1	// Choose the connection in which MS serves as a client.
OK	
AT+QICLOSE	// Close the session with the remote server.

```
CLOSE OK
AT+QIDEACT                  // Deactivate GPRS/CSD context.
DEACT OK
```

3.3. Multiple TCP or UDP connections as clients

The module supports to establish 6 connections at a time when **AT+QIMUX=1**. ***Please notice that it is only supported in non-transparent mode.***

Establish TCP or UDP connection by **AT+QIOPEN** (***AT+QIOPEN=<index>,"Protocol","IP Address","Port"***). **<index>, CONNECT OK** will be output after the module established a TCP or UDP connection. Then, use **AT+QISEND=<index>** to send data to remote server. For more details on how to send data in different ways, please refer to Section 5.2. In the multiple mode, **AT+QIOPEN**, **AT+QISEND**, **AT+QISACK** and **AT+QICLOSE** commands need connection index to indicate which connection it is. The following is an example to use related commands.

Step1: Start TCP or UDP connections and send/receive data.

```
AT+QIOPEN=0,"TCP", "124.79.167.121",7007
                            // Use the profile 0 to visit the remote TCP server. And the address
                               of the remote server is an IP address. The index of profile is 0~5.
OK

0, CONNECT OK               // 0, CONNECT OK means that the profile 0 has been successfully
                               connected to the remote TCP server.
AT+QISEND=0                 // Send data to the remote server by profile 0 connection.
> Welcome to use Quectel module.
                            // '>' from the UART indicates the following input data is
                               considered as data to be sent. After receiving '>', input data
                               ('Welcome to use Quectel module.'). The maximum length of the
                               data is 1460. The data beyond 1460 will be omitted.
<Ctrl+Z>                    // After input data, use <Ctrl+Z> to send data.
SEND OK                     // SEND OK means the data has been sent.
AT+QISACK=0                 // Query the total size of the data sent and acknowledged.
+QISACK: 30, 20, 10         // The total size of the data sent is 30, the total size of the data
                               acknowledged is 20 and the length of the data unacknowledged
                               is 10.

OK
+RECEIVE: 0, 30
Welcome to use Quectel module.
                            // 0 is the index of the profile to receive the data and 30 is the
                               length of the received data. 'Welcome to use Quectel module.'
                               is the received data. Besides, it is allowed to add some other
                               information at the beginning of the received data controlled by
                               the commands, please refer to Section 5.3.
```

Step2: Close the TCP or UDP connection and deactivate GPRS/CSD context.

```
AT+QICLOSE=0                    // Close the profile 0 connection with the remote server.
CLOSE OK
AT+QIDEACT                      // Deactivate GPRS/CSD context.
DEACT OK
```

3.4. Multiple TCP or UDP connections as a server

The module supports to listen to multiple clients as a TCP or UDP server, up to 5 clients. Customer also can establish n connection(s) as clients and listen to m connection(s) as a server (***n>=0&m>=0&n+m<=5***).

Note:

The server module also can establish a TCP or UDP connection with remote server as clients.

First, take module A as a server.

Second, use module B to establish TCP or UDP connection with the server by **AT+QIOPEN** (***AT+QIOPEN*= <index>,"Protocol","IP Address","Port"**). **<index>, CONNECT OK** will be output after the module established a TCP or UDP connection. **<index>, REMOTE IP: IP Address** will be indicated in the module A after connection is established successfully.

The following is an example to use related commands.

Step1: Enter listening state.

```
// Take module A as a server
AT+QISERVER=0,3                 // 0 means to set the module as a TCP server. 3 is the maximum
                                   clients to listen to, i.e. after the number of clients reaches 3, no
                                   more other clients can visit the module.
OK

SERVER OK                       // Enter listening state successfully.
AT+QILOCIP                      // Query the local IP address.
10.79.142.227
AT+QILPORT?                     // Query the local TCP and UDP port.
TCP: 2020
UDP: 3030

OK
```

Step2: Accept a remote client connection and start a connection to a remote server.

// Then, use module B to establish TCP or UDP connection with the module server (IP address is the module A's IP address, the TCP or UDP port is the first module's port.) (About how to establish TCP or UDP connection in non-transparent mode, please refer to Section3.1 or Section 3.3).

0, REMOTE IP: 10.79.141.26	// Profile 0 is used to establish the session with the client "10.79.141.26". It will output the following string through UART. It doesn't output the following string until receiving the first packet from the **UDP client** "10.79.141.26".
// The server module (module A) also supports to establish connection with remote server as a client after entering listening mode.	
AT+QIOPEN=3,"TCP", "124.79.167.121",7007	
	// Visit the remote TCP server. And the address of the remote server is an IP address.
OK	
3, CONNECT OK	// **CONNECT OK** means the module has been successfully connected to the remote TCP server.

Step3: Select to send data to remote client or remote server.

AT+QISRVC=2	// Choose the connection in which MS acts as a server.
OK	
AT+QISEND=0	// Send data to the remote server by profile 0 connection.
> Welcome to use Quectel module.	// '>' from the UART indicates the following input data is considered as data to be sent. After receiving '>', input data ('Welcome to use Quectel module.'). The maximum length of the data is 1460. The data beyond 1460 will be omitted.
<Ctrl+Z>	// After input data, use **<Ctrl+Z>** to send data.
SEND OK	// **SEND OK** means the data has been sent.
AT+QISACK=0	// Query the total size of the data sent and acknowledged.
+QISACK: 30, 20, 10	// The total size of the data sent is 30, the total size of the data acknowledged is 20 and the length of the data unacknowledged is 10.
OK	
AT+QISRVC=1	// Choose the connection in which MS serves as a client.
OK	
AT+QISEND=3	// Send data to the remote server by profile 0 connection.
> Welcome to use Quectel module.	// '>' from the UART indicates the following input data is considered as data to be sent. After receiving '>', input data ('Welcome to use Quectel module.'). The maximum length of the data is 1460. The data beyond 1460 will be omitted.
<Ctrl+Z>	// After input data, use **<Ctrl+Z>** to send data.
SEND OK	// **SEND OK** means the data has been sent.
AT+QISACK=3	// Query the total size of the data sent and acknowledged.
+QISACK: 30, 20, 10	// The total size of the data sent is 30, the total size of the data acknowledged is 20 and the length of the data unacknowledged

is 10.

