RALINK TECHNOLOGY, CORP.

RALINK RT2860 LINUX SOFTAP RELEASE NOTES & USER'S GUIDE

PCI/MINIPCI/CARDBUS/PCIE WIRELESS CARD

Ralink Technology Corporation (Taiwan)

5F, No.36, Tai-Yuen Street,

ChuPei City

HsinChu Hsien 302, Taiwan, ROC

Tel +886-3-560-0868

Fax +886-3-560-0818

Sales Taiwan: ~~Sales@ralinktech.com.tw~~

Technical Support Taiwan: ~~FAE@ralinktech.com.tw~~

http://www.ralinktech.com/

1 CONTENTS

2 RELEASE NOTES

2.1 Version History

2.1.1 Version 2.2.0.0

1. New generation schema for multiple OS porting
2. New chip support for RT3572
3. New chip support for RT3062/RT3562.
4. Restrict the encryption type in HT mode..
5. Support 802.1x reauthentication mechanism.
6. Limit the STA connection count per BSS.
7. Some variables support MBSS setting.
8. Support WDS entry life check function
9. Support Dot11K RRM for all testing cases of voice-enterprise testing event.

2.1.2 Version 2.0.0.0

10. Added Global Country Domain supported.
11. Fix bug: suspend/resume error when ra0 down, rax up
12. Add new UAPSD SP counting mechanism.
13. Fix bug: Ikanos WDS, AP Client, Mess interface get problem.
14. Add new WSC hardware push button function for PCI & USB.
15. Added a function allow user to sepcific Tx rate for Mcast packets.
16. Migrate Mesh supporting to Draft-2.0.
17. Support WAPI functionality
18. Modify the priority of BAR transmission to solve the connection issue with Intel 4965 11n STA.

2.1.3 Version 1.9.0.0

1. Replace iwpriv cmd "AccessControlList" by "ACLAddEntry" and "ACLClearAll"
2. Fixed the wrong usage of AtoH ().
3. Support new Windows ATE GUI.
4. Add a command "iwpriv ra0 set ATERE2P=1" to display all EEPROM content.
5. Correct the limitation of the length of fragment
6. Fix bug: Fail to transmit packets through AMPDU way except the case that AP to STA.
7. Wrong Hareware packet length calculation of Mesh packet if it has been fragmented.
8. Support SIGMA 8622/8624 platform.
9. Add WPS PBC Session Overlap Detecting.
10. Add WPS 4-PinCode Support.
11. Fixed WPS enable PSP can not associate AP when AP security set to WPA-PSK.
12. If 11n station operated in power save mode, the AP should transmit none AMPDU nor AMSDU to the station for the Ps-Poll.
13. 20/40 overlapping BSS scan mechanism and bandwidth adjustment.
14. Support 802.11n draft 4.0

2.1.4 Version 1.8.0.0

1. Show Tx/Rx statistics per MBSS.

2. 802.1x supports failover mechanism.
3. Add watchdog to prevent MAC/BBP into the deadlock condition.
4. Support pure 11n with 5G band.
5. Update Timer Functions
6. In multiple cards application, the interface name is changed to raxx_k, where xx means card ID (0 ~ 31) and k means the BSS number (0 ~ 7)
7. Support individual MCS per BSS.
8. Add IKANOS Vx160 and Vx180.
9. Add station keep alive detection function in AP mode.
10. The SIFS of CCK is changed to 16 micro seconds to fix the connection problem with INTEL 2200bg cards.
11. QBSS Load Element is added to provide channel utilization information to all STAs.
12. Fix bug : After AP re-key, the ping connection from client to AP would be time-out within several seconds.
13. Support Mesh function.
14. Support SNMP function.
15. Big-endian ATE supported.

2.1.5 Version 1.7.0.0

1. Support IDS notification mechanism.
2. Change IRQ LOCK to SEM LOCK.
3. Fix bug : When QoS(non-BE) and fragment packets are received, AP would calculate wrong MIC in TKIP mode.
4. Support Non-GPL MD5.
5. Update Group rekey mechanism.
6. Fix BA time-out issue for Intel wireless card 4965AGN with version 11.5.0.32.
7. Add command "iwpriv ra0 set ATELDE2P=1" to overwrite all EEPROM contents from "/etc/Wireless/RT2860(/70)AP(/STA)/e2p.bin".
8. Fix RTS threshold issue in 5G-band.
9. Add DLS Function.
10. IPV6 MLDv2 support.
11. Fix VLAN ID >= 256 can not be used.
12. Added PCIE MSI supporting for RT2890.
13. Added new channel list builder that create channel list according to country-code and channel Geography (in/out door).

2.1.6 Version 1.6.0.0

1. Fix bug: Before AP shutdown, AP doesn't noify those associated STA through dis-association.
2. Fix bug: The Atheros wireless STA card built in MacBook can't work normally when HT mode and the encryption is WEP or TKIP.
3. The support region in A band synchronizes with EEPROM.
4. It supports to initialize current wireless MAC address from E2PROM or module parameter.
5. Support maximum 8 MBSS and each beacon maximum length is 512 bytes.
6. Support 5-GHz band ATE.
7. Send DisAssoc frame to timeout STA.
8. Workaround for Atheros STA on AES mode.
9. Tx RTS/CTS when AP setup BA.
10. Driver sends IAPP L2 frame instread of Daemon.
11. Correct some timeout values of WPS.

12. Fix bug: The 802.1x daemon (rt2860apd) has some problem for parsing multiple parameters in MBSS.
13. Fix bug: The AP site survey signal isn't correct.
14. Provide some 11n statistics variables.
15. Fix bug: RT2561 module can not be removed after RT2860 module is inserted.
16. Added DEO (100 ~ 140) channel list for Ganmany.
17. Support wds phy mode and security setting for each wds link.
18. Fix bug: The Atheros legacy USB STA card can't connect to our AP in WPA-TKIP.
19. Modify rate adaptation for fast ramp-up tuning.
20. Fix WPS IOT issue with Atheros externnal registrar. Need Sync the user space daemon "wscd" to version 0.1.0
21. Fix bug: The BlackBerry/HTC can't connect to our AP.
22. Correct the default values of those WMM EDCA parameters.

2.1.7 Version 1.5.0.0

1. Added McastPhyMode and McastMcs iwpriv commands let user to specifice the rate for Multicast packets transmition.
2. Added two configrations of McastPhyMode and McastMcs.
3. Re-organize the WPA state machine in order to the consistency between AP and AP-Client.
4. Added DFS support.
5. Added Carrier-Sense suppport.
6. Fixed a bug about dissection issue about ';' in profile.
7. Fixed CountryRegion and channel map, when profile's channel have not on channel list.
8. Fixed 802.1x Authentication problem with 1x-WEP/WPA(2)-Enterprise when WPS is enabled.
9. Fix bug: Once the radio is off and on, the beacon frames disappear.

2.1.8 Version 1.4.0.0

1. Wireless IGMP snooping support for multimedia steaming.
2. Access control list support
3. Re-organize the Rx data path.
4. AP client WPS support.
5. Fix the Auto-selecting channel issue.
6. Add CountryString ioctl command function.
7. Buf fix for Atheros WPS STA can not config WPS AP when Athros JumpStart STA is external Registrar.
8. Merge for WCN test modify to WPS functions.
9. Patch for 11n requirement, if HT mode is set and BW is 40MHz in A-band, the supported Channel number must be the multiple of 2.
10. Fix bug: If STA card operated with zero-config, the group rekey negotiation of WPA2(PSK)-AES always fails.
11. Add Tx & Rx Stream functionality.
12. Support QA user interface for ATE function.

2.1.9 Version 1.3.0.0

1. Add vlan tag support for each BSS.
2. Add support for 32bit/64bit Linux.
3. Merge in plugfest code.
4. Support Ap-Client function.

5. Add new parameter "bWiFiTest" for WPA & WMM WiFi-Test.
6. Add the setting of Japan filter coefficients for ATE.
7. Fix bug for channel have not update when auto channel select was true.
8. Add protect for RTMP_IRQ_LOCK to avoid in spin_lock_irqsave call spin_lock_bh cause kernel waring messages.

2.1.10 Version 1.2.0.0

1. Fix bug for counterMeasures in WiFi test.
2. Write TXWI in ATE's way and disable any protection mechanism when ATE is running.
3. Disable ATE RSSI statistics when ATE is not running.
4. Select DAC according to HT or Legacy mode.
5. Support WPA2 Pre-authemtication.
6. Fix WDS panic bug.
7. Shift skb control block used by driver to offset 10 to avoid dirty cb[] from protocol stack.
8. Fix issue - 802.1X daemon may cause throughput reduction.
9. Support Wireless event log mechanism.
10. Add a 200ms-timer to enqueue EAPoL-Start for WPAPSK, not RTMPusecDelay.
11. Auto-selecting channel check.

2.1.11 Version 1.1.0.0

1. Add fast rate switch.
2. Modify fast rate switch timer form periodic to trigger by condition.
3. Fix UAPSD bugs for null frame was drop.
4. Fix management queue pass qos null frame ...
5. Code freeze for Wifi.
6. Merge code from Plugfest #6.
7. Add "iwpriv ra0 show driverinfo" to show the driver version.

2.1.12 Version 1.0.0.0:

1. Interface support and bugs fix for WMM (Under testing).
2. DFS support.
3. Support WPA over WDS.
4. Bug fix for two WPAPSK-STAs causes the AP to crash.
5. Bug fix for BG-STAs will link up with B-only-AP.
6. Fix compatiblility issue in 802.11d.

2.2 FEATURES

This RT2800 a/b/g/n SoftAP driver implements wireless Access Point (AP) function and supports 4 BSSIDs concurrently.

The AP can access the internet through other interfaces (e.g. Ethernet) through the bridge service in Linux.

This driver allows OPEN, SHARED, WPAPSK/WPA2PSK, and WPA/WPA2 authentication modes and also supports WEP, TKIP, AES, MIXED MODE or NONE encryption methods. It can also handle cerification negotiating through the 802.1x daemon.

Use NONE or WEP as the encryption method if using OPEN or SHARED authentication modes.

Use TKIP or AES encryption methods if using WPA/WPA2 or WPAPSK/WPA2PSK and their combinations as an authentication mode.

Other combinations are not yert supported by this driver.

For support 802.11n draft 4.0

1. AP receives PS-Poll behavior changed: If a 11n station operate in power save mode, the AP should transmit none AMPDU nor AMSDU to the station for the Ps-Poll.
2. 20/40 BSS Coexistence :
   A. Before an AP starts a 20/40 MHz BSS it shall perform overlapping BSS scans to search for existing BSSs and decide if it need to perform fallback to 20MHz bandwidth.
   B. In a 2.4GHz 40/20MHz BSS, a station that uses 40MHz in this BSS must support periodic scanning. The scanning channels include the channels that are affected by 40MHz transmission. The scanning dwell time and period are announced in AP's beacon.
   C. After each scanning, the station must send 20/40 BSS Coexistence Action frame report to AP. The action frame contains a list of legacy AP if scan any.
   D. If AP find a list of legacy AP in this report frame, the AP may make decision whether it should change my 20/40 BSS to operate in 20MHz-only.
   E. If the AP decides to change to 20MHz, the AP will either send Notify Bandwidth action frames to notify all Station to change to 20MHz. So in this 40/20MHz BSS, there are only 20MHz wireless traffic.
   F. If the AP continuously receive the 20/40 BSS Coexistence Action frame that with ZERO legacy AP in the list for Dot11BssWidthChanTranDelay minutes, the AP may decide to turn back to use 40MHz traffic with 40MHz-capable station.
   G. If the BSS is already configured as a 20MHz-only BSS or it's a BSS in 5GHz. No need to do this.
3. Reversed Direction Granted: RDG means the transmitter who already reserved the channel transmission opportunity for a period of time allows the receiver to send wireless packet in its reserved transmission opportunity interval too. Noted, without RDG, only the one who reserves the channel transmission opportunity can transmit wireless packet.
4. Support draft 4.0 IE:
   A. #define IE_2040_BSS_COEXIST 72 // 802.11n D3.03
   B. #define IE_2040_BSS_INTOLERANT_REPORT 73 // 802.11n D3.03
   C. #define IE_OVERLAPBSS_SCAN_PARM 74 // 802.11n D3.03
   D. #define IE_SECONDARY_CH_OFFSET 62 // 802.11n D3.03
   E. #define IE_EXT_CAPABILITY 127 // 802.11n D3.03
5. New functions:
   A. SendNotifyBWActionFrame()
   B. SendBSS2040CoexistMgmtAction()
   C. APOverlappingBSSScan()
   D. Set_OBSSScanParam_Proc()
   E. Update2040CoexistFrameAndNotify()
   F. ChannelSwitchAction()

2.3 USAGE

This source code package can be used with Linux versions released after RedHat Linux 7.3.

2.3.1 Scripts

load	load module to kernel
unload	unload module from kernel
Configure	config build Linux version
bridge_setup	script for bridge setup

2.3.2 Setup Sequence

1. Use the 'chmod' command to change the access rights of following script files: load; unload; Configure; bridge_setup.
2. Turn on or patch the Linux bridge package
3. $make config   # config build Linux os version
4. $make   # compile driver source code
5. $cp rt2860.bin /etc/Wireless/RT2860AP/   # copy firmware
6. $load   # load/insmod module
7. $bridge_setup   # configure bridge

2.3.3 bridge_setup

```
/usr/sbin/brctl addbr br0
/usr/sbin/brctl addif br0 eth0
/usr/sbin/brctl addif br0 ra0
/sbin/ifconfig eth0 0.0.0.0
/sbin/ifconfig ra0 0.0.0.0
/sbin/ip link set br0 up
/sbin/ip addr add 192.168.5.234/24 brd + dev br0
/sbin/ip route add default via 192.168.5.254
```

2.3.4 load

```
/sbin/insmod RT2860ap.o                Kernel 2.4.x
or
/sbin/insmod RT2860ap.ko               Kernel 2.6.x
/sbin/ifconfig ra0 inet 192.168.5.234 up
/sbin/route add default gw 192.168.5.254
```

2.3.5 unload

```
/sbin/ifconfig ra0 down
/sbin/rmmod RT2860ap
```

2.3.6 Support multicard in one platform

Introduction:

We provide three usages in RT28xxCard.dat, CARDTYPE, CARDID, or MAC

<<<<<<<<<< CARDTYPE >>>>>>>>>>

The only thing you need to know is that you have "how many 11ABGN cards and how many 11BGN cards".

If you have 3 USB/PCMCIA cards, we name them: card00 (bgn), card01 (abgn), card02 (abgn).So we can write their profile path in RT28xxCard.dat as below:

```
00CARDTYPEbgn=/etc/Wireless/RT2860AP/RT2860AP0.dat
01CARDTYPEabgn=/etc/Wireless/RT2860AP/RT2860AP1.dat
02CARDTYPEabgn=/etc/Wireless/RT2860AP/RT2860AP2.dat
```

1. After you plug-in the card01, it will find RT2860AP1.dat;

After you plug-in the card00, it will find RT2860AP0.dat;
After you plug-in the card02, it will find RT2860AP2.dat;
Then you plug-out card01 and card02.
Then you plug-in card02, card02 will find RT2860AP2.dat because it has ever pluged-in before.
2. If no any card is pluged in before, then you plug-in card02, card02 will find RT2860AP1.dat.

If it can not find its CARDTYPE in RT28xxCard.dat, it will use default path (/etc/Wireless/RT2860AP/RT2860AP.dat or /etc/Wireless/RT2870AP/RT2870AP.dat or /etc/Wireless/RT2860AP/RT2860STA.dat or /etc/Wireless/RT2860AP/RT2870STA.dat)
<<<<<<<<<< CARDID >>>>>>>>>>
The 1st plug-in card will match 1st profile except it is plug-in before when driver is not removed.

If you have 3 USB/PCMCIA cards, we name them: card00 (bgn), card01 (abgn), card02 (abgn).So we can write their profile path in RT28xxCard.dat as below:

```
00CARDID=/etc/Wireless/RT2860AP/RT2860AP0.dat
01CARDID=/etc/Wireless/RT2860AP/RT2860AP1.dat
02CARDID=/etc/Wireless/RT2860AP/RT2860AP2.dat
```

1. After you plug-in the card01, it will find RT2860AP0.dat;
   After you plug-in the card00, it will find RT2860AP1.dat;
   After you plug-in the card02, it will find RT2860AP2.dat;
   Then you plug-out card01 and card02.Then you plug-in card02, card02 will be still find RT2860AP2.dat because it has ever pluged-in before.
2. If no any card is pluged in before, then you plug-in card02, card02 will find RT2860AP0.dat, not RT2860AP2.dat.

If you want to use RT2860AP2.dat profile for card02, you must swap row 00CARDID and row 02CARDID or plug-in card00 and card01 before card02 is pluged-in.

If CARDIDxx list are not enough, it will use default path (/etc/Wireless/RT2860AP/RT2860AP.dat or /etc/Wireless/RT2870AP/RT2870AP.dat or /etc/Wireless/RT2860AP/RT2860STA.dat or /etc/Wireless/RT2860AP/RT2870STA.dat)

<<<<<<<<<< MAC >>>>>>>>>>
Every card will find its MAC address and get its profile path. (1 vs. 1 absolute mapping)

```
00MAC00:01:02:03:04:05=/etc/Wireless/RT2860AP/RT2860AP0.dat
01MAC00:01:02:03:04:06=/etc/Wireless/RT2860AP/RT2860AP1.dat
02MAC00:01:02:03:04:07=/etc/Wireless/RT2860AP/RT2860AP2.dat
```

If it can not find its MAC address, it will use default path (/etc/Wireless/RT2860AP/RT2860AP.dat or /etc/Wireless/RT2870AP/RT2870AP.dat or /etc/Wireless/RT2860AP/RT2860STA.dat or /etc/Wireless/RT2860AP/RT2870STA.dat)

Note:

3. When you have more than 1 RT2860 or RT2870 cards, you need to use RT28xxCard.dat
   to determine which profile is used by which card.

