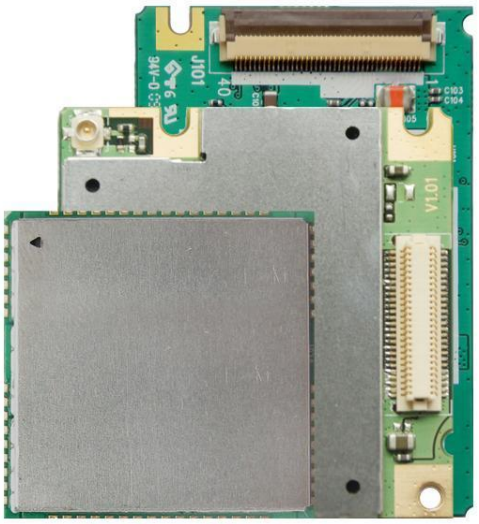
Quectel Cellular Engine

GSM TCPIP Recommended Process

GSM_TCPIP_Recommended_

Process_V1.0



Document Title	GSM TCPIP Recommended Process
Version	1.0
Date	2011-3-14
Status	Release
Document Control ID	GSM_TCPIP_Recommended_Process_V1.0

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 to Quectel by the customers. Quectel has not undertaken any independent search for additional relevant information, including 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.

Copyright

Contents

0. Revision history

Revision	Date	Author	Description of change
1.0	2011-3-14	Colin HU	Initial

1. Introduction

This document introduces the simple process on how to use the embedded TCPIP services and gives some recommendations for handling the abnormalities.

Please refer to [1] Mxx_ATC for AT commands description in Table 1. And refer to [2] GSM_TCPIP_AN for the details about TCPIP.

1.1. Reference

Table 1: Reference

SN	Document name	Remark
[1]	Mxx_ATC_Vx.x.pdf	GSM AT Commands set for Mxx
[2]	GSM_TCPIP_AN_V1.01.pdf	GSM TCPIP application notes

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
UDP	User Datagram Protocol
UART	Universal Asynchronous Receiver/Transmitter

2. Initialization

AT+CPIN?	//Make sure the SIM PIN is unlocked.
+CPIN:READY	//It shows SIM PIN is unlocked and ready
OK	
AT+CREG?	//Make sure the network is registered successfully
+CREG: 0,1	//It shows it's successful to register the network. If it's failed, execute command **AT+CREG?** continually. Here we can also set **AT+CREG=1** to enable the network registration unsolicited result code at the beginning. And then just wait the automatic report **+CREG: 1** or **+CREG: 5.**
OK	
AT+QIFGCNT=0	//Set the Context 0 as the foreground context. Then all the following operations are aimed at Context 0.
OK	
AT+QICSGP=1,"CMNET"	//Set APN for the current context. Here set GPRS as the bearer for TCPIP connection, and set APN as "CMNET". It's the default setting. If the operator of SIM card is China Mobile or China Unicom, and use GPRS as the bearer, this command can be omitted.
OK	

/**********

Other options:

A. **Set server address format.** By default, the module will adopt a dotted decimal IP address. If the server address is a domain name, user should execute command "**AT+QIDNSIP=1**" to change it to the domain name server format.

B. **Set the mode to handle the received data**. Currently the module supports two modes to handle the received TCP/IP data: Output all the received data through UART directly (default); or Output a notification statement "**+QIRD:**", not output them directly when receiving data, then user need retrieve the data by command **AT+QIRD**. If the latter is used, please execute the following command.

AT+QINDI=1	//Set mode: when receiving the data, output the notification:" **+QIRDI: <id>,<sc>,<sid>**". Then retrieve data by command "

AT+QIRD=<id>,<sc>,<sid>,<len>"

OK

C. **Set display format for received data**. When users select to output the received data through UART directly, following commands can be used to set the display format for the received data. User can select according to actual needs.

AT+QIHEAD=1 //Add head information "**IPD<len>:**" before the received data

OK

AT+QISHOWRA=1 // Display the IP address and port of the sender:
RECV FROM:<IP ADDRESS>:<PORT>

OK

AT+QISHOWPT=1 //Show the transmission layer protocol type, TCP or UDP. This application is rare.