OK

+RECEIVE: 0, 30
Welcome to use Quectel module. // **0** is the index of the profile to receive the data and **30** is the length of the received data. 'Welcome to use Quectel module' is the received data. Besides, it is allowed to add some other information at the beginning of the received data controlled by the commands, please refer to Section 5.3.

Step4: Close remote client connection or remote server connection. Deactivate GPRS/CSD context.

AT+QISRVC=2 // Choose the connection in which MS acts as a server.
OK
AT+QICLOSE=0 // Close the session with the remote client.
CLOSE OK
AT+QICLOSE // Close the server listening.
CLOSE OK

AT+QISRVC=1 // Choose the connection in which MS serves as a client.
OK
AT+QICLOSE=3 // Close the session with the remote server.
CLOSE OK

AT+QIDEACT // Deactivate GPRS/CSD context.
DEACT OK

4. Transparent mode

Use **AT+QIMODE** command to select TCPIP stack mode. It is transparent mode when **AT+QIMODE=1**. When **AT+QIMUX=0**, the module supports to establish a TCP connection or a UDP connection as a client (see Section 4.1.), and it also supports to listen to a client as a TCP/UDP server (see Section 4.2.). Establishing multiple connections is not supported in transparent mode.

Following is the configuration before establishing TCP or UDP connection in transparent mode.
First, configure the context and transmission mode.

AT+QIFGCNT=0	// Set the context 0 as FGCNT.
OK	
AT+QICSGP=1,"CMNET"	// Set APN
OK	
AT+QIMUX=0	// **0** disables the function of MUXIP.
OK	
AT+QIMODE=1	// Set the session mode as transparent.
OK	
AT+QITCFG=3,2,512,1	// **3** stands for resending times. This is an internal process, which doesn't have an obvious influence. **2** means to set wait interval as 200ms, which means system will wait for 200 ms before sending the data in the buffer when the data in the buffer is less than 512. **512** means that once the length of the data in the buffer reaches 512, the data will be sent out. **1** means to enable to escape data mode with '+++'.
OK	
AT+QIDNSIP=0	// Use IP address as the address to establish TCP/UDP session.
OK	

Second, activate the PDP context. (Those steps also can be ignored when start TCP or UDP connection by **AT+QIOPEN** or enter listening state by **AT+QISERVER**).

AT+QIREGAPP	// Register the TCP/IP stack.
OK	
AT+QIACT	// Activate FGCNT.
OK	
AT+QILOCIP	// Get Local IP address.
10.180.217.206	

In transparent mode, UART has two modes, AT command mode and data mode.

In AT command mode, all the input data through UART is considered as AT command and the received data from the remote end is saved in the buffer and cannot be output through UART until the UART is switched to data mode.

In data mode, all the input data through UART is considered as the data to be sent to the remote end and all the received data from the remote end is output through UART. When the session mode is transparent mode, the UART will enter data mode after the TCP/UDP session is established.

The internal TCP/IP stack supports to switch the UART mode between AT command mode and data mode. The method to switch data mode to AT command mode is to input '+++'. The interval time between the first "+" and the character before the first '+' **MUST NOT** be less than 500 ms and the interval time between the last '+' and the character next to the last '+' **MUST NOT** be less than 500 ms and the interval time between each '+' **MUST** be less than 20 ms. The method to switch AT command mode to data mode is to implement the command **ATO**.

Besides, the internal TCP/IP stack provides hardware method to switch the UART mode from data mode to AT mode. In order to adopt the hardware method, it is necessary to issue the command **AT&D1** at first. Then it is supported to change DTR from ON to OFF in data mode and to switch the UART mode to AT mode.

Note:

- ***If the connection is closed by remote ends or network, or the PDP context is deactivated during data transferring, the module will switch to command mode automatically.***
- ***If an incoming call or a new short message arrived in data mode, RI pin of serial port will give a 12-ms low pulse. To handle the event, it is needed to enter command mode first (+++/DTR) and then the module will give the normal unsolicited response, such as RING or +CMTI: "SM", 1.***

4.1. Single TCP or UDP connection as a client

Establish TCP or UDP connection by **AT+QIOPEN** (***AT+QIOPEN="Protocol","IP Address","Port"***). **CONNECT OK** will be output after the module established a TCP or UDP connection. The following is an example to use related commands.

The following is an example to use related commands.

Step1: Start a TCP or UDP connection and send/receive data.

AT+QIOPEN="TCP", "124.79.167.121",7007	// Visit the remote TCP server. And the address of the remote server is an IP address.
OK	
CONNECT	// **CONNECT** means the module has been connected to the remote TCP server and entered data mode.
Welcome to use Quectel module.	// Input data directly. It is invisible and will be transmit to the remote server. The module will output data (**Welcome to use Quectel module.**) which came from remote server.
OK	// '+++' is used to switch data mode to AT command mode. It will return **OK** after switching to AT command mode.

```
ATO                          // ATO is used to switch AT command mode to data mode.
CONNECT                      // It will return CONNECT after switching to data command mode.
….
OK                           // Input '+++'
AT+QISACK                    // Query the total size of the data sent and acknowledged.
+QISACK: 30, 20, 10          // The total size of the data sent is 30, the total size of the data
                               acknowledged is 20 and the length of the data unacknowledged is
                               10.

OK
```

Step2: Close the TCP or UDP connection and deactivate GPRS/CSD context.

```
// Note: Please ensure the module is in AT command mode before doing following operations.
AT+QICLOSE                   // Close the session with the remote server.
CLOSE OK
AT+QIDEACT                   // Deactivate GPRS/CSD context.
DEACT OK
```

4.2. Single TCP or UDP connection as a server

AT+QISERVER is used to enter listening mode.

