AT Command Application Note

This document provides information for controlling Ameba through external UART.

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	return	delimeter	payload
\r\n	OK	\r\n	Data
\r\n	Error type	\r\n	Usage

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
ATWA	Activate Access Point
ATWB	Start STA+AP
ATWC	Join a network
ATWD	Disconnect from a network
ATWE	Start web server
ATWI	Ping test
ATWL	SSL client
ATWM	Wlan Wi-Fi promisc
ATWP	Power on/off wifi module
ATWp	Power Saving control
ATWQ	Wi-Fi Simple Config
ATWR	Get RSSI of Associated Network Access Point
ATWS	Scan for Network Access Point
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
System	
ATSC	Clear OTA signature
ATSL	System wakelock control
ATSR	Recover OTA signature

3.2 AT command list

3.2.1 COMMON

3.2.1.1 'help' Print help message

Description: Print some commands description and usage
Command Format: AT??<CR>
Default Value: None
Response: TBD

3.2.1.2 'AT??' Print Log History

Description:
Command Format: AT??<CR>
Default Value: None
Response: TBD

3.2.1.3 'AT--' Exit Log Service

Description:
Command Format: AT--<CR>
Default Value: None
Response: TBD

3.2.2 WLAN

3.2.2.1 'ATW0' Wlan Set Network SSID

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

3.2.2.2 'ATW1' Wlan set Network Passphrase

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

3.2.2.3 'ATW2' Wlan Set Key ID

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

3.2.2.4 'ATWC' Wlan Join a Network

Description:
Command Format: ATWC<CR>
Default Value: None
Response: TBD

3.2.2.5 'ATWD' Wlan Disconnect from Network

Description:
Command Format: ATWD<CR>
Default Value: None
Response: TBD

3.2.2.6 'ATW3' Wlan Set Access Point SSID

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

3.2.2.7 'ATW4' Wlan Set Access Point Security Key

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

3.2.2.8 'ATW5' Wlan Set Access Point Channel

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

3.2.2.9 'ATWA' Wlan Activate Access Point

Description:
Command Format: ATWA<CR>
Default Value: None
Response: TBD

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

Description:
Command Format: ATWB<CR>
Default Value: None
Response: TBD

3.2.2.11 'ATW?' Wlan Show WiFi information

Description:
Command Format: ATW?<CR>
Default Value: None
Response: TBD

3.2.2.12 'ATWS' Wlan Scan for Network Access Point

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

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

Description:
Command Format: ATWR <CR>
Default Value: None
Response: TBD

3.2.2.14 'ATWM' Wlan Wi-Fi promisc

Description:
Command Format: ATWM=DURATION_SECONDS [with_len]<CR>
Default Value: None
Response: TBD

3.2.2.15 'ATWE' Wlan Start Web Server

Description:
Command Format: ATWE<CR>
Default Value: None
Response: TBD

3.2.2.16 'ATWQ' Wlan Wi-Fi Simple Config

Description:
Command Format: ATWQ=pin_code<CR>
Default Value: None
Response: TBD

3.2.2.17 'ATWP' Wlan Power on/off wifi module

Description:
Command Format: ATWP=0/1<CR>
Default Value: None
Response: TBD

WiFi Power	
Off	0
On	1

3.2.2.18 '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: TBD

3.2.2.19 'ATWO' Wlan OTA update

Description:
Command Format: ATWO=IP[PORT] <CR>
ATWO= REPOSITORY[FILE_PATH]<CR>
Default Value: None
Response: TBD

3.2.2.20 '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: TBD

3.2.2.21 '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: TBD

3.2.2.22 'ATWL' Wlan SSL client

Description:

Command Format: ATWL=SSL_SERVER_HOST<CR>

Default Value: None

Response: TBD

3.2.2.23 'ATWW' Wlan Wi-Fi Protected Setup

Description:
Command Format: ATWW=pbc/pin<CR>
Default Value: None
Response: TBD

3.2.2.24 'ATWZ' Wlan IWPRIV

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

3.2.2.25 '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: TBD

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]
Release wakelock on the bitmap provided
ATSL=?
Query current wakelock bitmap value

Default Value: None
Response: None

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

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.

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 print 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 print 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.

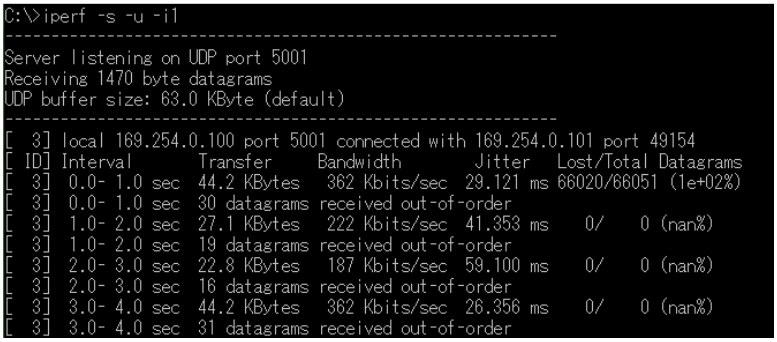
```
#ATWU=-s
[ATWU]: _AT_WLAN_UDP_TEST_

[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 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.



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.

5.9 Start Web Server

The “ATWE” command can be used to start webserver. Web server works only after Wi-Fi driver switched to AP mode or concurrent AP mode. After client associated with the AP and get right IP address, the client PC can open web browser and enter http://192.168.1.1 (http://192.168.1.1 in AP mode or http://192.168.43.1 in concurrent AP mode) to get or set AP settings. For details, please refer to the document UM0014 Realtek web server user guide.pdf.

5.10 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.

5.11 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.

5.12 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.

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

5.13Set 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.
```