4. The first item in RT28xxCard.dat must be from 00CARDID, 00MAC, 00CARDTYPE, not 01CARDID, 01MAC, 01CARDTYPE
5. You can not modify RT28xxCard.dat when you yet remove RT28xx module.
6. Multiple RT2860 cards configured as AP use "/etc/Wireless/RT2860AP/RT2860APCard.dat"
7. Multiple RT2860 cards configured as STA use "/etc/Wireless/RT2860AP/RT2860STACard.dat"
8. Multiple RT2870 cards configured as AP use "/etc/Wireless/RT2860AP/RT2870APCard.dat"
9. Multiple RT2870 cards configured as STA use "/etc/Wireless/RT2860AP/RT2870STACard.dat"
   Ex: When you have 2 RT2860 cards and 2 RT2870 cards, you need two
   RT28xxCard.dat for RT2860 and RT2870. Same RT28xxCard.dat can not be shared for AP and STATION or RT2860 and RT2870.

3 CONFIGURATION

1. The RT2800 SoftAP driver can be configured via two interfaces, i.e. 1) configuration file, 2). "iwpriv" command
   1.1. RT2860AP.dat is an example of configuration file.
   1.2. For instructions on iwpriv usage, please refer to iwpriv_usage.txt.
2. Please put RT2860AP.dat in /etc/Wireless/RT2860AP/RT2860AP.dat.
3. To change the file path, please change the definition in rt_Linux.h
   3.1. #define PROFILE_PATH "/etc/Wireless/RT2860AP/RT2860AP.dat"
4. To edit configuration file, please follow the rules below:
   4.1. add # at head for comment line
   4.2. syntax is 'Param'='Value'
5. A detailed description and the usage of each parameter is provided in the following sections.

3.1 RT2860AP.dat Parameter List

#The word of "Default" must not be removed
Default

1. Basic Parameters:

CountryRegion=5
CountryRegionABand=7
CountryCode=
BssidNum=1
SSID=AP1
WirelessMode=0
FixedTxMode=1
Channel=6
BasicRate=15
BeaconPeriod=100
DtimPeriod=1
TxPower=100
DisableOLBC=0
BGProtection=0
TxPreamble=0
RTSThreshold=2347
FragThreshold=2346
TxBurst=1
PktAggregate=0
NoForwarding=0
NoForwardingBTNBSSID=0
HideSSID=0
ShortSlot=1
AutoChannelSelect=0
WiFiTest=0
WirelessEvent=0
AccessPolicy0=0
AccessControlList0=
AccessPolicy1=0
AccessControlList1=
AccessPolicy2=0
AccessControlList2=
AccessPolicy3=0
AccessControlList3=
McastPhyMode
McastMcs
IdsEnable
AuthFloodThreshold
AssocReqFloodThreshold
ReassocReqFloodThreshold
ProbeReqFloodThreshold
DisassocFloodThreshold
DeauthFloodThreshold
EapReqFooldThreshold
StationKeepAlive
OBSSScanParam
WpaMixPairCipher
MaxStaNum

2. HT Parameters:

HT_HTC (Support the HT control field)
HT_RDG (Support reverse direction grant)
HT_LinkAdapt (Obsolete)
HT_OpMode
HT_MpduDensity (MPDU density)
HT_BW (Support channel width)
HT_EXTCHA (To locate the 40MHz channel in combination with the control)
HT_AutoBA (setup BA session automatically)
HT_AMSDU (Tx AMSDU)
HT_BAWinSize (Supported BA Windows Size)
HT_GI (Support Short/Long GI)
HT_MCS (MCS rate control)
HT_BADecline
HT_TxStream
HT_RxStream

3. WPS Parameters:

WscConfMode=0
WscConfStatus=1
WscConfMethods
WscKeyASCII

4. WMM Parameters:

WmmCapable=0
DLSCapable=0
APAifsn=3;7;1;1
APCwmin=4;4;3;2
APCwmax=6;10;4;3
APTxop=0;0;94;47
APACM=0;0;0;0
BSSAifsn=3;7;2;2
BSSCwmin=4;4;3;2
BSSCwmax=10;10;4;3
BSSTxop=0;0;94;47
BSSACM=0;0;0;0
AckPolicy=0;0;0;0
APSDCapable=0

5. IEEE802.1h+d, Spectrum Management

MaxTxPowerLevel=16
IEEE80211H=0
CSPeriod=10
RDRegion
CarrierDetect
ChGeography

6. Security Policy Parameters

AuthMode=OPEN
EncrypType=NONE
WPAPSK=
PreAuth=0
RekeyMethod=DISABLE
RekeyInterval=0
PMKCachePeriod=10
DefaultKeyID=1
Key1Type=0
Key1Str=
Key2Type=0
Key2Str=
Key3Type=0
Key3Str=
Key4Type=0
Key4Str=

7. WDS Parameters

WdsEnable=0
WdsEncrypType=NONE
WdsList=
WdsKey=

8. 802.1X Authenticator

IEEE8021X=0
RADIUS_Server=192.168.2.3
RADIUS_Port=1812
RADIUS_Key=ralink
own_ip_addr=192.168.5.234

EAPifname=br0
PreAuthifname=br0

9. AP Client Parameters

ApCliEnable=0
ApCliSsid=
ApCliBssid=
ApCliWPAPSK=
ApCliAuthMode=
ApCliEncrypType=
ApCliDefaultKeyID=
ApCliKey1Type=
ApCliKey1Str=
ApCliKey2Type=
ApCliKey2Str=
ApCliKey3Type=
ApCliKey3Str=
ApCliKey4Type=
ApCliKey4Str=

3.2 Iwpriv Command List

1. Basic Parameters:

DriverVersion
CountryRegion
CountryRegionABand
SSID
HideSSID
WirelessMode
FixedTxMode
Channel
BasicRate
BeaconPeriod
DtimPeriod
TxPower
BGProtection
DisableOLBC
TxPreamble
ShortSlot

4 BASIC PARAMETERS

These parameters are basic parameters and have to set, otherwise default value used.

4.1 Supported Parameters in RT2860AP.dat

4.1.1 CountryRegion=value

Value:

Region	Channels
0	1-11
1	1-13
2	10-11
3	10-13
4	14
5	1-14
6	3-9
7	5-13

4.1.2 CountryRegionABand=value

Value:

Region	Channels
0	36, 40, 44, 48, 52, 56, 60, 64, 149, 153, 157, 161, 165
1	36, 40, 44, 48, 52, 56, 60, 64, 100, 104, 108, 112, 116, 120, 124, 128, 132, 136, 140
2	36, 40, 44, 48, 52, 56, 60, 64
3	52, 56, 60, 64, 149, 153, 157, 161
4	149, 153, 157, 161, 165
5	149, 153, 157, 161
6	36, 40, 44, 48
7	36, 40, 44, 48, 52, 56, 60, 64, 100, 104, 108, 112, 116, 120, 124, 128, 132, 136, 140, 149, 153, 157, 161, 165
8	52, 56, 60, 64
9	36, 40, 44, 48, 52, 56, 60, 64, 100, 104, 108, 112, 116, 132, 136, 140, 149, 153, 157, 161, 165
10	36, 40, 44, 48, 149, 153, 157, 161, 165
11	36, 40, 44, 48, 52, 56, 60, 64, 100, 104, 108, 112, 116, 120, 149, 153, 157, 161

4.1.3 CountryCode=value

Value:

2 characters, like TW for Taiwan.

Please refer to ISO3166 code list for other countries and can be found at
http://www.iso.org/iso/en/prods-services/iso3166ma/02iso-3166-code-lists/list-en1.html#sz

4.1.4 BssidNum=value

Value:

1/2/4/8: multiple BSSID number

Note:

1. MAC Address alignment on MBSSID.
   1.1. Main BSSID have to insure MAC address is multiple of 2s on 2-BSSIDs' application.
   1.2. Main BSSID have to insure MAC address is multiple of 4s on 4-BSSIDs' application.
   1.3. Main BSSID have to insure MAC address is multiple of 8s on 8-BSSIDs' application.
2. Example 4 BSSIDs:

Align	1st	2nd	3rd	4th
0x00	AA-BB-CC-DD-EE-F0	AA-BB-CC-DD-EE-F1	AA-BB-CC-DD-EE-F2	AA-BB-CC-DD-EE-F3
0x04	AA-BB-CC-DD-EE-F4	AA-BB-CC-DD-EE-F5	AA-BB-CC-DD-EE-F6	AA-BB-CC-DD-EE-F7
0x08	AA-BB-CC-DD-EE-F8	AA-BB-CC-DD-EE-F9	AA-BB-CC-DD-EE-FA	AA-BB-CC-DD-EE-FB
0x0C	AA-BB-CC-DD-EE-FC	AA-BB-CC-DD-EE-FD	AA-BB-CC-DD-EE-FE	AA-BB-CC-DD-EE-FF

3. Refer to data sheet for detail.
   3.1. MAC_BSSID_DW1.
   3.2. Security Key Table Layout.

4.1.5 SSID=value

Value:

1~32 ASCII characters.

SSID1=value
SSID2=value
SSID3=value
SSID4=value
SSID5=value
SSID6=value
SSID7=value
SSID8=value
(Refer to Q&A – 7)

4.1.6 WirelessMode=value

Value:

0:	802.11 B/G mixed
1:	802.11 B only
2:	802.11 A only
4:	802.11 G only
6:	802.11 N only
7:	802.11 G/N mixed
8:	802.11 A/N mixed
9:	802.11 B/G/N mixed
10:	802.11 A/G/N mixed

11: 802.11 N in 5G band only

4.1.7 FixedTxMode=value

Fix Tx mode to CCK or OFDM for MCS rate selection.
Refer to Q&A - 6 (last page) for detail description and example.

Value:

0:	None (imply N is default)
1:	CCK
2:	OFDM

4.1.8 Channel=value

Value:

802.11b/g: 1~14 depends on CountryRegion setting

802.11a : 36~165 depends on CountryRegion setting

4.1.9 BasicRate=value

Value:

0 ~4095

Note:

A bitmap represent basic support rate (A mode not support)

1:	Basic rate-1Mbps
2:	Basic rate-2Mbps
3:	Basic rate-1Mbps, 2Mbps
4:	Basic rate-5.5Mbps
15:	Basic rate-1Mbps, 2Mbps, 5.5Mbps, 11Mbps

Examples:

Basic Rate Bit Map (max. 12-bit, represent max. 12 basic rates)												
Bit	11	10	9	8	7	6	5	4	3	2	1	0
Rate	54	48	36	24	18	12	9	6	11	5.5	2	1
Set	0	1	0	1	0	1	0	1	1	1	1	1
Hex	5				5				F			
Decimal	1375											

Note:

Set correct basic rates set before changing wireless mode.

11B/G Mixed, 11B/G/N Mixed, and 11N Only:
iwpriv ra0 set BasicRate=15 (0x0F: 1, 2, 5.5, 11 Mbps)
11B:
iwpriv ra0 set BasicRate=3 (0x03: 1, 2 Mbps)
11G-Only and 11G/N Mixed:

iwpriv ra0 set BasicRate=351 (0x15F: 1, 2, 5.5, 11, 6, 12, 24 Mbps)

4.1.10 BeaconPeriod=value

Value:

20 ~ 1024

4.1.11 DtimPeriod=value

Value:

1 ~ 255

4.1.12 TxPower=value

Value:

100 ~ 90 use value in E2PROM as default
90 ~ 60 default value -2
60 ~ 30 default value -6
30 ~ 15 default value -12
15 ~ 9 default value -18
9 ~ 0 default value -24

Note:

1. Range: 1 ~ 100 (unit in percentage)
2. This value restricted by HW characteristic.

4.1.13 BGProtection=value

Value:

0:	Auto
1:	Always On
2:	Always Off

4.1.14 DisableOLBC=value

Value:

0:	Enable
1:	Disable

4.1.15 TxPreamble=value

Value:

0:	Long Preamble
1:	Short Preambl

4.1.16 RTSThreshold=value

Value:

1 ~ 2347

4.1.17 FragThreshold=value

Value:

256 ~ 2346

4.1.18 TxBurst=value

Value:

0:	Disable
1:	Enable

4.1.19 PktAggregate=value

Value:

0:	Disable
1:	Enable

4.1.20 NoForwarding=value

Value:

0:	Disable
1:	Enable

4.1.21 NoForwardingBTNBSSID=value

Value:

0:	Disable
1:	Enable

4.1.22 HideSSID=value

Value:

0:	Disable
1:	Enable

4.1.23 ShortSlot=value

Value:

0:	Disable
1:	Enable

4.1.24 AutoChannelSelect=value

Value: (auto channel select when driver is loaded)

0:	Disable
1:	Enable

4.1.25 WiFiTest=value

Value:

0:	Disable
1:	Enable

4.1.26 WirelessEvent=value

Value:

0:	Disable
1:	Enable

4.1.27 AccessPolicy0=value

Value:

0:	Disable
1:	Allow all
2:	Reject all

4.1.28 AccessControlList0=value

Value:

[Mac Address];[Mac Address];...

E.g.

00:10:20:30:40:50;0A:0b:0c:0D:0e:0f;1a:2b:3c:4d:5e:6f

Note:

ACL for Bssid0, max=64

4.1.29 AccessPolicy1=value

Value:

0:	Disable
1:	Allow all
2:	Reject all

4.1.30 AccessControlList1=value

Value:

[Mac Address];[Mac Address];...

E.g.

00:10:20:30:40:50;0A:0b:0c:0D:0e:0f;1a:2b:3c:4d:5e:6f

Note:

ACL for Bssid1, max=64

4.1.31 AccessPolicy2=value

Value:

0:	Disable
1:	Allow all
2:	Reject all

4.1.32 AccessControlList2=value

Value:

[Mac Address];[Mac Address];...

E.g.

00:10:20:30:40:50;0A:0b:0c:0D:0e:0f;1a:2b:3c:4d:5e:6f

Note:

ACL for Bssid2, max=64

4.1.33 AccessPolicy3=value

Value:

0:	Disable
1:	Allow all
2:	Reject all

4.1.34 AccessControlList3=value

Value:

[Mac Address];[Mac Address];...

E.g.

00:10:20:30:40:50;0A:0b:0c:0D:0e:0f;1a:2b:3c:4d:5e:6f

Note:

ACL for Bssid3, max=64

4.1.35 McastPhyMode=value

Set PHY mode for Multicast frames

Value:

0:	Disable
1:	CCK
2:	OFDM
3:	HTMIX

4.1.36 McastMcs=value

Set MCS for Multicast frames

Value:

0 ~ 15

4.1.37 IdsEnable=value

Enable or disable IDS function

Value:

0:	Disable
1:	Enable

4.1.38 AuthFloodThreshold=value

Set Authentication frame flood threshold

Value:

0:	Disable this threshold
1 ~ 65535:	Enable this threshold

4.1.39 AssocReqFloodThreshold=value

Set Association request frame flood threshold

Value:

0:	Disable this threshold
1~65535:	Enable this threshold

4.1.40 ReassocReqFloodThreshold=value

Set Re-association request frame flood threshold

Value:

0:	Disable this threshold
1~65535:	Enable this threshold

4.1.41 ProbeReqFloodThreshold=value

Set Probe request frame flood threshold

Value:

0:	Disable this threshold
1~65535:	Enable this threshold

4.1.42 DisassocFloodThreshold=value

Set Disassociation frame flood threshold

Value:

0:	Disable this threshold
1~65535:	Enable this threshold

4.1.43 DeauthFloodThreshold=value

Set Deauthentication frame flood threshold

Value:

0:	Disable this threshold
1~65535:	Enable this threshold

4.1.44 EapReqFooldThreshold=value

Set EAP request frame flood threshold

Value:

0:	Disable this threshold
1~65535:	Enable this threshold

4.1.45 StationKeepAlive

Auto-detect the alive statue of the station periodically

Value:

0:	disable
1~65535:	(unit: seconds)

4.1.46 OBSSScanParam

This command used to set the 802.11n draft3 new information element "Overlapping BSS Scan Parameters element", this IE is used by an AP in a BSS to indicate the values to be used by BSS members (i.e., connected STAs) when performing overlapping BSS scan operations.

OBSSScanParam=PassiveScanDwell; ActiveScanDwell; TriggerScanInterval; PassiveScanTotalPerCh; ScanActiveTotalPerCh; TransDelayFactor; ScanActivityThre

1. PassiveScanDwell: uint in units of TU within range 5~1000, default value is 20 Define the minimum duration of each channel when a STA do an individual passively scan within an overlapping BSS scan operation.
2. ActiveScanDwell: uint in units of TU within range 10~1000, default value is 10. Define the minimum duration of each channel when a STA do an individual actively scan within an overlapping BSS scan operation.
3. TriggerScanInterval: uint in units of second, default value is 300. Define the max interval between scan operations to be performed to detect BSS channel width trigger events Support WPA over WDS.
4. PassiveScanTotalPerCh: uint in units of TU within range 200~10000, default value is 200. Define the minimium total amount of time that the STA scans each channel when performing a passive OBSS scan.
5. ScanActiveTotalPerCh: uint in units of TU within range 20~10000, default value is 20 Define the min total amount of time that the STA scans each channel when performing a active OBSS scan.
6. TransDelayFactor: uint in units of times, default value is 5. Define the minimum ratio between the delay time in performing a switch from 20 MHz BSS operation to 20/40 MHz BSS operation and the maximum.
7. ScanActivityThre: uint in units of %%, default value is 25, it means 0.25%.Define the max total time that a STA may be active on the medium during a period of (dot11BSSWidthChannelTransactionDelayFactor * dot11BSSWidthTriggerScanInterval) seconds without being obligated to perform OBSS Scan operations.

Example:

```
OBSSScanParam=20; 10; 300; 200; 20; 5; 25
```

Note:
(1) It only supported when enable the compile flag "DOT11N_DRAFT3".
(2) By default, we didn't suggest user use this "iwpriv cmd"/"profile entity" to modify those values unless they have specific requirements.

4.1.47 WpaMixPairCipher

[Description]
It provides a more flexible cipher combination.
In WPA-WPA2 with TKIP/AES mode, we provide a more flexible cipher combination.
If users want to operate the command, please make sure that the AuthMode is WPAWPA2 mixed mode and the encryption is TKIPAES mixed mode.

The definition of the cipher combination

WPA		WPA2		
TKIP	AES	TKIP	AES	
0	1	1	0	WPA-AES and WPA2-TKIP
0	1	1	1	WPA-AES and WPA2-TKIPAES
1	0	0	1	WPA-TKIP and WPA2-AES
1	0	1	1	WPA-TKIP and WPA2-TKIPAES
1	1	0	1	WPA-TKIPAES and WPA2-AES
1	1	1	0	WPA-TKIPAES and WPA2-TKIP
1	1	1	1	WPA-TKIPAES and WPA2-TKIPAES (default)

[Usage]

```
WpaMixPairCipher=Value
```

Value:

WPA_AES_WPA2_TKIPAES
WPA_AES_WPA2_TKIP
WPA_TKIP_WPA2_AES
WPA_TKIP_WPA2_TKIPAES
WPA_TKIPAES_WPA2_AES
WPA_TKIPAES_WPA2_TKIPAES
WPA_TKIPAES_WPA2_TKIP

[Example]

WpaMixPairCipher=WPA_AES_WPA2_TKIPAES

4.1.48 MaxStaNum

[Description]
To limit the maximum number of associated clients per BSS.

