PK_Module_AT_User_Guide_V01.1.1

This document provides information for controlling PK module through external UART.

COPYRIGHT

Realsil reserves the right to make corrections, enhancements, improvements and other changes to its products and services. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete.

Buyers and others who are developing systems that incorporate Realsil products (collectively, "Customers") understand and agree that Customers remain responsible for using their independent analysis, evaluation and judgment in designing their applications and that customers have full and exclusive responsibility to assure the safety of Customers' applications and compliance of their applications (and of all Realsil products used in or for Customers' applications) with all applicable regulations, laws and other applicable requirements. Designer represents that, with respect to their applications, Customer has all the necessary expertise to create and implement safeguards that, (1) anticipate dangerous consequences of failures, (2) monitor failures and their consequences, and (3) lessen the likelihood of failures that might cause harm and take appropriate actions. Customer agrees that prior to using or distributing any applications that include Realsil products, Customer will thoroughly test such applications and the functionality of such Realsil products as used in such applications.

Realsil's provision of technical, application or other design advice, quality characterization, reliability data or other services or information, including, but not limited to, reference designs and materials relating to evaluation kits, (collectively, "Resources") are intended to assist designers who are developing applications that incorporate Realsil products; by downloading, accessing or using Realsil's Resources in any way, Customer (individually or, if Customer is acting on behalf of a company, Customer's company) agrees to use any particular Realsil Resources solely for this purpose and subject to the terms of this Notice.

Realsil's provision of Realsil Resources does not expand or otherwise alter Realsil's applicable published warranties or warranty disclaimers for Realsil's products, and no additional obligations or liabilities arise from Realsil providing such Realsil Resources. Realsil reserves the right to make corrections, enhancements, improvements and other changes to its Realsil Resources. Realsil has not conducted any testing other than that specifically described in the published documentation for a particular Realsil Resource.

Customer is authorized to use, copy and modify any individual Realsil Resource only in connection with the development of applications that include the Realsil product(s) identified in such Realsil Resource. No other license, express or implied, by estoppel or otherwise to any other Realsil intellectual property right, and no license to any technology or intellectual property right of Realsil or any third party is granted herein, including but not limited to any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which Realsil products or services are used. Information regarding or referencing third-party products or services does not constitute a license to use such products or services, or a warranty or endorsement thereof. Use of Realsil Resources may require a license from a third party under the patents or other intellectual property of the third party, or a license from Realsil under the patents or other Realsil's intellectual property.

Realsil's Resources are provided "as is" and with all faults. Realsil disclaims all other warranties or representations, express or implied, regarding resources or use thereof, including but not limited to accuracy or completeness, title, any epidemic failure warranty and any implied warranties of merchantability, fitness for a particular purpose, and non-infringement of any third party intellectual property rights.

Realsil shall not be liable for and shall not defend or indemnify Customer against any claim, including but not limited to any infringement claim that related to or is based on any combination of products even if described in Realsil Resources or otherwise. In no event shall Realsil be liable for any actual, direct, special, collateral, indirect, punitive, incidental, consequential or exemplary damages in connection with or arising out of Realsil's Resources or use thereof, and regardless of whether Realsil has been advised of the possibility of such damages. Realsil is not responsible for any failure to meet such industry standard requirements.

Contents

1 Introduction

1.1 Abstract

This article describes the AT commands supported in PK module, and the format of each AT command.

(1) Users can connect the device to networks, cloud services, and implement IoT services, with these AT commands.

(2) Users can use the device to do some TCP/IP sevice as a server or client.

(3) Users can control the device as a Bluetooth central or Bluetooth peripheral, with these AT commands.

(4) Users can use the common commands to check version, reset device, set GPIO, restore parameters back to outing-factory, etc.

Users can deal the connections as Figure 1.1, in order to test the AT commands.

The module can be powered by PC with USB cable.

The AT commands are inputted and outputted with UART_RX, UART_TX by USB2TTL converter. The baud rate of this cable is set to 38400 as default, the data bits = 8, parity = none, encoding = ASCII.

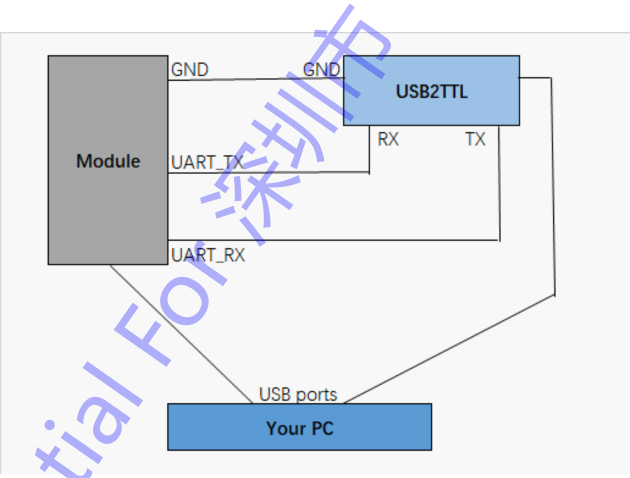

Figure 1.1 Abstract of connection

In order to deal these connections, the user need to make some preparations, as table 1.1.

Hardware	Illustration
Module	Device.
PC	To input AT commands, and check response.
USB cable (x 2)	To connect device to PC.
Some dupont lines (at least 3)	To connect the device to USB2TTL converter.

USB2TTL converter	To connect the UART port to USB port of PC.

Table 1.1 Required hardware

1.2 Command description

1. Comma (,), quotation marks (“ ”), square brackets ([ ]) and backslash (\) are used as delimiter in this version AT command, so if they are needed, use escape character “\”. For example, if need to input “[”, it should be “\[” instead.
2. Every normal command should end with “\n”, except data command.
3. Each AT command is started with “AT”, and all the pending letters of command (excluding parameters) are all in uppercase.

1.3 AT command list

Description	AT Command
Common command	
Test AT command ready	AT+TEST
Print all AT command	AT+LIST
Restart module	AT+RST
Query version info	AT+GMR
Set AT commands echo mode	AT+ECHOLEVEL
Factory Reset	AT+RESTORE
UART configuration	AT+UARTCFG
OTA upgrade	AT+OTA
Choose activated image	AT+OTASET
GPIO control	AT+GPIO
Wifi command	

Connect to AP (STA mode)	AT+WLCONN
Disconnect from AP	AT+WLDISCONN
Scan AP	AT+WLSCAN
Query the RSSI value	AT+WLRSSI
Set AP mode	AT+WLSOFTAP
Wifi information	AT+WLSTATE
Set static IP for STA	AT+WLSTATICIP
Set Auto connect	AT+WLAUTOCONN
Set MAC address	AT+WLMAC
TCP/IP command	
Get errno	AT+SKTGETERR
TCP/UDP/SSL Server	AT+SKTSERVER
TCP/UDP/SSL Client	AT+SKTCLIENT
Close TCP/UDP/SSL connection	AT+SKTDEL
Send packet	AT+SKTSEND
Receive packet	AT+SKTREAD
Enable auto receive data mode	AT+SKTRECVCFG
Check network connection status	AT+SKTSTATE
Ping	AT+PING
Set transparent transmission mode	AT+SKTTT
Save translink and enable autolink	AT+SKTAUTOLINK
Send http/https client request	AT+HTTPCLIENT
Read or set CA cert/pk key	AT+SSLCRET
MQTT Command	

Open a new mqtt connection	AT+MQTTOPEN
Close a mqtt connection	AT+MQTTCLOSE
Connect to mqtt server	AT+MQTTCONN
Disconnect from mqtt server	AT+MQTTDISCONN
Subscribe a mqtt topic	AT+MQTTSUB
Unsubscribe a mqtt topic	AT+MQTTUNSUB
Publish mqtt messages	AT+MQTTPUB
Configure or inquire mqtt parameters	AT+MQTTCFG
Reset all mqtt connections	AT+MQTTRESET
Bluetooth Command	
Set the BT peripheral mode	AT+BLEPMODE
Set the BT central mode	AT+BLECMODE
Set or get the BT MAC address	AT+BLEMAC
Set or get the BT GAT mtu size	AT+BLEMTU
Configure authentication information	AT+BLEPAIR
Setup or inquire the pairing code	AT+BLEPASSKEY
Send user confirmation	AT+BLEUSERCONF
Update connection parameters	AT+BLECONNPARAM
Clear or inquire the pairing information	AT+BLECLRINQ
Set or inquire the adv name	AT+BLENAME
Set or inquire the adv status	AT+BLEADV
Set or inquire the adv interval	AT+BLEADVINTV
Send indication/notification from peripheral BT	AT+BLEINDNTF
Create connection	AT+BLECONN

Close connection	AT+BLEDISCONN
Get all connection information	AT+BLECONNINFO
Scan BLE adv	AT+BLESCAN
Read characteristic value	AT+BLEREAD
Write characteristic value	AT+BLEWRITE
Modify whitelist	AT+BLEWHITELIST
Modify scan interval/window	AT+BLESCANPARAM
BLE auto reconnect	AT+BLEAUTOCONN
Start or stop ibeacon	AT+BLEIBEACON
Set or get ibeacon adv data	AT+BLEIBCNDATA
Set or get ibeacon uuid	AT+BLEIBCNUUID

2 Common Command

2.1 AT+TEST – Test AT command ready

AT+TEST	
Description	This command is used to test system boot successfully, and user can execute AT commands.
Response	+TEST:OK

2.2 AT+LIST – Print all AT command

AT+LIST	
Description	This command is used to output all supported AT commands right now.