First, take module A as a server.

Second, use module B to establish TCP or UDP connection with the server by **AT+QIOPEN** (***AT+QIOPEN="Protocol","IP Address","Port"***). **CONNECT OK** will be output after the module established a TCP or UDP connection. **REMOTE IP: IP Address** will be indicated in the module A after connection is established successfully.

Note:
The server module (Module A) also can establish a TCP or UDP connection with remote server as a client.

Step1: Enter listening state.

```
// Take module A as a server
AT+QISERVER                  // Enter listening state to listen to a TCP client and it is also
                               supported to implement AT+QISERVER=1 to listen to a UDP
                               client.
OK

SERVER OK                    // Enter listening state successfully
AT+QILOCIP                   // Query the local IP address.
10.79.142.227
```

```
AT+QILPORT?                    // Query the local TCP and UDP port.
TCP: 2020
UDP: 3030

OK
```

Step2: Accept a remote client connection and start a connection to a remote server.

```
// Then, use module B to establish TCP or UDP connection with the module server (Module A)(IP address
is the first module's IP address, the TCP or UDP port is the first module's port.) (About how to establish
TCP or UDP connection in transparent mode, please refer to Section 4.1.)
REMOTE IP: 10.79.141.26        // The TCP client whose address is "10.79.141.26" tries to visit the
                                  module. After the module listens to the visit and accepts the visit
                                  successfully, it will output the following string through UART. It
                                  doesn't output the following string until receiving the first packet
                                  from the UDP client "10.79.141.26".
CONNECT                        // CONNECT means the module has been connected to the remote
                                  TCP server and entered data mode.
Welcome to use Quectel module. // Input data directly. It is invisible and will be transmit to the
                                  remote server. The module will output data (Welcome to use
                                  Quectel module.) which came from remote server.
OK                             // '+++' is used to switch data mode to AT command mode. It will
                                  return OK after switching to AT command mode.
ATO                            // ATO is used to switch AT command mode to data mode.
CONNECT                        // It will return CONNECT after switching to data command mode.
....
OK                             // Input '+++'.
AT+QISACK                      // Query the total size of the data sent and acknowledged.
+QISACK: 30, 20, 10            // The total size of the data sent is 30, the total size of the data
                                  acknowledged is 20 and the length of the data unacknowledged
                                  is 10.

OK
```

Step3: Close the TCP or UDP connection and deactivate GPRS/CSD context.

```
// Please ensure the module is in AT command mode before doing following operations.
AT+QICLOSE                     // Close the session with the remote server.
CLOSE OK
AT+QIDEACT                     // Deactivate GPRS/CSD context.
DEACT OK
```

5. Related configuration

5.1. Accept connection

When **AT+QIMUX=0**, the TCP client whose address is "10.79.141.26" tries to visit the module. After the module listens to the visit and accepts the visit successfully, it will output the following string through UART.

```
REMOTE IP: 10.79.141.26
[CONNECT]                    // It will output CONNECT here in transparent mode.
```

When **AT+QIMUX=1** (Only in non-transparent mode), the TCP client whose address is "10.79.141.26" tries to visit the module. After the module listens to the visit and accepts the visit successfully, it will output the following string through UART.

```
0, REMOTE IP: 10.79.141.26   // Profile 0 is used to establish the session with the client
                                '10.79.141.26'.
```

5.2. Send data

5.2.1. Send data in different ways

The module supports three ways to send data to remote ends: sending data with changeable length, sending data with fixed length and timed sending.

5.2.2.1. Sending data with changeable length

When **AT+QIMUX=0**, after establishing TCP or UDP connection, send data as below.

```
AT+QISEND                    // Send data to the remote server.
> Welcome to use Quectel module.   // '>' from the UART indicates the following input data is
                                   considered as data to be sent. After receiving '>', input data
                                   ('Welcome to use Quectel module.'). The maximum length
                                   of the data is 1460. The data beyond 1460 will be omitted.
<Ctrl+Z>                     // After input data, use <Ctrl+Z> to send data.
SEND OK                      // SEND OK means the data has been sent.
```

When **AT+QIMUX=1**, after establishing TCP or UDP connection by profile 0, send data as below.

```
AT+QISEND=0                  // Use the connection of profile 0 to send data to the remote end.
> Welcome to use Quectel module.   // '>' from the UART indicates the following input data is
                                   considered as data to be sent. After receiving '>', input data
                                   ('Welcome to use Quectel module.'). The maximum length of
```

	the data is 1460. The data beyond 1460 will be omitted.
<Ctrl+Z>	// After input data, use **<Ctrl+Z>** to send data.
SEND OK	// **SEND OK** means the data has been sent.

Note:

In this method, the sending operation can be escaped through inputting <ESC> before sending out.

5.2.2.2. Sending data with fixed length

When **AT+QIMUX=0**, after establishing TCP or UDP connection, send data as below.

AT+QISEND=30	// Send 30 bytes data to the remote server.
> Welcome to use Quectel module.	// '>' from the UART indicates the following input data is considered as data to be sent. After receiving '>', input data ('Welcome to use Quectel module.'). The maximum length of the data is 1460. The data beyond 1460 will be omitted.
SEND OK	// The data will be sent when input data length equals to 30. **SEND OK** means the data has been sent.

When **AT+QIMUX=1**, after establishing TCP or UDP connection by profile 0, send data as below.

AT+QISEND=0,30	// Send 30 bytes data to the remote server.
> Welcome to use Quectel module.	// '>' from the UART indicates the following input data is considered as data to be sent. After receiving '>', input data ('Welcome to use Quectel module.'). The maximum length of the data is 1460. The data beyond 1460 will be omitted.
SEND OK	// The data will be sent when input data length equals to 30. **SEND OK** means the data has been sent.

Note:

In this method, <Ctrl+Z> is considered as a common character to be sent. The sending operation can be escaped through inputting <ESC> before sending out.