[Usage]

MaxStaNum=Value
Value:

0	: no limit
1~255	

4.2 iwpriv ra0 set [parameters]=[Value]

Syntax:	Example
Section# parameters Explanation Value: 0: ... 1: ... .: ...	3.2.1 DriverVersion Get Driver Version Value: 0

4.2.1 DriverVersion

Get driver version.

Value:

0

4.2.2 CountryRegion

Set country region

Value:

Region	Channels
0	1-11
1	1-13

2	10-11
3	10-13
4	14
5	1-14
6	3-9
7	5-13

4.2.3 CountryRegionABand

Set country region for A band.

Value:

Region	Channels
0	36, 40, 44, 48, 52, 56, 60, 64, 149, 153, 157, 161, 165
1	36, 40, 44, 48, 52, 56, 60, 64, 100, 104, 108, 112, 116, 120, 124, 128, 132, 136, 140
2	36, 40, 44, 48, 52, 56, 60, 64
3	52, 56, 60, 64, 149, 153, 157, 161
4	149, 153, 157, 161, 165
5	149, 153, 157, 161
6	36, 40, 44, 48
7	36, 40, 44, 48, 52, 56, 60, 64, 100, 104, 108, 112, 116, 120, 124, 128, 132, 136, 140, 149, 153, 157, 161, 165
8	52, 56, 60, 64
9	36, 40, 44, 48, 52, 56, 60, 64, 100, 104, 108, 112, 116, 132, 136, 140, 149, 153, 157, 161, 165
10	36, 40, 44, 48, 149, 153, 157, 161, 165
11	36, 40, 44, 48, 52, 56, 60, 64, 100, 104, 108, 112, 116, 120, 149, 153, 157, 161

4.2.4 CountryCode

Set country code on 802.11d.

Value:

2 characters, like TW for Taiwan.
Please refer to ISO3166 code list for other countries and can be found at
http://www.iso.org/iso/en/prods-services/iso3166ma/02iso-3166-code-lists/list-en1.html#sz

4.2.5 AccessPolicy

Set Access control policy.

Value:

0: Disble,

1: Allow All,
2: Reject All

4.2.6 Debug

Set Debug level

Value:

0: Disable
1: Error
2: Warn
3: Trace
4: Info
5: Loud

4.2.7 ResetCounter

Reset all statistics counter.

Value:

0

4.2.8 RadioOn

Turn radio on or off

Value:

0: Off
1: On

4.2.9 SiteSurvey

Issue a site survey command to driver.

Value:

1

4.2.10 CountryString

Set country string on 802.11d.

Value:

32 characters, like Taiwan, case insensitive
Please refer to ISO3166 code list for other countries and can be found at
http://www.iso.org/iso/en/prods-services/iso3166ma/02iso-3166-code-lists/list-en1.html#sz

Item	Country Number	ISO Name	Country Name (CountryString)	Support 802.11A	802.11A Country Region	Support 802.11G	802.11G Country Region
	0	DB	Debug	Yes	A_BAND_REGION_7	Yes	G_BAND_REGION_5

	8	AL	ALBANIA	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	12	DZ	ALGERIA	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	32	AR	ARGENTINA	Yes	A_BAND_REGION_3	Yes	G_BAND_REGION_1
	51	AM	ARMENIA	Yes	A_BAND_REGION_2	Yes	G_BAND_REGION_1
	36	AU	AUSTRALIA	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	40	AT	AUSTRIA	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	31	AZ	AZERBAIJAN	Yes	A_BAND_REGION_2	Yes	G_BAND_REGION_1
	48	BH	BAHRAIN	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	112	BY	BELARUS	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	56	BE	BELGIUM	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	84	BZ	BELIZE	Yes	A_BAND_REGION_4	Yes	G_BAND_REGION_1
	68	BO	BOLIVIA	Yes	A_BAND_REGION_4	Yes	G_BAND_REGION_1
	76	BR	BRAZIL	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	96	BN	BRUNEI DARUSSALAM	Yes	A_BAND_REGION_4	Yes	G_BAND_REGION_1
	100	BG	BULGARIA	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	124	CA	CANADA	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_0
	152	CL	CHILE	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	156	CN	CHINA	Yes	A_BAND_REGION_4	Yes	G_BAND_REGION_1
	170	CO	COLOMBIA	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_0
	188	CR	COSTA RICA	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	191	HR	CROATIA	Yes	A_BAND_REGION_2	Yes	G_BAND_REGION_1
	196	CY	CYPRUS	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	203	CZ	CZECH REPUBLIC	Yes	A_BAND_REGION_2	Yes	G_BAND_REGION_1
	208	DK	DENMARK	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	214	DO	DOMINICAN REPUBLIC	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_0
	218	EC	ECUADOR	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	818	EG	EGYPT	Yes	A_BAND_REGION_2	Yes	G_BAND_REGION_1
	222	SV	EL SALVADOR	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	233	EE	ESTONIA	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	246	FI	FINLAND	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	250	FR	FRANCE	Yes	A_BAND_REGION_2	Yes	G_BAND_REGION_1
	268	GE	GEORGIA	Yes	A_BAND_REGION_2	Yes	G_BAND_REGION_1
	276	DE	GERMANY	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	300	GR	GREECE	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	320	GT	GUATEMALA	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_0
	340	HN	HONDURAS	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	344	HK	HONG KONG	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	348	HU	HUNGARY	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	352	IS	ICELAND	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	356	IN	INDIA	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	360	ID	INDONESIA	Yes	A_BAND_REGION_4	Yes	G_BAND_REGION_1
	364	IR	IRAN	Yes	A_BAND_REGION_4	Yes	G_BAND_REGION_1
	372	IE	IRELAND	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	376	IL	ISRAEL	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	380	IT	ITALY	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	392	JP	JAPAN	Yes	A_BAND_REGION_9	Yes	G_BAND_REGION_1
	400	JO	JORDAN	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	398	KZ	KAZAKHSTAN	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	408	KP	KOREA DEMOCRATIC	Yes	A_BAND_REGION_5	Yes	G_BAND_REGION_1
	410	KR	KOREA REPUBLIC OF	Yes	A_BAND_REGION_5	Yes	G_BAND_REGION_1
	414	KW	KUWAIT	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	428	LV	LATVIA	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	422	LB	LEBANON	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	438	LI	LIECHTENSTEIN	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	440	LT	LITHUANIA	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	442	LU	LUXEMBOURG	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1

	446	MO	MACAU	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	807	MK	MACEDONIA	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	458	MY	MALAYSIA	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	484	MX	MEXICO	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_0
	492	MC	MONACO	Yes	A_BAND_REGION_2	Yes	G_BAND_REGION_1
	504	MA	MOROCCO	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	528	NL	NETHERLANDS	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	554	NZ	NEW ZEALAND	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	578	NO	NORWAY	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_0
	512	OM	OMAN	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	586	PK	PAKISTAN	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	591	PA	PANAMA	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_0
	604	PE	PERU	Yes	A_BAND_REGION_4	Yes	G_BAND_REGION_1
	608	PH	PHILIPPINES	Yes	A_BAND_REGION_4	Yes	G_BAND_REGION_1
	616	PL	POLAND	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	620	PT	PORTUGAL	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	630	PR	PUERTO RICO	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_0
	634	QA	QATAR	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	642	RO	ROMANIA	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	643	RU	RUSSIA FEDERATION	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	682	SA	SAUDI ARABIA	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	702	SG	SINGAPORE	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	703	SK	SLOVAKIA	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	705	SI	SLOVENIA	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	710	ZA	SOUTH AFRICA	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	724	ES	SPAIN	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	752	SE	SWEDEN	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	756	CH	SWITZERLAND	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	760	SY	SYRIAN ARAB REPUBLIC	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	158	TW	TAIWAN	Yes	A_BAND_REGION_3	Yes	G_BAND_REGION_0
	764	TH	THAILAND	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	780	TT	TRINIDAD AND TOBAGO	Yes	A_BAND_REGION_2	Yes	G_BAND_REGION_1
	788	TN	TUNISIA	Yes	A_BAND_REGION_2	Yes	G_BAND_REGION_1
	792	TR	TURKEY	Yes	A_BAND_REGION_2	Yes	G_BAND_REGION_1
	804	UA	UKRAINE	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	784	AE	UNITED ARAB EMIRATES	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	826	GB	UNITED KINGDOM	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	840	US	UNITED STATES	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_0
	858	UY	URUGUAY	Yes	A_BAND_REGION_5	Yes	G_BAND_REGION_1
	860	UZ	UZBEKISTAN	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_0
	862	VE	VENEZUELA	Yes	A_BAND_REGION_5	Yes	G_BAND_REGION_1
	704	VN	VIET NAM	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	887	YE	YEMEN	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	716	ZW	ZIMBABWE	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1

4.2.11 SSID

Set SoftAP SSID

Value:

0~z, less than 32 characters

4.2.12 WirelessMode

Set Wireless Mode

Value:

0:	802.11 B/G mixed
1:	802.11 B only
2:	802.11 A only
4:	802.11 G only
6:	802.11 N only
7:	802.11 G/N mixed
8:	802.11 A/N mixed
9:	802.11 B/G/N mixed
10:	802.11 A/G/N mixed
11:	802.11 N in 5G band only

4.2.13 FixedTxMode=value

Fix Tx mode to CCK or OFDM for MCS rate selection.
Refer to Q&A - 6 (last page) for detail description and example.

Value:

CCK
OFDM

4.2.14 2: OFDMBasicRate

Be careful to set this value, if you don’t know what this is, please don’t set this field.

Value:

0 ~ 4095

e.g.

Basic Rate Bit Map (max. 12-bit, represent max. 12 basic rates)												
Bit	11	10	9	8	7	6	5	4	3	2	1	0
Rate	54	48	36	24	18	12	9	6	11	5.5	2	1
Set	0	1	0	1	0	1	0	1	1	1	1	1
Hex	5				5				F			
Decimal	1375											

4.2.15 Channel

Set channel number.

Value:

802.11b/g:	1 ~ 14 (it must agree with the CountryRegion setting)
802.11a:	36~165 (it must agree with the CountryRegionABand setting)

4.2.16 BeaconPeriod

Set beacon period.

Value:

20 ~ 1024 (unit is in milli-seconds)

4.2.17 DtimPeriod

Set Dtim period.

Value:

1 ~ 255

4.2.18 TxPower

Set AP Tx power percentage.

Value:

1 ~ 100

4.2.19 BGProtection

Set 11B/11G Protection.

Value:

0:	Auto,
1:	Always on,
2:	Always off

4.2.20 DisableOLBC

Set OLBC detection.

Value:

0:	Enable
1:	Disable

4.2.21 TxPreamble

Set TxPreamble.

Value:

0:	Long Preamble
1:	Short Preamble
2:	Auto

4.2.22 RTSThreshold

Set RTS Threshold.

Value:

1~2347

4.2.23 FragThreshold

Set Fragment Threshold.

Value:

256~2346

4.2.24 TxBurst

Set TxBurst Enable or Disable.

Value:

0:	Disable
1:	Enable

4.2.25 PktAggregate

Set Ralink proprietary packet aggregate Enable or Disable.

Value:

0:	Disable
1:	Enable

4.2.26 NoForwarding

Set No Forwarding Enable or Disable.

Value:

0:	Disable
1:	Enable

4.2.27 NoForwardingBTNBSSID

Set No Forwarding between each BSSID interface.

Value:

0:	Disable
1:	Enable

4.2.28 HideSSID

Set Hide SSID Enable or Disable.

Value:

0:	Disable
1:	Enable

4.2.29 ShortSlot

Set Short Slot Time Enable or Disable

Value:

0:	Disable
1:	Enable

4.2.30 VLANID

Set Vlan ID, 0: disable Vlan

Value:

0~

4.2.31 VLANPriority

Set Vlan Priority

Value:

0~

4.2.32 DisConnectSta

Use to disassociate one STA manually

Value:

[Mac Address]

E.g.

00:11:22:33:44:55

4.2.33 McastPhyMode

Use to set multicast pyhsical mode

Value:

0:	Disable
1:	CCK
2:	OFDM
3	HTMIX

4.2.34 McastMcs

Use to set multicast MCS

Value:

0 ~ 15

4.2.35 WscVendorPinCode

Set vendor pin code as pin code of WPS AP's enrollee

Value:
8 valid decimal digital pin code

4.2.36 ACLAddEntry

To insert one or several MAC addresses into Access control MAC table list, up to 64 MAC address at one time

iwpriv ra0 set ACLAddEntry=Value

Value:
"[MAC address];[MAC address];...;[MAC address]"

Example:
iwpriv ra0 set ACLAddEntry="00:0c:43:28:aa:12;00:0c:43:28:aa:11;00:0c:43:28:aa:10"

4.2.37 ACLClearAll

To clear all the MAC address entries in an Access control MAC table list

iwpriv ra0 set ACLClearAll=Value

Value: 1

Example:
iwpriv ra0 set ACLClearAll=1

4.2.38 FixedTxMode

To limit the transmisson mode, CCK or OFDM

iwpriv ra0 set FixedTxMode=Value

Value:
CCK
OFDM

4.2.39 BDInfo

It is used for engineer debug use.
It will display all tx/rx buffer descriptor information

4.2.40 MeasureReq

To trigger AP to issue a measure request action. It's just for engineer debugging

4.2.41 TpcReq

To trigger AP to issue a TPC request action. It's just for engineer debugging

4.2.42 OBSSScanParam

This command used to set the 802.11n draft3 new information element "Overlapping BSS Scan Parameters element", this IE is used by an AP in a BSS to indicate the values to be used by BSS members (i.e., connected STAs) when performing overlapping BSS scan operations.

OBSSScanParam=PassiveScanDwell; ActiveScanDwell; TriggerScanInterval; PassiveScanTotalPerCh; ScanActiveTotalPerCh; TransDelayFactor; ScanActivityThre

1. PassiveScanDwell: uint in units of TU within range 5~1000, default value is 20 Define the minimum duration of each channel when a STA do an individual passively scan within an overlapping BSS scan operation.
2. ActiveScanDwell: uint in units of TU within range 10~1000, default value is 10. Define the minimum duration of each channel when a STA do an individual actively scan within an overlapping BSS scan operation.
3. TriggerScanInterval: uint in units of second, default value is 300. Define the max interval between scan operations to be performed to detect BSS channel width trigger events Support WPA over WDS.
4. PassiveScanTotalPerCh: uint in units of TU within range 200~10000, default value is 200. Define the minimium total amount of time that the STA scans each channel when performing a passive OBSS scan.
5. ScanActiveTotalPerCh: uint in units of TU within range 20~10000, default value is 20 Define the min total amount of time that the STA scans each channel when performing a active OBSS scan.
6. TransDelayFactor: uint in units of times, default value is 5. Define the minimum ratio between the delay time in performing a switch from 20 MHz BSS operation to 20/40 MHz BSS operation and the maximum.
7. ScanActivityThre: uint in units of %%, default value is 25, it means 0.25%.Define the max total time that a STA may be active on the medium during a period of (dot11BSSWidthChannelTransactionDelayFactor * dot11BSSWidthTriggerScanInterval) seconds without being obligated to perform OBSS Scan operations.

Example:

```
iwpriv ra0 set OBSSScanParam=20-10-300-240-20-5-25
```

Note:
(1) It only supported when enable the compile flag "DOT11N_DRAFT3".
(2) By default, we didn't suggest user use this "iwpriv cmd"/"profile entity" to modify those values unless they have specific requirements.

4.2.43 WpaMixPairCipher

[Description]

It provides a more flexible cipher combination.

[Usage]

WpaMixPairCipher=Value
Value:
WPA_AES_WPA2_TKIPAES
WPA_AES_WPA2_TKIP
WPA_TKIP_WPA2_AES
WPA_TKIP_WPA2_TKIPAES
WPA_TKIPAES_WPA2_AES
WPA_TKIPAES_WPA2_TKIPAES
WPA_TKIPAES_WPA2_TKIP

[Example]

```
iwpriv ra0 set WpaMixPairCipher=WPA_AES_WPA2_TKIPAES
```

4.2.44 stasecinfo

[Description]

Display the security setting of associated stations

[Usage]

```
iwpriv ra0 show stasecinfo
```

4.2.45 MaxStaNum

[Description]

To limit the maximum number of associated clients per BSS.

[Usage]

MaxStaNum=Value
Value:

0	: no limit
1~255	

4.2.46 PwrConstraint

[Description]

Used to set value of power constraint.

[Usage]

PwrConstraint=Value
Value:
0~30 (unit is dB)

4.3 iwpriv ra0 get_site_survey

Display the site survey result after issuing “iwpriv ra0 set SiteSurvey=1”.

4.4 iwpriv ra0 get_mac_table

Display associated STA's MAC address

4.5 iwpriv ra0 stat

Display statistics counter.

4.6 iwpriv ra0 get_wsc_profile

Display WSC Profile.

4.7 iwpriv ra0 get_ba_table

Get Block ACK Table. (Raw data)

4.8 iwpriv ra0 show [command]

Command List:

1. stainfo - Show associated STA's MAC address
2. descinfo - Show Descriptor information.
3. driverinfo - Show driver version.
4. wdsinfo - Show WDS list information.
5. bainfo - Show Block ACK Table. (String message)
6. stat - Show statistics counter.
7. stat_reset - Show, then reset statistics counter.
8. igmpinfo - Show all entrys in IGMP table.
9. mcastrate - Show multicast phy mode and MCS rate.