Response	Common AT Command: AT+TEST *// … … followed by other common command list.* Wi-Fi AT Command: AT+WLSOFTAP *// … … followed by other wifi command list.* TCP/IP AT Command: AT+SKTGETERR *// … … followed by other tcp/ip command list.* BT AT command: AT+BLECMODE *// … … followed by other BT command list.* MQTT AT command: AT+MQTTOPEN *// … … followed by other mqtt command list.* +LIST:OK

2.3 AT+RST – Restart module

AT+RST	
Description	This command is used to restart the module
Response	+RST:OK *// Then the system should restart right now.*

2.4 AT+GMR – Query version info

AT+GMR	
Description	This command is used to query module AT version as well as SDK version
Response	+GMR:<at-version>,<sdk-version>(<compile_time>)

2.5 AT+ECHOLEVEL – Set UART echo mode and debug mode

AT+ECHOLEVEL=<echo>[,< debug mask >,< debug level >]		
Description	This command is used to enable/disable UART echo and set debug mask and level	
Response	+ECHOLEVEL:OK +ECHOLEVEL:ERROR:<error_no>	
Parameter	<echo>	0 : disable echo 1 : enable echo (default)
	[<debug mask>]	BIT 6: OS (default on) BIT 7: LWIP (default on) BIT 8: COMMON (default on)
	[<debug level>]	0: OFF 1: ALWAYS 2: ERROR (default) 3: WARNING 4: INFO
Error Number	1: There should be parameters. 2: parameter number error 3: echo should be '0' or '1' only	
Example	*// Disable echo and debug message* AT+ECHOLEVEL=0,0x0,0x0	

2.6 AT+RESTORE – Factory Reset

AT+RESTORE	
Description	This command is used to clean flash data, module will restore to factory setting
Response	+RESTORE:OK +RESTORE:ERROR:<error_no>
Error Number	1: restore default data fail 2: restore default image fail

NOTE	AT+RESTORE +RESTORE:OK *// System will reboot*

2.7 AT+UARTCFG – UART configuration

AT+UARTCFG =<baudrate>,<databits>,<stopbits>,<parity>,<flowcontrol>,<configmode>		
Description	This command is used to setup uart mode	
Response	+UARTCFG:OK +UARTCFG:ERROR:<error_code>	
Parameter	<baudrate>	2400, 4800, 9600, 19200, 38400(default), 57600, 115200, 921600, 1152000
	<databits>	5: 5 bit data 6: 6 bit data 7: 7 bit data 8: 8 bit data (default)
	<stopbits>	1: 1 bit stop (default) 2: 2 bit stop
	<parity>	0: None parity (default) 1: Odd parity 2: Even parity
	<flowcontrol>	0: disable flowcontrol (default) 1: enable RTS and CTS
	<configmode>	0: set the current configuration and will not save to flash 1: save configuration to flash and take effect immediately 2: save configuration to flash and take effect after reboot
Error number	1: command format error 2: command parameter error	

2.8 AT+OTA – OTA upgrade

AT+OTA=<ip>,<port>		
Description	This command is used to upgrade firmware	
Response	+OTA:OK +OTA:ERROR:<error_code>	
Parameter	<ip>	Download server ip address
	<port>	Download server port number
Error number	1: command format error 2: command parameter error 3: can not connect to this ip:port.	
NOTE	1: download server should run first. 2: module should connect to the same network as download server.	

2.9 AT+OTASET – Choose Activated Image

AT+OTASET=<image ID>		
Description	This command is used to choose the activated image	
Response	+OTASET:OK +OTASET:ERROR:<error_code>	
Parameter	<image ID>	0: default image 1: OTA upgrade image
Error number	1: command format error 2: command parameter error	
NOTE	System will reboot	

2.10AT+GPIO – GPIO control

AT+GPIO=<R/W>,<PORT>[,<DATA>,<DIR>,<PULL>]	
Description	This command is used to control gpio pin

Response	+GPIO:OK:<val> *//val is the value read from gpio or write to gpio* +GPIO:ERROR:<error_code>	
Parameter	<R/W>	“R”: read gpio “W”: write gpio
	<PORT>	Px_x, ex: PC_4
	[<DATA>]	0 or 1 when write gpio
	[<DIR>]	Pin direction: 0: PIN_INPUT 1: PIN_OUTPUT
	[<PULL>]	Pin mode: 0: PullNone/PullDefault 1: PullUp 2: PullDown 3: OpenDrain
Error number	1: command format error 2: command parameter error 3: invalid pin name	

3 Wifi command

3.1 AT+WLCONN – Connect to AP

AT+WLCONN=<ssid>,<pwd>[,<key_id>,<bssid>]		
Description	This command is used to connect to AP for station	
Response	+WLCONN:OK +WLCONN:ERROR:<error_code>	
Parameter	<ssid>	This parameter can't be empty Format: "ssid" Must add prefix '\' for special character(',' , '\' , '"', ‘[’, ‘]’)
	<pwd>	1. WPA/WPA2 : length is 8~64 2. WEP : length is 5 or 13
	[<key_id>]	For WEP security, must be 0~3. If not set, it will use id 0 as default
	[<bssid>]	Format : 6 bytes hex number e.g. 112233445566
	[<async>]	0 : synchronized network connection (default) 1 : non-synchronized network connection
Error number	1: command format error 2: command parameter error 3: wifi initial error 4: connect to AP failed 5: wifi mode error 6: get ap security type failed 7: dhcp timeout, use static ip 192.168.1.80	
NOTE	*// If no password, set the parameter <pwd> NULL* AT+WLCONN=“SSID” AT+WLCONN=“SSID”,,,112233445566 *// If need non-synchronized network connection* AT+WLCONN=”SSID”,”PWD”,,,1	

3.2 AT+WLDISCONN - Disconnect from AP

AT+WLDISCONN	
Description	This command is used to disconnect with AP for station
Response	+WLDISCONN:OK +WLDISCONN:ERROR:<error_code>
Error number	1,2: reserved 3: operation failed 4: disconnect timeout

3.3 AT+WLSCAN - Scan AP

AT+WLSCAN	
Description	This command is used to scan AP in the air
Response	AP : <num>,<ssid>,<chl>,<sec>,<rssi>,<bssid> +WLSCAN:OK +WLSCAN:ERROR:<error_no>
Error number	1: Input wrong parameters. 2, 3: Memory failure. 4: Failed when setting scan channel. 5: Failed when calling scan app.
NOTE	The information of AP in order are number, SSID, channel, security mode, strength of signal, BSSID

3.4 AT+WLRSSI – Query the RSSI value

AT+WLRSSI	
Description	This command is used to read the RSSI value of connected wifi.
Response	RSSI = <read_val> +WLRSSI:OK

Error number	NULL
Example	*// Connect to an AP.* AT+WLCONN=test,12345678 *+WLCONN:OK* *// Read the RSSI* AT+WLRSSI *RSSI = -66* *+WLRSSI:OK* *// Disconnect.* AT+WLDISCONN *+WLDISCONN:OK* *// Read the RSSI again. It should be 0 now.* AT+WLRSSI *RSSI = 0* *+WLRSSI:OK*

3.5 AT+WLSOFTAP - Set AP mode

AT+WLSOFTAP=<ssid>,<pwd>,<chl>,<hidden>[,<max_conn>]		
Description	This command is used to config AP mode	
Response	+WLSOFTAP:OK +WLSOFTAP:ERROR:<error_no>	
Parameter	<ssid>	This parameter can't be empty Format: "ssid" Must add prefix '\' for special character(',' , '\' , '"', '[', ']')
	<pwd>	WPA/WPA2 : length is 8~64
	<chl>	Channel : 1~11
	<hidden>	0 : Not hidden SSID 1 : hidden SSID
	[<max_conn>]	Max number of STAs, should be [1,3], default is 3

Error number	1: command format error 2: command parameter error 3: wifi initial error 4: start AP failed 5: wifi mode error
NOTE	*// If no password, remain the parameter NULL.* AT+WLSOFTAP=”SSID”,,11,0