OK

D. **Set the session mode**. The default mode is non-transparent mode, and the following section from Chapter 3 to Chapter 7 is the introduction of creating connection, sending and receiving data in this mode. Execute the following commands to set transparent mode and refer to Chapter 8 for the specific applied example in this mode.

AT+QIMODE=1 //Set transparent mode

OK

AT+QITCFG=3,2,512,1 //Configure transparent transfer mode. Please focus on the two middle parameters. The second parameter **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. The third parameter **512** means that once the length of the data in the buffer reaches 512, the 512 bytes data will be sent out.

OK

**********/

3. Establish TCP session

AT+QIOPEN="TCP","116.226.39.202","7007"	//Visit the remote TCP server: 116.226.39.202:7007
OK	//Format is right and current status is permitted to establish TCP session.

/***

Analysis and treatment of other error response:

A. "**ERROR**", two reasons are possible:

1. The command format is wrong. If all format are right here, verify if QIMUX is set as 0. Send command "**AT+QIMUX?**", if the response is 1, reset it as 0 by command **AT+QIMUX=0**.
2. Current TCPIP connection status is not **IP INITIAL, IP STATUS** and **IP CLOSE** (Query by command **AT+QISTAT).** If current status is **TCP CONNECTING**, execute command **AT+QICLOSE** to close current failed TCP connection. If current status is others, execute command **AT+QIDEACT** to deactivate current failed GPRS context.

B. "**ALREADY CONNECT**", this means one TCP or UDP connection has been established. If new connection is required, send command **AT+QICLOSE** to close current connection.

***/

CONNECT OK	// Successfully connect to the remote TCP server

/***

Analysis and treatment of other error response:

A. "**CONNECT FAIL**", this shows the TCP connection is failed to establish. The correct processing method is to send command **AT+QISTAT** to query the current status at first.

If current status is "**TCP CONNECTING**", it's recommended to close the failed connection by command **AT+QICLOSE**. In this way GPRS is still active and there is no need to restart the GPRS context, so the program speed is imporved.

If current status is not "**TCP CONNECTING**", it's recommended to execute command **AT+QIDEACT** to deactivate GPRS context because these status are usually caused by failure to activate GPRS context. More details about the treatment of response for **AT+QIDEACT**, please refer to the description in Chapter 7.

The longest waiting time for this command, theoretically, is about 2.5 to 3.5 minutes. Customer can set timeout value which is less than 2 minutes in their own application according to their need. If it is timeout, it's recommended to restart the module by **EMERG_OFF** pin. In the same way, the GPRS context is still not activated after repeating several times **AT+QIDEACT** and **AT+QIOPEN**, please restart the module by **EMERG_OFF** pin.

***/

/* Similarly, the module supports UDP mode. Just asTCP, it's needed to establish UDP session before using UDP as below.*/

AT+QIOPEN="UDP","116.226.39.202","7007"	//Visit the remote UDP server: 116.226.39.202:7007.
OK	
CONNECT OK	**//S**uccessfully connect to the remote UDP server. In fact, UDP needs not establish a connection. Here just configure the target IP address and port which the data will be sent to.

4. Send data

4.1. Send data in unfixed length

AT+QISEND	//Ready to send data
> TEST<Ctrl+Z>	//Send data "TEST", **<Ctrl+Z>** is used to end the input and begin to send all input data. Please note that the maximum length of the data to input at one time is 1460. And some special characters can not be sent in this mode: 0x08 (back space), when the module receives this character, it will delete the last input character. 0x1A (**<Ctrl+Z>**), when the module receives this character, it will stop receiving the following input and begin to send all previously received data. 0x1B (**ESC**), when the module receives this character, it will escape from the sending operation.
SEND OK	//The data has been sent to TCP protocol layer successfully.
AT+QISACK	// Check if data has been sent successfully by querying the data information.
+QISACK: 4, 4, 0	//The former two parameters are total size and acknowledged data size. The last parameter is unacknowledged data size. If this size is 0, it means all data has been sent successfully. But this command is moot for UDP mode. Because UDP is connectionless, it will not confirm whether data has been sent successfully, thus the last parameter is always 0.
OK	

