AT Command Application Note

This document provides information for controlling Ameba through external UART.

COPYRIGHT

DISCLAIMER

Please Read Carefully:

Realtek Semiconductor Corp., (Realtek) 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.

Reproduction of significant portions in Realtek data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Realtek is not responsible or liable for such reproduced documentation. Information of third parties may be subject to additional restrictions.

Buyers and others who are developing systems that incorporate Realtek 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 Realtek 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 Realtek products, Customer will thoroughly test such applications and the functionality of such Realtek products as used in such applications.

Realtek’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 Realtek products; by downloading, accessing or using Realtek’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 Realtek Resources solely for this purpose and subject to the terms of this Notice.

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

Customer is authorized to use, copy and modify any individual Realtek Resource only in connection with the development of applications that include the Realtek product(s) identified in such Realtek Resource. No other license, express or implied, by estoppel or otherwise to any other Realtek intellectual property right, and no license to any technology or intellectual property right of Realtek 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 Realtek 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 Realtek Resources may require a license from a third party under the patents or other intellectual property of the third party, or a license from Realtek under the patents or other Realtek’s intellectual property.

Realtek's Resources are provided "as is" and with all faults. Realtek 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.

Realtek 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 Realtek Resources or otherwise. In no event shall Realtek be liable for any actual, direct, special, collateral, indirect, punitive, incidental, consequential or exemplary damages in connection with or arising out of Realtek's Resources or use thereof, and regardless of whether Realtek has been advised of the possibility of such damages. Realtek is not responsible for any failure to meet such industry standard requirements.

Where Realtek specifically promotes products as facilitating functional safety or as compliant with industry functional safety standards, such products are intended to help enable customers to design and create their own applications that meet applicable functional safety standards and requirements. Using products in an application does not by itself establish any safety features in the application. Customers must ensure compliance with safety-related requirements and standards applicable to their applications. Designer may not use any Realtek products in life-critical medical equipment unless authorized officers of the parties have executed a special contract specifically governing such use. Life-critical medical equipment is medical equipment where failure of such equipment would cause serious bodily injury or death. Such equipment includes, without limitation, all medical devices identified by the U.S.FDA as Class III devices and equivalent classifications outside the U.S.

Customers agree that it has the necessary expertise to select the product with the appropriate qualification designation for their applications and that proper product selection is at Customers' own risk. Customers are solely responsible for compliance with all legal and regulatory requirements in connection with such selection.

Customer will fully indemnify Realtek and its representatives against any damages, costs, losses, and/or liabilities arising out of Designer's non-compliance with the terms and provisions of this Notice.

TRADEMARKS

Realtek is a trademark of Realtek Semiconductor Corporation. Other names mentioned in this document are trademarks/registered trademarks of their respective owners.

USING THIS DOCUMENT

Though every effort has been made to ensure that this document is current and accurate, more information may have become available subsequent to the production of this guide.

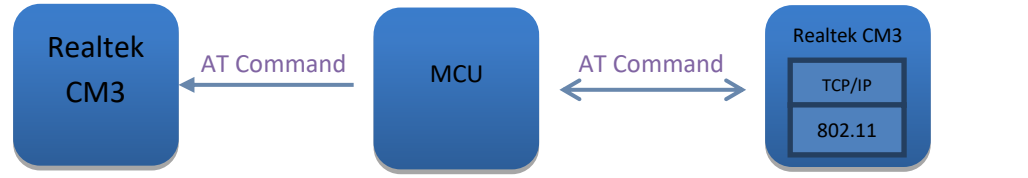
Revision History

Revision	Release Date	Summary
0.1	2020/06/10	First Version
0.2	2022/03/02	Updated list of AT Command as of 22/03/02 and updated all TBD in command responses
0.5	2024/04/29	Update BSSID format for ATW6
0.6	2024/08/23	Update ATW7 for z2/z2plus

Table of Contents

1 System Architecture

Realtek Low Power Wi-Fi SoC can be a standalone system with Wi-Fi internet capability or a Wi-Fi interface that connect to an existing MCU.



Standalone System

8195A as Wi-Fi Internet connectivity

Realtek CM3 attaches to MCU through UART or SPI, and MCU control Realtek CM3 through AT command.

2 Command Format

Command	Delimiter	Payload	Delimiter
AT CMD(4 chars)	=	Req Data	\r
AT CMD(4 chars)	\r		

Response Formats					
Delimiter (2 chars)	Command (4 chars)	Delimiter (2 chars)	Return (1 char)	Delimiter (2 chars)	Payload
\r\n	AT CMD	\r\n	OK	\r\n	Data
\r\n	AT CMD	\r\n	Error type	\r\n	

3 AT command

3.1 AT command list

AT Command	Description
LOG Common Command	
AT??	Print cmd history
AT--	Exit Log service
WLAN	
ATW0	Network set SSID
ATW1	Network set passphrase
ATW2	Network set Key ID
ATW3	Set Access Point SSID
ATW4	Set Access Point Security Key
ATW5	Set Access Point Channel
ATW6	Set BSSID of the WLAN
ATW7	Set WLAN security
ATWA	Activate Access Point
ATWB	Start STA+AP
ATWC	Join a network
ATWD	Disconnect from a network
ATWI	Ping test
ATWL	SSL client
ATWM	Wlan Wi-Fi promisc
ATWO	Wlan OTA update
ATWP	Power on/off wifi module
ATWQ	Wi-Fi Simple Config
ATWR	Get RSSI of Associated Network Access Point
ATWS	Scan for Network Access Point
ATWs	Scan with SSID/channel
ATWT	TCP T/RX throughput test
ATWU	UDP
ATWW	Wi-Fi Protected Setup
ATWZ	Wlan iwpriv
ATW?	Show network information
ATXP	Wlan Power Saving Control
ATWY	Get SNR value

Socket	
ATP1	Set Transport Protocol
ATP2	Set Transport Local Port Number
ATP3	Set Transport Remote Host Port IP Address
ATP4	Set Transport Remote Port Number
ATP5	Stop/Start Transport Server
ATP6	Stop/Start Transport Client
ATP?	Show Transport Settings
ATR0	Read Transport Data
ATR1	Set Read Transport Packet Size
ATRA	Write Transport Data
ATRB	Set Write Transport Packet Size
ATPE	Set static IP, gateway and mask for STA
System	
ATSC	Clear OTA signature
ATSL	System wakelock control
ATSR	Recover OTA signature
ATSK	Secure boot enable
ATSG	MP GPIO test
ATSS	Show CPU status
ATS@	Debug message setting
ATS!	Debug config setting
ATS?	SW version and compile time

3.2 AT command

3.2.1 COMMON

3.2.1.1 'AT??' Print Log History

Description: To display the log of AT Command run previously
Command Format: AT??<CR>
Default Value: None
Response: Log History of AT Command
Note: To use this command, please set **CONFIG_LOG_HISTORY** to 1 in platform_opts.h

3.2.1.2 'AT--' Exit Log Service

Description:
Command Format: AT--<CR>
Default Value: None
Response: Leave LOG Service

3.2.2 WLAN

3.2.2.1 Quick Start

This section introduces common usage, some use AT command and some use API.
The detail of API usage can reference \doc\api_doc.

There is also wlan scenario example for reference.
\component\common\example\wlan_scenario\example_wlan_scenario.c

3.2.2.1.1 How does station connect to AP?

Use ATCMD with SSID

ATW0=<ssid>
ATW1=<password>
ATW2=<key_id>
ATWC

Use ATCMD with BSSID

ATW6=<bssid>
ATW1=<password>
ATWC

Call API in wifi_conf.c

wifi_connect : use SSID to connect to AP
wifi_connect_bssid : use bssid to connect to AP

3.2.2.1.2 How does station disconnect from AP?

Use ATCMD

ATWD

Call API in wifi_conf.c

wifi_disconnect

3.2.2.1.3 How to scan with SSID/channel

Use ATCMD

ATW0=<ssid>
ATWs=<Num_of_channel><Channel1,Channel2…>

3.2.2.1.4 How to register wifi event callback function?

Search “wifi_reg_event_handler” as reference in wifi_conf.c

3.2.2.1.5 How to detect wlan condition of connect or disconnect event?

Register wifi event callback function for specific event

WIFI_EVENT_CONNECT : association done
WIFI_EVENT_FOURWAY_HANDSHAKE_DONE : fourway handshake done
WIFI_EVENT_BEACON_AFTER_DHCP : Get IP from DHCP
WIFI_EVENT_DISCONNECT : wifi disconnect