3.6 AT+WLSTATE - Wifi information

AT+WLSTATE	
Description	This command is used to list wifi information
Response	<mode>,<SSID>,<chl>,<sec>[,<key_id>],<pwd>,<mac>,<ip>,<gw> CLIENT : <num>,<mac> +WLSTATE:OK
NOTE	The information in order are wifi mode, SSID, channel, security mode, (key id for WEP), password, device mac, device IP, gateway. In AP mode, show extra client information, number and the BSSID of client

3.7 AT+WLSTATICIP - Set static IP for STA

AT+WLSTATICIP=<ip>[,<gateway>,<mask>]		
Description	This command is used to set static IP for station	
Response	+WLSTATICIP:OK +WLSTATICIP:ERROR:<error_no>	
Parameter	<ip>	Static station IP, e.g. 192.168.1.2
	[<gateway>]	[optional] set gateway IP
	[<mask>]	[optional] set mask IP

Error number	1: command format error 2: command parameter error
Example	*// Set static IP for station to 192.168.1.150* AT+WLSTATICIP:=192.168.1.150 *// Connect to iot_newifi* AT+WLCONN=iot_newifi,abcdef1234 *// query wifi information* AT+WLSTATE *STA,iot_newifi,11,AES,abcdef1234,ec:f0:0e:4e:75:0b,192.168.99.150,192.168.99.1* *+WLSTATE:OK*
NOTE	Default static IP of station is 192.168.1.80

3.8 AT+WLAUTOCONN - Set Auto connect

AT+WLAUTOCONN=<enable>		
Description	This command is used to set the auto connection when device booting. Default disable.	
Response	+WLAUTOCONN:OK +WLAUTOCONN:ERROR:<error_no>	
Parameter	<enable>	0 : disable auto connect 1 : enable auto connect
Error number	1: command format error 2: command parameter error	
Example	*// connect to "iot_newifi", device will store this information into flash* AT+WLCONN=iot_newifi,abcdef1234 *// enable auto connect, this will be store in flash* AT+WLAUTOCONN=1 *>> reboot device* *>> device will read connection information from flash and auto connect to "iot_newifi"*	

3.9 AT+WLMAC - Set MAC address

AT+WLMAC=<mac>

Description	This command is used to set the mac address of device	
Response	+WLMAC:OK +WLMAC:ERROR:<error_no>	
Parameter	<mac>	Format : 6 bytes hex number e.g. 00e04cb72300
Error number	1: command format error 2: command parameter error	
NOTE	Must restart system for effecting new MAC. Do not modify this MAC value frequently unless necessary.	

4 TCP/IP command

4.1 AT+SKTGETERR – Get LWIP errno

AT+SKTGETERR	
Description	This command is used to get errno in LwIP
Response	+SKTGETERR:OK:<errno> *// errno isn't enabled in FW* *+SKTGETERR:ERROR*

4.2 AT+SKTSERVER – Create TCP/UDP/SSL Server

AT+SKTSERVER =<mode>,<Local Port>		
Description	This command is used to create TCP/UDP/SSL Server.	
Response	+SKTSERVER:OK *// (x=[1,9], con_id 0 is reserved)* +SKTSERVER:con_id=x *// Under TCP mode, if a client connects, there will be response as below:* +SKTSERVER:A client connected to server[<server_id>] con_id:<x>,seed,tcp,address:xxx.xxx.xxx.xxx,port:<x>,socket:<x> (response format refer to section 4.8 ATPI) +SKTSERVER:ERROR:<error_no>	
Parameter	<Mode>	0 : TCP mode 1 : UDP mode 2 : SSL mode
	<Local Port>	1~65535
Error Number	1: parameter number error 2: local port should be 1~65535 3: create con_id error 4: create server task error 5: create socket error 6: set socket option error 7: bind error 8: listen error	

	9: tcp server already exists error 10: accept error 11: create con_id for seed error 12: udp server already exists error 13: server can't start under TT(transparent transmission) mode 14: connection type is unknown (SSL isn't supported) 15: listening socket on bind_ip:port failed for ssl server 16: malloc failed for server certificate 17: malloc failed for server key 18: x509_crt_parse failed for server certificate 19: x509_crt_parse failed for server ca list 20: pk_parse_key failed for server key 21: hang node failed for ssl server 22: accept error for ssl server 23: malloc failed for ssl seed 24: initialization failed for ssl context 25: ssl_set_own_cert error 26: ssl handshake failed for ssl seed 27: create node failed for ssl seed
Example	*// create a TCP server on PORT 5001* AT+SKTSERVER=0,5001 *+SKTSERVER:OK* *+SKTSERVER:con_id=1* *// when a client connects to TCP server[con_id=1]* *+SKTSERVER:A client connected to server[1]* *con_id:2,seed,tcp,address:192.168.99.185,port:64068,socket:1* *// create a UDP server on PORT 5002* AT+SKTSERVER=1,5002 *+SKTSERVER:OK* *+SKTSERVER:con_id=3* *// query connection information* AT+SKTSTATE *con_id:1,server,tcp,address:192.168.99.143,port:5001,socket:0* *con_id:2,seed,tcp,address:192.168.99.185,port:64089,socket:1* *con_id:3,server,udp,address:192.168.99.143,port:5002,socket:2* *+SKTSTATE:OK*
NOTE	This command will assign a con_id to this TCP/UDP/SSL Server.

4.3 AT+SKTCLIENT – Create TCP/UDP/SSL Client

AT+SKTCLIENT =<mode>,< Remote Addr>,< Remote Port>[,<Local Port>,<Auth Mode>,<SNI>]

Description	This command is used to create TCP/UDP/SSL Client.	
Response	+SKTCLIENT:OK *// (x=[1,9], con_id 0 is reserved)* +SKTCLIENT:con_id=x +SKTCLIENT:ERROR:<error_no>	
Parameter	<Mode>	0 : TCP mode 1 : UDP mode 2 : SSL mode
	<Remote Addr>	xxx.xxx.xxx.xxx *Or* “www.xxx.com”
	< Remote Port>	1~65535
	[<Local Port>]	Local port to bind, only valid for UDP
	[<Auth Mode>]	Option for SSL connection. Default: 0
	[<SNI>]	Option for SNI feature.
Error Number	1: parameter number error 2: remote IP format or host unfound error 3: remote port should be 1~65535 error 4: create con_id error (none available) 5: create client task error 6: inet_ntoa_r remote address error 7: create socket error 8: hang node error for tcp client 9: connect error for tcp client 10: hang node error for udp client 11: local port should be 1~65535 12: bind local port error 13: connection already exists for TT(transparent transmission) mode 14: set broadcast on socket failed 15: set multicast add membership on socket failed 16: set multicast interface failed 17: connection type is unknown (SSL isn’t supported) 18: Initiate a TCP connection with host:port failed for ssl client 19: memory allocation failed for ssl context structure 20: ssl context initialization failed 21: ssl handshake failed 22: hang node failed for ssl client	

	23: mbedtls_ssl_conf_max_frag_len fail 24: ssl cert setup failed 25: sni setup failed 26: ssl auth mode invalid
Example	*//Create a TCP client and connect to TCP server IP 192.168.99.185 on server's port 5001* AT+SKTCLIENT=0,192.168.99.101,5001 *+SKTCLIENT:OK* *+SKTCLIENT:con_id=4* *// Create a UDP client targeting to server "www.google.com" on server's port 8080* AT+SKTCLIENT=1,"www.google.com",8080 *+SKTCLIENT:OK* *+SKTCLIENT:con_id=5* *// query connection information* AT+SKTSTATE *con_id:1,server,tcp,address:192.168.99.143,port:5001,socket:0* *con_id:2,seed,tcp,address:192.168.99.185,port:64089,socket:1* *con_id:3,server,udp,address:192.168.99.143,port:5002,socket:2* *con_id:4,client,tcp,address:192.168.99.185,port:5001,socket:3* *con_id:5,client,udp,address:64.233.189.104,port:8080,socket:4* *+SKTSTATE:OK* *// Test SNI* AT+SKTCLIENT=2,www.google.com,443,,2,www.google.com
NOTE	This command will assign a con_id to this TCP/UDP/SSL Client.