5.2.2.3. Timed sending

Use **AT+QIAUTOS=1, <period>** to set timed sending.

AT+QIAUTOS=1,10	// Set the 10s to wait before sending data to the remote end.
OK	

When **AT+QIMUX=0**, after establishing TCP or UDP connection, send data as below.

AT+QISEND	// After this command, then the preset timer starts.
>	// Echo from the UART indicates the following input data is considered as data to be sent.
Welcome to use Quectel module.	// Input data and the maximum length of the data is 1460. The data beyond 1460 will be omitted. After the timer expires, the data will be sent.
SEND OK	// The data has been sent. If the length of the input data is 0, it

returns **SEND FAIL** here

When **AT+QIMUX=1**, after establishing TCP or UDP connection by profile 0, send data as below.

AT+QISEND=0	// After this command, then the preset timer starts.
>	// Echo from the UART indicates the following input data is considered as data to be sent.
Welcome to use Quectel module.	// Input data and the maximum length of the data is 1460. The data beyond 1460 will be omitted. After the timer expires, the data will be sent.
SEND OK	// The data has been sent. If the length of the input data is 0, it returns **SEND FAIL** here.

Note:

In this method, if <Ctrl+Z> is inputted before the timer expires, the data before <Ctrl+Z> will be sent and the timer stops; if sending data with fixed length, the data will be sent when time or length arrived, and the sending operation can be escaped through inputting <ESC> before sending out.

5.2.2. Query the information of the sending data

It is optional to query the total size of the data sent and acknowledged by the command **AT+QISACK**, which is only used for TCP connection. In UDP connection, the acknowledged data is always 0. The following is an example to use the command **AT+QISACK**.

When **AT+QIMUX=0,** after establishing TCP or UDP connection, query the data sent as below.

AT+QISACK	// Query the total size of the data sent and acknowledged.
+QISACK: 30, 20, 10	// The total size of the data sent is 30, the total size of the data acknowledged is 20 and the length of the data unacknowledged is 10.
OK	

When **AT+QIMUX=1**, after establishing TCP or UDP connection by profile 0, query the data sent as below.

AT+QISACK=0	// Query the total size of the data sent and acknowledged.
+QISACK: 30, 20, 10	// The total size of the data sent is 30, the total size of the data acknowledged is 20 and the length of the data unacknowledged is 10.
OK	

5.3. Receive data

5.3.1. Output the data through UART directly

In default setting, the received data from the server will be output through UART without any information. Besides, it is allowed to add some information at the beginning of the received data controlled by some commands. Please refer to Document [1] for the detailed information on these commands mentioned below.

The command **AT+QISHOWRA** is used to set whether to add the address and port of the remote server.

```
AT+QISHOWRA=1          // Add the address and port of the remote server (RECV FROM:<IP ADDRESS>:<PORT>) before the received data.
OK
```

The command **AT+QISHOWLA** is used to set whether to add the local address. The following is the example to use this command.

```
AT+QISHOWLA=1          // Add the local address before the received data (TO:<IP ADDRESS>).
OK
```

The command **AT+QIHEAD** is used to set whether to add the header information (data length and transmission layer type if necessary). The following is the detailed explanation to use the command.

```
AT+QIHEAD=1            // Add the header information before the received data (IPD (data length) :).
OK
```

The command **AT+QISHOWPT** is used to set whether to add the transmission layer protocol type (TCP or UDP) at the end of header information. The following is the example to use the command.

```
// Establish a TCP connection while AT+QIMUX=0.
AT+QIHEAD=1
OK
AT+QISHOWPT=1          // Add the transmission layer protocol type (IPD (data length) (TCP/UDP) :).
                          It is invalid if AT+QIHEAD=0, it is invalid in AT+QIMUX=1 mode.
OK
IPD30TCP: Welcome to use Quectel module.   // The module has received data; data length is 30;
                                              TCP connection; data content is "Welcome to use
                                              Quectel module."
```

In MUXIP mode (**AT+QIMUX=1**), the received data from the server will be output through UART with only '**+RECEIVE: <index>, <length>**'. (**<index>** is the index of the profile to receive data and **<length>** is the length of the received data). Besides, it is also allowed to add some other information at the beginning of the received data controlled by the commands **AT+QISHOWRA**, **AT+QISHOWLA**, **AT+QIHEAD** and **AT+QISHOWPT**.

5.3.2. Retrieve the received data via AT command

The module in this mode, compared with the above mode, outputs a URC '**+QIRD:**' to inform the user that TCP/UDP data has been received and user can retrieve the data by command **AT+QIRD**. This mode is not supported by default. Execute command **AT+QINDI=1** to enable this mode before the TCP/UDP connection is established. An example when the module receives the data in this mode is shown as below.

Suppose the module receives the TCP data below from the remote end:
"1234567890abcdefghijklmnopqrstuvwxyz"

```
+QIRDI:0,1,0                    //The module receives the data based on Context 0, and the module acts as
                                  the client.
```

Now retrieve data by the following command.

```
AT+QIRD=0,1,0,1024              //Retrieve the data from the module's socket buffer. The maximum length to
                                  retrieve is 1024. If the data length in the buffer is less than 1024, retrieve all
                                  the data from the buffer.
+QIRD:116.228.146.250:7070,TCP,36<CR><LF>
1234567890abcdefghijklmnopqrstuvwxyz
```

It is recommended to use **AT+QIRD** to read all received data when there is '**+QIRD:**' arrived.

5.4. Close connection

When **AT+QIMUX=0**, after establishing TCP or UDP connection, close connection as below.

```
AT+QICLOSE                      // Close the session with the remote server.
CLOSE OK
```

When **AT+QIMUX=1**, after establishing TCP or UDP connection, use **AT+QICLOSE=<index>** to close one connection as below.

```
AT+QICLOSE=0                    // Close the session with the remote server.
0, CLOSE OK
```

Note:
When connection is established as a server, it is needed to select and control server connection or client contention by AT+QISRVC.