4.9 Examples

4.9.1 Example I

```
iwpriv ra0 set CountryRegion=6
iwpriv ra0 set SSID=SoftAP-1
iwpriv ra0 set WirelessMode=0
iwpriv ra0 set Channel=1
iwpriv ra0 set BeaconPeriod=100
iwpriv ra0 set BGProtection=1
iwpriv ra0 set TxPreamble=0
iwpriv ra0 set RTSThreshold=2347
iwpriv ra0 set FragThreshold=2346
iwpriv ra0 set TxBurst=0
iwpriv ra0 set TurboRate=0
iwpriv ra0 set NoForwarding=0
iwpriv ra0 set NoForwardingBTNBSSID=0
iwpriv ra0 set HideSSID=0
iwpriv ra0 set ShortSlot=0
iwpriv ra0 set AuthMode=SHARED
iwpriv ra0 set EncrypType=WEP
iwpriv ra0 set DefaultKeyID=1
iwpriv ra0 set Key1=1234567890
iwpriv ra0 set Key2=passd
iwpriv ra0 set Key3=12345678901234567890123456
iwpriv ra0 set key4=enterpassword
iwpriv ra0 set AccessPolicy=1
iwpriv ra0 set AccessControlList="00:03:A0:10:0E:10; 00:08:0c:FD:e1:00; 1a:28:40:42:ce:6f"
iwpriv ra0 set WPAPSK=0123456789
iwpriv ra0 set Debug=0
iwpriv ra0 set ResetCounter=1
iwpriv ra0 set DisConnectSta=00:11:22:33:44:55
```

4.9.2 Example II

One iwpriv command sets two parameters.

```
iwpriv ra0 set Channel=8
iwpriv ra0 set SSID=SoftAP-1
```

5 HT PARAMETERS

5.1 Supported Parameters in RT2860AP.dat

5.1.1 HT_AutoBA=value

Value:

0:	Disable, setup BA session manually.
1:	Enable, setup BA session automatically after connected, recommended.

Reference:

9.2.8a BlockAck procedure
9.10 Block Acknowledgment (Block Ack)
9.10.1 to 9.10.5
9.10.7.2 (HT-immediate BlockAck architecture) to 9.10.7.9 (Originator's support of recipient STAs' partial state)

5.1.2 HT_HTC=value

Support the HT control field.

Value:

0:	Disable Tx_+HTC frame
1:	Enable Tx_+HTC frame

Note:

HTC Control field(4-octet) is following QOS field.
An MPDU that contains the HT control field is referred to as a +HTC frame.

Reference:

7.1.3.5a HT Control field

5.1.3 HT_RDG=value

Value:

0:	Disable Reverse Direction Grant, recommended.

1:	Enable Reverse Direction Grant

Note:

1. If HT Reverse Direction Grant is enabled, Tx_+HTC will also been enabled; No matter what value HT_HTC is set.
2. During a response burst, only the responder may transmit – i.e. there are no transmissions by other STA, including the initiator.
3. During an RDG, the RD responder shall not transmit any frames that are not addressed to the RD initiator as the RA.
4. Within RDG period, if responder no frame to transmit or frame corrucpt, initiator can transmit frame when RDG period stopped on PIFS' channel idle time. (On normal condition, responder will received frame after SIFS time.)

5.1.4 HT_LinkAdapt=value

Value:

0:	Disable HT Link Adaptation Control
1:	Enable HT Link Adaptation Control

Note:

If HT Link Adaptation Control is enabled, Tx_+HTC will also been enabled; No matter what value HT_HTC is set.

5.1.5 HT_BW=value

Support channel width.

Value:

0:	Channel Width = 20 MHz
1:	Channel Width = 20/40 MHz

5.1.6 HT_EXTCHA=value

To locate the 40MHz channel in combination with the control.

Value:

0: Extension channel below the control channel.

e.g.:

HT_BW=1, Channel=11, HT_EXTCHA=0 control channel=11, extension channel=7

1: Extension channel above the control channel.

e.g.:

HT_BW=1, Channel=7, HT_EXTCHA=1 control channel=7, extension channel=11

Note:

```
1.    If (HT_BW = 1) and (CommonChannel <= 4):
          BBPCurrentBW = 40MHz;
      HT_EXTCHA MUST be 1
          CentralChannel = CommonChannel + 2;
          ControlChannel = CommonChannel;
2.    Else if (CommonChannel > 4) and (CommonChannel < 8) and (HT_BW = 1):
          BBPCurrentBW = 40MHz;
          If(HT_EXTCHA = 0)
              CentralChannel = CommonChannel – 2;
          Else if(HT_EXTCHA = 1)
              CentralChannel = CommonChannel + 2;
          ControlChannel = CommonChannel;
3.    Else if (HT_BW = 1) and (CommonChannel >= 8):
          If ((ChannelListNum – CommonChannel) < 4)
              BBPCurrentBW = 40MHz;
      HT_EXTCHAN MUST be 0
              CentralChannel = CommonChannel - 2;
          Else
              BBPCurrentBW = 40MHz;
               If (HT_EXTCHA = 0)
                      CentralChannel = CommonChannel – 2;
               Else if (HT_EXTCHA = 1)
                      CentralChannel = CommonChannel + 2;
          ControlChannel = CommonChannel;
4.    Else
          BBPCurrentBW = 20MHz;
          CentralChannel = CommonChannel;
          ControlChannel = CommonChannel;
```

5. ControlChannel is used for control frames and management frames.
6. CentralChannel is used by AsicSwitchChannel() and AsicLockChannel().

5.1.7 HT_OpMode=value

Value:

0: Mixed Mode
1: Green Field

Note:

Mixed Mode:

In this mode packets are transmitted with a preamble compatible with the legacy 802.11a/g – the legacy Short Training Field (STF), the legacy Long Training Field (LTF) and the legacy signal field are transmitted so they can be decoded by legacy 802.11a/g devices. The rest of the packet has a new format. In this mode the receiver shall be able to decode both the Mixed Mode packets and legacy packets.

Green Field:

In this mode high throughput packets are transmitted without a legacy compatible part. This mode is optional. In this mode the receiver shall be able to decode both Green Field mode packets, Mixed Mode packets and legacy format packets.

5.1.8 HT_MpduDensity=value

Value:

0:	no restriction
1 ~ 7:	MPDU Density = (2(value – 1))*1/8 µsec
Other:	MPDU Density = 2 µsec(default 5)

Note:

1. Minimum separation of MPDUs in an A-MPDU,i.e. MPDU density, is negotiable (MPDU density).
2. This limitation shall be measured at the PHY_SAP; the number of bytes between the start of two consecutive MPDUs in A-MPDU shall be equal or greater than MPDU-density*PHY-bit-rate/8.
3. PHY_SAP is the interface between MAC Sublayer and PLCPSublayer.

5.1.9 HT_AMSDU=value

Value:

0:	Disable Tx AMSDU
1:	Enable Tx AMSDU

Note:

1. A Frame aggregation format that allows aggregation of multiple MSDUs in one MPDU.
2. Recipient shall receive and deaggregate an A-MSDU.
3. Be aware that, driver has to ensure only frame of the same TID can be aggregated in this way and different SA/DA frames can be aggregated together (as long as they’re toward

the same RA). The “same TID” constraint is to ensure QOS characteristics is consistent in this A-MSDU.

4. In addition, driver has to indicate A-MSDU by setting this in QOS Control field bit 7.
5. Each A-MSDU subframe has its own 802.3 header used by receiver to recover the real SA and DA.

5.1.10 HT_GI=value

Support Short/Long GI.

Value:

0: Long Guard Interval, 800 nsec
1: Short Guard Interval, 400 nsec

Note:

1. MCS 0 through 15 are mandatory in 20 MHz with 800 ns guard interval at an access point (AP). MCS 0 through 7 are mandatory in 20 MHz with 800 ns guard interval at all STAs. All other MCSs and modes are optional, specifically including Tx (transmit) and Rx (receive) support of 400 ns guard interval, operation in 40 MHz, and support of MCSs with indices 16 through 76.
2. In telecommunications, guard intervals are used to ensure that distinct transmissions do not interfere with one another. These transmissions may belong to different users (as in TDMA) or to the same user (as in OFDM).
3. The purpose of the guard interval is to introduce immunity to propagation delays, echoes and reflections, to which digital data is normally very sensitive.
4. Longer guard periods allow more distant echoes to be tolerated. However, longer guard intervals reduce the channel efficiency.

5.1.11 HT_BAWinSize=value

Value:

1 ~ 64: Recommand 64 for IOT
Other: BA Windw Size = 8(default)

Note:

1. The Originator contains a Tx Buffer control that uses WinStart, WinSize to submit MPDUs for transmission and releases the Tx Buffers getting related Block Acknowledgements from Recipient.
2. WinStart and WinSize are the starting position (sequence number) of the transmit window and the number of buffers negotiated in the BA agreement.

5.1.12 HT_MCS=value

Value:

0 ~ 15, 32:	Fix MCS rate for HT rate.
33:	Auto Rate Adaption, recommended.

Note:

The Modulation and Coding Scheme (MCS) is a value that determines the modulation, coding and number of spatial channels.

5.1.13 HT_BADecline=value

Reject peer BA-Request.

Value:

0:	Disable
1:	Enable

5.1.14 HT_TxStream=value

Value:

1:	Support 1-Tx Stream for MCS0 ~ MCS7
2:	Support 2-Tx Stream for MCS0 ~ MCS15

5.1.15 HT_RxStream=value

Value:

1:	Support 1-Rx Stream for MCS0 ~ MCS7
2:	Support 2-Rx Stream for MCS0 ~ MCS15
3:	Support 3-Rx Stream for MCS0 ~ MCS15

5.2 iwpriv ra0 set [parameters]=[Value]

Syntax:	Example
Section# parameters Explanation Value: 0: … 1: … .: …	4.2.7 HtOpMode HtOpMode. Value: 0: Mixed Mode 1: Green Field

5.2.1 BASetup

Setup Block Ack MAC address.

Value:

[MAC address]

5.2.2 SendMIMOPS

Send MIMO Power Save Action frame by MAC address.

Value:

[MAC address]

5.2.3 BAOriTearDown

Stop Originator Session of Block Ack by MAC address.

Value:

[MAC address]

5.2.4 BARecTearDown

Stop Recipient Session of Block Ack by MAC address.

Value:

[MAC address]

5.2.5 HtBw

Stop Recipient Session of Block Ack by MAC address.

Value:

0:	Channel Width = 20 MHz

1:	Channel Width = 20/40 MHz

5.2.6 HtGi

Set guard interval.

Value:

0:	800 ns long guard interval
1:	400 ns short guard interval

5.2.7 HtOpMode

Set HT mode.

Value:

0:	Mixed Mode
1:	Green Field

5.2.8 HtMcs

Set modulation coding scheme.

Value:

0 ~ 15, 32, 33

HT Mixed Mode, Refer to IEEE P802.11n Figure n67 HT Greenfield, Refer to IEEE P802.11n Figure n68	
MCS = 0 (1S)	(BW=0, SGI=0) 6.5Mbps
MCS = 1	(BW=0, SGI=0) 13Mbps
MCS = 2	(BW=0, SGI=0) 19.5Mbps
MCS = 3	(BW=0, SGI=0) 26Mbps
MCS = 4	(BW=0, SGI=0) 39Mbps
MCS = 5	(BW=0, SGI=0) 52Mbps
MCS = 6	(BW=0, SGI=0) 58.5Mbps
MCS = 7	(BW=0, SGI=0) 65Mbps
MCS = 8 (2S)	(BW=0, SGI=0) 13Mbps
MCS = 9	(BW=0, SGI=0) 26Mbps
MCS = 10	(BW=0, SGI=0) 39Mbps
MCS = 11	(BW=0, SGI=0) 52Mbps
MCS = 12	(BW=0, SGI=0) 78Mbps
MCS = 13	(BW=0, SGI=0) 104Mbps
MCS = 14	(BW=0, SGI=0) 117Mbps
MCS = 15	(BW=0, SGI=0) 130Mbps
MCS = 32	(BW=1, SGI=0) HT duplicate 6Mbps

Notes:
When BW=1, PHY_RATE = PHY_RATE * 2
When SGI=1, PHY_RATE = PHY_RATE * 10/9
The effects of BW and SGI are accumulative.
When MCS=0~7(1S, One Tx Stream), SGI option is supported. BW option is supported.
When MCS=8~15(2S, Two Tx Stream), SGI option is supported. BW option is supported.
When MCS=32, only SGI option is supported. BW option is not supported. (BW =1)
Other MCS code in HT mode are reserved.

5.2.9 HtHtc

Enable HS control.

Value:

0:	Disable
1:	Enable

5.2.10 HtExtcha

Set extension channel.

Value:

0:	Below
1:	Above

5.2.11 HtMpduDensity

Set MPDU density, (Refer to 7.3.2.49.3 A-MPDU Parameters field).

Value:

0:	no restriction
1:	1/4 µs
2:	1/2 µs
3:	1 µs
4:	2 µs
5:	4 µs
6:	8 µs
7:	16 µs

5.2.12 HtBaWinSize

Set Block Ack windows size.

Value:

0 ~ 64

5.2.13 HtMIMOPS

Set MIMO Power Save.

Value:

0:	Static
1:	Dynamic
2:	Reserved
3:	No Limit

5.2.14 HtRdg

Enable reverse direction grant.

Value:

0:	Disable
1:	Enable

5.2.15 HtLinkAdapt

Enable Link Adaption.

Value:

0:	Disable
1:	Enable

5.2.16 HtAmsdu

Enable A-MSDU.

Value:

0:	Disable
1:	Enable

5.2.17 HtAutoBa

Enable Auto Block Ack.

Value:

0:	Disable
1:	Enable

5.2.18 HtProtect

Enable HT Protection.

Value:

0:	Disable
1:	Enable

5.2.19 HtMimoPs

Enable MIMO Power Save.

Value:

0:	Disable
1:	Enable

5.2.20 BADecline=value

Reject peer BA-Request.

Value:

0:	Disable
1:	Enable

5.2.21 HtTxStream=value

Value:

1:	Support 1-Tx Stream for MCS0 ~ MCS7
2:	Support 2-Tx Stream for MCS0 ~ MCS15

5.2.22 HtRxStream=value

Value:

1:	Support 1-Rx Stream for MCS0 ~ MCS7
2:	Support 2-Rx Stream for MCS0 ~ MCS15
3:	Support 3-Rx Stream for MCS0 ~ MCS15

6 WPS – WI-FI PROTECTED SETUP

6.1 Simple Config Architectural Overview

This section presents a high-level description of the Simple Config architecture. Much of the material is taken directly from the Simple Config specification.

Figure 1 depicts the major components and their interfaces as defined by Wi-Fi Simple Config Spec. There are three logical components involved: the Registrar, the access point (AP), and the Enrollee.

The Enrollee is a device seeking to join a WLAN domain. Once an Enrollee obtains a valid credential, it becomes a member.

A Registrar is an entity with the authority to issue and revoke domain credentials. A registrar can be integrated into an AP.

The AP can be either a WLAN AP or a wireless router.

Figure 1. Components and Interfaces

Registration initiation is ordinarily accomplished by a user action such as powering up the Enrollee and, optionally, running a setup wizard on the Registrar (PC).

6.1.1 Interface E

This interface is logically located between the Enrollee and the Registrar (physically, the AP can work as a proxy to convey the messages). The functionality of Interface E is to enable the Registrar to discover and issue WLAN Credentials to the Enrollee. Interface E may include only WLAN communication or it may also include communication across an out-of-band channel.

6.1.1.1 ENROLLEE

The Enrollee implements Interface E by:

1. Including a Simple Config IE in 802.11 probe messages.
2. Including a device password on a display or printed label for in-band configuration.
3. Optionally supporting one or more out-of-band configuration channels.
4. Implementing the “Device” part of the Registration Protocol.
5. Optionally receiving ad-hoc probe-responses from wireless Registrars.

6.1.1.2 REGISTRAR

The Registrar implements Interface E by:

1. Processing Enrollee (device or AP) Discovery data in Probe messages (for wireless Registrars) and/or UPnP (for Ethernet-based Registrars).
2. Implementing the "Registrar" part of the Registration Protocol.
3. Optionally supporting one or more out-of-band configuration channels.
4. Configuring the AP with the Enrollee's MAC address and Credential using Interface M.
5. Optionally respond to Enrollee Probe-Requests via an ad-hoc Probe-Response.

6.1.2 Interface M

This interface is between the AP and the Registrar. Interface M enables an external Registrar to manage a Wi-Fi Simple Config AP. Wi-Fi Simple Config uses a similar protocol for setting up the AP Management interface as for issuing credentials to Enrollee devices.

6.1.2.1 AP

The AP implements Interface M by:

1. Acting as the Enrollee in the Registration Protocol for initial setup with one or more external Registrars. This includes sending its own Discovery message across all appropriate channels (Ethernet and/or 802.11 probe response over Wi-Fi). Support for at least three external Registrars is required.
2. Implementing the Management Interface described in the WFADevice and WFAWLANConfig Service documents. This requires the AP to be a UPnP device that includes support for the Wi-Fi Simple Config proxy service.
3. Monitoring 802.11 probe request and EAP messages from Enrollees and converting them to UPnP Event messages according to the method described in the WFAWLANConfig Service document.

6.1.2.2 REGISTRAR

The Registrar implements Interface M by:

1. Processing AP Discovery messages across 802.11 and/or Ethernet.
2. Receiving and processing Enrollee Discovery and Registration messages forwarded by the AP.
3. Optionally receiving and processing Enrollee Discovery and Registration messages sent in ad hoc mode.
4. Implementing the Registrar side of the Registration Protocol to gain management rights over the AP or to issue WLAN credentials to Enrollees
5. Configuring the AP with the MAC address and/or per-device Credential of the Enrollee.
6. Implementing the Management Interface described in the WFADevice and WFAWLANConfig Service documents. This requires the Registrar to function as a UPnP control point.

6.1.3 Interface A

This interface is between the Enrollee and the AP. The function of Interface A is to enable discovery of the Simple Config WLAN and to enable communication between the Enrollee and Ethernet-only Registrars.

6.1.3.1 AP

The AP implements Interface A by:

1. Sending out 802.11 beacons indicating support for Simple Config and generating Probe Response messages containing a description of the AP.
2. Implementing an 802.1X authenticator and the Simple Config EAP method.
3. Proxying 802.11 probe request and EAP messages between Enrollees and external Registrars as described in the WFADevice and WFAWLANConfig Service documents.

6.1.3.2 ENROLLEE

The Enrollee implements Interface A by:

1. Discovering a Simple Config AP and/or wireless external Registrar and sending it 802.11 probe requests including the Enrollee Discovery data.
2. Implementing an 802.1X supplicant and the Simple Config Registration Protocol EAP method.

6.2 Supported Parameters in RT2860AP.dat

6.2.1 WscConfMode=value

Set WPS function, bitwise.