4.4 AT+SKTDEL – Close TCP/UDP/SSL connection

AT+SKTDEL=<con_id>		
Description	This command is used to close TCP/UDP/SSL connection	
Response	+SKTDEL:OK +SKTDEL:ERROR:<error_no>	
Parameter	< con_id >	con_id=[1,9] for certain connection con_id=0 to close all connections
Error Number	1: command format error 2: command parameter error 3: no con_id is found	

Example	*// query connection information* AT+SKTSTATE *con_id:1,server,tcp,address:192.168.99.143,port:5001,socket:0* *con_id:2,seed,tcp,address:192.168.99.185,port:64089,socket:1* *con_id:3,server,udp,address:192.168.99.143,port:5002,socket:2* *con_id:4,client,tcp,address:192.168.99.185,port:5001,socket:3* *con_id:5,client,udp,address:64.233.189.104,port:8080,socket:4* *+SKTSTATE:OK* *// close con_id 5 (udp client)* AT+SKTDEL=5 *+SKTDEL:OK* *// query connection information* AT+SKTSTATE *con_id:1,server,tcp,address:192.168.99.143,port:5001,socket:0* *con_id:2,seed,tcp,address:192.168.99.185,port:64089,socket:1* *con_id:3,server,udp,address:192.168.99.143,port:5002,socket:2* *con_id:4,client,tcp,address:192.168.99.185,port:5001,socket:3* *+SKTSTATE:OK* *// close con_id 1 (TCP server), and its seed(con_id=2) will be also closed* AT+SKTDEL=1 *+SKTDEL:OK* *// query connection information* AT+SKTSTATE *con_id:3,server,udp,address:192.168.99.143,port:5002,socket:2* *con_id:4,client,tcp,address:192.168.99.185,port:5001,socket:3* *+SKTSTATE:OK* *// close all connections* AT+SKTDEL=0 *+SKTDEL:OK* *// query connection information* # AT+SKTSTATE *+SKTSTATE:OK*
NOTE	Use the AT+SKTSTATE command to show the connection id.

4.5 AT+SKTSEND – Send data

AT+ SKTSEND =<data_size>,<con_id>[,<dst_ip>,<dst_port>]:<data>	
Description	This command is used to send data to a specific connection

Response	+ SKTSEND:OK,<con_id> + SKTSEND:ERROR:<error_no>	
Paramter	<data_size>	Data length
	<con_id>	(1~9, con_id 0 is reserved)
	[<dst_ip>]	[optional]xxx.xxx.xxx.xxx (only need for udp server mode)
	[<dst_port>]	[optional]1~65535 (only need for udp server mode)
	<data>	Payload data
Error Number	1: parameter number error 2: <Buffer Size> exceeds ATPT send buffer size 3: con_id is not found 4: <UDP Client IP> or <UDP Client Port> error for udp server case 5: sendto() error for udp server 6: sendto() error for udp client 7: TCP server should send data to the seed 8: write error for tcp client/server	
Example	*// query connection information* AT+SKTSTATE *con_id:1,server,tcp,address:192.168.99.143,port:5001,socket:0* *con_id:2,seed,tcp,address:192.168.99.185,port:64089,socket:1* *con_id:3,server,udp,address:192.168.99.143,port:5002,socket:2* *con_id:4,client,tcp,address:192.168.99.185,port:5001,socket:3* *con_id:5,client,udp,address:64.233.189.104,port:8080,socket:4* *+SKTSTATE :OK* *// send data to TCP client(Seed) (con_id 2)* AT+ SKTSEND=14,2:Hello Realsil! *+ SKTSEND:OK,2* *// send data to UDP Server via UDP client(con_id 5)* AT+ SKTSEND=14,5:Hello Realsil! *+ SKTSEND:OK,5* *// send data to TCP Server via TCP client(con_id 4)* AT+ SKTSEND=14,4:Hello Realsil! *+ SKTSEND:OK,4* *// send data to UDP client(ip: 192.168.99.185, port:55339) via UDP Server(con_id 3)* AT+ SKTSEND=14,3,192.168.99.185,55339:Hello Realsil! *+ SKTSEND:OK,3*	

NOTE	1. Use the AT+SKTSTATE command to show the connection status. 2. The AT+ SKTSEND command can't send data via TCP server created at localhost. 3. After delimiter ":", any input will count

4.6 AT+SKTREAD – Receive data

<table>
<tr><th colspan="3">AT+ SKTREAD=<con_id>,<Buffer Size&gt;</th></tr>
<tr><td>Description</td><td colspan="2">This command is used to receive data from a specific connection id, and FW can also be configured to auto receive mode which means any packet received on any connection will return to host automatically(refer to command AT+SKTRECVCFG)</td></tr>
<tr><td>Response</td><td colspan="2">+ SKTREAD:OK,<data size>,<con_id>[,<dst_ip>,<dst_port>]:<data>
+ SKTREAD:ERROR:<error_no&gt;</td></tr>
<tr><td rowspan="2">Parameter</td><td>&lt;con_id&gt;</td><td>(1~9, con_id 0 is reserved)</td></tr>
<tr><td>&lt;Buffer Size&gt;</td><td>Data length</td></tr>
<tr><td>Error Number</td><td colspan="2">1: command format error
2: <Buffer Size> error (should be 1 ~ MAX_BUFFER(default 1600))
3: <con_id> is not found
4: recvfrom() error for udp server
5: recvfrom() error for udp client/seed
6: TCP server should receive from seed
7: connection lost
8: read() error for tcp con_id</td></tr>
<tr><td>Example</td><td colspan="2"><i>// query connection information</i>
AT+SKTSTATE
<i>con_id:1,server,tcp,address:192.168.99.143,port:5001,socket:0</i>
<i>con_id:2,seed,tcp,address:192.168.99.185,port:64089,socket:1</i>
<i>con_id:3,server,udp,address:192.168.99.143,port:5002,socket:2</i>
<i>con_id:4,client,tcp,address:192.168.99.185,port:5001,socket:3</i>
<i>con_id:5,client,udp,address:64.233.189.104,port:8080,socket:4</i>
<i>+SKTSTATE:OK</i>

<i>// receive data "12345678" via TCP seed (con_id 2)</i>
AT+ SKTREAD=2,1500
<i>+ SKTREAD:OK,8,2:12345678</i>

<i>// receive data "12345678" via UDP server(con_id 3)</i>
AT+ SKTREAD=3,1500
<i>+ SKTREAD:OK,8,3,192.168.99.185,52795:12345678</i></td></tr>
</table>

	// receive data "12345678" via TCP client(con_id 4) # AT+ SKTREAD=4,1500 *+ SKTREAD:OK,8,4:12345678*
NOTE	1. Use the AT+ SKTREAD command to receive data from the specific connection id. 2. The AT+ SKTREAD command can't receive data via TCP server created at localhost. 3. [,<dst_ip>,<dst_port>] will append only if receive data via UDP server created at localhost.

4.7 AT+SKTRECVCFG – Set auto receive data mode

AT+SKTRECVCFG =<enable>		
Description	This command is used to set auto receive data mode	
Response	+SKTRECVCFG:OK +SKTRECVCFG:ERROR:<error_no>	
Parameter	<enable>	0 : disable auto receive data mode (default) 1 : enable auto receive data mode
Error Number	1: command parameter error 2: start auto receive task fail	
NOTE	Once the auto receive mode is enabled, any packet received on any connection will return to host automatically in the same format as AT+ SKTREAD (refer to AT+ SKTREAD, response of command AT+ SKTREAD) in normal transmission mode. But if under transparent transmission mode, received data will return to host without any information in the head. Normal mode: *+ SKTREAD:OK,8,3,192.168.99.185,52795:12345678* TT(transparent transmission) mode: *12345678*	

4.8 AT+SKTSTATE – Check network connection status

AT+SKTSTATE	
Description	This command is used to print network connection status

Response	con_id :<con_id >,<server/seed(TCP client)/client>,\ <tcp/udp>,address:<IP ADDRESS>,port:<PORT>,socket:<socket id> … +SKTSTATE:OK
Error Number	NULL
Example	*// If there are some connections.* AT+SKTSTATE *con_id:1,server,tcp,address:192.168.99.143,port:5001,socket:0* *con_id:2,seed,tcp,address:192.168.99.185,port:64089,socket:1* *con_id:3,server,udp,address:192.168.99.143,port:5002,socket:2* *con_id:4,client,tcp,address:192.168.99.185,port:5001,socket:3* *con_id:5,client,udp,address:64.233.189.104,port:8080,socket:4* *+SKTSTATE:OK* *// If there is not any connection.* AT+SKTSTATE *+SKTSTATE:OK*

4.9 AT+PING – PING Command

AT+PING=<xxxx.xxxx.xxxx.xxxx>,[y/loop] **Or** **AT+PING=<con_id>, [y/loop]**		
Description	This command is used to PING a specific connection id, or PING a specific network address	
Response	+PING:OK +PING:ERROR:<error_no>	
Parameter case 1	<Remote IP>	xxx.xxx.xxx.xxx
	[y/loop]	No assign: Only five ping requests will be sent. Loop: loop, no count Count: loop with count
Parameter case 2	<con_id>	1~NUM_NS(default 10)
	[y/loop]	No assign: Only five ping requests will be sent. Loop: loop, no count Count: loop with count
Error Number	1: command format error 2: con_id is not found 3: partially lost of packets.	