5.5. Deactivate PDP context

Use **AT+QIDEACT** command to deactivate current PDP context.

```
AT+QIDEACT                      // If the context will not be used in a long period (for example more
                                  than one hour), it's recommended to close this context by
                                  command AT+QIDEACT. Normally the response time for this
                                  command is about 2-5 seconds. But when the network is very
```

	bad or in some abnormal conditions, the longest waiting time will achieve about 2.5 minutes. It's recommended to set timeout value which is less than 1 minute or less according to their applications. If it is timeout, users have not received **DEACT OK**, user can restart the module by **EMERG_OFF** pin.
DEACT OK	

Note:
Make sure to receive the corresponding response (or ERROR) and then execute the next command.

5.6. GPRS status

5.6.1. GPRS status of TCP/IP stack

While **AT+QIMUX=0** is enabled, there are 9 status for single connection. Please refer to ***Figure 3. GPRS status of single connection*.**

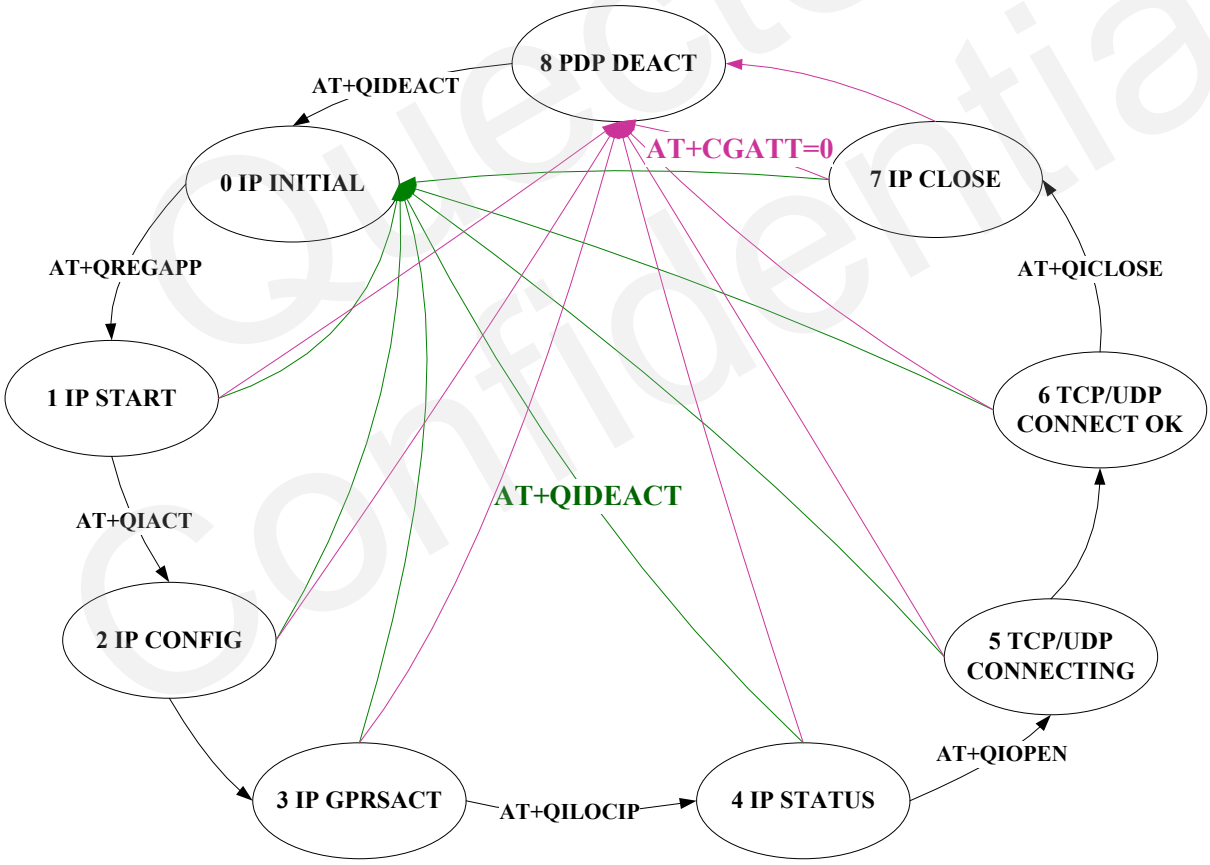


Figure 3: GPRS status of single connection

While **AT+QIMUX=1** is enabled, there are 7 status for multiple connections. Please refer to ***Figure 4.***

GPRS status of multiple connections.