Value:

0x0: Disable
0x1: Enrollee
0x2: Proxy
0x4: Registrar

6.2.2 WscConfStatus=value

Set WPS AP SC (Simple Config) State.

Value:

1: AP is un-configured
2: AP is configured

6.2.3 WscConfMethods

[Description]
The Config Methods Data component lists the configuration methods the Enrollee or Registrar supports. The list is a bitwise OR of values from the table below. If you don’t know what this is, please don’t set this field.

[Usage]

WscConfMethods=Value

Value:

1	- USBA (Flash Drive)
2	- Ethernet
4	- Label
8	- Display
16	- External NFC Token
32	- Integrated NFC Token
64	- NFC Interface
128	- PushButton
256	- Keypad

[Example]

WscConfMethods=16

6.2.4 WscKeyASCII

[Description]

Define WPS WPAPSK format and key length for un-configured internal WPS Registrar AP.

[Usage]

WscKeyASCII=Value

Value:

0: Hex (64-bytes). Default is 0.

1: ASCII(random length)

8 ~ 63: ASCII length

6.3 iwpriv ra0 set [parameters]=[value]

Syntax:	Example
Section# parameters Explanation Value: 0: ... 1: ... .: ...	5.3.1 W scConfMode Set WPS function Value: 0x0: Disable 0x1: Enrollee .....: ...

6.3.1 WscConfMode

Set WPS function, bitwise.

Value:

0x0: Disable
0x1: Enrollee
0x2: Proxy
0x4: Registrar

6.3.2 WscConfStatus

Set WPS AP SC (Simple Config) State.

Value:

1: AP is un-configured
2: AP is configured

6.3.3 WscMode

Set WPS Configured Methods.

Value:

1: use PIN code (Personal Identification Number)
2: use PBC (Push Button Communication)

6.3.4 WscStatus

Get WPS Configured Methods.

Value:

0:	Not Used
1:	Idle
2:	WSC Process Fail
3:	Start WSC Process
4:	Received EAPOL-Start
5:	Sending EAP-Req(ID)
6:	Receive EAP-Rsp(ID)
7:	Receive EAP-Req with wrong WSC SMI Vendor Id
8:	Receive EAPReq with wrong WSC Vendor Type
9:	Sending EAP-Req(WSC_START)
10:	Send M1
11:	Received M1
12:	Send M2
13:	Received M2
14:	Received M2D
15:	Send M3
16:	Received M3
17:	Send M4
18:	Received M4
19:	Send M5
20:	Received M5
21:	Send M6
22:	Received M6
23:	Send M7
24:	Received M7
25:	Send M8
26:	Received M8
27:	Processing EAP Response (ACK)
28:	Processing EAP Request (Done)
29:	Processing EAP Response (Done)
30:	Sending EAP-Fail
31:	WSC_ERROR_HASH_FAIL
32:	WSC_ERROR_HMAC_FAIL
33:	WSC_ERROR_DEV_PWD_AUTH_FAIL
34:	Configured

6.3.5 WscPinCode

Input Enrollee's Pin Code to AP-Registrar.

Value:

8-digits

6.3.6 WscOOB

Reset WPS AP to the OOB (out-of-box) configuration.

Value:

0: Disable
1: Enable

6.3.7 WscGetConf

Trigger WPS AP to do simple config with WPS Client.

Value:

0: Disable
1: Enable

6.4 Examples

6.4.1 Disable WPS function support

- iwpriv ra0 set WscConfMode=0

6.4.2 Enable WPS function support

- iwpriv ra0 set WscConfMode =7 (Binary: 111)

(AP could be Registrar(0x4), Proxy(0x2) or Enrollee(0x1))

6.4.3 WPS AP SC (Simple Config) State

- iwpriv ra0 set WscConfStatus=1 (AP is un-configured)
- iwpriv ra0 set WscConfStatus=2 (AP is configured)

6.4.4 WPS Configured Methods

- iwpriv ra0 set WscMode =1 (use PIN code)
- iwpriv ra0 set WscMode =2 (use PBC)

6.4.5 Input Enrollee's Pin Code to AP-Registrar

- iwpriv ra0 set WscPinCode=xxxxxxxx

6.4.6 Reset WPS AP to the OOB configuration

- iwpriv ra0 set WscOOB=1

(Security: WPAPSK/TKIP, psk: "RalinkInitialAPxx1234" ; SC state: 0x1)
(SSID: RalinkInitialAPxxxxxx, last three characters of AP MAC address)

6.4.7 Trigger WPS AP to do simple config with WPS Client

- iwpriv ra0 set WscGetConf=1

6.4.8 AP services as Enrollee by using PIN code

- iwpriv ra0 set WscMode=1
- iwpriv ra0 set WscGetConf=1

6.4.9 AP services as Enrollee by using PBC

- iwpriv ra0 set WscMode=2
- iwpriv ra0 set WscGetConf=1

6.4.10 AP services as Internal Registrar using PIN code

- iwpriv ra0 set WscMode=1
- iwpriv ra0 set WscPinCode=xxxxxxxx (PIN code from Enrollee, len=8)
- iwpriv ra0 set WscGetConf=1

6.4.11 AP services as Internal Registrar using PBC

- iwpriv ra0 set WscMode=2
- iwpriv ra0 set WscGetConf=1

6.4.12 Get WPS Profile from external registrar

- iwpriv ra0 get_wsc_profile

6.5 Ralink WPS AP Setup Procedure

6.5.1 Introduction

Currently we provide support to run the Access Point (as Enrollee or with Registrar capabilities). The following scenarios are currently supported:

1. Initial Access Point (AP) setup, with the Registrar configuring the Access Point
   1.1. One WiFi-enabled laptop is setup as the AP acting as an Enrollee
   1.2. Another WiFi-enabled laptop is setup as a station acting as the Registrar
   1.3. Two sub cases are 1a) using EAP transport and 1b) using UPnP transport
2. Configuration of a WiFi client, using an AP with a built-in registrar
   2.1. One WiFi-enabled laptop is setup as the AP with registrar functionality Another WiFi-enabled laptop is setup as a station acting as an Enrollee
3. Configuration of a WiFi client using an external registrar. AP acts as a proxy and communicates with the client over EAP and with the Registrar over UPnP.
   3.1. One WiFi-enabled laptop is setup as a station acting as an Enrollee
   3.2. Second WiFi-enabled laptop is setup as the AP with proxy functionality
   3.3. Third laptop is setup as the registrar. The registrar and the AP are connected over Ethernet.

6.5.2 Running the WPS command-line application

Run the protocol from the console.

First, run UPNP deamon like below:

wscd -w /etc/xml -m 1 -d 3 & (if your xml file in /etc/xml)

use iwpriv command trigger wps, like below:

```
iwpriv ra0 set WscConfMode=7
iwpriv ra0 set WscConfStatus=1
iwpriv ra0 set WscMode=1
iwpriv ra0 set WscPinCode=31668576
iwpriv ra0 set WscGetConf=1
iwpriv ra0 set WscStatus=0
```

Note:

1. AP services as Enrollee:
   1.1. If AP-Enrollee SC state is 0x1, AP will restart with new configurations.
   1.2. If AP-Enrollee SC state is 0x2, AP sends own configurations to external-registrar and ignores configurations from external-registrar.
2. AP services as Registrar:
   2.1. If AP-Registrar SC state is 0x1, the security mode will be WPAPSK/TKIP and generate random 64bytes psk; after process, AP will restart with new security.
3. WPS AP only services one WPS client at a time.
   3.1. WPS AP only can work in ra0.
   3.2. After WPS configuration finishes, Ralink AP driver writes new configuration to Cfg structure and DAT file.
4. Write items to MBSSID Cfg structure are as below:
   4.1. Ssid
   4.2. AuthMode
   4.3. WepStatus
   4.4. PMK
   4.5. DefaultKeyId.
5. Write items to SharedKey table are as below:
   5.1. Key
   5.2. CipherAlg
6. Write items to DAT file are as belw:
   6.1. SSID
   6.2. AuthMode
   6.3. EncrypType
   6.4. WPAPSK
   6.5. WscConfStatus
   6.6. DefaultKeyID

6.5.3 Initial AP setup with Registrar Configuring AP (EAP/UPnP)

To run command-line console in this mode do:

[Unconfigured AP] ←←EAP/UPnP [Registrar]

Note:

Please make sure upnp deamon is running. After the success of WPS registration, Configured AP will act as a proxy forward EAP and Upnp.)

1. PIN

(1) on AP side

```
iwpriv ra0 set WscConfMode=7
iwpriv ra0 set WscConfStatus=1
iwpriv ra0 set WscMode=1
iwpriv ra0 set WscGetConf=1
```

(2) on Registrar side

When prompted for the enrollee's PIN, Enter the AP's PIN. Enter the new SSID and new Security for the AP when prompted.
The registration process will start, and the application will display the result of the process on completion.

2. PBC

(1) on AP side

```
iwpriv ra0 set WscConfMode=7
iwpriv ra0 set WscConfStatus=1
iwpriv ra0 set WscMode=2
iwpriv ra0 set WscGetConf=1
```

(2) on Registrar side

Select push-button".
The registration process will start, and the application will display the result of the process on completion.

The security config will be written out to the AP and registrar config files.

6.5.4 Adding an Enrollee to AP+Registrar (EAP)

To run command-line console in this mode do:

[AP+Registrar] ←←AP [Client]

Note:

Please make sure WPS AP configure status is configured, if AP is un-configure, when WPS AP configure client, it will change configure status to configured and auth mode are WPA-PSK)

1. PIN

(1) on AP side

```
iwpriv ra0 set WscConfMode=7
iwpriv ra0 set PinCode=31668576 (enter the enrollee's PIN, the PIN from WPS client)
iwpriv ra0 set WscMode=1
iwpriv ra0 set WscGetConf=1.
```

The registration process will begin, and the console will display the result of the process on completion.

(2) on Client (Enrollee) side

Select PIN process.
The process will start, and the application will display the result of the process on completion

2. PBC

(1) on AP side

```
iwpriv ra0 set WscConfMode=7
iwpriv ra0 set WscMode=2
iwpriv ra0 set WscGetConf=1.
```

The registration process will start, and the application will display the result of the process on completion.

(2) on Client (Enrollee) side

Select PBC process.
The process will start, and the application will display the result of the process on completion

If the registration is successful, on the client will be re-configured with the new parameters, and will connect to the AP with these new parameters.

6.5.5 Adding an Enrollee with Eternal Registrar (UPnP/EAP)

To run command-line console in this mode do:

[Registrar] ←←UPnP [AP] ←←EAP [Client]

1. PIN

(1) on Registrar side

When prompted for the enrollee's PIN, Enter the enrollee's PIN.
AP Nothing to be selected..
The registration process will begin, and the application will display the result of the process on completion.

(2) on Client (Enrollee) side

Select PIN process
The process will start, and the application will display the result of the process on completion

2. PBC

(1) on Registrar side

Select "push-button".
AP Nothing to be selected.
The registration process will begin, and the application will display the result of the process on completion.

(2) on Client (Enrollee) side

Select PBC process
The registration process will start, and the application will display the result of the process on completion.

6.6 WPS Config status

6.6.1 Overview

The 'Simple Config State' of WPS attribute in WPS IEs contained in beacon and probe response indicates if a device is configured.If an AP is shipped from the factory in the Not-Configured state (Simple Config State set to 0x01), then the AP must transition to the Configured state (Simple Config State set to 0x02) if any of the following occur:

1. Configuration by an external registrar.

The AP sends the WSC_Done message in the External Registrar configuration process.

2. Automatic configuration by internal registrar.

The AP receives the WSC_Done response in the Enrollee Registration Process from the first Enrollee.

Note:

The internal registrar waits until successful completion of the protocol before applying the automatically generated credentials to avoid an accidental transition from unconfigured to configured in the case that a neighbouring device tries to run WSC before the real enrollee, but fails. A failed attempt does not change the configuration of the AP, nor the Simple Config State.

3. Manual configuration by user.

A user manually configures the AP using whatever interface(s) it provides to modify any one of the following:

- the SSID
- the encryption algorithm
- the authentication algorithm
- any key or pass phrase

If the AP is shipped from the factory in the Not Configured state (Simple Config State set to 0x01), then a factory reset must revert the Simple Config State to Not Configured.

If the AP is shipped from the factory pre-configured with WPA2-Personal mixed mode and a randomly generated key, the Simple Config State may be set to 'Configured' (0x2) to prevent an external registrar from overwriting the factory settings. A factory reset must restore the unit to the same configuration as when it was shipped.

6.7 Basic operation of Ralink WPS AP

6.7.1 Configure APUT using PIN method through a WLAN external Registrar

1. [Ralink AP] - Turn on the Ralink AP
2. [Ralink AP] - To change AP ability “iwpriv ra0 set WscConfMode=7”
3. [Ralink AP] - To change from configured to un-configured state: “iwpriv ra0 set WscConfStatus=1 “
4. [Ralink AP] - To change config method to PIN “iwpriv ra0 set WscMode=1”
5. [Ralink AP] - Trigger Ralink AP start process WPS protocol “iwpriv ra0 set WscGetConf=1”
6. [Intel WPS STA] - The Registrar on Intel STA will be configured with the new parameters (SSID = “scaptest4.1.2ssid” and WPA(2)-PSK=”scaptest4.1.2psk”) which should be entered when prompted
7. [Intel WPS STA] - Read AP's PIN from console and enter the PIN at Intel STA.
8. [Intel WPS STA] - Verify that Intel STA successes to ping to Ralink AP
9. [Ralink STA] - Manually configure Ralink STA with the new parameters (SSID = “scaptest4.1.2ssid” and WPA (2)-PSK = “scaptest4.1.2psk”).
10. [Intel WPS STA] - Verify that Intel STA successes to ping to Ralink STA

6.7.2 Configure APUT using PIN method through a wired external registrar

1. [Ralink AP] - Turn on the Ralink AP
2. [Ralink AP] - Connect the Ethernet cable between AP and extern registrar(Windows Vista) and make sure you can pin our device from extern registrar first!
3. [Ralink AP] - To change AP ability “iwpriv ra0 set WscConfMode=7”
4. [Ralink AP] - To change from configured to un-configured state: “iwpriv ra0 set WscConfStatus=1 “
5. [Ralink AP] - To change config method to PIN “iwpriv ra0 set WscMode=1”
6. [Ralink AP] - Trigger Ralink AP start process WPS protocol “iwpriv ra0 set WscGetConf=1”
7. [Microsoft STA] - The Registrar on Microsoft STA will be configured with the new wireless configuration settings (SSID = “scaptest4.1.3ssid” and WPA (2)-PSK=”scaptest4.1.3psk”), which should be entered when prompted.

Please refer to below figures [7-1] to [7-6].

1. [Microsoft STA] - Read AP's PIN from console and enter the PIN at Microsoft STA.

Please refer to below figures [8-1] to [8-2].

2. [Ralink STA] - Manually configure Ralink STA with the new parameters (SSID = “scaptest4.1.3ssid” and WPA (2)-PSK passphrase= “scaptest4.1.3psk”).
3. [Ralink STA] - Verify that Ralink STA successes to ping to Microsoft STA.

6.8 Add devices using external Registrars

1. [Ralink AP] - Turn on the APUT.
2. [Ralink STA] - Turn on the Ralink STA.
3. [Ralink STA] - Push PIN button.

4. [Microsoft STA] - Search will be configure enrollee (you can in control->network and internet->network and sharing center->add a device to the network). Enter the enrollee’s PIN(Ralink STA) at Microsoft STA when prompted.
5. [Ralink AP] - Do not thing.
6. [Ralink STA] - Verify that Ralink STA successes to ping Ralink A.

6.9 How to know WPS AP services as Internal Registrar, Enrollee or Proxy

It depends on the content of EAP-Response/Identity from WPS Client.

When identity is “WFA-SimpleConfig-Registrar-1-0”:

WPS AP would service as Enrollee. (After set trigger command)

When identity is “WFA-SimpleConfig-Enrollee-1-0”:

WPS AP would service as Internal Registrar and Proxy.

Without trigger command, WPS AP services as proxy only.

6.10 How to know WPS AP PinCode

Use ioctl query RT_OID_WSC_PIN_CODE OID to get AP PinCode.

6.11 Notes

1. AP services as Enrollee:
   1.1. If AP-Enrollee SC state is 0x1, AP’s configuration is changeable and will restart with new configurations.
   1.2. If AP-Enrollee SC state is 0x2, AP’s configuration is un-changeable. AP sends own configurations to external-registrar and ignores configurations from external-registrar.
2. AP services as Registrar:
   2.1. If AP-Registrar SC state is 0x1, the security mode will be WPAPSK/TKIP and generate random 64bytes psk; after process, AP will restart with new security.
3. AP services as Proxy:
   3.1. The value of SC state has no effect in proxy mode.
   3.2. WPS AP only services one WPS client at a time.
   3.3. WPS AP only can work in ra0.

6.12 New files for WPS AP

- wsc.c
- wsc_tlv.c
- sha2.c
- hmac.c
- dh_key.c
- evp_enc.c

6.13 New compile flag for WPS AP

```
WFLAGS += -DWSC_SUPPORT
```

6.14 New items for RT2860AP.dat file

```
WscConfMode=0
WscConfStatus=1
```

6.15 Related Documents

1. ~~Wi-Fi Protected Setup Specification v1.0~~ (member only)
2. ~~Wi-Fi Protected Setup White Paper~~
3. ~~Introducing Wi-Fi Protected Setup~~
4. ~~WSC Linux* Reference Implementation~~
5. ~~How to Use Windows Connect Now Configuration to Enable Simple Setup for Consumer Wi-Fi Networks [WinHEC 2006; 5.83 MB]~~
6. ~~Network Infrastructure Device Implementer's Guide~~

6.16 UPNP Daemon HOWTO

6.16.1 Build WPS UPnP Daemon

6.16.1.1 REQUIREMENTS:

1. Linux platform
2. Ralink wireless driver version which support WPS
3. Libupnp

   You can download the libupnp source code from the following URL:
   http://upnp.sourceforge.net/
   libupnp-1.3.1 is preferred version. For other versions, you may need to patch our modification to the library yourself.
4. POSIX thread library

   Both libupnp and our WPS UPnP daemon need the POSIX thread library, following are recommended pthread library version.

   For uCLibc, need the version >= 0.9.27
   For GLIBC, need the version >= 2.3.2

   If your pthread library is older than upper list, you may need to upgrade it.