	4: totally lost of packets.
Example	*// parameter case 1* AT+PING=192.168.1.1 *// Only five ping requests will be sent* AT+PING=192.168.1.1,loop *// loop, no count* AT+PING=192.168.1.1,10 *// loop 10 times* *// parameter case 2* AT+SKTSTATE *con_id:1,server,tcp,address:192.168.99.143,port:5001,socket:0* *con_id:2,seed,tcp,address:192.168.99.185,port:64089,socket:1* *con_id:3,server,udp,address:192.168.99.143,port:5002,socket:2* *con_id:4,client,tcp,address:192.168.99.185,port:5001,socket:3* *con_id:5,client,udp,address:64.233.189.104,port:8080,socket:4* *+SKTSTATE:OK* AT+PING=1 AT+PING=2 AT+PING=3
NOTE	Use the AT+ SKTREAD command to receive data from the specific connection id.

4.10AT+SKTTT – Set transparent transmission mode

AT+SKTTT=<enable>		
Description	This command is used to set transparent transmission(TT) mode	
Response	+SKTTT:OK +SKTTT:ERROR:<error_no>	
Parameter	<enable>	1 : enable TT mode (only “1” is valid by now)
Error Number	1: command parameter error 2: no connection found when try to enter TT mode 3: cannot enter TT mode if it’s server connection 4: more than one connection when try to enter TT mode 5: start TT task failed	
Example	*// For TT(transparent transmission) mode* AT+SKTDEL=0 *//close all connectiosn* *+SKTDEL:OK* *// create TCP client, single connection* AT+SKTCLIENT=0,192.168.99.101,5001 *+SKTCLIENT:OK*	

	[AT+SKTCLIENT] con_id=1 AT+SKTTT=1 *//enter TT mode* *//20ms interval between sending packets* *//auto recv mode is also enabled* *+SKTTT:OK* *//enter data transmission mode, any input is treated as data to send,* *//besides the uart echo is turned off, which means any input character* *//won't have uart echo* Hello Realsil! *// first packet* *// (wait for 20ms)* Hello Realsil! *// second packet* *(wait for 20ms)* ---- *// input four hyphens("-") to return to command mode* *# // return to command mode now, auto recv is disabled, uart echo is turned on*
NOTE	Once the TT mode is enabled, only one TCP/UDP client connection can be created.

4.11 AT+SKTAUTOLINK – Save translink and enable autolink

AT+SKTAUTOLINK=<enable>		
Description	This command is used to save connection information to flash and enable auto connect while booting up	
Response	+SKTAUTOLINK:OK +SKTAUTOLINK:ERROR:<error_no>	
Parameter	<enable>	0 : erase translink info in flash and disable autolink 1 : save translink and enable autolink
Error Number	1: command parameter error 2: parameter number error 3: no connection found	
Example	*// close all connections, if there are.* AT+SKTDEL=0 *+SKTDEL:OK* *// connect to AP* AT+WLCONN=iot_test,12345678 *+WLCONN:OK*	

	// enable auto connect, this will be store in flash AT+WLAUTOCONN=1 *+WLAUTOCONN:OK* *// create TCP client, single connection* AT+SKTCLIENT=0,192.168.99.101,5001 *+SKTCLIENT:OK* *+SKTCLIENT:con_id=1* *// save information into flash* AT+SKTAUTOLINK=1 *+SKTAUTOLINK:OK* *// reboot device* AT+RST *+RST:OK* *AT COMMAND READY* *> // start data transmission from here, 20ms between packets* ---- *// input four hyphens(“-”) to return to command mode* *# //return to command mode*
NOTE	Device will auto establish connection by using the information stored in flash, and enter data transparent transmission mode.

4.12 AT+HTTPCLIENT – Send http/https client request

AT+HTTPCLIENT=<HTTP:1/HTTPS:2>,<host>,<port>,<GET:2/POST:3>,<path>,<ca:1:N/2:Y>,<content-type>,<data>		
Description	Send http/https post packet.	
Response	+HTTPCLIENT:OK +HTTPCLIENT:ERROR: <error_code>	
Parameter	<HTTP:1/HTTPS:2>	1: HTTP 2: HTTPS
	<host>	The host name of server.
	<port>	The port value.
	<GET:2/POST:3>	Http type 2: Get 3: Post
	<path>	A string of path name.
	<ca:1:N/2:Y>	Need SSL verify? 1: No 2: Yes
	<content-type>	A string of http content.

	<data>	The post data, valid when http type is post.
Error number	1: command format error. 2: error to signal http or https. 3: invalid port. 4: invalid host. 5: invalid http type. 6: invalid path. 7: invalid post content. 8: invalid post data. 9: failed to set verify for https. 10: failed to create http task. 11: failed to create https task.	
Example	AT+HTTPCLIENT=2,httpbin.org,443,3,/post,2,application/json,param1=test_data1&param2=test_data2 AT+HTTPCLIENT=1,httpbin.org,80,2,/get?param1=test_data1&param2=test_data2,0,0,0	

4.13 AT+SSLCRET – Read or set CA cert/pk key

AT+SSLCRET=<TYPE>,[<LENGTH>,<CRT>]		
Description	Read or set CA cert/pk key	
Response	+SSLCRET:OK +SSLCRET:ERROR: <error_code>	
Parameter	<TYPE>	1: client CA. 2: private key. 3: server root CA. 4: public key.
	<LENGTH>	The cert_length.
	<CRT>	The string of output cert.
Error number	1: There is no parameter. 2: There is no cert type. 3: Failed when parse one or more PEM certificates from a buffer and add them to the chained list. For client crt, 4: Failed when parse one or more PEM certificates from a buffer and add them to the chained list. For CA crt.	
Example	AT+SSLCRET=1 AT+SSLCRET=1,10,1234567890	

5 MQTT command

5.1 AT+MQTTOPEN – Create (open) a new mqtt connection

AT+MQTTOPEN=<conn_id>,<host>,<port>		
Description	Create (open) a new mqtt connection with a conn_id	
Response	OK ERROR <error_number>	
Parameter	conn_id	0~3 in order to distinguish different connections, there are 4 at most.
	host	A string of host name, with 100 bytes at most.
	port	1 ~ 65535 The port of this connection. It is optional. If absent, the default value is 1883.
Error_number	1: common error. 2: input invalid parameter. 3: conflict conn_id. 4: memory failure. 5: has not attached. 6: the conn_id has not been created. 7: can not connect to the URL. 8: can not be authorized. 9: rejected by the server. 10: the conn_id is not connected. 11: the conn_id has been connected. 12: publish message failed. 13: subscribe topic failed. 14: this topic has been subscribed. 15: this topic has not been subscribed. 16: failed to unsubscribe this topic. 17: time out when subscribe or connect. 18: failed to create this conn_id task. 19: the wifi is not connected.	
Example	*// Create a connetion with ID 0, without port value.* AT+MQTTOPEN=0,adqqqkk.iot.gz.baidubce.com *+MQTTOPEN:OK* *// Create a connetion with ID 1, port value 1883* AT+MQTTOPEN=1,adqqqkk.iot.gz.baidubce.com,1883 *+MQTTOPEN:OK* *// Create a connetion with ID 0, which has been created before.* AT+MQTTOPEN=0,adqqqkk.iot.gz.baidubce.com	

	+MQTTOPEN:ERROR: 3

5.2 AT+MQTTCLOSE – Delete (close) a connection

AT+MQTTCLOSE=<conn_id>		
Description	Delete (close) a connection.	
Response	OK ERROR <error_number>	
Parameter	conn_id	0~3
Error_number	1: common error. 2: input invalid parameter. 3: conflict conn_id. 4: memory failure. 5: has not attached. 6: the conn_id has not been created. 7: can not connect to the URL. 8: can not be authorized. 9: rejected by the server. 10: the conn_id is not connected. 11: the conn_id has been connected. 12: publish message failed. 13: subscribe topic failed. 14: this topic has been subscribed. 15: this topic has not been subscribed. 16: failed to unsubscribe this topic. 17: time out when subscribe or connect. 18: failed to create this conn_id task. 19: the wifi is not connected.	
Example	AT+MQTTCLOSE=0	

5.3 AT+MQTTCONN – Connect to the mqtt server

AT+MQTTCONN=<conn_id>,clientid,<the_string_of_clientid> AT+MQTTCONN=<conn_id>,username,<the_string_of_username> AT+MQTTCONN=<conn_id>,password,<the_string_of_password> AT+MQTTCONN=<conn_id>,send	
Description	Connect to the mqtt server. The command (AT+MQTTCONN=<conn_id>,”send”) should be executed at last. The clientid should be set at first. The connection may be anonymous, so the username and password may be not needed. The result “OK” just means the command is executed successfully. When receiving connection_ack, there will be an “ACK” response.