3.2.2.1.6 How to enable/disable power saving mode in station mode?

Call API in wifi_conf.c

wifi_enable_powersave
wifi_disable_powersave

3.2.2.1.7 How to start soft AP mode?

Use ATCMD

ATW3=<ssid>
ATW4=<password>
ATW5=<channel>
ATWA

Call API in wifi_conf.c

wifi_start_ap

3.2.2.1.8 How to start soft AP mode with hidden ssid?

Call API in wifi_conf.c

wifi_start_ap_with_hidden_ssid

3.2.2.1.9 How to create concurrent mode?

Use ATCMD, start AP first then Station

ATW3=<ssid>
ATW4=< password>
ATW5=<channel>
ATWB
ATW0=<ssid>
ATW1=<password>
ATW2=<key_id>
ATWC

3.2.2.1.10 How to set client number in AP mode?

Call API in wifi_util.c

wext_set_sta_num

3.2.2.1.11 How to delete station in AP mode?

Call API in wifi_util.c

wext_del_station

3.2.2.1.12 How to get auto-scan channel?

Call API in wifi_util.c

wext_get_auto_chl

3.2.2.1.13 How to set partial scan channel in station mode?

Call API in wifi_conf.c

wifi_set_pscan_chan

3.2.2.1.14 How to set auto-reconnect in station mode?

Call API in wifi_conf.c

wifi_config_autoreconnect

3.2.2.1.15 How to get TX power?

Call API in wifi_util.c

wext_get_tx_power

3.2.2.1.16 How to get RX RSSI?

Call API in wifi_conf.c

wifi_get_rssi

3.2.2.2 'ATW0' Wlan Set Network SSID

Description:
Command Format: ATW0=SSID<CR>
Default Value: None
Response: None

3.2.2.3 'ATW1' Wlan set Network Passphrase

Description:
Command Format: ATW1=password<CR>
Default Value: None
Response: None

3.2.2.4 'ATW2' Wlan Set Key ID

Description:
Command Format: ATW2=Key_ID<CR>
Default Value: None
Response: None

3.2.2.5 'ATWC' Wlan Join a Network

Description:
Command Format: ATWC<CR>
Default Value: None
Response: Status of connection to the AP

3.2.2.6 'ATWD' Wlan Disconnect from Network

Description:
Command Format: ATWD<CR>
Default Value: None
Response: `WIFI disconnect succeed`

3.2.2.7 'ATW3' Wlan Set Access Point SSID

Description:
Command Format: ATW3=AP_SSID<CR>
Default Value: None
Response: None

3.2.2.8 'ATW4' Wlan Set Access Point Security Key

Description:
Command Format: ATW4=key<CR>
Default Value: None
Response: None

3.2.2.9 'ATW5' Wlan Set Access Point Channel

Description:
Command Format: ATW5=channel<CR>
Default Value: None
Response: None

3.2.2.10 'ATW6' Wlan Set BSSID

Description:
Command Format: ATW6=BSSID<CR>
BSSID format as XX:XX:XX:XX:XX:XX
Default Value: None
Response: None

3.2.2.11 'ATW7' Wlan Set Security

Description: 0=RTW_SECURITY_OPEN
1=RTW_SECURITY_WEP_PSK
2=RTW_SECURITY_WPA2_TKIP_PSK
3=RTW_SECURITY_WPA2_AES_PSK
Command Format: ATW7=0/1/2/3<CR>
Where 0 corresponds to open, 1 to WEP, 2 to TKIP and 3 to AES
Default Value: None
Response: None

For z2/z2plus:

Description: 0=RTW_SECURITY_OPEN
1=RTW_SECURITY_WEP_PSK
2=RTW_SECURITY_WPA2_AES_PSK
Command Format: ATW7=0/1/2<CR>
Where 0 corresponds to open, 1 to WEP, 2 to WPA2 AES
Default Value: None
Response: None

3.2.2.12 'ATWA' Wlan Activate Access Point

Description:
Command Format: ATWA<CR>
Default Value: None
Response: Status of AP Activated

3.2.2.13 'ATWB' Wlan Activate Access Point mode and Station mode

Description:
Command Format: ATWB<CR>
Default Value: None
Response: Status of AP & Station mode

3.2.2.14 'ATW?' Wlan Show WiFi information

Description:
Command Format: ATW?<CR>
Default Value: None
Response: Current connection status

3.2.2.15 'ATWS' Wlan Scan for Network Access Point

Description:
Command Format: ATWS<CR>
ATWS=num_channels[channel1, channel2,…]
Default Value: None
Response: List of AP scanned

3.2.2.16 'ATWs' Wlan Scan with SSID/Channel

Description: WIFI scan with SSID, it can also scan partial channel usage with channel number included
Command Format: ATWs=<BUFFER_LENGTH><CR>
ATWs= num_of_channels, [channel1, channel2,…]<CR>
Default Value: BUFFER_LENGTH = 500
Response:
Note: To use this command, please set **SCAN_WITH_SSID to 1** in atcmd_wifi.c
Before this command, use ATW0 to set the SSID to scan.

3.2.2.17 'ATWR' Wlan Get RSSI of Associated Network Access Point

Description:
Command Format: ATWR <CR>
Default Value: None
Response: wifi_get_rssi: rssi = <value>

3.2.2.18 'ATWM' Wlan Wi-Fi promisc

Description:
Command Format: ATWM=DURATION_SECONDS [with_len]<CR>
Default Value: None
Response: Enters promisc mode for DURATION_SECONDS

3.2.2.19 'ATWQ' Wlan Wi-Fi Simple Config

Description:
Command Format: ATWQ=pin_code<CR>
Default Value: None
Response: Enters Simple Config mode

3.2.2.20 'ATWP' Wlan Power on/off wifi module

Description:
Command Format: ATWP=0/1<CR>
Default Value: 1
Response: Wifi module Activated/Deactivated

WiFi Power	
Off	0
On	1

3.2.2.21 'ATWI' Wlan ping test

Description:
Command Format: ATWI=[host],[options]<CR>
-t Ping the specified host until stopped
-n # Number of echo requests to send (default 4 times)
-l # Send buffer size (default 32 bytes)
Default Value: Number of echo requests is 4 times
Send buffer size is 32 bytes
Response: Response of pinging a host

3.2.2.22 'ATWO' Wlan OTA update

Description:
Command Format: ATWO=IP[PORT] <CR>
ATWO= REPOSITORY[FILE_PATH]<CR>
Default Value: None
Response: Log of OTA receiving data and performing check sum

3.2.2.23 'ATWT' Wlan TCP throughput test

Description:

Command Format: ATWT=[-s|-c,host|stop],[options] <CR>

Client/Server:

stop	terminate client & server
-p #	server port to listen on/connect to (default 5001)

Server specific:

-s	run in server mode

Client specific:

-c <host>	run in client mode, connecting to <host>
-t #	time in seconds to transmit for (default 10 secs)
-n #[KM]	number of bytes to transmit (instead of -t)

Default Value: Port is 5001

Time is 10 seconds

Response: Response of a throughput test from the host

Note: To use this command, please set **`CONFIG_BSD_TCP` to 1** in `platform_opts.h`

3.2.2.24 'ATWU' Wlan UDP test

Description:

Command Format: ATWU=[-s|-c,host|stop][options] <CR>

Client/Server:

stop	terminate client & server
-p #	server port to listen on/connect to (default 5001)

Server specific:

-s	run in server mode

Client specific:

-b #[KM]	for UDP, bandwidth to send at in bits/sec
-c <host>	run in client mode, connecting to <host>
-t #	time in seconds to transmit for (default 10 secs)
-n #[KM]	number of bytes to transmit (instead of -t)

Default Value: Port is 5001

Time is 10 seconds

Bandwidth is 1Mbit/sec

Response: UDP Server Start & Log of throughput test receiving data

UDP Client Start & Log of throughput test sending data

Note: To use this command, please set **`CONFIG_BSD_TCP` to 1** in `platform_opts.h`

3.2.2.25 'ATWL' Wlan SSL client

Description: The parentheses "[]" is required to define user name and password if needed
Command Format: ATWL=SSL_SERVER_HOST[SRP_USER_NAME,SRP_PASSWORD]<CR>
Default Value: None
Response: Connection to SSL server
Note: To use this command, please set **`CONFIG_SSL_CLIENT` to 1** in `platform_opts.h`