/**********

Note:

A. The send buffer in the underlying socket is 7300. Therefore, don't need to wait all data has been acknowledged (that is the last parameter in the response of **AT+QISACK** is 0) and then send next package. Given that the maximum length of one TCP package is 1448, it's recommended to set a threshold value as 3000. If the unacknowledged data length (the last parameter in the response of **AT+QISACK**) is less than 3000, continue to send the next package by command **AT+QISEND.** Else if it exceeds 3000, stop sending data, and then query by **AT+QISACK** every 5 seconds until the last parameter is less than 3000. If the query times achieves a certain number (for example 20 times, equivalent to 100 seconds

timeout) and the last parameter is always greater than 3000, it can be considered an anomaly occurred in the TCP connection. In this case close this connection, re-establish TCP connection, and then continue to send all previously unacknowledged data and next data package.

B. If users want to send hexadecimal characters in non-transparent mode, especially 0x08, 0x1A, 0x1B, it's recommended to send data with fixed length, which described as below.

**********/

4.2. Send data in fixed length

AT+QISEND=3	//Specify the data length to be sent is 3.
>0x080x1A0x1B	//Send special characters 0x08, 0x1A, 0x1B in turn. When actually input the data, be sure to input these special data 0x08, 0x1A, 0x1B in hexadecimal format, other than input a string of "0x080x1A0x1B". The example just gives a demonstration to let user see clearly.
SEND OK	
AT+QISACK	
+QISACK: 17, 17, 0	
OK	

5. Receive data

5.1. Output the data through UART directly

The received data from the server will be output through UART without any head information in the defaulting setting. In order to distinguish the received data from the responses of AT commands or any URC, it is allowed to add some information at the beginning of the received data controlled by some commands. The details have been described in Chapter 2.

It's better to set display format for the received data before the TCP connection has been established. Below is a sample of the received data.

RECV FROM:116.228.146.250:7070<CR><LF>	//The module receives data from the remote server 116.228.146.250:7070. This line will not be displayed if not set **AT+QISHOWRA=1** in the initialization phase.
IPD36TCP:1234567890abcdefghijklmnopqrstuvwxyz	//Receive TCP data with the length of 36. These data is 36 characters after the colon symbol ":". The header "**IPD36TCP:**" will not be displayed if not set **AT+QIHEAD=1.** The header "**TCP**" will not be displayed if not set **AT+QISHOWPT=1.**

Note:

In this mode, the data will be output immediately once received, so it's unavoidable that the data appears in the middle of AT commands. In the current design we have done some timeout processing to avoid the received data from breaking AT command or its response (for example, separate AT command directly). But it's still unavoidable that the AT command is separated with its response.

5.2. Retrieve the received data by 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 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**	

6. TCP connection maintenance and detection

Most of the modules are connected to the Internet through GPRS gateway. GPRS gateway here has the similar functions as the router in LAN. Every time when the module tries to connect with one server in the internet, GPRS getaway should assign a port to the module. A problem will be caused when the connection between the module and the Internet is established via this port. Since the port resource of GPRS gateway is limited, so it will have some restrictions on these ports for the terminals within the GPRS network. If there is no data transmission on the TCP connection for a period of time, it will release the port to other connection to achieve a rational allocation of port resource. So the TCP connection between the module and the server may be disconnected by GPRS network without any notification.

There is no clear value at present about how long the resource will be released without data transmission. The test result in Shanghai is the port resource will not be released within 10 minutes.

In order to avoid the TCP connection from disconnection by GPRS gateway without any notification, it is recommended to periodically send a small data packet to the remote end through which the TCP connection can be maintained and detected.

Example:
Set an interval of 10 minutes to send the heart beating package according to our test result.