6.16.1.2 BUILD AND RUN:

1. Modify the "$(work_directory)/wsc_upnp/Makefile" and change the compile flags depends on your target platform.

   Ex. For arm-Linux target platform, you may need to set the following fags:

   CROSS_COMPILE = arm-Linux-
   TARGET_HOST = arm-Linux
   WIRELESS_H_INCLUDE_PATH = /usr/src/kernels/2.6.11-1.1369_FC4-smp-i686/include/
2. Modify the "$(work_directory)/wsc_upnp/libupnp-1.3.1/Makefile.src" and change the configure parameters.

   Ex. For big-endian system, you may need to add CFAGS as following:

   ./configure --host=$(TARGET_HOST) CFLAGS="-mbig-endian"
3. Compile it

   Run "make" in "$(work_directory)/wsc_upnp", after successful compilation, you will get an execution file named "wscd".
4. Install

   Create a sub-directory named "xml" in the "/etc" of your target platform
   Copy all files inside in "$(work_directory)/wsc_upnp/xml" to "/etc/xml"

   Copy the "wscd" to the target platform.
5. Run it

Before run it, be sure the target platform already has set the default route or has a route entry for subnet 239.0.0.0 (For UuPnP Multicast) . Or the WPS daemon will failed when do initialization.
Now you can run it by following command:

```
/bin/wscd –m 1 –d 3
```

6.16.2 Related Documents

1. WPS Specification (Simple_Config_v1.0g.pdf)
2. UPnP Device Architecture 1.0
3. Windows Connect Now-NET Version 1.0
4. WFAWLANConfig:1 Service Template Version 1.01
5. WFA Device:1 Device Template Version 1.01

7 WMM PARAMETERS

7.1 Setting Parameters

1. Set 'WmmCapable' as 1 to turn on WMM QoS support
2. Parameters of 'APAifsn', 'APCwmin', 'APCwmax', 'APTxop', 'APACM' are WMM parameter for AP
3. Parameters of 'BSSAifsn', 'BSSCwmin', 'BSSCwmax', 'BSSTxop', 'BSSACM' are WMM parameter for station
4. Parameter of AckPolicy is for Ack policy which support normal Ack or no Ack
5. Default WMM parameters for STA and AP

1. All WMM parameters do not support iwpriv command but 'WmmCapable'', please store all parameter to RT2800AP.dat, and restart driver.
2. The format for WMM parameter is as followed,
   APAifsn=3;7;1;1 //AC_BE, AC_BK, AC_VI, AC_VO

7.2 How to turn on WMM test in RT2800 SoftAP

1. WmmCapable=1

   For each BSSID:

   0: Disable WMM,

   1: Enable WMM

(If the parameter sets to 1, the relative BSSID will turn on WMM)

2. TxBurst=0

3. Parameters for AP (for each AC (access category))

```
APAifsn=3;7;1;1                    // AC_BE;AC_BK;AC_VI;AC_VO
APCwmin=4;4;3;2                // AC_BE;AC_BK;AC_VI;AC_VO
APCwmax=6;10;4;3               // AC_BE;AC_BK;AC_VI;AC_VO
APTxop=0;0;94;47               // AC_BE;AC_BK;AC_VI;AC_VO
APACM=0;0;0;0                      // AC_BE;AC_BK;AC_VI;AC_VO
```

4. Parameters for all STAs (for each AC (access category))

```
BSSAifsn=3;7;2;2                   // AC_BE;AC_BK;AC_VI;AC_VO
BSSCwmin=4;4;3;2               // AC_BE;AC_BK;AC_VI;AC_VO
BSSCwmax=10;10;4;3                 // AC_BE;AC_BK;AC_VI;AC_VO
BSSTxop=0;0;94;47              // AC_BE;AC_BK;AC_VI;AC_VO
BSSACM=0;0;0;0                     // AC_BE;AC_BK;AC_VI;AC_VO
```

5. Ack policy

```
AckPolicy=0;0;0;0              // AC_BE;AC_BK;AC_VI;AC_VO;
                                // 0: Normal ACK, 1: No ACK
```

All default values comply with Wi-Fi spec.

1. WmmCapable=1

For each BSSID:

0: Disable WMM,
1: Enable WMM

(If the parameter sets to 1, the relative BSSID will turn on WMM)

2. TxBurst=0
3. Parameters for AP (for each AC (access category))

```
APAifsn=3;7;1;1      // AC_BE;AC_BK;AC_VI;AC_VO
APCwmin=4;4;3;2      // AC_BE;AC_BK;AC_VI;AC_VO
APCwmax=6;10;4;3 // AC_BE;AC_BK;AC_VI;AC_VO
APTxop=0;0;94;47     // AC_BE;AC_BK;AC_VI;AC_VO
APACM=0;0;0;0        // AC_BE;AC_BK;AC_VI;AC_VO
```

4. Parameters for all STAs (for each AC (access category))

```
BSSAifsn=3;7;2;2     // AC_BE;AC_BK;AC_VI;AC_VO
BSSCwmin=4;4;3;2     // AC_BE;AC_BK;AC_VI;AC_VO
BSSCwmax=10;10;4;3        // AC_BE;AC_BK;AC_VI;AC_VO
```

```
BSSTxop=0;0;94;47   // AC_BE;AC_BK;AC_VI;AC_VO
BSSACM=0;0;0;0      // AC_BE;AC_BK;AC_VI;AC_VO
```

5. Ack policy

```
AckPolicy=0;0;0;0   // AC_BE;AC_BK;AC_VI;AC_VO;
// 0: Normal ACK, 1: No ACK
```

- All default values comply with Wi-Fi spec.

7.3 The ACKs

1. Current driver of RT2800AP only support NORMAL_ACK and NO_ACK.

   Section 11.1, item 4

   Parameter of AckPolicy is for an Ack policy which supports normal Ack or no Ack .

   The other two ack types have to be supported by the hardware.

2. The difference of ACKs

   a. NORMAL_ACK is used to ACK data packet.

   b. NO_ACK is used never ACK any data packet.

   c. NO_EXPLICIT_ACK have two ways to implement,

      By received packet count threshold to ACK.

      By timeing period threshold to ACK.

   d. BLOCK_ACK is used to ACK data packet per ACK request packet received.

      If peer didn't request to ACK then never ACK.

      This type of ACK is depends on what AIR quality is.

      1.) AIR quality is bad, then the ACK should be mostly required.

      2.) AIR quality is good, then the ACK period maybe longer or even needn't ACK.

3. Reference:

   Below table is pasted from IEEE802.11e-D13.0 for your reference.(Page 27 and 28)

Table 3.2—Ack policy field in QoS control field of QoS data frames

Bits in QoS Control field		Meaning
Bit 5	Bit 6	
0	0	Normal acknowledgement. The addressed recipient returns an ACK or QoS +CF-Ack frame after a SIFS period, according to the procedures defined in 9.2.8, 9.3.3 and 9.9.2.3. The Ack Policy field is set to this value in all directed frames in which the sender requires acknowledgement.

		For QoS Null (no data) frames, this is the only permissi-ble value for the Ack Policy field.
1	0	No Acknowledgement. The addressed recipient takes no action upon receipt of the frame. More details are provided in 9.11. The Ack Policy is set to this value in all directed frames in which the sender does not require acknowledgement. This combination is also used for broadcast and multicast frames that use the QoS frame format.
0	1	No Explicit Acknowledgement. There may be a response frame to the frame that is received, but it is neither the ACK nor any Data frame of subtype +CF-Ack. For Data frames of subtype QoS CF-Poll and subtype QoS CF-Ack+CF-Poll, this is the only permissible value for the Ack Policy field.
1	1	Block Acknowledgement. The addressed recipient takes no action upon the receipt of the frame except for recording the state. The recipient can expect a BlockAckReq frame in the future to which it responds using the procedure described in 9.10.

7.4 Access Precedence and Outgoing Frame Classification

1. 802.1e-D13

1.1. Section 7.3.2.16 Traffic Classification (TCLAS) Element

Table 20.7—Frame classifier type

Classifier Type	Classifier Parameters
0	Ethernet parameters
1	TCP/UDP IP parameters
2	IEEE 802.1D/Q Parameters
3-255	Reserved

1.2. Section 9.1.3.1 HCF contention-based channel access (EDCA)

Table 20.23—User priority to Access Category mappings

Priority	User priority (UP - Same as 802.1D User Priority)	802.1D Designation	Access Category (AC)	Designation (Informative)
lowest	1	BK	AC_BK	Background
	2	-	AC_BK	Background
	0	BE	AC_BE	Best Effort
	3	EE	AC_BE	Best Effort
	4	CL	AC_VI	Video
	5	VI	AC_VI	Video
	6	VO	AC_VO	Voice
highest	7	NC	AC_VO	Voice

2. 802.1Q-2003

2.1. Section 8.9 VLAN classification

3. 802.1q-rev-d4.0-2005-05-19

3.1. Section 6.8 Protocol VLAN classification

3.2. Section 9. Tagged frame format

Table 9-1—802.1Q Ethernet Type allocations		
Tag Type	Name	Value
VLAN TAG	802.1Q Tag Protocol Type (802.1QTagType)	81-00

4. RFC 2474

Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers
(802.11e - Differentiated Services Code Point (DSCP))

5. RFC 791

Internet Protocol

6. RFC 795

6.1. Service mappings – TOS of IP Header

The IP Type of Service has the following fields:

Bit 0-2	Precedence.
Bit 3	0 = Normal Delay, 1 = Low Delay.
Bit 4	0 = Normal Throughput, 1 = High Throughput.
Bit 5	0 = Normal Relibility, 1 = High Relibility.
Bit 6-7	Reserved for Future Use.

0	1	2	3	4	5	6	7
PRECEDENCE			D	T	R	0	0

111 - Network Control				
110 - Internetwork Control				
101 - CRITIC/ECP				
100 - Flash Override				
011 - Flash				
010 - Immediate				
001 – Priority				
000 - Routine				

7.5 Supported Parameters in RT2860AP.dat

7.5.1 WmmCapable=Value

Value:

0:	Disable
1:	Enable

7.5.2 DLSCapable=Value

Value:

0:	Disable
1:	Enable

7.5.3 APAifsn=Value

Value:

```
APAifsn=3;7;1;1            // AC_BE, AC_BK, AC_VI, AC_VO
```

7.5.4 APCwmin=Value

Value:

```
APCwmin=4;4;3;2 // AC_BE, AC_BK, AC_VI, AC_VO
```

7.5.5 APCwmax =Value

Value:

```
APCwmax=6;10;4;3       // AC_BE, AC_BK, AC_VI, AC_VO
```

7.5.6 APTxop =Value

Value:

```
APTxop=0;0;94;47 // AC_BE, AC_BK, AC_VI, AC_VO
```

7.5.7 APACM =Value

Value:

```
APACM=0;0;0;0           // AC_BE, AC_BK, AC_VI, AC_VO
```

7.5.8 BSSAifsn =Value

Value:

```
BSSAifsn=3;7;2;2        // AC_BE, AC_BK, AC_VI, AC_VO
```

7.5.9 BSSCwmin =Value

Value:

```
BSSCwmin=4;4;3;2// AC_BE, AC_BK, AC_VI, AC_VO
```

7.5.10 BSSCwmax =Value

Value:

```
BSSCwmax=10;10;4;3      // AC_BE, AC_BK, AC_VI, AC_VO
```

7.5.11 BSSTxop =Value

Value:

```
BSSTxop=0;0;94;47       // AC_BE, AC_BK, AC_VI, AC_VO
```

7.5.12 BSSACM =Value

Value:

```
BSSACM=0;0;0;0          // AC_BE, AC_BK, AC_VI, AC_VO
```

7.5.13 AckPolicy =Value

Value:

```
AckPolicy=0;0;0;0       // AC_BE, AC_BK, AC_VI, AC_VO
```

7.5.14 APSDCapable=Value

Value [Valid on WmmCapable=1]

0:	Disable
1:	Enable

7.5.15 EthWithVLANTag=Value [RTL865x Only]

Value:

0:	Disable
1:	Enable

7.6 iwpriv ra0 set [parameters]=[Value]

Syntax:	Example
Section# parameters Explanation Value: 0: … 1: …	6.6.1 WmmCapable Set WmmCapable Enable or Disable Value: 0: Disable 0: Enrollee

7.6.1 WmmCapable

Set WmmCapable Enable or Disable

Value:

0:	Disable
1:	Enable

8 IEEE802.11H+D

DFS - Dynamic Frequency Selection

8.1 IEEE802.11d

Regulatory Domains

1. To turn on IEEE802.11d, just fill up the parameter of ‘CountryCode’, according to ISO3166 code list. This parameter can work in A/B/G band.

2. The parameter of “CountryCode’ needs to match with ‘CountryRegion’ or ‘CountryRegionABand’ depends on A or B/G band

3. Wi-Fi test requirement for IEEE802.11d

   Country code IE(0x07) includes in beacon frame and probe response

   Power constraint IE(32) includes in beacon frame and probe response

8.2 IEEE802.11h

Spectrum and Transmit Power Management

1. To turn on IEEE802.11h, just fill up the parameters of ‘IEEE80211H’, ‘AutoChannelSelect’ as 1, WirelessMode set as 3 to support A band. This parameter can work in only A band.

2. Use ‘CSPeriod’ to determine how many beacons before channel switch

3. Driver will turn off BBP tuning temporarily in radar detection mode

4. If turn on IEEE802.11h, AP will have 60sec to do channel available check, and will not send beacon and can not be connect.

5. Wi-Fi test requirement for IEEE802.11h

   Force AP switch channel, AP will stop beacon transmit between 15 sec

   At least five beacon includes channel switch announcement IE (37)in beacon frame

6. ETSI test requirement, please refer to ETSI EN 301 893 for V1.2.3 detail

8.3 Supported Parameters in RT2860AP.dat

8.3.1 IEEE80211H=Value

Spectrum management. This field can only be enabled in A band

Value:

0:	Disable
1:	Enable

8.3.2 CSPeriod=Value

Value:

0 ~ 255

Note:

Channel switch period (Beacon count), unit is based on Beacon interval.

8.3.3 RDRegion

Set radar detection duration region.

Value:

CE
FCC
JAP
JAP_W53
JAP_W56

8.3.4 CarrierDetect

Enable or Disable Carrier Detection.

Value:

0:	Disable
1:	Enable

8.3.5 ChannelGeography

For channel list builder.

Value:

0:	Outdoor
1:	Indoor
2:	Both

8.4 iwpriv ra0 set [parameters]=[Value]

Syntax:	Example
Section# parameters	7.4.1 IEEE8021H
Explanation	Spectrum management.
Value:	Value:
0: ...	0: Disable
1: ...	0: Enrollee

8.4.1 IEEE80211H

Spectrum management. This field can only be enabled in A band

Value:

0:	Disable
1:	Enable

8.4.2 CSPeriod

Channel switch period (Beacon count), unit is based on Beacon interval.
The value indicate how many Channel-Switch Announcements will be sent.

Value:

0 ~ 255

8.4.3 FastDfs

Enable or Disable Fast Radar Detection.

Value:

0:	Disable
1:	Enable

8.4.4 ChMovTime

Change channel moving time for DFS testing

Value:

0 ~ 255 sec

8.4.5 CarrierDetect

Enable or Disable Carrier Detection.

Value:

0:	Disable
1:	Enable

8.4.6 ChGeography

For channel list builder.

Value:

0:	Outdoor
1:	Indoor
2:	Both

9 SECURITY POLICY

9.1 All possible combinations of security policy

Type I. No Radius

(Must set parameter of IEEE8021X as FALSE)

	OPEN	SHARED	WEPAUTO
NONE	V	X	X
WEP	V	V	V
802.1x daemon	Off	Off	Off

Type II. With Radius (Non WiFi standard)

(Must set parameter of IEEE8021X as TRUE)

	OPEN
NONE	V
WEP	V
802.1x daemon	On

Type III. With WPA

(Must set parameter of IEEE8021X as FALSE)

	WPAPSK	WPA2PSK	WPAPSK WPA2PSK	WPA	WPA2	WPA WPA2
TKIP	V	V	V	V	V	V
AES	V	V	V	V	V	V
BOTH	V	V	V	V	V	V
802.1x daemon	Off	Off	Off	On	On	On

The “off” of 802.1x daemon means may be off, it also can be “on”

However “on” of 802.1x daemon means must be “on”

There are no relationship between the parameter of IEEE8021X and 802.1x daemon (RT2860apd).

9.2 WPA2 setting

All settings are same as WPA, but modify attributes --- AuthMode, EncrypType, PreAuth, PMKCachePeriod.