3.2.2.26 'ATWW' Wlan Wi-Fi Protected Setup

Description:
Command Format: ATWW=pbc/pin<CR>
Default Value: None
Response: WPS scan done!
Note: To use this command, please set **`CONFIG_ENABLE_WPS` to 1** in `platform_opts.h`

3.2.2.27 'ATWZ' Wlan IWPRIV

Description:
Command Format: ATWZ=command[parameter]<CR>
Default Value: None
Response: Command response

3.2.2.28 'ATXP' Wlan Power Saving Control

Description: Provide detail setting of wlan power saving. Please note that setting other than ips and lps are note effect immediately. 'tdma' and 'dtim' only works after next time enter LPS.
Command Format: ATXP=ips[ips_mode]<CR>
ips_mode: 0:off, 1:on (default)
ATXP =lps[lps_mode]<CR>
lps_mode: 0:off, 1:legacy (default), 3:tdma
ATXP =tdma[slot_period,rf_on_len_1, rf_on_len_3, rf_on_len_3]
ATXP =dtim[dtim_value]<CR>
Default Value: None
Response: Power Mode <on/off>

3.2.2.29 'ATWY' Get SNR value

Description: Get the current SNR value
Command Format: ATWY<CR>
Default Value: None
Response: Snr value
Note: Not available for AmebaD

3.2.3 System

3.2.3.1 'ATSC' System Clear OTA Signature

Description: Clear OTA signature so that boot code load default image.
Command Format: ATSC<CR>
Default Value: None
Response: None

3.2.3.2 'ATSL' System wakelock control

Description: In FreeRTOS tickless mode, we can check and control wakelock status
Command Format: ATSL=a,acquire_wakelock_bitmap<CR>
Acquire wakelock on the bitmap provided
ATSL=r,release_wakelock_bitmap<CR>
Release wakelock on the bitmap provided
ATSL=?<CR>
Query current wakelock bitmap value
ATSL=c<CR>
Clear the statics and recalculate again
Default Value: None
Response: None
Note: Please set **configUSE_TICKLESS_IDLE** to 1, **configUSE_CUSTOMIZED_TICKLESS_IDLE** to 1 and **configUSE_WAKELOCK_PMU** to 1 in **FreeRTOSConfig.h**

3.2.3.3 'ATSR' System Recover OTA Signature

Description:	Recover OTA signature so that boot code load upgraded image(ota image).
Command Format:	ATSR<CR>
Default Value:	None
Response:	None

3.2.3.4 'ATSK' Set RDP/RSIP or secure boot

Description:	Set RDP/RSIP enable and key for AmebaZ, secure boot for AmebaZ2. To enable secure boot, the user needs to define enc_key, hash_key. Refer to AN0500.
Command Format:	ATSK=ENC_KEY[value(string)]<CR> ATSK=HASH_KEY[value(string)]<CR> ATSK=SB_KEY[value(string)]<CR> ATSK=ROOT_KEY [value(string)]<CR> ATSK=SEC_BOOT_EN[offetset,offset,length]<CR> ATSK=CHECK_MAC <CR>
Default Value:	None
Response:	None

3.2.3.5 'ATSG' MP GPIO test

Description:	PA_7 ~ PA_12 only available on RTL8720CF, PA_0 ~ PA_4 should not be tested when they are used as JTAG/SWD. PA_15 ~ PA_16 or PA_0 ~ PA_1 or PA_2 ~ PA_3 or PA_13 ~ PA_14 should not be tested when they are used UART RX and TX. PA_5, PA_6, PA_21 and PA_22 cannot be tested
Command Format:	ATSG=Pinname (eg A17) <CR>
Default Value:	None
Response:	Pinname=1/-1

3.2.3.6 'ATSS' show CPU status

Description:	This command allows use to read the status of the CPU
Command Format:	ATSS <CR>
Default Value:	None
Response:	None
Note:	To use this command, please set **configGENERATE_RUN_TIME_STATS** to 1 in **FreeRTOSConfig.h** and **FreeRTOSConfig_AFQP.h**

3.2.3.7 'ATS@' Debug message setting

Description:	Level [AT_DBG_OFF = 0, AT_DBG_ALWAYS = 1, AT_DBG_ERROR = 2, AT_DBG_WARNING = 3, AT_DBG_INFO = 4] Flag [AT_FLAG_DUMP = 1, AT_FLAG_EDIT = 2, AT_FLAG_ADC = 4, AT_FLAG_GPIO = 8, AT_FLAG_OTA = 10, AT_FLAG_NFC = 20, AT_FLAG_OS =40, AT_FLAG_LWIP = 80, AT_FLAG_COMMON = 100, AT_FLAG_WIFI = 200, AT_FLAG_RDP = 400]
Command Format:	ATS@=level.flag <CR>
Default Value:	`level = 0, flag = 0x00000000`
Response:	Return Level and Flag set for that level
Note:	Each bit represents 1 flag. You can set multiple flags at a time. For Example, flag=F will set AT_DBG_OFF, AT_DBG_ALWAYS, AT_DBG_ERROR, AT_DBG_WARNING

3.2.3.8 'ATS!' Debug config setting

Description:	Config[LEVEL_ERROR = 0, LEVEL_INFO = 1, LEVEL_WARN = 2] Flag [AT_FLAG_DUMP = 1, AT_FLAG_EDIT = 2, AT_FLAG_ADC = 4, AT_FLAG_GPIO = 8, AT_FLAG_OTA = 10, AT_FLAG_NFC = 20, AT_FLAG_OS =40, AT_FLAG_LWIP = 80, AT_FLAG_COMMON = 100, AT_FLAG_WIFI = 200, AT_FLAG_RDP = 400]
Command Format:	ATS!=config,flag <CR>

Default Value:	`ConfigDebugErr  = 0xFFFFFFFF`
	`ConfigDebugInfo = 0x00000000`
	`ConfigDebugWarn = 0x00000000`

Response:	Return flag for each debug level
Note:	Each bit represents 1 flag. You can set multiple flags at a time. For Example, flag=F will set AT_DBG_OFF, AT_DBG_ALWAYS, AT_DBG_ERROR, AT_DBG_WARNING

3.2.3.9 'ATS?' Display SW version and compile time

Description:	Returns Compile time and Software Version
Command Format:	ATS? <CR>
Default Value:	None
Response:	SW version and compile time

3.2.4 Socket AT Command

To use this set of commands, please set **CONFIG_TRANSPORT** to 1 in platform_opts.h

3.2.4.1 'ATP1' Set Transport Protocol

Description: 0 = TCP Enabled; 1 = UDP Enabled
Command Format: ATP1=1<CR>
Default Value: 0
Response: None

3.2.4.2 'ATP2' Set Transport Local Port Number

Description: Define the local port that the wifi module will listen on for Transport communication connections
Command Format: ATP2=LOCAL_PORT_NO<CR>
Default Value: 0
Response: None

3.2.4.3 'ATP3' Set Transport Remote Host Port IP Address

Description: Used to contact a Transport server on the network
Command Format: ATP3=xxx.xxx.xxx.xxx<CR>
Default Value: None
Response: None

3.2.4.4 'ATP4' Set Transport Remote Port Number

Description: Define the port number for a Transport Server on the network that the wifi module will user for communications with that server
Command Format: ATP4=REMOTE_PORT_NO<CR>
Default Value: 0
Response: None

3.2.4.5 'ATP5' Stop/Start Transport Server

Description: Start or stop the wifi module's Transport Server mode.
0=Disable; 1=Enable
Command Format: ATP5=1<CR>
Default Value: None
Response: None

3.2.4.6 'ATP6' Stop/Start Transport Client

Description: Start or stop the wifi module's Transport Client mode.
0=Disable; 1=Enable
Command Format: ATP6=1<CR>
Default Value: None
Response: None

3.2.4.7 'ATP?' Show Transport Settings

Description: Return current Transport Communication Settings.
Command Format: ATP?<CR>
Default Value: None
Response: Protocol; Local IP; Local Port; Remote IP; Remote Port

3.2.4.8 'ATR0' Read Transport Data

Description: Read the available receive data
Command Format: ATR0<CR>
Default Value: None
Response: DATA RECEIVING

3.2.4.9 'ATR1' Set Read Transport Packet Size

Description: Defined value for the packet size of data to return a data read
Command Format: ATR1=PACKET_SIZE <CR>
Default Value: None
Response: None

3.2.4.10 'ATRA' Write Transport Data

Description: Write data to a Transport Server or Client
Command Format: ATRA=[DATA]<CR>
Default Value: None
Response: None