AT+QISACK	// Check the information of the sending data
+QISACK: 1448, 1448, 0	//Suppose 1448 bytes data has been sent to the server successfully.
OK	
AT+QISEND	//Send the normal data
> **1234567890abcdefghijklmnopqrstuvwxyz<0x1A>**	//Input the data to be sent, <0x1A> is used to end the input and require to send data
SEND OK	//The data has been sent to TCP protocol layer successfully

/**********

Assuming no data is needed to be sent for a long time, it is recommended to send a heart beating small packet to maintain the TCP connection. Before sending the heart beating packet to maintain the TCP connection, confirm if the current TCP connection is normal by querying whether the current package has been sent successfully. If the package is still not acknowledges after two minutes (query every 5 seconds, 24 times in total), the TCP connection may be abnormal, execute **AT+QICLOSE** to close the current connection, and then execute **AT+QIOPEN** to re-establish the TCP connection. After the connection has been established successfully, resent the unacknowledged data.

**********/

AT+QISACK	//Check the information of the sending data
+QISACK: 1484,1448,36	//The total length of the data which has been sent is 1484, but only 1448 bytes has been acknowledged, and 36 bytes has not been acknowledged (can not confirm whether it has been sent successfully) yet. Check again after 5 seconds.
OK	
.....	//Wait for 5 seconds
AT+QISACK	//Check the information of the sending data
+QISACK: 1484,1484,0	//All 1484 bytes data has been acknowledged to sent successfully. This means the TCP connection is normal until now.
OK	
……	//Sleep 8 minutes, no data is sent or received
AT+QISEND	//Send a heart beating small packet to server
> Heart01<0x1A>	//Send small packet **Heart 01**. It can be simpler. Any other data is OK, and users can define specific format to avoid server misunderstanding
SEND OK	
…….	//Wait for 5 seconds, and then check if the heart beating data has been sent successfully
AT+QISACK	// Check the information of the sending data
+QISACK: 1491,1491,0	//The packet has been received successfully, the TCP connection is still OK, so stop the check operation. If the packet has not been acknowledged (the third parameter of the response of "**AT+QISACK**" is not 0), please wait for 5 seconds and check again. Repeat the former operation for several times until the packet has been acknowledged or the times of repeat reaches to a limitation (for example, 24 times, i.e. 2 minutes). If the times of repeat reaches to the limitation, the TCP connection may be abnormal, so it is suggested to re-establish the TCP connection.
OK	

7. Close TCP session

AT+QICLOSE	//Close the current TCP connection after all data has been sent.
CLOSE OK	
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 responsing 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 application. 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.

8. Transparent session mode

If **AT+QIMODE=1** is executed in the initialization phase, the UART will enter data mode after the TCP/UDP session is established. In data mode, all the input data through UART is considered as the data needed to send to the remote server and all the output data through UART is received from the remote server except some special strings such as "**CLOSED**"，"**+PDP DEACT**" . An example of TCP transparent session is as below.

8.1. Example

AT+QIOPEN="TCP","116.226.39.202","7007"	///Visit the remote TCP server: 116.226.39.202:7007
OK	//Format is right and current status is permitted to establish TCP session. Other error response here can be referred to the Chapter 3.
CONNECT	// Successfully connect to the remote TCP server and the UART has entered the data mode. Other error response here can be referred to the Chapter 3.
……..	//User can send data now. For example, when "1234567890abcdefghijklmnopqrstuvwxyz"is inputted, these data will not be echoed by UART. Because the data length is less than 512, the module will wait 200ms to send the data (see configuration in Chapter 2).
abcdefghijklmnopqrstuvwxyz1234567890	//These data are received from the server.
…….	//Input "+++" to quit the data mode
OK	//The response for "+++". "OK" means the UART has switched to the command mode successfully. Then if user wants to close the TCP connection, please refer to the example in Chapter 7.

8.2. Treatment for abnormality

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 responsing 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 application. If it is timeout **DEACT OK** is still not received, user can restart the module by **EMERG_OFF** pin.
DEACT OK	

Shanghai Quectel Wireless Solutions Co., Ltd.

Room 501, Building 9, No.99, TianZhou Road, Shanghai, China 200233

Tel: +86 21 5108 2965

Mail: info@quectel.com