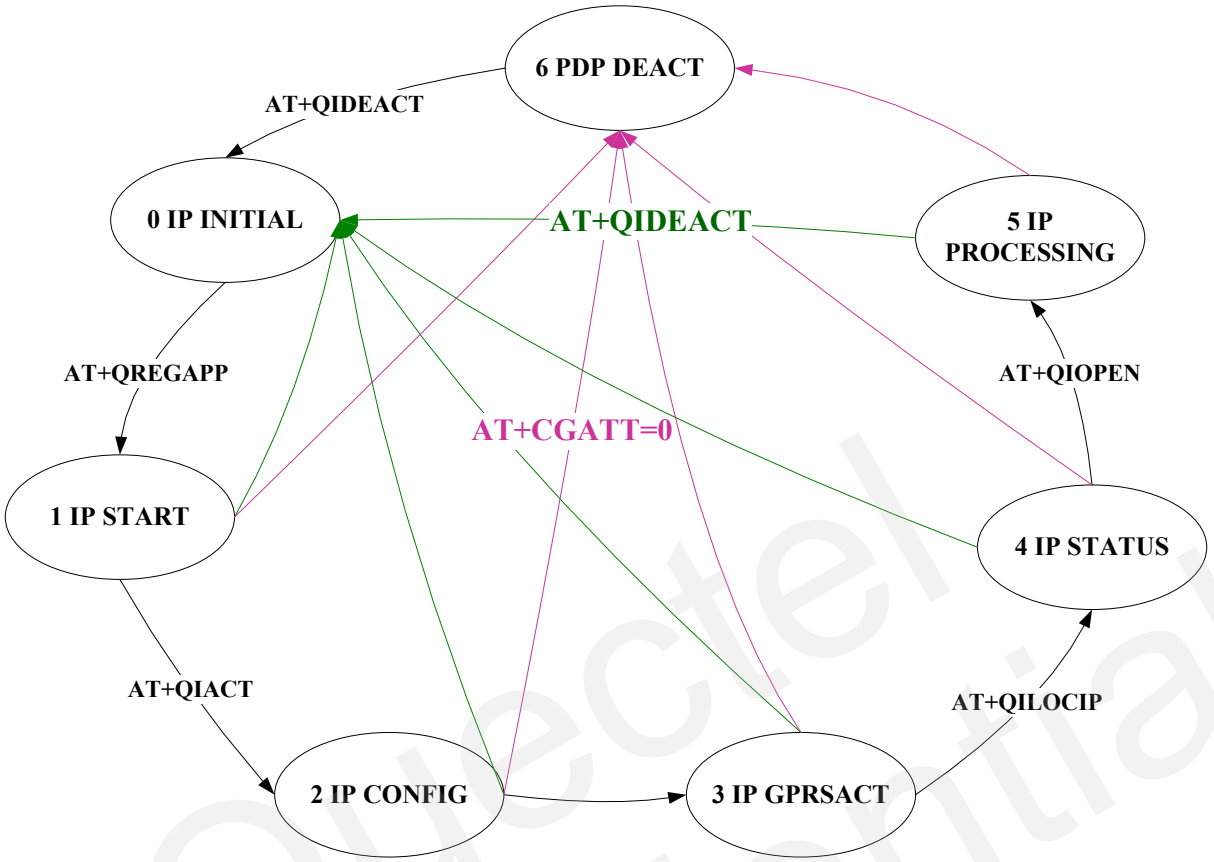


Figure 4: GPRS status of multiple connections

Following is the details of each status.

"IP INITIAL"	The TCPIP stack is in idle state.
"IP START"	The TCPIP stack has been registered.
"IP CONFIG"	It has been start-up to activate GPRS/CSD context.
"IP IND"	It is activating GPRS/CSD context.
"IP GPRSACT"	GPRS/CSD context has been activated successfully.
"IP STATUS"	The local IP address has been gotten by the command **AT+QILOCIP**.
"TCP CONNECTING"	It is trying to establish a TCP connection.
"UDP CONNECTING"	It is trying to establish a UDP connection.
"CONNECT OK"	The TCP/UDP connection has been established successfully.
"IP CLOSE"	The TCP/UDP connection has been closed.
"PDP DEACT"	GPRS/CSD context was deactivated because of unknown reason.
"IP PROCESSING"	Processing the existing connection now.

5.6.2. Query status

The module supports to use **AT+QISTAT** or **AT+QISTATE** to query the TCP/IP stack status.

When **AT+QIMUX=0**, use **AT+QISTAT** to query the TCP/IP stack status.

```
AT+QISTAT

OK

STATE: IP INITIAL
```

When **AT+QIMUX=1**, use **AT+QISTATE** to query the TCP/IP stack status and connection status.

```
AT+QISTATE
OK

STATE: IP INITIAL

+QISTATE: 0,"","",,"INITIAL"

+QISTATE: 1,"","",,"INITIAL"

+QISTATE: 2,"","",,"INITIAL"

+QISTATE: 3,"","",,"INITIAL"

+QISTATE: 4,"","",,"INITIAL"

+QISTATE: 5,"","",,"INITIAL"
```

In listening state, use **AT+QISSTAT** to query the server status.

```
AT+QISSTAT
OK

S: LISTENNING
```

Note:

In transparent mode, DCD pin also can be used for querying current connection status. If TCP or UDP connection exists, the DCD pin will be low. If the connection is dropped, DCD pin will go high.

6. TCP connection maintenance and detection

TCP is a reliable stream delivery service that guarantees delivery of a data stream sent from one host to another without duplication or losing data. But GSM network is very complicated so that some unknown errors occur to the internal TCP/IP stack based on GSM. This seriously affects the capability of TCP connection. So it is necessary to check whether the TCP connection is OK and whether the sent data has been received by the server. The chapter intends to recommend several methods to maintain TCP connection and detect the status of TCP connection.

6.1. TCP connection maintenance

The internal TCP/IP stack is based on GSM network and the resource of GSM network is limited. So the TCP connection may be disconnected by GSM network without any notification if there is no data transmission on the TCP connection for a period of time. Here is a method recommended by Quectel.

In order to maintain a TCP connection, ***it is recommended to send a small data packet to the remote end through the TCP connection and then wait for a moment to use AT+QISACK command to check whether the packet has been received by the remote end.*** Please refer to the following example for the details.

```
// Establish a TCP connection at first.
AT+QISTAT                   // Check the status of the TCP connection by the command
                               AT+QISTAT at first.
OK

STATE: CONNECT OK           // The TCP connection is still OK. This just means the internal
                               TCP/IP stack didn't receive any abnormal report for the TCP
                               connection.
……                          // No data was transmitted through the TCP connection for a period
                               of time, such as 8 minutes.
AT+QISACK                   // Check the information for the sending data. The purpose is to
                               check whether some sent data still wait for the acknowledgement
                               from the remote end. If some sent data still wait for the
                               acknowledgement from the remote end, the TCP connection
                               MUST be abnormal. Then it is strongly recommended to
                               re-establish the TCP connection.
+QISACK: 0, 0, 0            // The third parameter 0 means no data wait for the
                               acknowledgement from the remote end. If the third parameter is
                               not 0, please re-establish the TCP connection.

OK
```