3.2.4.11 'ATRB' Set Write Transport Packet Size

Description:	Define the packet size of data to write to a connected Transport Server or Client.
Command Format:	ATRB=PACKET_SIZE<CR>
Default Value:	None
Response:	None

3.2.4.12 'ATPE' Set static IP, gateway and mask for STA

Description:	Update the STA ip gateway and mask
Command Format:	ATPE=ip,gateway,mask
Default Value:	None
Response:	None
Note:	To use this command, please set **WIFI_LOGO_CERTIFICATION_CONFIG** to 1 in lwipopts.h file Can use ATW? Command to check that the ip, gateway and mask has been updated

3.2.4.13 'ATMF' Set station PMF

Description:	Supported PMF mode= none, optional, required
Command Format:	ATMF=[PMF mode]<CR>
Default Value:	Optional
Response:	None
Note:	To use this command, please set **WIFI_LOGO_CERTIFICATION_CONFIG** to 1 & **CONFIG_IEEE80211W** to 1 in autoconf.h file

4 Common AT command

4.1 help

The help command can be used to get description and usage of supported commands.

```
# help

WLAN AT COMMAND SET:
============================
1. Wlan Scan for Network Access Point
   # ATWS
2. Connect to an AES AP
   # ATW0=SSID
   # ATW1=PASSPHRASE
   # ATWC
3. Create an AES AP
   # ATW3=SSID
   # ATW4=PASSPHRASE
   # ATW5=CHANNEL
   # ATWA
4. Ping
   # ATWI=xxx.xxx.xxx.xxx
[MEM] After do cmd, available heap 42752
```

4.2 Log history

The “AT??” command prints history of commands which have been made, in order to confirm command information as expected.

```
# AT??
#AT?? match AT??, search cnt 1
[AT]log history:

  ATW3=realtek
  ATW5=1
  ATWA
  ATW?

[MEM] After do cmd, available heap 47896
```

4.3 Exit

The “AT--” command makes leaving from UART interactive mode. The stack used by interactive task is released to get more memory.

```
# AT--
AT-- match AT--, search cnt 1

Leave LOG SERVICE
```

5 WIFI AT Command Usage

UART interactive mode provides some commands to control Wi-Fi. Users can also implement their commands and add them into command table. The following is the description of built-in commands.

5.1 Disable/Enable WI-FI

The “ATWP=0/1” commands are used to initialize and de-initialize Wi-Fi driver correspondingly. Before using the functionality of Wi-Fi driver, it needs to be initialized. After Wi-Fi driver is initialized, it will be in station mode. The following are the output when executing “ATWP” commands.

```
# ATWP=0
ATWP match ATWP, search cnt 1
[ATWP]: _AT_WLAN_POWER_[OFF]

LwIP_DHCP: dhcp stop.
Deinitializing WIFI ...lextra_bus_dma_Interrupt(80)

WIFI deinitialized
[MEM] After do cmd, available heap 89080
```

```
# ATWP=1
ATWP match ATWP, search cnt 1
[ATWP]: _AT_WLAN_POWER_[ON]

reg 002: 0x3 WIFI ...
reg 01F: 0xea
reg 0b0: 0x0
reg 0b4: 0x0
reg 11c: 0

[_freertos_usleep_os] _freertos_usleep_os: Please Implement micro-second delay

WIFI initialized
[MEM] After do cmd, available heap 47264
```

5.2 Network Connection

The “ATWC” command can be used to connect to an access point. To process the connection, an SSID should be set first. Meanwhile a password must be set except in open mode, and a key id is also required for WEP mode.

To disconnect AP, type “ATWD”.

WPA2 mode

Command sequence: (refer to 3.2.1)

#ATW0=SSID
#ATW1=passphrase
#ATWC

```
# ATW0=rtk
ATW0 match ATW0, search cnt 2
[ATW0]: _AT_WLAN_SET_SSID_ [rtk]

[MEM] After do cmd, available heap 47264


# ATW1=12345678
ATW1 match ATW1, search cnt 1
[ATW1]: _AT_WLAN_SET_PASSPHRASE_ [12345678]

[MEM] After do cmd, available heap 47264


# ATWC
ATWC match ATWC, search cnt 2
[ATWC]: _AT_WLAN_JOIN_NET_

Joining BSS ...RTL8195A[Driver]: set ssid [rtk]
RTL8195A[Driver]: start auth
RTL8195A[Driver]: auth success, start assoc
RTL8195A[Driver]: association success(res=2)

wifi_handshake_done_hdl 31
CCConnected after 1261ms.
RTL8195A[Driver]: set group key to hw: alg:4(WEP40-1 WEP104-5 TKIP-2 AES-4) keyid:1
RTL8195A[Driver]: set pairwise key to hw: alg:4(WEP40-1 WEP104-5 TKIP-2 AES-4)

IP address : 192.168.1.100

GGGot IP after 2782ms.


[MEM] After do cmd, available heap 46616
```

#ATWD

```
# ATWD
ATWD match ATWD, search cnt 1
[ATWD]: _AT_WLAN_DISC_NET_

Deassociating AP ...
ioctl[SIOCGIWESSID] ssid = NULL, not connected
WIFI disconnected

[MEM] After do cmd, available heap 47376
```

WEP mode

Command sequence: (refer to 3.2.1)

#ATW0=SSID
#ATW1=Password
#ATW2=Key id
#ATWC

The WEP key can be 5 ASCII characters for WEP 40 or 13 ASCII characters for WEP 104. The key ID should be 0, 1, 2 or 3. The following is an example to connect network by using WEP 40 with key ID 0.

```
# ATW0=rtk
ATW0 match ATW0, search cnt 2
[ATW0]: _AT_WLAN_SET_SSID_ [rtk]

[MEM] After do cmd, available heap 47480


# ATW1=12345
ATW1 match ATW1, search cnt 1
[ATW1]: _AT_WLAN_SET_PASSPHRASE_ [12345]

[MEM] After do cmd, available heap 47480


# ATW2=0
ATW2 match ATW2, search cnt 2
[ATW2]: _AT_WLAN_SET_KEY_ID_ [0]

[MEM] After do cmd, available heap 47480


# ATWC
ATWC match ATWC, search cnt 2
[ATWC]: _AT_WLAN_JOIN_NET_

Joining BSS ...RTL8195A[Driver]: set ssid [rtk]
RTL8195A[Driver]: set group key to hw: alg:1(WEP40-1 WEP104-5 TKIP-2 AES-4) keyid:0
RTL8195A[Driver]: start auth
RTL8195A[Driver]: auth success, start assoc
RTL8195A[Driver]: association success(res=1)

wifi_connected_hdl 31
CCConnected after 1286ms.

IP address : 192.168.1.100

GGGot IP after 1801ms.


[MEM] After do cmd, available heap 46616
```

5.3 Wi-Fi Information

The “ATW?” command can be used to get the information of Wi-Fi driver, including some Wi-Fi statistic, setting, status and memory usage. The following is an example of the output of “ATW?” command when Wi-Fi is disabled. The Wi-Fi status information shows nothing about the Wi-Fi module.

```
# ATW?
ATW? match ATW?, search cnt 1
[ATW?]: _AT_WLAN_INFO_

[MEM] After do cmd, available heap 102752
```

The following is the output of “ATW?” command when Wi-Fi driver is enabled and disconnected. The Wi-Fi status shows the Wi-Fi driver is running without SSID connected. The wlan statistic includes the memory usage that wlan heap used.

```
# ATW?
ATW? match ATW?, search cnt 1
[ATW?]: _AT_WLAN_INFO_


WIFI wlan0 Status: Running
==============================
[rltk_wlan_statistic] tx stat: tx_packets=4, tx_dropped=0, tx_bytes=884
[rltk_wlan_statistic] rx stat: rx_packets=10, rx_dropped=10, rx_bytes=4186
[rltk_wlan_statistic] min_free_heap_size=46096, current heap free size=47480
[rltk_wlan_statistic] max_skbbuf_used_num=20, skbbuf_used_num=16
[rltk_wlan_statistic] max_skbdata_used_num=20, skbdata_used_num=16
[rltk_wlan_statistic] max_timer_used_num=7
ioctl[SIOCGIWESSID] ssid = NULL, not connected

WIFI  wlan0 Setting:
==============================
      MODE => STATION
      SSID =>
   CHANNEL => 3
  SECURITY => OPEN
  PASSWORD =>

Interface (wlan0)
==============================
        MAC => 00:e0:4c:87:00:00
        IP  => 192.168.1.100
        GW  => 192.168.1.254

[MEM] After do cmd, available heap 47480
```