Response	+MQTTCONN:OK ACK +MQTTCONN:ERROR <error_number>	
Parameter	conn_id	0~3
	clientid username password send	These parameters should be inputted with lowercase.
	<the_string_of_clientid>	The string of clientid, with 100 bytes at most.
	<the_string_of_username>	The string of username, with 100 bytes at most.
	<the_string_of_password>	The string of username, with 100 bytes at most.
Error_number	1: common error. 2: input invalid parameter. 3: conflict conn_id. 4: memory failure. 5: has not attached. 6: the conn_id has not been created. 7: can not connect to the URL. 8: can not be authorized. 9: rejected by the server. 10: the conn_id is not connected. 11: the conn_id has been connected. 12: publish message failed. 13: subscribe topic failed. 14: this topic has been subscribed. 15: this topic has not been subscribed. 16: failed to unsubscribe this topic. 17: time out when subscribe or connect. 18: failed to create this conn_id task. 19: the wifi is not connected.	
Example	*// Set mqtt server url at first.* AT+MQTTCONN=0,clientid,the_string_of_real_clientid *+MQTTCONN:OK* *// Set the username. If access with anonymous, it can be ignored.* AT+MQTTCONN=0,username,the_string_of_real_username *+MQTTCONN:OK* *// Set the password. If access with anonymous, it can be ignored.* AT+MQTTCONN=0,password,the_string_of_real_password *+MQTTCONN:OK* *// Connect to the server.* AT+MQTTCONN=0,send *+MQTTCONN:OK* *ACK*	

5.4 AT+MQTTDISCONN – Disconnect from the mqtt server

AT+MQTTDISCONN=<conn_id>		
Description	Disconnect from the mqtt server.	
Response	OK ERROR <error_number>	
Parameter	conn_id	0~3
Error_number	1: common error. 2: input invalid parameter. 3: conflict conn_id. 4: memory failure. 5: has not attached. 6: the conn_id has not been created. 7: can not connect to the URL. 8: can not be authorized. 9: rejected by the server. 10: the conn_id is not connected. 11: the conn_id has been connected. 12: publish message failed. 13: subscribe topic failed. 14: this topic has been subscribed. 15: this topic has not been subscribed. 16: failed to unsubscribe this topic. 17: time out when subscribe or connect. 18: failed to create this conn_id task. 19: the wifi is not connected.	
Example	AT+MQTTDISCONN=0 *+MQTTDISCONN:OK*	

5.5 AT+MQTTSUB – Subscribe topic

AT+MQTTSUB=<conn_id>,<topic_string>,<QoS>		
Description	Subscribe topic. The result "OK" just means the command is executed successfully. When receiving subscribe_ack, there will be an "ACK" response.	
Response	+MQTTSUB:OK ACK +MQTTSUB:ERROR <error_number>	
Parameter	conn_id	0~3

	topic_string	The string of subscribed topic, with 100 bytes at most.
	QoS	0~2 This parameter is optional, if absent, the default value is 2.
Error_number	1: common error. 2: input invalid parameter. 3: conflict conn_id. 4: memory failure. 5: has not attached. 6: the conn_id has not been created. 7: can not connect to the URL. 8: can not be authorized. 9: rejected by the server. 10: the conn_id is not connected. 11: the conn_id has been connected. 12: publish message failed. 13: subscribe topic failed. 14: this topic has been subscribed. 15: this topic has not been subscribed. 16: failed to unsubscribe this topic. 17: time out when subscribe or connect. 18: failed to create this conn_id task. 19: the wifi is not connected.	
Example	*// Subscribe the topic without QoS value.* AT+MQTTSUB=0,the_string_of_your_topic *+MQTTSUB:OK* *ACK* *// Subscribe the topic with QoS 0.* AT+MQTTSUB=0,the_string_of_your_topic,0 *+MQTTSUB:OK* *ACK*	

5.6 AT+MQTTUNSUB – Unsubscribe topic

AT+MQTTUNSUB=<conn_id>,<topic_string>		
Description	Unsubscribe topic. The result “OK” just means the command is executed successfully. When receiving unsubscribe_ack, there will be an “ACK” response.	
Response	+MQTTUNSUB:OK ACK +MQTTUNSUB:ERROR <error_number>	
Parameter	conn_id	0~3

	topic_string	The string of topic to be unsubscribed, with 100 bytes at most.
Error_number	1: common error. 2: input invalid parameter. 3: conflict conn_id. 4: memory failure. 5: has not attached. 6: the conn_id has not been created. 7: can not connect to the URL. 8: can not be authorized. 9: rejected by the server. 10: the conn_id is not connected. 11: the conn_id has been connected. 12: publish message failed. 13: subscribe topic failed. 14: this topic has been subscribed. 15: this topic has not been subscribed. 16: failed to unsubscribe this topic. 17: time out when subscribe or connect. 18: failed to create this conn_id task. 19: the wifi is not connected.	
Example	AT+MQTTUNSUB=0,the_string_of_your_topic *+MQTTUNSUB:OK* *ACK*	

5.7 AT+MQTTPUB – Publish message

AT+MQTTPUB=<conn_id>,<message_id>,qos<qos_value> AT+MQTTPUB=<conn_id>,<message_id>,retain,<retain_value> AT+MQTTPUB=<conn_id>,<message_id>,topic ,<the_string_of_topic> AT+MQTTPUB=<conn_id>,<message_id>,message ,<the_string_of_message> AT+MQTTPUB=<conn_id>,<message_id>,send		
Description	Publish message to the server. The command (AT+MQTTPUB=<conn_id>,<message_id>,“send”) should be executed at last. The qos and retain are optional, if absent, the default value of qos is 2, the default value of retain is 0. The result “OK” just means the command is executed successfully. When receiving publish_ack, there will be an “ACK” response.	
Response	+MQTTPUB:OK ACK +MQTTPUB:ERROR <error_number>	
Parameter	conn_id	0~3

	message_id	0~65535
	qos retain topic message send	These parameters should be inputted with lowercase.
	qos_value	0~2
	retain_value	0~1
	the_string_of_topic	The string of topic, with the length of 100 bytes at most.
	the_string_of_message	The string of message, with the length of 100 bytes at most.
Error_number	1: common error. 2: input invalid parameter. 3: conflict conn_id. 4: memory failure. 5: has not attached. 6: the conn_id has not been created. 7: can not connect to the URL. 8: can not be authorized. 9: rejected by the server. 10: the conn_id is not connected. 11: the conn_id has been connected. 12: publish message failed. 13: subscribe topic failed. 14: this topic has been subscribed. 15: this topic has not been subscribed. 16: failed to unsubscribe this topic. 17: time out when subscribe or connect. 18: failed to create this conn_id task. 19: the wifi is not connected.	
Example	*// Set the topic string.* AT+MQTTPUB=0,1,topic,the_string_of_your_topic *+MQTTPUB:OK* *// Set the message string.* AT+MQTTPUB=0,1,msg,the_string_of_your_message *+MQTTPUB:OK* *// Set the qos value.* AT+MQTTPUB=0,1,qos,0 *+MQTTPUB:OK* *// Set the retain value.* AT+MQTTPUB=0,1,retain,0 *+MQTTPUB:OK* *// Send publish message.* AT+MQTTPUB=0,1,send *+MQTTPUB:OK*	

	ACK

5.8 AT+MQTTCFG – Configure or inquire the parameters

AT+MQTTCFG=<conn_id>,? AT+MQTTCFG=<conn_id>,version,<version_value> AT+MQTTCFG=<conn_id>,keepalive,<keepalive_value> AT+MQTTCFG=<conn_id>,session,<session_value> AT+MQTTCFG=<conn_id>,timeout,<timeout_value> AT+MQTTCFG=<conn_id>,will,<will_value_0> AT+MQTTCFG=<conn_id>,will,<will_value_1>,<will_value_qos>,<will_retain>,<will_topic>,<will_message> AT+MQTTCFG=<conn_id>,ssl,<ssl_value>		
Description	Configure or inquire the parameters. The configure will work before creating connection.	
Response	OK ERROR <error_number>	
Parameter	conn_id	0~3
	? version keepalive session timeout will ssl	If the second parameter is '?', it means inquire command, otherwise, it means configure command. The input parameters should be input with lowercase.
	version_value	3 or 4
	keepalive_value	1~3600
	session_value	0~1
	timeout_value	10000 ~ 60000 (means millisecond)
	will_value	0~1
	will_qos	The quality of service setting for the LWT message.
	will_retain	The retained flag for the LWT message.
	will_topic	The LWT topic to which the LWT message will be published.
	will_message	The LWT payload.
Error_number	1: common error. 2: input invalid parameter. 3: conflict conn_id. 4: memory failure.	