AT+QISEND	// No data was transmitted through the TCP connection for so long time, it is recommended to send a small packet to the remote end to maintain the TCP connection.
> TCP detect	// This is the small packet to be sent to the remote end. Of course, other data is OK.
any	
SEND OK	// The internal TCP/IP stack has sent the data through the TCP connection.
……	// Wait for a moment (for example, 30 seconds) to check whether the packet has been received by the remote end.
AT+QISACK	// Check the information for the sending data.
+QISACK: 10, 10, 0	// The third parameter is 0, the packet has been received by the remote end successfully. This means the TCP connection is still normal.
OK	

In the above example, it waits for a period of time after sending the small packet '**TCP detect**' to issue the command **AT+QISACK** to check whether the packet has been received by the remote end. Here is another method. After sending the packet, wait for 5 seconds, and then issue the command **AT+QISACK**. If the packet has been received by the remote end successfully, the TCP connection is still OK, so stop the check operation. If the packet has not been received by the remote end (according to the value of the third parameter of the response of **AT+QISACK**), please wait for 10 seconds again, and repeat the former operation. Repeat the former operation for several times until the packet has been received by the remote end successfully or the time of repeat reaches to a limitation (for example, 12 times). If the time of repeat reaches to the limitation, the TCP connection is abnormal, so it is suggested to re-establish the TCP connection.

6.2. TCP connection detection

The principle to detect TCP connection is similar to the method of maintaining TCP connection. After a TCP connection has been established, here is a sample process to detect the status of the TCP connection.

// Establish a TCP connection at first.	
AT+QISTAT	// Check the status of the TCP connection by the command **AT+QISTAT** at first.
OK	
STATE: CONNECT OK	// The TCP connection is still OK. This just means the internal TCP/IP stack doesn't receive any abnormal report for the TCP connection. If it is not **CONNECT OK**, the TCP connection is abnormal. Please refer to Section 5.6 for the details of status.
……	// No data is transmitted through the TCP connection for a period of

	time, such as 8 minutes.
AT+QISACK	// Check the information for the sending data. The purpose is to check whether some sent data still wait for the acknowledgement from the remote end. If some sent data still wait for the acknowledgement from the remote end, the TCP connection MUST be abnormal.
+QISACK: 10, 10, 0	// The third parameter 0 means no data wait for the acknowledgement from the remote end. If the third parameter is not 0, the TCP connection is abnormal.
OK	
AT+QISEND	// No data was transmitted through the TCP connection for so long time, it is recommended to send a small packet to the remote end to check whether the connection is still OK.
> TCP detect	// This is the small packet to be sent to the remote end. Of course, any other data is OK.
SEND OK	// The internal TCP/IP stack has sent the data through the TCP connection.
……	// Wait for 5 seconds to check whether the packet has been received by the remote end.
AT+QISACK	// Check the information for the sending data.
+QISACK: 20, 20, 0	// The third parameter is **0**. The packet has been received by the remote end successfully. This means the TCP connection is still normal. If it isn't 0, the packet hasn't been received by the remote end and it is strongly recommended to repeat checking the information for the sending data by the command **AT+QISACK** for at most 12 times. If the third parameter of the response of **AT+QISACK** becomes 0 in some time of repeat, the packet has been sent to the remote end successfully which indicates the TCP connection is still OK, and then please stop the repeat operation. If the third parameter is always not 0, it means the TCP connection is abnormal.
OK	

Note:
If the module was set to enable MUXIP (AT+QIMUX=1), please detect the specified TCP connection, i.e. use "AT+QISACK=<index>" to replace "AT+QISACK" in the above examples.

6.3. Exception handling

If the module receives **CLOSED** or **+PDP DEACT** in the data mode, it means TCP connection has been broken or some abnormalities have happened. But it's also possible that these data are from the remote end. In this case, it's recommended to input '+++' to confirm if the UART is still in the data mode. When input

'+++', **OK** is returned, which means the UART has switched to the command mode successfully and the previously received **CLOSED** or **+PDP DEACT** are TCP data from server. Otherwise, when input '+++' just '+++' echoes (when echo mode is open) and **OK** is not returned, which means the module has already entered in command mode, that is, the previously received **CLOSED** or **+PDP DEACT** means the TCP session has been disconnected or GPRS context has been deactivated. In this case, the TCP session is needed to re-establish as the following steps. (Of course, if it's certain that the remote server will not send a string as **CLOSED** and **+PDP DEACT**, re-establish the TCP session directly and not require to input '+++' to judge the current state.)

AT+QIDEACT	//Deactivate the current GPRS context. The response time for this command is normally about 2-5 seconds. But when the network is very bad or in some abnormal conditions, the longest waiting time will achieve about 2.5 minutes. It's recommended to set timeout value which is less than 1 minute or less according to their applications. If it is timeout **DEACT OK** is still not received, user can restart the module by **EMERG_OFF** pin.
DEACT OK	

7. Two local contexts

7.1. Introduction on two local contexts

The internal TCP/IP stack of Quectel supports to activate two GPRS contexts at the same time. In another word, the internal TCP/IP stack supports to establish TCP/UDP sessions based on two different activated GPRS contexts. For example, establish a TCP/UDP session based on the GPRS context whose APN is "CMNET" and it is OK to establish another TCP/UDP session based on another GPRS context whose APN is "CMWAP". The command **AT+QIFGCNT** is used to select a context as FGCNT. The purpose of the command is to select a GPRS context.

Note: The two contexts activated at the same time should be both GPRS.

7.2. Establish TCP/UDP sessions in two local contexts

The following is an example to activate two GPRS contexts one by one.