The following is the output of “ATW?” Command when Wi-Fi is connected. Wi-Fi setting shows the Wi-Fi driver is in station mode and connecting to a SSID. The connection information in Wi-Fi setting also includes current channel and security.

```
# ATW?
ATW? match ATW?, search cnt 1
[ATW?]: _AT_WLAN_INFO_


WIFI wlan0 Status: Running
==============================
[rltk_wlan_statistic] tx stat: tx_packets=4, tx_dropped=0, tx_bytes=884
[rltk_wlan_statistic] rx stat: rx_packets=2, rx_dropped=2, rx_bytes=1236
[rltk_wlan_statistic] min_free_heap_size=46096, current heap free size=46616
[rltk_wlan_statistic] max_skbbuf_used_num=20, skbbuf_used_num=16
[rltk_wlan_statistic] max_skbdata_used_num=20, skbdata_used_num=16
[rltk_wlan_statistic] max_timer_used_num=7

WIFI  wlan0 Setting:
==============================
      MODE => STATION
      SSID => rtk
   CHANNEL => 3
  SECURITY => WEP
 KEY INDEX => 0
  PASSWORD =>

Interface (wlan0)
==============================
        MAC => 00:e0:4c:87:00:00
        IP  => 192.168.1.100
        GW  => 192.168.1.254

[MEM] After do cmd, available heap 46616
```

5.4 Start AP

The Wi-Fi driver can be switched from station mode to AP mode. The wifi_ap command can be used to start a Wi-Fi AP with indicated SSID, channel and password. If password is not given, this command starts AP in open mode. Otherwise, it starts AP with WPA2 security.

Command sequence: (refer to 3.2.1)

#ATW3=SSID
#ATW4=Password (no need for OPEN mode)
#ATW5=Channel
#ATWA

```
# ATW3=bonjour
ATW3 match ATW3, search cnt 2
[ATW3]: _AT_WLAN_AP_SET_SSID_ [bonjour]

[MEM] After do cmd, available heap 47480


# ATW5=1
ATW5 match ATW5, search cnt 1
[ATW5]: _AT_WLAN_AP_SET_CHANNEL_ [channel 1]

[MEM] After do cmd, available heap 47480


# ATWA
ATWA match ATWA, search cnt 1
[ATWA]: _AT_WLAN_AP_ACTIVATE_

LwIP_DHCP: dhcp stop.
Deinitializing WIFI ...lextra_bus_dma_Interrupt(80)

WIFI deinitialized
reg 002: 0x3 WIFI ...
reg 01F: 0xea
reg 0b0: 0x0
reg 0b4: 0x0
reg 11c: 0

[_freertos_usleep_os] _freertos_usleep_os: Please Implement micro-second delay

WIFI initialized
Starting AP ...
bonjour started

[MEM] After do cmd, available heap 47840
```

The following is the output of "ATW?" command when AP mode. The Wi-Fi setting shows the Wi-Fi driver is operating in AP mode with SSID, channel, security.

```
# ATW?
ATW? match ATW?, search cnt 1
[ATW?]: _AT_WLAN_INFO_


WIFI wlan0 Status: Running
==============================
[rltk_wlan_statistic] tx stat: tx_packets=0, tx_dropped=0, tx_bytes=0
[rltk_wlan_statistic] rx stat: rx_packets=0, rx_dropped=0, rx_bytes=0
[rltk_wlan_statistic] min_free_heap_size=46936, current heap free size=47896
[rltk_wlan_statistic] max_skbbuf_used_num=17, skbbuf_used_num=16
[rltk_wlan_statistic] max_skbdata_used_num=17, skbdata_used_num=16
[rltk_wlan_statistic] max_timer_used_num=8

WIFI  wlan0 Setting:
==============================
      MODE => AP
      SSID => bonjour
   CHANNEL => 1
  SECURITY => OPEN
  PASSWORD =>

Interface (wlan0)
==============================
        MAC => 00:e0:4c:87:00:00
        IP  => 192.168.1.1
        GW  => 192.168.1.1

[MEM] After do cmd, available heap 47896
```

To switch back from AP to STA mode, set Wi-Fi connection command set (refer to 5.2).

5.5 Start STA+AP

The Wi-Fi driver can start station mode and AP mode concurrently. The "ATWB" command can be used to start a Wi-Fi AP with indicated SSID, channel and password and start a station mode together. If password is not given, this command starts AP in open mode. Otherwise, it starts AP with WPA2 security. And the Wi-Fi connection command set (refer to 5.2) is used to connect with an AP.

Note :

(1) The MAC address of AP will be the MAC address of station plus one.

For example : If the MAC address of station is 00:e0:4c:87:00:00, the MAC address of AP will be 00:e0:4c:87:00:01.

(2) In AP mode or Concurrent mode, when Ameba doing wifi scan, WPS, the client which connects to Ameba AP might disconnect cause of switching channel in the process.

Command sequence: (refer to 3.2.1)

Start AP:
#ATW3=SSID
#ATW4=Password (no need for OPEN mode)
#ATW5=Channel
#ATWB
Connect to an AP:
#ATW0=SSID
#ATW1=Password
#ATW2=Key_id(only needed for WEP mode)
#ATWC

5.6 Ping

The “ATWI” command continues sending 4 ping packets, each in one second, to an indicated IP address. Please note that if DHCP client is not enabled, it is required to pre-configured default IP in main.h. It is useful when testing the network connection.

```
#ATWI=169.254.0.103
[ATWI]: _AT_WLAN_PING_TEST_

[ping_test] PING 169.254.0.103 32(60) bytes of data

[ping_test] 20 bytes from 169.254.0.103: icmp_seq=1 time=43 ms
[ping_test] 20 bytes from 169.254.0.103: icmp_seq=2 time=22 ms
[ping_test] 20 bytes from 169.254.0.103: icmp_seq=3 time=179 ms
[ping_test] 20 bytes from 169.254.0.103: icmp_seq=4 time=26 ms
[MEM] After do cmd, available heap 62032
```

To ping [x] packets, type “ATWI=[host],-n,[x]”

```
#ATWI=169.254.0.103,-n,2
[ATWI]: _AT_WLAN_PING_TEST_

[ping_test] PING 169.254.0.103 32(60) bytes of data

[ping_test] 20 bytes from 169.254.0.103: icmp_seq=1 time=19 ms
[ping_test] 20 bytes from 169.254.0.103: icmp_seq=2 time=25 ms
[MEM] After do cmd, available heap 62032
```

To ping continuously, type “ATWI=[host],-t”. Please note that currently, exiting infinite ping loop by UART command is not supported yet.

```
#ATWI=169.254.0.103,-t
[ATWI]: _AT_WLAN_PING_TEST_

[ping_test] PING 169.254.0.103 32(60) bytes of data

[ping_test] 20 bytes from 169.254.0.103: icmp_seq=1 time=669 ms
[ping_test] 20 bytes from 169.254.0.103: icmp_seq=2 time=43 ms
[ping_test] 20 bytes from 169.254.0.103: icmp_seq=3 time=278 ms
[ping_test] 20 bytes from 169.254.0.103: icmp_seq=4 time=104 ms
[ping_test] 20 bytes from 169.254.0.103: icmp_seq=5 time=415 ms
[ping_test] 20 bytes from 169.254.0.103: icmp_seq=6 time=13 ms
[ping_test] 20 bytes from 169.254.0.103: icmp_seq=7 time=417 ms
[ping_test] 20 bytes from 169.254.0.103: icmp_seq=8 time=209 ms
[ping_test] 20 bytes from 169.254.0.103: icmp_seq=9 time=843 ms
[ping_test] 20 bytes from 169.254.0.103: icmp_seq=10 time=296 ms
[ping_test] 20 bytes from 169.254.0.103: icmp_seq=11 time=221 ms
[ping_test] 20 bytes from 169.254.0.103: icmp_seq=12 time=304 ms
[ping_test] 20 bytes from 169.254.0.103: icmp_seq=13 time=30 ms
[ping_test] 20 bytes from 169.254.0.103: icmp_seq=14 time=198 ms
[ping_test] 20 bytes from 169.254.0.103: icmp_seq=15 time=7 ms
[ping_test] 20 bytes from 169.254.0.103: icmp_seq=16 time=305 ms
[ping_test] 20 bytes from 169.254.0.103: icmp_seq=17 time=325 ms
[ping_test] 20 bytes from 169.254.0.103: icmp_seq=18 time=516 ms
[ping_test] 20 bytes from 169.254.0.103: icmp_seq=19 time=717 ms
[ping_test] 20 bytes from 169.254.0.103: icmp_seq=20 time=316 ms
[ping_test] 20 bytes from 169.254.0.103: icmp_seq=21 time=212 ms
[ping_test] 20 bytes from 169.254.0.103: icmp_seq=22 time=104 ms
```

To set sending buffer size [x] bytes, type “ATWI=[host],-l,[x]”.

```
#ATWI=169.254.0.103,-l,128
[ATWI]: _AT_WLAN_PING_TEST_

[ping_test] PING 169.254.0.103 128(156) bytes of data

[ping_test] 116 bytes from 169.254.0.103: icmp_seq=1 time=11 ms
[ping_test] 116 bytes from 169.254.0.103: icmp_seq=2 time=46 ms
[ping_test] 116 bytes from 169.254.0.103: icmp_seq=3 time=10 ms
[ping_test] 116 bytes from 169.254.0.103: icmp_seq=4 time=182 ms
[MEM] After do cmd, available heap 62032
```

5.7 TCP RX/TX Throughput Test

TCP transmit and receive throughput can be measured by iperf.exe tool which you can get from $sdk/tools/iperf.exe.