9.3 Supported Parameters in RT2860AP.dat

9.3.1 PreAuth=Value

Value:

0:	Disable
1:	Enable

Note:

Set WPA2 PMKID cache timeout period, after time out, the cached key will be delete

9.3.2 AuthMode=Value

Value:

OPEN	
SHARED	
WEPAUTO	
WPAPSK	
WPA	
WPA2PSK	
WPA2	
WPA1WPA2	:WPA/WPA2 mix mode
WPAPSKWPA2PSK	:WPAPSK/WPA2PSK mix mode

Note:

1. WPA and analogous only support TKIP and AES as encryption method.
2. SHARED only supports Wep as encryption method.
3. WEPAUTO means AP can accept STA connect to it using OPEN-WEP or SHARED-WEP

9.3.3 EncrypType=Value

Value:

NONE:	For AuthMode=OPEN
WEP:	For AuthMode=OPEN or AuthMode=SHARED
TKIP:	For AuthMode=WPAPSK/WPA2PSK, WPA/WPA2, mix mode
AES:	For AuthMode=WPAPSK/WPA2PSK, WPA/WPA2, mix mode
TKIPAES:	TKIP/AES mix mode

9.3.4 DefaultKeyID=Value

Value:

1 ~ 4

9.3.5 Key1Type=Value

Value:

0:	Hexadecimal
1:	ASCII

9.3.6 Key1Str=Value

Key1Str1=Value
Key1Str2=Value
Key1Str3=Value
Key1Str4=Value
Key1Str5=Value
Key1Str6=Value
Key1Str7=Value
Key1Str8=Value
(Refer to Q&A – 7)

Value:

10 or 26 hexadecimal characters, eg: 012345678
5 or 13 ASCII characters, eg: passd

9.3.7 Key2Type=Value

Value:

0:	Hexadecimal
1:	ASCII

9.3.8 Key2Str=Value

Key2Str1=Value
Key2Str2=Value
Key2Str3=Value
Key2Str4=Value
Key2Str5=Value
Key2Str6=Value
Key2Str7=Value
Key2Str8=Value
(Refer to Q&A – 7)

Value:

10 or 26 hexadecimal characters, eg: 012345678
5 or 13 ASCII characters, eg: passd

9.3.9 Key3Type=Value

Value:

0:	Hexadecimal
1:	ASCII

9.3.10 Key3Str=Value

Key3Str1=Value
Key3Str2=Value
Key3Str3=Value
Key3Str4=Value
Key3Str5=Value
Key3Str6=Value
Key3Str7=Value
Key3Str8=Value
(Refer to Q&A – 7)

Value:

10 or 26 hexadecimal characters, eg: 012345678
5 or 13 ASCII characters, eg: passd

9.3.11 Key4Type=Value

Value:

0:	Hexadecimal
1:	ASCII

9.3.12 Key4Str=Value

Key4Str1=Value
Key4Str2=Value
Key4Str3=Value
Key4Str4=Value
Key4Str5=Value
Key4Str6=Value
Key4Str7=Value
Key4Str8=Value
(Refer to Q&A – 7)

Value:

10 or 26 hexadecimal characters, eg: 012345678
5 or 13 ASCII characters, eg: passd

9.3.13 WPAPSK=Value

WPAPSK1=Value
WPAPSK2=Value
WPAPSK3=Value
WPAPSK4=Value
WPAPSK5=Value
WPAPSK6=Value
WPAPSK7=Value
WPAPSK8=Value
(Refer to Q&A – 7)

Value:

8 ~ 63 ASCII characters

or
64 hexadecimal characters

9.3.14 RekeyMethod=Value

Value (for WPA/WPA2):

TIME:	Time rekey
PKT:	Packet rekey
DISABLE:	Disable rekey

9.3.15 RekeyInterval=Value

Value (for WPA/WPA2)

0 ~ 0x3fffff
unit: 1 seconds/1000packets

9.3.16 PMKCachePeriod=Value

Value (for WPA2):

0 ~
unit:minute

9.4 iwpriv ra0 set [parameters]=[Value]

Syntax:	Example
Section# parameters	8.4.1 PreAuth
Explanation	Set WPS function
Value:	Value:
0: ...	0: Disable
1: ...	0: Enrollee

9.4.1 PreAuth

Set WPA2 pre-authentication mode.

Value:

0:	Disable
1:	Enable

9.4.2 AuthMode

Set Authentication mode.

Value:

OPEN
WEPAUTO
SHARED
WPAPSK

WPA
WPA2PSK
WPA2
WPA1WPA2
WPAPSKWPA2PSK

9.4.3 EncrypType

Set the Encryption Type.

Value:

NONE
WEP
TKIP
AES
TKIPAES

9.4.4 DefaultKeyID

Set Default Key ID.

Value:

1 ~ 4

9.4.5 Key1

Set Key1 String.

Value:

5 ASCII characters, or
10 hex number, or
13 ASCII characters, or
26 hex numbers

9.4.6 Key2

Set Key2 String.

Value:

5 ASCII characters, or
10 hex number, or
13 ASCII characters, or
26 hex numbers

9.4.7 Key3

Set Key3 String.

Value:

5 ASCII characters, or
10 hex number, or
13 ASCII characters, or
26 hex numbers

9.4.8 Key4

Set Key4 String.

Value:

5 ASCII characters, or
10 hex number, or
13 ASCII characters, or
26 hex numbers

9.4.9 WPAPSK

WPA Pre-Shared Key.

Value:

8~63 ASCII or 64 HEX characters

9.4.10 RekeyMethod

Set group rekey interval-unit's type.

Value:

TIME
PKT
NONE

9.4.11 RekeyInterval

Set group rekey interval. 0 to disable rekey. Unit:1seconds/1000packets dependent on Rekeytype.

Value:

0~0x3FFFFFFF

9.4.12 PMKCachePeriod

Set WPA2 PMKID cache timeout period, after time out, the cached key will be deleted.

Value:

0~   ; unit: minute

9.5 Examples

9.5.1 Example I

On Step-by-Step setting of how to set SoftAP using WPAPSK security mechanism with encryption method TKIP. Assume RT2800 SoftAP set PreShared Key as "myownpresharedkey". Please ensure to set SSID, before/after set WPAPSK.

1. load RT2800ap driver
2. iwpriv ra0 set AuthMode=WPAPSK
3. iwpriv ra0 set EncrypType=TKIP
4. iwpriv ra0 set IEEE8021X=0
5. iwpriv ra0 set SSID=myownssid
6. iwpriv ra0 set WPAPSK=myownpresharedkey
7. iwpriv ra0 set DefaultKeyID=2
8. iwpriv ra0 set SSID=myownssid

9.5.2 Example II

On Step-by-Step setting of how to set SoftAP using WEP security mechanism. Assume RT2800 SoftAP uses user-defined key.

1. load RT2800ap driver
2. iwpriv ra0 set AuthMode=SHARED
3. iwpriv ra0 set EncrypType=WEP
4. iwpriv ra0 set IEEE8021X=0
5. iwpriv ra0 set Key1=0123456789
6. iwpriv ra0 set DefaultKeyID=1
7. iwpriv ra0 set SSID=myownssid

9.5.3 Example III

On Step-by-Step setting of how to set SoftAP using OPEN security mechanism.

1. load RT2800ap driver
2. iwpriv ra0 set AuthMode=OPEN
3. iwpriv ra0 set EncrypType=NONE
4. iwpriv ra0 set IEEE8021X=0
5. iwpriv ra0 set SSID=myownssid

9.5.4 Example IV

Change setting to WPAPSK with AES.

1. iwpriv ra0 set AuthMode=WPAPSK
2. iwpriv ra0 set EncrypType=AES
3. iwpriv ra0 set IEEE8021X=0
4. iwpriv ra0 set SSID=MySsid
5. iwpriv ra0 set WPAPSK=MyPassword
6. iwpriv ra0 set DefaultKeyID=2
7. iwpriv ra0 set SSID=MySsid

Note:

Step 3 is a must for calculating WPAPSK Key, which requires both SSID and WPAPSK.
Step 5 will make driver to reload all settings. step5 must be the same with step3.

9.5.5 Example V

Change setting to OPEN, no 802.1x.

1. iwpriv ra0 set AuthMode= OPEN
2. iwpriv ra0 set EncrypType= NONE
3. iwpriv ra0 set IEEE8021X=0
4. iwpriv ra0 set SSID=MySsid

Note:

Step 3 will make driver to reload all setting.

10 WDS

Wireless Distribution System

10.1 WDS Setup

1. edit file in /etc/Wireless/RT2860AP/RT2860AP.dat to add

   (a). WdsEnable=1

   (b). WdsList=00:10:20:30:40:50; ;Another AP's MAC address

   (c). WdsEncrypType=NONE ;the encryption type in WDS interface

2. edit script file bridge_setup according to the number of WDS-AP

   add "/usr/sbin/brctl addif br0 wds0" and "/sbin/ifconfig wds0 0.0.0.0" to relative place.

3. re-load driver(rt2860ap.o)

4. run bridge_setup

10.2 WDS Usage

1. Each WDS APs need seting as same channel, encryption type.(not support mixed mode, like WPAPSKWPA2PSK).

2. WDS Security support up to pre-shared key, this is inter AP’s security and no 802.1x support.

3. In case want have auto-learning WDS peers, Lazy mode is the one. But have to note that can’t set each AP to Lazy mode, otherwise no addr4 will be carried by each AP. This means that there at least has one AP have to fill WDS list.

10.3 WDS Individual Encryption

If the WDS mode is enabled and set as LAZY mode, the all WDS-link shall share the same encryption type and key material(based on wds0 setting). Otherwise, each WDS-link has own individual security setting.

No matter what WDS mode is set, it has no any relation to the encryption of BSSIDs.

Although the new WDS implementation has been provided, it alos supports previous WDS configuration.

A: WdsKey:

WdsKey is used for all WDS interface and support AES or TKIP encryption only. WEP key will follow main-AP’s setting. Wds0Key/Wds1Key/Wds2Key/Wds3Key is used to support all of the encryption per WDS interface, WEP, TKIP, and AES.

B: AuthMode:

Follows the main-AP’s setting.

Case 1: main AP choose open mode, and WDS choose WEP or AES

AuthMode: OPEN, take from main-AP

EncrypType: WDS = WEP or AES

WEP key will follow main-AP’s setting,

Or, take from Wds0Key… depend on which WDs interface.

AES key will take from WdsKey or Wds0Key, depend on which WDs interface.

Please use PING to check the data whether encrypted or not.

Case 2: main AP is wep mode, and WDS is AES mode

AuthMode: WEP

10.4 Supported Parameters RT2860AP.dat

10.4.1 WdsEnable=Value

Value:

0: Disable - Disable all WDS function.
1: Restrict mode - Same as Repeater mode.
2: Bridge mode - Turn on WDS function, the peer WDS APs are according to the mac address listed in "WdsList" field below. In this mode, AP will not send beacon out and will not deal with probe request packets, therefore STA will not possible to connect with it.
3: Repeater mode - Turn on WDS function, the peer WDS APs are according to the mac address listed in "WdsList" field below.
4: Lazy mode - Turn on WDS function, and auto learning from WDS packet which with addr4 field.

10.4.2 WdsList=Value

Value:

[Mac Address];[Mac Address];...

E.g.

00:10:20:30:40:50;0A:0b:0c:0D:0e:0f;1a:2b:3c:4d:5e:6f

Note:

It supports the maximum WDS-link is 4.

10.4.3 WdsEncrypType=Value;Value;Value;Value

Value:

NONE
WEP
TKIP

AES

E.g.

OPEN;TKIP;WEP;AES
The encrptytion of wds0 is OPEN
The encrptytion of wds1 is TKIP
The encrptytion of wds2 is WEP
The encrptytion of wds3 is AES

10.4.4 WdsKey=Value

The key material of WDS link.

Value:

10 or 26 hexadecimal characters (eg: 1234567890) for WEP
5 or 13 ASCII characters (eg: 12345) for WEP
8 ~ 63 ASCII characters for TKIP or AES
64 hexadecimal characters for TKIP or AES

Depends on the setting of WdsEncrypType.

Main BSSID's EncrypType	WDS's WdsEncrypType	Peer AP WDS's WdsEncrypType	Remark
NONE	NONE	NONE	
WEP	WEP	WEP	Using legacy key setting method
TKIP	TKIP	TKIP	WDS's key is from WdsKey
TKIP	AES	AES	WDS's key is from WdsKey
AES	TKIP	TKIP	WDS's key is from WdsKey
AES	AES	AES	WDS's key is from WdsKey
TKIPAES	TKIP	TKIP	WDS's key is from WdsKey
TKIPAES	AES	AES	WDS's key is from WdsKey

10.4.5 Wds0Key=Value

The key material of wds0 link.

Value:

10 or 26 hexadecimal characters (eg: 1234567890) for WEP
5 or 13 ASCII characters (eg: 12345) for WEP
8 ~ 63 ASCII characters for TKIP or AES
64 hexadecimal characters for TKIP or AES

10.4.6 Wds1Key=Value

The key material of wds1 link.

Value:

10 or 26 hexadecimal characters (eg: 1234567890) for WEP
5 or 13 ASCII characters (eg: 12345) for WEP
8 ~ 63 ASCII characters for TKIP or AES

64 hexadecimal characters for TKIP or AES

10.4.7 Wds2Key=Value

The key material of wds2 link.

Value:

10 or 26 hexadecimal characters (eg: 1234567890) for WEP
5 or 13 ASCII characters (eg: 12345) for WEP
8 ~ 63 ASCII characters for TKIP or AES
64 hexadecimal characters for TKIP or AES

10.4.8 Wds3Key=Value

The key material of wds3 link.

Value:

10 or 26 hexadecimal characters (eg: 1234567890) for WEP
5 or 13 ASCII characters (eg: 12345) for WEP
8 ~ 63 ASCII characters for TKIP or AES
64 hexadecimal characters for TKIP or AES

10.4.9 WdsDefaultKeyID=Value

The default key index setting.

Value:

1~4

E.g.

1;2;3;4
The key index of wds0 is 1
The key index of wds1 is 2
The key index of wds2 is 3
The key index of wds3 is 4

10.4.10 WdsPhyMode=Value

Value:

CCK
OFDM
HTMIX
GREENFIELD

11 AUTHENTICATOR

rt2860apd - user space IEEE 802.1X Authenticator

11.1 Introduction

rt2860apd is an optional user space component for RT2800 SoftAP driver.

It adds 802.1x Authenticator feature using external RADIUS Authentication Server (AS).

11.1.1 IEEE 802.1X features in rt2860apd

IEEE Std 802.1X-2001 is a standard for port-based network access control. It introduces a extensible mechanism for authenticating and authorizing users.

rt2860apd implements partial IEEE 802.1x features that helps AS authorizing Supplicant and in the mean time proves itself a valid Authenticator for AS.

Noticed that Key management state machine is not included in rt2860apd. And those keys management is included in RT2800 SoftAP driver.

rt2860apd relays the frames between the Supplicant and the AS. Not until either one timeout or Success or Fail frame indicated does rt2860apd finish the authentication process. The port control entity is implemented in SoftAP driver for RT2800.

11.1.2 How to start rt2860apd

Manually start rt2860apd, type “$rt2860apd”.

11.1.3 rt2860apd configuration for IEEE 802.1X

When rt2860apd starts, it reads the configuraion file to derive parameters. For any changes to make, one need to first edit the configuration file, then restart rt2860apd.

Please add 4 required parameters in the configuration file for RT2800 a/b/g SoftAP driver.

```
RADIUS_Server='192.168.2.3'
RADIUS_Port='1812'
RADIUS_Key='password'
own_ip_addr='your_ip_addr'
```

The word in ' ' must be replaced with your own correct setting. Please make sure 'your_ip_addr' and RADIUS_Server is connected and RADIUS_Server's IAS (or related) services are started.

The optional variables as below,

session_timeout_interval is for 802.1x reauthentication setting.
set to zero to disable 802.1x reauthentication service for each session.
session_timeout_interval unit is second and must be larger than 60.
For example,

- session_timeout_interval = 120

  reauthenticate each session every 2 minutes.

- session_timeout_interval = 0

  disable reauthenticate service.

EAPifname is assigned as the binding interface for EAP negotiation.
Its default value is "br0". But if the wireless interface doesn't attach to bridge interface or the bridge interface name isn't "br0", please modify it.
For example,

- EAPifname=br0

PreAuthifname is assigned as the binding interface for WPA2 Pre-authentication.
Its default value is "br0". But if the ethernet interface doesn't attach to bridge interface or the bridge interface name isn't "br0", please modify it.
For example,

- PreAuthifname=br0

11.1.4 Support Multiple RADIUS Server

We use complier option to turn on/off the multiple RADIUS servers for 802.1x.

If you want to enable the feature, make sure that "MULTIPLE_RADIUS" is defined in Makefile. Default is disabled. Besides, you must modify the file "RT2860AP.dat" to co-operate with 802.1x. We extend some variables to support individual RADIUS server IP address, port and secret key for MBSS.

E.g.

RADIUS_Server=192.168.2.1;192.168.2.2;192.168.2.3;192.168.2.4

RADIUS_Port=1811;1812;1813;1814

RADIUS_Key=ralink_1;ralink_2;ralink_3;ralink_4

Or

RADIUS_Key1=ralink_1

RADIUS_Key2=ralink_2

RADIUS_Key3=ralink_3

RADIUS_Key4=ralink_4
For backward compatibility, the driver parses "RADIUS_Key" or RADIUS_KeyX"(X=1~4) for radius key usage. But the paramter "RADIUS_Key" has the first priority.

This implies,

The RADIUS server IP of ra0 is 192.168.2.1, its port is 1811 and its secret key is ralink_1.

The RADIUS server IP of ra1 is 192.168.2.2, its port is 1812 and its secret key is ralink_2.

The RADIUS server IP of ra2 is 192.168.2.3, its port is 1813 and its secret key is ralink_3.

The RADIUS server IP of ra3 is 192.168.2.4, its port is 1814 and its secret key is ralink_4.
If your wireless interface prefix is not "ra", please modify these variables.

11.1.5 Enhance dynamic wep keying

In OPEN-WEP with 802.1x mode, the authentication process generates broadcast and unicast key. The unicast key is unique for every individual client so it is always generated randomly by 802.1x daemon. But the broadcast key is shared for all associated clients; it can be pre-set manually by users or generated randomly by 802.1x daemon.

Through the parameter "DefaultKeyID" and its corresponding parameter "KeyXStr"(i.e. X = the value of DefaultKeyID) in RT2860Ap.dat, the 802.1x daemon would use it as the broadcast key material. But if the corresponding parameter "KeyXStr" is empty or unsuitable, the broadcast key would be generated randomly by the 802.1x daemon.

The 802.1x daemon need to read RT2860AP.dat to decide whether the broadcast key is generated randomly or not, so please update the RT2860AP.dat and restart rt2860apd if those correlative parameters are changed.

11.2 Supported Parameters in RT2860AP.dat

11.2.1 IEEE8021X=Value

Value:

0: Disable
1: Enable

Note:

This field is enable only when Radius-WEP mode on, otherwise must disable

11.2.2 EAPifname=Value

Value:

br0
The binding interface for EAP negotiation.

11.2.3 PreAuthifname=Value

Value:

br0
The binding interface for WPA2 Pre-authentication.

11.2.4 RADIUS_Server=xxx.xxx.xx.xx

IP for Radius server

11.2.5 RADIUS_Port=Value

Value:

1812 (Default)
This is port number for IAS service in Authentication Server(AS).

11.2.6 RADIUS_Key=Value

RADIUS_Key1=Value
RADIUS_Key2=Value
RADIUS_Key3=Value
RADIUS_Key4=Value
RADIUS_Key5=Value
RADIUS_Key6=Value
RADIUS_Key7=Value
RADIUS_Key8=Value

Value:

It is suggested that you set the string to longer than 8 ASCII characters.
This is Radius Secret shared with Authenticator and AS.

11.2.7 own_ip_addr=xxx.xxx.xx.xx

This is the ip address of our SoftAP.

11.2.8 session_timeout_interval = Value

Value:

0, or >=60
0 to disable reauthentication for every session.
>=60 to set reauthenticaion interval with unit of second.