	5: has not attached. 6: the conn_id has not been created. 7: can not connect to the URL. 8: can not be authorized. 9: rejected by the server. 10: the conn_id is not connected. 11: the conn_id has been connected. 12: publish message failed. 13: subscribe topic failed. 14: this topic has been subscribed. 15: this topic has not been subscribed. 16: failed to unsubscribe this topic. 17: time out when subscribe or connect. 18: failed to create this conn_id task. 19: the wifi is not connected.
Example	*// Query the current parameters of connect-id 0.* AT+MQTTCFG=0,? *+MQTTCFG:MQTTVersion 4* *+MQTTCFG:keepAliveInterval 60* *+MQTTCFG:cleansession 1* *+MQTTCFG:command_timeout_ms 60000 (ms)* *+MQTTCFG:willFlag 0* *+MQTTCFG:useSsl 0* *+MQTTCFG:OK* *// Set the version to 3.* AT+MQTTCFG=0,version,3 *+MQTTCFG:OK* *// Query the current parameters of connect-id 0 again.* AT+MQTTCFG=0,? *+MQTTCFG:MQTTVersion 3* *+MQTTCFG:keepAliveInterval 60* *+MQTTCFG:cleansession 1* *+MQTTCFG:command_timeout_ms 60000 (ms)* *+MQTTCFG:willFlag 0* *+MQTTCFG:useSsl 0* *+MQTTCFG:OK*

5.9 AT+MQTTRESET – Reset all connections

AT+MQTTRESET	
Description	Reset all connections.
Response	OK ERROR <error_number>

Parameter	None parameter.
Error_number	NULL

6 BlueTooth command

6.1 AT+BLEPMODE – Set the BT peripheral mode

AT+BLEPMODE=<peripheral_mode>		
Description	Set the BT peripheral mode. If you want to use this module as BT peripheral, this command should be set to 1 at first.	
Response	OK ERROR <error_number>	
Parameter	peripheral_mode	0: Disable 1: Enable
Error_number	1: There should be some parameters. 2: The number of parameters is wrong, or input wrong parameters. 3: Command type error.	
Example	AT+BLEPMODE=0 AT+BLEPMODE=1	

6.2 AT+BLECMODE – Set the BT central mode

AT+BLECMODE=<central_mode>		
Description	Set the BT central mode. If you want to use this module as BT central, this command should be set to 1 at first.	
Response	OK ERROR <error_number>	
Parameter	central_mode	0: Disable 1: Enable
Error_number	1: There should be some parameters. 2: The number of parameters is wrong, or input wrong parameters. 3: Command type error.	
Example	AT+BLECMODE=0 AT+BLECMODE =1	

6.3 AT+BLEMAC – Set or get BT MAC address

AT+BLEMAC=? AT+BLEMAC=<mac>	
Description	AT+BLEMAC=? AT+BLEMAC=<mac> The set command will work after next initialisation.

Response	OK ERROR <error_number>	
Parameter	mac	A hexadecimal string with length of 12 bytes.
Error_number	1: There should be some parameters. 2: The number of parameters is wrong, or input wrong parameters. 3: Command type error.	
Example	AT+BLEMAC=? AT+BLEMAC=2a3f2d10e429	
NOTE	The OTP area shall be written while setting the BT MAC address. As the OTP space is limited, please do not modify this MAC value unless necessary.	

6.4 AT+BLEMTU – Set or get BT GATT MTU size

AT+BLEMTU=? AT+BLEMTU=<mtu>		
Description	AT+BLEMTU=? AT+BLEMTU=<mtu>	
Response	OK ERROR <error_number>	
Parameter	mtu	The mtu means maximum transfer unit. 23 ~ 512
Error_number	1: There should be some parameters. 2: The number of parameters is wrong, or input wrong parameters. 3: Command type error.	
Example	AT+BLEMTU=? AT+BLEMTU=200	

6.5 AT+BLEPAIR – Configure authentication information

AT+BLEPAIR=KEY,<conn_id>,<passcode> AT+BLEPAIR=SEND,<conn_id> AT+BLEPAIR=MODE,<auth_flags>,<io_cap>,<sec_enable>,<oob_enable>	
Description	Configure authentication information
Response	OK

	ERROR <error_number>	
Parameter	KEY SEND MODE	These parameters should be inputted with uppercase.
	conn_id	0~2
	passcode	0~999999
	auth_flags	A hexadecimal string, such as “0x2A”.
	io_cap	0~255
	sec_enable	0~1
	oob_enable	0~1
Error_number	1: There should be some parameters. 2: The number of parameters is wrong, or input wrong parameters. 3: Command type error.	
Example	AT+BLEPAIR=SEND,0 AT+BLEPAIR=KEY,0,123456 AT+BLEPAIR=MODE,0x5,2,1,0	

6.6 AT+BLEPASSKEY – Setup or inquire the pairing code

AT+BLEPASSKEY=? AT+BLEPASSKEY=<passkey>		
Description	Setup or inquire the pairing code	
Response	OK ERROR <error_number>	
Parameter	?	means inquire the pairing code.
	passkey	000000~999999 means the paring code.
Error_number	1: There should be some parameters. 2: The number of parameters is wrong, or input wrong parameters. 3: Command type error.	
Example	AT+BLEPASSKEY=? AT+BLEPASSKEY=000001	
NOTE	The passkey must be in 6 digits. If the user need set the value less than 6 digits, please write more 0s padding at left.	

6.7 AT+BLEUSERCONF – Send user confirmation

AT+BLEUSERCONF=<conn_id>,<conf>

Description	Send user confirmation.	
Response	OK ERROR <error_number>	
Parameter	conn_id	0~2
	conf	0-(Reject),1-(Accept)
Error_number	1: There should be some parameters. 2: The number of parameters is wrong, or input wrong parameters. 3: Command type error.	
Example	AT+BLEUSERCONF=0,1	

6.8 AT+BLECONNPARAM – Update connection parameters

AT+BLECONNPARAM=<conn_id>,<interval_min>,<interval_max>,<latency>,<supervision_timeout>		
Description	Update connection parameters. The interval_min, interval_max, latency, supervision_timeout are all string of a hexadecimal value, such as “0x0A20”.	
Response	OK ERROR <error_number>	
Parameter	conn_id	0~2
	interval_min	0x0006 ~ 0x0C80 (Range is 7.5ms to 4 seconds)
	interval_max	0x0006 ~ 0x0C80 (Range is 7.5ms to 4 seconds) interval_max > interval_min
	latency	0x0000 - 0x01F3
	supervision_timeout	0x000A - 0x0C80 (Range is 100ms to 32 seconds)
Error_number	1: There should be some parameters. 2: The number of parameters is wrong, or input wrong parameters. 3: Command type error.	
Example	AT+BLECONNPARAM=0,0x30,0x40,0x0,0x1F4	

6.9 AT+BLECLRINQ – Clear or inquire the pairing information

AT+BLECLRINQ=CLEAR AT+BLECLRINQ=INFO	
Description	Clear or inquire the paring information.

Response	OK ERROR <error_number>	
Parameter	CLEAR	Clear all the paring informations.
	INFO	List all the paring informations.
Error_number	1: There should be some parameters. 2: The number of parameters is wrong, or input wrong parameters. 3: Command type error.	

6.10AT+BLENAME – Set or inquire the adv name

AT+BLENAME=? AT+BLENAME=<name>		
Description	Set or inquire the adv name.	
Response	OK ERROR <error_number>	
Parameter	?	Inquire the adv name.
	name	Set the adv name, it is a string with 22 bytes at most. It will work when the adv is not going.
Error_number	1: There should be some parameters. 2: The number of parameters is wrong, or input wrong parameters. 3: Command type error. 4: The ble adv is on going, please stop it.	

6.11AT+BLEADV – Set or inquire the adv status

AT+BLEADV=? AT+BLEADV=<status>		
Description	Set or inquire the adv status.	
Response	OK ERROR <error_number>	
Parameter	?	Inquire the adv status.
	status	0~1
Error_number	1: There should be some parameters. 2: The number of parameters is wrong, or input wrong parameters. 3: Command type error.	

6.12AT+BLEADVINTV – Set or inquire the adv interval

AT+BLEADVINTV=? AT+BLEADVINTV=<adv_interval_max>,<adv_interval_min>		
Description	Set or inquire the adv interval.	
Response	OK ERROR <error_number>	
Parameter	?	Inquire the adv interval.
	adv_interval_max	0x0020 - 0x4000 (20ms - 10240ms, 0.625ms/step)
	adv_interval_min	0x0020 - 0x4000 (20ms - 10240ms, 0.625ms/step) adv_interval_max > adv_interval_min
Error_number	1: There should be some parameters. 2: The number of parameters is wrong, or input wrong parameters. 3: Command type error.	
Example	AT+BLEADVINTV=1600,1600	

6.13AT+BLEINDNTF – Send indication/notification from GATT server

AT+BLEINDNTF=<conn_id>,<service_id>,<attribute_index>,<type>,<length>,<p_value>		
Description	Send indication/notification from GATT server.	
Response	OK ERROR <error_number>	
Parameter	conn_id	0~2
	service_id	1.
	attribute_index	0xa or 0x7
	type	0: any PDU type. 1: notification PDU type. 2: indication PDU type.
	length	1~23.
	p_value	A hexadecimal value stream after the parameter “length”, and the number of value is not larger than length. If the number is less than length, the end will be filled with 0xFF as padding.
Error_number	1: There should be some parameters.	