5.7.1 Receive Throughput Test

Receive test measures receive throughput of the development board. Start TCP server in the development board, listen to port 5001 and wait for connection from iperf client. Iperf on the Windows platforms connects to the TCP server via AP and transmits data to it. Iperf client running on the Windows platforms computes bytes of data transmitted, and prints it out every 1 second. A sample session is illustrated as bellow:

Type the following command to start TCP server on the console of development board:

ATWT=-s

The “-s” command-line option starts a TCP server.

```
#ATWT=-s
[ATWT]: _AT_WLAN_TCP_TEST_

[MEM] After do cmd, available heap 60920


#
TCP: Start TCP server!
tcp_server_func: Create socket fd = 0
tcp_server_func: Bind socket successfully
tcp_server_func: Listen port 5001
```

Type the following command to start Iperf client on Windows platforms:

~:> iperf .exe –c 169.254.0.101 –i 1 –t 60 –w 256k

The “-c” command-line option means starting a TCP client and connecting to “169.254.0.101”, “-i” is seconds between periodic bandwidth reports, “-t” is time in seconds to transmit for (default 10 seconds).

```
C:\>iperf -c 169.254.0.101 -i 1 -t 60 -w 256k
------------------------------------------------------------
Client connecting to 169.254.0.101, TCP port 5001
TCP window size:  256 KByte
------------------------------------------------------------
[  3] local 169.254.0.100 port 61322 connected with 169.254.0.101 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 1.0 sec   512 KBytes  4.19 Mbits/sec
[  3]  1.0- 2.0 sec   128 KBytes  1.05 Mbits/sec
[  3]  2.0- 3.0 sec   256 KBytes  2.10 Mbits/sec
[  3]  3.0- 4.0 sec   128 KBytes  1.05 Mbits/sec
[  3]  4.0- 5.0 sec   256 KBytes  2.10 Mbits/sec
[  3]  5.0- 6.0 sec   128 KBytes  1.05 Mbits/sec
```

5.7.2 Transmit Throughput Test

Transmit test measures the transmission throughput of the development board. Start TCP Client in the development board and connect to Iperf server on the Windows platforms via AP. TCP client can set connect port and send packet total size with length 1460 one timet. Iperf server running on the Windows platforms computes bytes of data received, and prints it out every 1 second. A sample session is illustrated as below:

Type the following command to start Iperf server on Windows platforms:

~:> iperf.exe –s –i 1

The “-s” command-line option starts a TCP server, “-i” is seconds between periodic bandwidth reports.

```
C:\>iperf -s -i1
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size: 63.0 KByte (default)
------------------------------------------------------------
[  4] local 169.254.0.100 port 5001 connected with 169.254.0.101 port 49155
[ ID] Interval       Transfer     Bandwidth
[  4]  0.0- 1.0 sec  54.2 KBytes   444 Kbits/sec
[  4]  1.0- 2.0 sec  49.9 KBytes   409 Kbits/sec
[  4]  2.0- 3.0 sec  85.5 KBytes   701 Kbits/sec
[  4]  3.0- 4.0 sec  57.0 KBytes   467 Kbits/sec
[  4]  4.0- 5.0 sec  69.9 KBytes   572 Kbits/sec
[  4]  5.0- 6.0 sec  89.8 KBytes   736 Kbits/sec
[  4]  6.0- 7.0 sec  62.7 KBytes   514 Kbits/sec
[  4]  7.0- 8.0 sec  54.2 KBytes   444 Kbits/sec
[  4]  8.0- 9.0 sec  88.4 KBytes   724 Kbits/sec
[  4]  9.0-10.0 sec   124 KBytes  1.02 Mbits/sec
[  4] 10.0-11.0 sec  87.0 KBytes   712 Kbits/sec
[  4] 11.0-12.0 sec  49.9 KBytes   409 Kbits/sec
[  4] 12.0-13.0 sec  65.6 KBytes   537 Kbits/sec
[  4] 13.0-14.0 sec  87.0 KBytes   712 Kbits/sec
[  4]  0.0-14.0 sec  1.00 MBytes   599 Kbits/sec
```

Type the following command to start TCP client on the development board:

ATWT=–c,192.168.0.100,-n,1m

The “-c” command-line option starts a TCP client, “192.168.0.100” is IP address of the Windows platforms, the “–n” is to set transmit size, and the “1m” is the size of packets transmitted to Iperf Server.

```
#ATWT=-c,169.254.0.100,-n,1m
[ATWT]: _AT_WLAN_TCP_TEST_

[MEM] After do cmd, available heap 60920


#
TCP: Start TCP client!
tcp_client_func: Server IP=169.254.0.100, port=5001
tcp_client_func: Create socket fd = 0
tcp_client_func: Connect to server successfully
tcp_client_func: Send 1049740 Bytes packets
tcp_client_func: Close client socket
TCP: TCP client stopped!
```

Stop TCP test by typing the following command:

#ATWT=stop

```
#ATWT=stop
[ATWT]: _AT_WLAN_TCP_TEST_

[MEM] After do cmd, available heap 58944


#
tcp_server_func: Receive 1345784 Bytes packets
TCP: TCP server stopped!
```

5.7.3 Transmit and Receive Throughput Test

The concurrent throughput test measures receive and transmit throughput concurrently. The development board run “ATWT=-s” to start a TCP server and communicate with iperf client on Windows platform, run “ATWT= –c,169.254.0.100,-n,1m” to start a TCP client and communicate with iperf server on Windows platform. A sample session is illustrated as bellow:

Step 1: Start Iperf server on Windows platforms:

~:> iperf.exe -s -i 1

Step 2: Start TCP server on the development board:

ATWT=-s

Step 3: Start Iperf client on Windows platforms:

~:> iperf.exe –c 169.254.0.101 –i 1 –t 60 –w 256k

Step 4: Start TCP client on the development board:

ATWT=–c,169.254.0.100,-n,1m

```
#ATWT=-s
[ATWT]: _AT_WLAN_TCP_TEST_

[MEM] After do cmd, available heap 60920


#
TCP: Start TCP server!
tcp_server_func: Create socket fd = 0
tcp_server_func: Bind socket successfully
tcp_server_func: Listen port 5001
tcp_server_func: Accept connection successfully
#
#
#
#ATWT=-c,169.254.0.100,-n,1m
[ATWT]: _AT_WLAN_TCP_TEST_

[MEM] After do cmd, available heap 57832


#
TCP: Start TCP client!
tcp_client_func: Server IP=169.254.0.100, port=5001
tcp_client_func: Create socket fd = 2
tcp_client_func: Connect to server successfully
tcp_client_func: Send 200020 Bytes packets
tcp_client_func: Close client socket
TCP: TCP client stopped!
```

```
C:\>iperf -s -i1
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size: 63.0 KByte (default)
------------------------------------------------------------
[  4] local 169.254.0.100 port 5001 connected with 169.254.0.101 port 49155
[ ID] Interval       Transfer     Bandwidth
[  4]  0.0- 1.0 sec  54.2 KBytes   444 Kbits/sec
[  4]  1.0- 2.0 sec  49.9 KBytes   409 Kbits/sec
[  4]  2.0- 3.0 sec  85.5 KBytes   701 Kbits/sec
[  4]  3.0- 4.0 sec  57.0 KBytes   467 Kbits/sec
[  4]  4.0- 5.0 sec  69.9 KBytes   572 Kbits/sec
[  4]  5.0- 6.0 sec  89.8 KBytes   736 Kbits/sec
[  4]  6.0- 7.0 sec  62.7 KBytes   514 Kbits/sec
[  4]  7.0- 8.0 sec  54.2 KBytes   444 Kbits/sec
[  4]  8.0- 9.0 sec  88.4 KBytes   724 Kbits/sec
[  4]  9.0-10.0 sec   124 KBytes  1.02 Mbits/sec
[  4] 10.0-11.0 sec  87.0 KBytes   712 Kbits/sec
[  4] 11.0-12.0 sec  49.9 KBytes   409 Kbits/sec
[  4] 12.0-13.0 sec  65.6 KBytes   537 Kbits/sec
[  4] 13.0-14.0 sec  87.0 KBytes   712 Kbits/sec
[  4]  0.0-14.0 sec  1.00 MBytes   599 Kbits/sec
```

```
C:\>iperf -c 169.254.0.101 -i 1 -t 60 -w 256k
------------------------------------------------------------
Client connecting to 169.254.0.101, TCP port 5001
TCP window size:  256 KByte
------------------------------------------------------------
[  3] local 169.254.0.100 port 61322 connected with 169.254.0.101 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 1.0 sec   512 KBytes  4.19 Mbits/sec
[  3]  1.0- 2.0 sec   128 KBytes  1.05 Mbits/sec
[  3]  2.0- 3.0 sec   256 KBytes  2.10 Mbits/sec
[  3]  3.0- 4.0 sec   128 KBytes  1.05 Mbits/sec
[  3]  4.0- 5.0 sec   256 KBytes  2.10 Mbits/sec
[  3]  5.0- 6.0 sec   128 KBytes  1.05 Mbits/sec
```

5.8 UDP RX/TX Throughput Test

UDP transmit and receive throughput test can be performed with iperf tool on Windows platform and ATWU command on device.

5.8.1 Receive Throughput Test

The following is the ATWU command executed on device to start a UDP server for throughput test. When UDP client is transmitting data for throughput test, the throughput information will be shown per second.

```
#ATWU=-s
[ATWU]: _AT_WLAN_UDP_TEST_

[MEM] After do cmd, available heap 60920


#
UDP: Start UDP server!
udp_server_func: Create socket fd = 0, port = 5001
udp_server_func: Bind socket successfully
udp_server_func: Receive 8820 Bytes in 1051 ticks, 67 bits/sec
udp_server_func: Receive 294000 Bytes in 1050 ticks, 2240 bits/sec
udp_server_func: Receive 132300 Bytes in 1026 ticks, 1031 bits/sec
udp_server_func: Receive 213150 Bytes in 1003 ticks, 1700 bits/sec
udp_server_func: Receive 211680 Bytes in 1011 ticks, 1675 bits/sec
udp_server_func: Receive 318990 Bytes in 1002 ticks, 2546 bits/sec
udp_server_func: Receive 458640 Bytes in 1015 ticks, 3614 bits/sec
udp_server_func: Receive 637980 Bytes in 1191 ticks, 4285 bits/sec
udp_server_func: Receive 380730 Bytes in 1003 ticks, 3036 bits/sec
udp_server_func: Receive 570360 Bytes in 1001 ticks, 4558 bits/sec
udp_server_func: Receive 745290 Bytes in 1005 ticks, 5932 bits/sec
udp_server_func: Receive 568890 Bytes in 1006 ticks, 4523 bits/sec
udp_server_func: Receive 705600 Bytes in 1003 ticks, 5627 bits/sec
```

A UDP client on Windows platform should also be started with iperf command as the following. UDP client is transmitting data to the specified UDP server (169.254.0.101 is the IP address of server on device in this example) for throughput test based on the setting of transmit time and bandwidth in iperf command.

```
C:\>iperf -c 169.254.0.101 -u -t10 -i1 -b20m
------------------------------------------------------------
Client connecting to 169.254.0.101, UDP port 5001
Sending 1470 byte datagrams
UDP buffer size: 63.0 KByte (default)
------------------------------------------------------------
[  3] local 169.254.0.100 port 52611 connected with 169.254.0.101 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 1.0 sec  2.38 MBytes  20.0 Mbits/sec
[  3]  1.0- 2.0 sec  2.38 MBytes  20.0 Mbits/sec
[  3]  2.0- 3.0 sec  2.38 MBytes  20.0 Mbits/sec
[  3]  3.0- 4.0 sec  2.38 MBytes  20.0 Mbits/sec
[  3]  4.0- 5.0 sec  2.38 MBytes  20.0 Mbits/sec
[  3]  5.0- 6.0 sec  2.38 MBytes  20.0 Mbits/sec
[  3]  6.0- 7.0 sec  2.38 MBytes  20.0 Mbits/sec
[  3]  7.0- 8.0 sec  2.38 MBytes  20.0 Mbits/sec
[  3]  8.0- 9.0 sec  2.38 MBytes  20.0 Mbits/sec
[  3]  9.0-10.0 sec  2.42 MBytes  20.3 Mbits/sec
[  3]  0.0-10.0 sec  23.8 MBytes  20.0 Mbits/sec
[  3] Sent 17008 datagrams
[  3] WARNING: did not receive ack of last datagram after 10 tries.
```

5.8.2 X`Transmit Throughput Test

The following is the iperf command executed on Windows platform to start a UDP server for throughput test. When UDP client is transmitting data for throughput test, the throughput information will be shown per second.

```
C:\>iperf -s -u -i1
------------------------------------------------------------
Server listening on UDP port 5001
Receiving 1470 byte datagrams
UDP buffer size: 63.0 KByte (default)
------------------------------------------------------------
[  3] local 169.254.0.100 port 5001 connected with 169.254.0.101 port 49154
[ ID] Interval       Transfer     Bandwidth        Jitter   Lost/Total Datagrams
[  3]  0.0- 1.0 sec  44.2 KBytes   362 Kbits/sec  29.121 ms 66020/66051 (1e+02%)
[  3]  0.0- 1.0 sec  30 datagrams received out-of-order
[  3]  1.0- 2.0 sec  27.1 KBytes   222 Kbits/sec  41.353 ms    0/    0 (nan%)
[  3]  1.0- 2.0 sec  19 datagrams received out-of-order
[  3]  2.0- 3.0 sec  22.8 KBytes   187 Kbits/sec  59.100 ms    0/    0 (nan%)
[  3]  2.0- 3.0 sec  16 datagrams received out-of-order
[  3]  3.0- 4.0 sec  44.2 KBytes   362 Kbits/sec  26.356 ms    0/    0 (nan%)
[  3]  3.0- 4.0 sec  31 datagrams received out-of-order
```

A UDP client on device should also be started with ATWU command as the following. UDP client is transmitting data to the specified UDP server (169.254.0.100 is the IP address of server on Windows platform in this example) for throughput test based on the setting of buffer length and packet count in ATWU command.

```
#ATWU=-c,169.254.0.100,-n,1m
[ATWU]: _AT_WLAN_UDP_TEST_

[MEM] After do cmd, available heap 60920


#
UDP: Start UDP client!
udp_client_func: Server IP=, port=5001
udp_client_func: Create socket fd = 0
```

5.9 Wi-Fi Simple Config

This “ATWQ” command provides a simple way for device to associate to AP. For details, please refer to the document AN0011 Realtek wlan simple configuration.pdf.

5.10 Wi-Fi Protected Setup

The “ATWW” command provides another simple way for device to associate to AP. After pressing WPS button on the AP, execute “ATWW=pbc” in the command line, then the device will automatically associate with the AP. PIN method also supported. Please refer to the document AN0011 Realtek wlan simple configuration.pdf for more detail.

5.11Set MAC address

The ATWZ command can be used to read/write MAC address. There are two examples for reading and writing MAC address as below:

Read MAC address:
#ATWZ=read_mac

Write MAC address:
#ATWZ=write_mac[00e04c870102]

6 System AT Command Usage

6.1 Clear OTA Signature

Read back OTA signature value. The value of 81958711 at first time shows OTA image is *valid*. After clear the signature, read back OTA signature again and it is 00000000.

```
#ATSC

[ATSC]: _AT_SYSTEM_CLEAR_OTA_SIGNATURE_
OTA offset = 0x00044000
Signature = 81958711
Signature = 00000000
Clear OTA signature success.
```

6.2 Restore OTA Signature

Read back OTA signature value. The value of 00000000 at first time shows OTA image is *invalid*. After set OTA signature to valid, (that is, 81958711), write this value to flash and read back again for double check.

```
#ATSR

[ATSR]: _AT_SYSTEM_RECOVER_OTA_SIGNATURE_
OTA offset = 0x00044000
Signature = 00000000
Signature = 81958711
Recover OTA signature success.
```

7 Transporting AT Command Usage

To enable the Transporting AT Command, enable the #define CONFIG_TRANSPORT in platform_opt.h

Before using the transporting AT Command, please connect WiFi first.