Establish a TCP session to visit a remote server based on the context 0.
Firstly, use the command **AT+QIFGCNT** to select context 0 as FGCNT as the following.

```
AT+QIFGCNT=0                    // Set context 0 as FGCNT, then all the operations in the UART are for
                                   the context 0.
OK
```

Secondly, initialize some configuration as the former chapters.

```
AT+QICSGP=1,"CMNET"             //Set the APN as "CMNET".
OK
AT+QIMUX=0                      // Disable MUXIP, which means it visits only one remote server based
                                   on the context 0. It also supports to enable MUXIP here.
OK
AT+QIMODE=0                     // Set the session mode as non-transparent. It doesn't support transparent
                                   mode if trying to activate two GPRS contexts on the same UART.
OK                              // Implement the command successfully.
AT+QIDNSIP=0                    // Use IP address as the address to establish TCP/UDP session
OK                              // Implement the command successfully.
```

Thirdly, activated the context 0 and establish a session based on the context 0.
Here take **AT+QIOPEN** as an example to establish a session with a remote server. Of course, it supports to establish a session by the steps introduced in Chapter 3 and Chapter 4.

```
AT+QIOPEN="TCP", "124.79.167.121",7007      // Visit the remote TCP server
                                               '124.79.167.121:7007'.
OK
```

CONNECT OK	// Connect to the remote TCP server successfully.
AT+QILOCIP	// Query the local IP address.
10.79.141.224	// The local IP for the context 0 is '10.79.141.224'.

Establish a TCP server based on context 1.

Firstly, use the command **AT+QIFGCNT** to select context 1 as FGCNT as the following.

AT+QIFGCNT=1	// Set context 1 as FGCNT, and then all the operations in the UART are for the context 1.
OK	

Secondly, initialize some configuration as the former chapters.

AT+QICSGP=1,"CMNET"	// Set the APN as "CMNET".
OK	
AT+QIMUX=1	// Enable MUXIP, which means it listens to multiple remote clients based on the context 1.
OK	
AT+QIMODE=0	// Set the session mode as non-transparent. It is not supported in transparent mode here.
OK	// Implement the command successfully.

Thirdly, activate the context 1 and enter listening state.

AT+QISERVER=0,3	// **0** means to enter the listening state to listen to TCP clients and if here is 1, it means to listen to UDP clients. **3** is the maximum clients to listen to, i.e. after the number of clients reaches 3, no more other clients can visit the module.
OK	
SERVER OK	// Enter listening state successfully.
AT+QILOCIP	// Query the local IP address.
10.79.140.214	// The local IP for the context 1 is "10.79.140.214".

The TCP client whose address is '10.79.141.26' tries to visit the module. After the module listens to the visit and accepts the visit successfully, it will output the following string through UART.

0, REMOTE IP: 10.79.141.26	// Profile 0 is used to establish the session with the client '10.79.141.26'.

8. DNS function

8.1. Visit a remote server with domain name

Besides to visit a remote server with its IP address, the internal TCP/IP stack also supports to visit a remote server with its domain name. The following is an example to visit "www.quectel.com" directly.

Firstly, initialize some configuration as the former chapters.

```
AT+QIFGCNT=0                // Set context 0 as the FGCNT.
OK
AT+QICSGP=1,"CMNET"         // Set the APN as "CMNET".
OK
AT+QIMUX=0                  // Disable MUXIP.
OK
AT+QIMODE=0                 // Set the session mode as non-transparent.
OK
AT+QIDNSIP=1                // Use domain name as the address to establish TCP/UDP session.
OK
```

The former configuration is just an example. It is unnecessary to be completely same as the former configuration. For example, it is OK to enable and set session mode as transparent mode.

Secondly, visit the remote server "www.quectel.com" by the command **AT+QIOPEN**.

```
AT+QIOPEN="TCP", "www.quectel.com", 80
OK

CONNECT OK
```

Then, it is allowed to send data to or receive data from the server "www.quectel.com:80" in the same steps introduced in Section 5.2. And it is supported to close the session by the command **AT+QICLOSE** or **AT+QIDEACT**.

8.2. Get IP according to the domain name

The internal TCP/IP stack supports to get IP address according to the given domain name. The following is an example to parse the domain name "www.google.com".

Initialize GPRS/CSD context

```
AT+QIFGCNT=0                // Set context 0 as the FGCNT.
OK
AT+QICSGP=1,"CMNET"         // Set the APN as "CMNET".
```

```
OK
```

Activate GPRS/CSD context and parse the domain name.

```
AT+QIREGAPP
OK
AT+QIACT                                    // Activate FGCNT.
OK
AT+QILOCIP
10.79.69.228
AT+QIDNSCFG="211.136.18.171","211.136.112.50"
                                            // '211.136.18.171' is the primary DNS server, and
                                               '211.136.112.50' is the secondary DNS server.
OK
AT+QIDNSGIP="www.google.com"                // Get the IP address of the domain name "www.google.com".
OK

64.233.189.99

64.233.189.104

64.233.189.147
```

Actually, the internal TCP/IP stack negotiates DNS servers automatically after activating GPRS/CSD. Hereby, it is unnecessary to implement the command **AT+QIDNSCFG** to set DNS servers. And the **AT+QIDNSGIP** can activate FGCNT automatically, so it is unnecessary to implement **AT+QIREGAPP** and **AT+QIACT** to activate FGCNT, either. So, the former commands can be simplified as the following.

```
AT+QIDNSGIP="www.google.com"                // Get the IP address of the domain name "www.google.com".
OK

64.233.189.99

64.233.189.104

64.233.189.147
```

9. Slow clock in TCP or UDP connection

The module can enter sleep mode to save power consumption with existing connection. Follow the steps below to enter sleep mode.

First, use **AT+QSCLK=1** to enable the sleep mode.
Second, keep the DTR PIN in HIGH status.
Last, ensure there is no H/W interruption, no S/W interruption or no network events.

After the module enters sleep mode, there are many ways to wake up the module.

1. Pull DTR PIN to LOW status. The serial port will be active after DTR is pulled to low status for 20ms.
2. RTC alarm expires.
3. H/W or S/W interruption. Such as a new message arrived, etc.

Shanghai Quectel Wireless Solutions Co., Ltd.

Room 501, Building 13, No.99, Tianzhou Road, Shanghai, China 200233

Tel: +86 21 5108 6236

Mail: info@quectel.com