	2: The number of parameters is wrong, or input wrong parameters. 3: Command type error.
Example	AT+BLEINDNTF=0,1,0xa,2,0x1,0x1 AT+BLEINDNTF=0,1,0x7,1,0x2,0x1,0x2
NOTE	Before peripheral sending indication/notification to central, central should enable CCCD at first. For central, execute "AT+BLEWRITE=0,1,0x14,0x02,0x01,0x00" to enable peripheral notification. For peripheral, execute "AT+BLEWRITE=0,1,0x17,0x02,0x02,0x00" to enable peripheral indication.

6.14AT+BLECONN – Create connection

AT+BLECONN=P/R,<ble_bd_addr>		
Description	Create connection.	
Response	OK ERROR <error_number>	
Parameter	P	public device address type.
	R	random device address type.
	ble_bd_addr	device address, a hexadecimal value string, with length of 12 bytes.
Error_number	1: There should be some parameters. 2: The number of parameters is wrong, or input wrong parameters. 3: Command type error.	
Example	AT+BLECONN=P,001122334455	

6.15AT+BLEDISCONN – Close connection

AT+BLEDISCONN=<conn_id>		
Description	Close connection.	
Response	OK ERROR <error_number>	
Parameter	conn_id	0~2
Error_number	1: There should be some parameters. 2: The number of parameters is wrong, or input wrong parameters. 3: Command type error.	
Example	AT+BLEDISCONN=0	

6.16AT+BLECONNINFO – Get all connection information

AT+BLECONNINFO	
Description	Get all connection information.
Response	Return the information of all connections, including active link number, active link information, idle link number.
Parameter	NULL
Error_number	1: The number of parameters is wrong, or input wrong parameters. 2: Command type error.

6.17AT+BLESCAN – Scan BLE adv

AT+BLESCAN=<scan_enable>,<filter_policy>,<filter_duplicate>		
Description	Scan BLE adv	
Response	OK ERROR <error_number>	
Parameter	<scan_enable>	0: stop scanning. 1: start scanning.
	<filter_policy>	0: any. 1: whitelist.
	<filter_duplicate>	0: disable. 1: enable.
Error_number	1: There should be some parameters. 2: The number of parameters is wrong, or input wrong parameters. 3: Command type error.	
Example	AT+BLESCAN=1,0,1 AT+BLESCAN=0	

6.18AT+BLEREAD – Read characteristic value

AT+BLEREAD=<conn_id>,<handle> AT+BLEREAD=<conn_id>,<start_handle>,<end_handle>,<uuid_type>,<uuid>		
Description	1. Read characteristic value. 2. Read characterristic value by uuid.	
Response	OK ERROR <error_number>	
Parameter	<conn_id>	0~2
	<handle>	Request handle, a hexadecimal value in 0x1 ~ 0xFFFF.

	<start_handle>	Start handle of range to be searched, a hexadecimal value in 0x1 ~ 0xFFFF.
	<end_handle>	End handle of range to be searched, a hexadecimal value in 0x1 ~ 0xFFFF.
	<uuid_type>	0~1
	<uuid>	A hexadecimal value stream. If uuid_type is 0, it is a hexadecimal value stream with 4 bytes, such as "2A3F", else, it is a hexadecimal value stream with 32 bytes.
Error_number	1: There should be some parameters. 2: The number of parameters is wrong, or input wrong parameters. 3: Command type error.	
Example	AT+BLEREAD=0,0x1,0xFFFF,0,B001	

6.19AT+BLEWRITE – Write characteristic value

AT+BLEWRITE=<conn_id>,<type>,<handle>,<length>,<value>		
Description	Write characteristic value.	
Response	OK ERROR <error_number>	
Parameter	<conn_id>	0~2
	<type>	0x1: Write request. 0x2: Write command.
	<handle>	0x11
	<length>	If type is 0x1, range of length is from 0 to 512. If type is 0x2, range of length is from 0 to (mtu_size - 3).
	<value>	A hexadecimal value stream after the parameter "length", and the number of value is not larger than length. If the number is less than length, the end will be filled with 0xFF as padding.
Error_number	1: There should be some parameters. 2: The number of parameters is wrong, or input wrong parameters. 3: Command type error.	
Example	AT+BLEWRITE=0,1,0x11,0x1,0x02	

	AT+BLEWRITE=0,2,0x11,0x2,0x10,0x20

6.20 AT+BLEWHITELIST – Modify whitelist

AT+BLEWHITELIST=0 AT+BLEWHITELIST=1,P/R,<addr> AT+BLEWHITELIST=2,P/R,<addr>		
Description	1. Clear the stored whitelist. 2. Add a new element into whitelist. 3. Delete an element from whitelist.	
Response	OK ERROR <error_number>	
Parameter	0	Clear the stored whitelist.
	1	Add a new element into whitelist.
	2	Delete an element from whitelist.
	P	Public address type.
	R	Random address type.
	<addr>	Address, a hexadecimal value string with length of 12 bytes.
Error_number	1: There should be some parameters. 2: The number of parameters is wrong, or input wrong parameters. 3: Command type error.	

6.21 AT+BLESCANPARAM – Modify scan interval/window

AT+BLESCANPARAM=1,<scan_interval> AT+BLESCANPARAM=2,<scan_window>		
Description	1. Modify scan interval. 2. Modify scan window.	
Response	OK ERROR <error_number>	
Parameter	1	Modify scan interval.
	2	Modify scan window.
	<scan_interval>	0x0004 - 0x4000 (2.5ms - 10240ms, 0.625ms/step)
	<scan_window>	0x0004 - 0x4000 (2.5ms - 10240ms, 0.625ms/step)
Error_number	1: There should be some parameters.	

	2: The number of parameters is wrong, or input wrong parameters. 3: Command type error.
Example	AT+BLESCANPARAM=1,0x190 AT+BLESCANPARAM=2,0xC8

6.22 AT+BLEAUTOCONN – BLE auto reconnect

AT+BLEAUTOCONN=<status> AT+BLEAUTOCONN=P/R,<ble_bd_addr>		
Description	1. Enable/Disable BLE auto reconnect. 2. Set BLE auto reconnect remote address.	
Response	OK ERROR <error_number>	
Parameter	<status>	0: Discable. 1: Enable.
	P	Public address type.
	R	Random address type.
	<ble_bd_addr>	Auto reconnect address, a hexadecimal value string, with length of 12 bytes.
Error_number	1: There should be some parameters. 2: The number of parameters is wrong, or input wrong parameters. 3: Command type error.	
Example	1. AT+BLEAUTOCONN=1 //Enable Ble Auto Connect 2. AT+BLEAUTOCONN=P/R,BLE_BD_ADDR //Set Ble Auto Connect Remote Mac, start scan the remote adv. When scanning the adv for the specified address, establish a connection. 3. When the GATT is connected, please input AT+BLEAUTOCONN=0, to disable Ble Auto Connect function.	

6.23 AT+BLEIBEACON– Start or stop ibeacon

AT+BLEIBEACON=<status>		
Description	Start or stop ibeacon	
Response	OK ERROR <error_number>	
Parameter	<status>	0: Disable 1: Enable
Error_number	1: There should be some parameters.	

	2: The number of parameters is wrong, or input wrong parameters. 3: Command type error.
Example	AT+BLEIBEACON=0 AT+BLEIBEACON=1

6.24AT+BLEIBCNDATA – Set or get ibeacon adv data

AT+BLEIBCNDATA=<companyID>,<major>,<minor>,<power> AT+BLEIBCNDATA=?		
Description	Set or get ibeacon adv data.	
Response	OK ERROR <error_number>	
Parameter	<companyID>	A hexadecimal value in 0x1 ~ 0xFFFF.
	<major>	A hexadecimal value in 0x1 ~ 0xFFFF.
	<minor>	A hexadecimal value in 0x1 ~ 0xFFFF.
	<power>	A hexadecimal value in 0x1 ~ 0xFF.
	?	Get ibeacon adv data.
Error_number	1: There should be some parameters. 2: The number of parameters is wrong, or input wrong parameters. 3: Command type error.	

6.25AT+BLEIBCNUUID – Set or get ibeacon uuid

AT+BLEIBCNUUID=<uuid> AT+BLEIBCNUUID=?		
Description	Set or get ibeacon uuid.	
Response	OK ERROR <error_number>	
Parameter	<uuid>	A hexadecimal value string with length of 32 bytes.
	?	Get ibeacon uuid.
Error_number	1: There should be some parameters. 2: The number of parameters is wrong, or input wrong parameters. 3: Command type error.	

7 Release History

Release	Time	Notes
The first release.	2024-01-29	-