7.1 Start/stop TCP server

The “ATP5=0/1” commands are used to start or stop the transport server. And before that, some necessary settings should be done.

ATP1=0<CR>(Set the protocol to TCP. The default value is also 0)

ATP2=LOCAL_PORT<CR>(Set the local port of the server)

ATP?(Can show the setting information)

```
#ATP1=0
[ATP1]: _AT_TRANSPORT_MODE_ [0]

[MEM] After do cmd, available heap 87136


# ATP2=5000
[ATP2]: _AT_TRANSPORT_LOCAL_PORT_ [5000]

[MEM] After do cmd, available heap 87136


# ATP?


The current Transport settings:
==============================
 Protocol: TCP
 LOCAL_IP   => 192.168.1.101
 LOCAL_PORT => 5000
 REMOTE_IP  =>
 REMOTE_PORT => 0

[MEM] After do cmd, available heap 87136
```

After done these, can use “ATP5=1<CR>” to start server.

```
# ATP5=1
[ATP5]: _AT_TRANSPORT_START_SERVER_ [1]

[MEM] After do cmd, available heap 78856


#
Start Server
        [IP]: 192.168.1.101
        [PORT]:5000


         The TCP SERVER START OK!
```

Then the Client can connect to this server and start communicating.

To Stop the TCP server, using the “ATP5=0<CR>”.

```
#ATP5=0
[ATP5]: _AT_TRANSPORT_START_SERVER_ [0]

[MEM] After do cmd, available heap 78856


#
```

7.2 Start/stop UDP server

The “ATP5=0/1” commands are used to start or stop the transport server. And before that, some necessary settings should be done.

ATP1=1<CR>(Set the protocol to UDP)

ATP2=LOCAL_PORT<CR>(Set the local port of the server)

ATP?(Can show the setting information)

```
#ATP1=1
[ATP1]: _AT_TRANSPORT_MODE_ [1]

[MEM] After do cmd, available heap 87136


# ATP2=5000
[ATP2]: _AT_TRANSPORT_LOCAL_PORT_ [5000]

[MEM] After do cmd, available heap 87136


# ATP?


The current Transport settings:
==============================
 Protocol: UDP
 LOCAL_IP   => 192.168.1.101
 LOCAL_PORT => 5000
 REMOTE_IP  =>
 REMOTE_PORT => 0

[MEM] After do cmd, available heap 87136


#
```

After done these, can use “ATP5=1<CR>” to start server.

```
#
#ATP5=1
[ATP5]: _AT_TRANSPORT_START_SERVER_ [1]

[MEM] After do cmd, available heap 78856


#
Start Server
        [IP]: 192.168.1.101
        [PORT]:5000


         The UDP SERVER START OK!

```

Then the Client can connect to this server and start communicating.

To Stop the TCP server, using the “ATP5=0<CR>”.

```
#ATP5=0
[ATP5]: _AT_TRANSPORT_START_SERVER_ [0]

[MEM] After do cmd, available heap 78856


#
```

7.3 Start/stop TCP client

The “ATP6=0/1” commands are used to start or stop the transport client. And before that, some necessary settings should be done.

ATP1=0<CR>(Set the protocol to TCP. The default value is also 0)

ATP3=REMOTE_IP<CR>(Set the remote IP address of the server)

ATP4=REMOTE_PORT<CR>(Set the remote port of the server)

ATP?(Can show the setting information)

```
#ATP1=0
[ATP1]: _AT_TRANSPORT_MODE_ [0]

[MEM] After do cmd, available heap 87136


# ATP3=192.168.1.100
[ATP3]: _AT_TRANSPORT_REMOTE_IP_ [192.168.1.100]

[MEM] After do cmd, available heap 87136


# ATP4=5000
[ATP4]: _AT_TRANSPORT_REMOTE_PORT_ [5000]

[MEM] After do cmd, available heap 87136


# ATP?


The current Transport settings:
==============================
 Protocol: TCP
 LOCAL_IP   => 192.168.1.101
 LOCAL_PORT => 0
 REMOTE_IP  => 192.168.1.100
 REMOTE_PORT => 5000

[MEM] After do cmd, available heap 87136


#
```

After setting these parameters and make sure they are correct, using “ATP6=1<CR>” to start client and connect to server.

```
# ATP6=1
[ATP6]: _AT_TRANSPORT_START_CLIENT_ [1]

[ATP6]TCP Client mode will start

[MEM] After do cmd, available heap 78856


#
        Start Client
        [IP]: 192.168.1.100
        [PORT]:5000


OK to create sock_fd!

Connect to Server successful!
```

Then this client can start communicating with server.

To Stop the TCP client, using the “ATP6=0<CR>”.

```
#ATP6=0
[ATP6]: _AT_TRANSPORT_START_CLIENT_ [0]

[MEM] After do cmd, available heap 87136


#
```

7.4 Start/stop UDP client

The “ATP6=0/1” commands are used to start or stop the transport client. And before that, some necessary settings should be done.

ATP1=1<CR>(Set the protocol to UDP.)

ATP3=REMOTE_IP<CR>(Set the remote IP address of the server)

ATP4=REMOTE_PORT<CR>(Set the remote port of the server)

ATP? <CR> (Can show the setting information)

```
#ATP1=1
[ATP1]: _AT_TRANSPORT_MODE_ [1]

[MEM] After do cmd, available heap 87136


# ATP3=192.168.1.100
[ATP3]: _AT_TRANSPORT_REMOTE_IP_ [192.168.1.100]

[MEM] After do cmd, available heap 87136


# ATP4=5000
[ATP4]: _AT_TRANSPORT_REMOTE_PORT_ [5000]

[MEM] After do cmd, available heap 87136


# ATP?


The current Transport settings:
==============================
 Protocol: UDP
 LOCAL_IP   => 192.168.1.101
 LOCAL_PORT => 0
 REMOTE_IP  => 192.168.1.100
 REMOTE_PORT => 5000

[MEM] After do cmd, available heap 87136


#
```

After setting these parameters and make sure they are correct, using “ATP6=1<CR>” to start client and prepare to connect to server.

```
#ATP6=1
[ATP6]: _AT_TRANSPORT_START_CLIENT_ [1]

[MEM] After do cmd, available heap 78856


#
        Start Client
        [IP]: 192.168.1.100
        [PORT]:5000


OK to create sock_fd!

Udp client setup Server's information successful!

#
```

Then this client can start communicating with server.

To Stop the UDP client, using the “ATP6=0<CR>”.

```
#ATP6=0
[ATP6]: _AT_TRANSPORT_START_CLIENT_ [0]

[MEM] After do cmd, available heap 87136

#
```

7.5 Receiving data

To receive the data send to this board, using “ATR0<CR>” command to complete. Before that, the “ATR1=PACKET_SIZE<CR>” decides the receiving packet size. If no need to set the packet and using the “ATR0<CR>” command directly, the packet size will be the MAX_BUFFER = 256.

In this example, server send to client “Hello Ameba”, then using ATR0, Ameba receive this string.

```
# ATR0
[ATR0]Notice: Didn't set the value of packet_size, will using the MAX_BUFFER: 256
[ATR0]Receive the data:Hello Ameba
 with packet_size: 256

[MEM] After do cmd, available heap 86904

#
```

7.6 Sending data

To send data to server or the client that already connected, using “ATRA=[DATA]<CR>” command to complete. Before that, the “ATRB=PACKET_SIZE<CR>” decides the sending packet size. If there’s no need to set the packet and using the “ATRA=[DATA]<CR>” command directly, the packet size will be the MAX_BUFFER = 100.

```
#ATRA=[This is a message to server]
[ATRA]: _AT_TRANSPORT_WRITE_DATA_ [This is a message to server]
[ATRA] Sending data:This is a message to server
 with packet_size:256

[MEM] After do cmd, available heap 86904

#
```

After doing this, the server will receive the message “This is a message to server”.