Note:

xxx.xxx.xx.xx is a IP address
* represents the parameters for 802.1x daemon-RT2860apd

11.3 iwpriv ra0 set [parameters]=[Value]

Syntax:	Example
Section# parameters Explanation Value: 0: ... 1: ...	10.3.1 IEEE8021X Enable 802.1x Value: 0: Disable 0: Enable

11.3.1 IEEE8021X

Set 8021X-WEP mode on, this field is enabled only when Radius-WEP or Radius-NONE mode on, otherwise must disable.

Value:

0: Disable
1: Enable

11.4 Examples

11.4.1 Example I

This is a step-by-step guide to set SoftAP using WPA security mechanism. Assume RT2800 SoftAP has ip address 192.168.1.138, AS (Authentication Server) has IP address 192.168.1.1, Radius Secret is myownkey.

1. load RT2800ap driver

   ```
   $insmod rt2860ap.o
   ```

2. First edit configuration file with correct value, esp. the following parameters that relate to the authentication features of RT2800AP

   ```
   RADIUS_Server=192.168.1.1

   RADIUS_Port=1812

   RADIUS_Key=myownkey

   own_ip_addr=192.168.1.138
   ```

3. start RT2800apd daemon by typing.

   ```
   $rt2860apd
   ```

4. iwpriv ra0 set AuthMode=WPA
5. iwpriv ra0 set EncrypType=TKIP
6. iwpriv ra0 set DefaultKeyID=2
7. iwpriv ra0 set IEEE8021X=0
8. iwpriv ra0 set SSID=myownssid

11.4.2 Example II

Change 802.1x settings to WPA with TKIP, using 802.1x authentication.

1. Modify 4 parameters

   ```
   RADIUS_Server=192.168.2.3

   RADIUS_Port=1812

   RADIUS_Key=password

   own_ip_addr=192.168.1.123
   ```
   in the RT2860AP.dat and save.

2. iwpriv ra0 set AuthMode=WPA
3. iwpriv ra0 set EncrypType=TKIP
4. iwpriv ra0 set IEEE8021X=0
5. iwpriv ra0 set SSID=myownssid

Note:

Step 4 restarts the rt2860apd, and is essential.

11.4.3 Example III

Change setting to OPEN/WEP with 802.1x.

1. iwpriv ra0 set AuthMode= OPEN
2. iwpriv ra0 set EncrypType= WEP
3. iwpriv ra0 set IEEE8021X=1

Note:

"IEEE8021X=1" only when Radius-WEP or Radius-NONE mode on, otherwise must "IEEE8021X=0".

11.4.4 Example IV

Change setting to OPEN/NONE with 802.1x.

1. iwpriv ra0 set AuthMode= OPEN
2. iwpriv ra0 set EncrypType= NONE
3. iwpriv ra0 set IEEE8021X=1

Note:

"IEEE8021X=1" only when Radius-WEP or Radius-NONE mode on , otherwise must "IEEE8021X=0".

12 ATE TEST COMMAND FORMAT

IF YOU ARE NOT FAMILIAR WITH HARDWARE, IT IS RECOMMANDED NOT TO MODIFY HARDWARE DEFAULT VALUE.

12.1 iwpriv ra0 set [parameters]=[val]

Syntax:	Example
Section# parameters Explanation Value: 0: ... 1: ... .: ...	11.1.5 ATECHANNEL Set ATE channel. Value: 1: 2: .:

12.1.1 ATE

Set ATE actions.

Value:

ATESTART	- Stop AP & ATE function.
ATESTOP	- Start AP function.
TXCONT	- Start AP continuous TX, for power mask.
TXCARR	- Start AP carrier test, for frequency calibration.
TXFRAME	- Transmit frame, for EVM.
RXFRAME	- Continuous RX, for PER/FER.

12.1.2 ATEDA

Set ATE frame header addr1.

Value:

xx:xx:xx:xx:xx:xx ; hex

12.1.3 ATESA

Set ATE frame header addr2.

Value:

xx:xx:xx:xx:xx:xx ; hex

12.1.4 ATEBSSID

Set ATE frame header addr3.

Value:

xx:xx:xx:xx:xx:xx ; hex

12.1.5 ATECHANNEL

Set ATE Channel, deimal.

Value:

802.11b/g: 1 ~ 14 depends on CountryRegion setting

12.1.6 ATETXPOW0

Set ATE Tx power for Antenna 1.

Value:

0 ~ 31	; 2.4GHz,5-bits only, deimal
-7 ~15	; 5GHz,5-bits only, deimal

12.1.7 ATETXPOW1

Set ATE Tx power for Antenna 2.

Value:

0 ~ 31	; 5-bits only, decimal
-7 ~15	; 5GHz,5-bits only, deimal

12.1.8 ATETXFREQOFFSET

Set ATE RF frequency offset.

Value:

0 ~ 63	; unit: 2KHz, decimal

12.1.9 ATETXLEN

Set ATE frame length.

Value:

24 ~ 1500	; decimal

12.1.10 ATETXCNT

Set ATE frame Tx count.

Value:

1 ~	; 32-bit, decimal

12.1.11 ATETXMODE (Refer to TxMode)

Set ATE Tx Mode.

Value:

0:	CCK	802.11b
1:	OFDM	802.11g
2:	HT_MIX	802.11b/g/n

3:	Green Field	802.11n

12.1.12 ATETXBW (Refer to TxMode)

Set ATE Tx and Rx Bandwidth.

Value:

0:	20MHz
1:	40MHz

12.1.13 ATETXGI (Refer to TxMode)

Set ATE Tx Guard Interval.

Value:

0:	Long
1:	Short

12.1.14 ATETXMCS (Refer to TxMode)

Set ATE Tx MCS type.

Value:

0 ~ 15

12.1.15 ATETXANT

Set ATE TX antenna.

Value:

0:	All
1:	Antenna one
2:	Antenna two

12.1.16 ATERXANT

Set ATE RX antenna.

Value:

0:	All
1:	Antenna one
2:	Antenna two
3:	Antenna three

12.1.17 ATERXFER

Set ATE to periodically reset and show up RxCount (per-second) and RxTotalCount.

Value:

0:	Disable counter visability
1:	Enable counter visability

12.1.18 ATESHOW

Show all parameters of ATE.

Value:

1

12.1.19 ATEHELP

List all commands of ATE.

Value:

1

12.1.20 ResetCounter

Reset statistic counter.

Value:

0

12.1.21 ATERRF

Read all of the RF registers.

Value:

1

12.1.22 ATEWRF1

Write the RF register 1.

Value:

xxxxxxxx ;32-bit, hex

12.1.23 ATEWRF2

Write the RF register 2.

Value:

xxxxxxxx ;32-bit, hex

12.1.24 ATEWRF3

Write the RF register 3.

Value:

xxxxxxxx ;32-bit, hex

12.1.25 ATEWRF4

Write the RF register 4.

Value:

xxxxxxxx ;32-bit, hex

12.1.26 ATELDE2P

Overwrite all EEPROM contents from "/etc/Wireless/RT2860/(70)AP(/STA)/e2p.bin".

Value:

1

E.g.

iwpriv ra0 set ATELDE2P=1

12.2 Tx Mode, MCS, BW and GI Selection Table

MODE = 0, Legacy CCK	
MCS = 0	Long Preamble CCK 1Mbps
MCS = 1	Long Preamble CCK 2Mbps
MCS = 2	Long Preamble CCK 5.5Mbps
MCS = 3	Long Preamble CCK 11Mbps
MCS = 8	Short Preamble CCK 1Mbps, * illegal rate
MCS = 9	Short Preamble CCK 2Mbps
MCS = 10	Short Preamble 5.5Mbps
MCS = 11	Short Preamble 11Mbps
Notes: Other MCS codes are reserved in legacy CCK mode. BW, SGI and STBC are reserved in legacy CCK mode.	
MODE = 1, Legacy OFDM	
MCS = 0	6Mbps
MCS = 1	9Mbps
MCS = 2	12Mbps
MCS = 3	18Mbps
MCS = 4	24Mbps
MCS = 5	36Mbps
MCS = 6	48Mbps
MCS = 7	54Mbps
Notes: Other MCS code in legacy CCK mode are reserved. When BW = 1, duplicate legacy OFDM is sent. SGI, STBC are reserved in legacy OFDM mode.	

MODE = 2, HT Mixed Mode
MODE = 3, HT Greenfield	

MCS = 0 (1S)	(BW=0, SGI=0) 6.5Mbps
MCS = 1	(BW=0, SGI=0) 13Mbps
MCS = 2	(BW=0, SGI=0) 19.5Mbps
MCS = 3	(BW=0, SGI=0) 26Mbps
MCS = 4	(BW=0, SGI=0) 39Mbps
MCS = 5	(BW=0, SGI=0) 52Mbps
MCS = 6	(BW=0, SGI=0) 58.5Mbps
MCS = 7	(BW=0, SGI=0) 65Mbps
MCS = 8 (2S)	(BW=0, SGI=0) 13Mbps
MCS = 9	(BW=0, SGI=0) 26Mbps
MCS = 10	(BW=0, SGI=0) 39Mbps
MCS = 11	(BW=0, SGI=0) 52Mbps
MCS = 12	(BW=0, SGI=0) 78Mbps
MCS = 13	(BW=0, SGI=0) 104Mbps
MCS = 14	(BW=0, SGI=0) 117Mbps
MCS = 15	(BW=0, SGI=0) 130Mbps
MCS = 32	(BW=1, SGI=0) HT duplicate 6Mbps
Notes: When BW=1, PHY_RATE = PHY_RATE * 2 When SGI=1, PHY_RATE = PHY_RATE * 10/9 The effects of BW and SGI are accumulative. When MCS=0~7(1S, One Tx Stream), STBC option is supported. SGI option is supported. BW option is supported. When MCS=8~15(2S, Two Tx Stream), STBC option is NOT supported. SGI option is supported. BW option is supported. When MCS=32, only SGI option is supported. BW and STBC option are not supported. (BW =1, STBC=0) Other MCS code in HT mode are reserved. When STBC is supported. Only STBC = 1 is allowed. STBC will extend the transmission range but will not increase transmission rate.	

12.3 Examples

*Note : Setting the ATE commands in sequence is strongly suggested.

12.3.1 Check EVM & Power

```
iwpriv ra0 set ATE=ATESTART
iwpriv ra0 set ATEDA=00:11:22:33:44:55
iwpriv ra0 set ATESA=00:aa:bb:cc:dd:ee
iwpriv ra0 set ATEBSSID=00:11:22:33:44:55
iwpriv ra0 set ATECHANNEL=1              ; set Channel
iwpriv ra0 set ATETXMODE=1         ; set TX-Mode.
iwpriv ra0 set ATETXMCS=7          ; set MCS type.
iwpriv ra0 set ATETXBW=0                 ; set Bandwidth
iwpriv ra0 set ATETXGI=0                 ; set Long GI.
iwpriv ra0 set ATETXLEN=1024             ; set packet length.
iwpriv ra0 set ATETXPOW0=18
iwpriv ra0 set ATETXPOW1=18
iwpriv ra0 set ATETXCNT=100000
iwpriv ra0 set ATE=TXFRAME
…
iwpriv ra0 set ATETXPOW0=19
```

```
…
iwpriv ra0 set ATETXPOW0=20
…
iwpriv ra0 set ATE=ATESTART
```

12.3.2 Check Carrier

```
iwpriv ra0 set ATE=ATESTART
iwpriv ra0 set ATECHANNEL=1              ; set Channel
iwpriv ra0 set ATETXMODE=1         ; set TX-Mode.
iwpriv ra0 set ATETXMCS=7          ; set MCS type.
iwpriv ra0 set ATETXBW=0                 ; set Bandwidth
iwpriv ra0 set ATETXCNT=200        ; Tx frame count(decmial)
iwpriv ra0 set ATE=TXFRAME         ; Start Tx Frame(inform BBP to change, modulation mode)
iwpriv ra0 set ATE=TXCARR                ; Start Tx carrier, Measure carrier with instrument
iwpriv ra0 set ATETXPOW0=05
iwpriv ra0 set ATETXPOW1=05
iwpriv ra0 set ATETXFREQOFFSET=19
iwpriv ra0 set ATE=ATESTART
```

12.3.3 Check specturm mask

```
iwpriv ra0 set ATE=ATESTART
iwpriv ra0 set ATECHANNEL=1        ; set Channel
iwpriv ra0 set ATETXMODE=1         ; set TX-Mode.
iwpriv ra0 set ATETXMCS=7          ; set MCS type.
iwpriv ra0 set ATETXBW=0           ; set Bandwidth
iwpriv ra0 set ATETXCNT=200        ; Tx frame count(decmial)
iwpriv ra0 set ATE=TXFRAME         ; Start Tx Frame(inform BBP to change, modulation mode)
iwpriv ra0 set ATE=TXCONT          ; Start continuous TX, Measure specturm mask with instrument
iwpriv ra0 set ATETXPOW0=5
iwpriv ra0 set ATETXPOW1=5
iwpriv ra0 set ATE=ATESTART
```

12.3.4 Frequency offset tuning

```
iwpriv ra0 set ATE=ATESTART
iwpriv ra0 set ATECHANNEL=1        ; set Channel
iwpriv ra0 set ATETXMODE=1         ; set TX-Mode.
iwpriv ra0 set ATETXMCS=7          ; set MCS type.
iwpriv ra0 set ATETXCNT=200        ; Tx frame count(decmial)
iwpriv ra0 set ATETXFREQOFFSET=0         ; Set frequency offset 0(decimal)
iwpriv ra0 set ATE=TXFRAME         ; Start Tx Frame
iwpriv ra0 set ATE=TXCARR          ; Start Tx carrier, Measure carrier frequency with instrument
iwpriv ra0 set ATETXFREQOFFSET=10        ; Dynamic turning frequency offset, 10(decimal)
iwpriv ra0 set ATETXFREQOFFSET=20        ; Dynamic turning frequency offset, 20(decimal)
iwpriv ra0 set ATE=ATESTART        ; Stop, Store the tuning result to EEPROM
```

12.3.5 Rx

```
iwpriv ra0 set ATE=ATESTART
iwpriv ra0 set ATECHANNEL=1        ; set Channel
iwpriv ra0 set ResetCounter=0        ; Reset statistic counter
iwpriv ra0 set ATETXFREQOFFSET=value           ;To use the “value” (decimal) you got in tx calibration
iwpriv ra0 set ATETXMODE=1           ; set TX-Mode.
iwpriv ra0 set ATETXMCS=7            ; set MCS type.
iwpriv ra0 set ATETXBW=0             ; set Bandwidth
iwpriv ra0 set ATE=RXFRAME           ; Start Rx,
iwpriv ra0 set ATERXFER=1            ; show RxCnt and RSSI/per-antenna, Transmit test packets
iwpriv ra0 set ATE=ATESTART          ; Stop
iwpriv ra0 stat                      ; get statistics counter
iwpriv ra0 set ATERXFER=1
iwpriv ra0 set ATERXANT=1

iwpriv ra0 set ATE=ATESTART
iwpriv ra0 set ATERXANT=0
iwpriv ra0 set ATE=RXFRAME
```

12.3.6 Show all ate parameters

```
iwpriv ra0 set ATESHOW=1

Mode=4
TxPower0=0
TxPower1=0
TxAntennaSel=0
RxAntennaSel=0
BBPCurrentBW=0
GI=0
MCS=7
TxMode=1
Addr1=00:11:22:aa:bb:cc
Addr2=00:11:22:aa:bb:cc
Addr3=00:11:22:aa:bb:cc
Channel=1
TxLength=1024
TxCount=40000
TxRate=11
RFFreqOffset=0
```

12.3.7 Online help

```
iwpriv ra0 set ATEHELP=1

ATE=ATESTART, ATESTOP, TXCONT, TXCARR, TXFRAME, RXFRAME
ATEDA
ATESA
ATEBSSID
ATECHANNEL, range:0~14
ATETXPOW0, set power level of antenna 1.
ATETXPOW1, set power level of antenna 2.
ATETXANT, set TX antenna. 0: all, 1: antenna one, 2: antenna two.
ATERXANT, set RX antenna.0: all, 1: antenna one, 2: antenna two, 3: antenna three.
ATETXFREQOFFSET, set frequency offset, range 0~63
ATETXBW, set BandWidth, 0:20MHz, 1:40MHz.
ATETXLEN, set Frame length, range 24~1500
```

ATETXCNT, set how many frame going to transmit.
ATETXRATE, set rate, reference to rate table.
ATETXMCS, set MCS, reference to rate table.
ATETXMODE, set Mode 0: CCK, 1: OFDM, 2: HT-Mix, 3: GreenField, reference to rate table.
ATETXGI, set GI interval, 0: Long, 1: Short
ATERXFER, 0: disable Rx Frame error rate. 1: enable Rx Frame error rate.
ATESHOW, display all parameters of ATE.
ATEHELP, online help.

12.3.8 Display Rx Packet Count and RSSI

```
iwpriv ra0 set ATE=RXFRAME            Start Rx
iwpriv ra0 set ATERXANT=0             Enable All Three Rx Antennas
iwpriv ra0 set ATERXFER=1             Enable Rx Frame Error Rate: RxCnt/RxTotal
MlmePeriodicExec: Rx packet cnt = 2/4
MlmePeriodicExec: Rx AvgRssi0=-88, AvgRssi1=-80, AvgRssi2=-91
MlmePeriodicExec: Rx packet cnt = 2/6
MlmePeriodicExec: Rx AvgRssi0=-86, AvgRssi1=-77, AvgRssi2=-89…
…
iwpriv ra0 set ATE=RXFRAME            Start Rx
iwpriv ra0 set ATERXANT=1             Enable Three Rx Antenna-1
iwpriv ra0 set ATERXFER=1             Enable Rx Frame Error Rate: RxCnt/RxTotal
MlmePeriodicExec: Rx packet cnt = 0/7
MlmePeriodicExec: Rx AvgRssi=-87
MlmePeriodicExec: Rx packet cnt = 7/14
MlmePeriodicExec: Rx AvgRssi=-90
…
…
```

12.4 iwpriv ra0 bbp [parameters]=[Value]

Read/Write BBP registers by ID number.

12.4.1 BBPID

Read BBP register, BBPID only, no “=” symbol.
BBPID:
0 ~ xx ; decimal, 8-bit

12.4.2 BBPID=Value

Write BBP register.
BBPID:
0 ~ xx ; decimal, 8-bit

Value:

00 ~FF ; hexdecimal, 8-bit

12.5 iwpriv ra0 mac [parameters]=[val]

Read/Write MAC registers by offset.

12.5.1 MAC_OFFSET

Read MAC register, MAC_OFFSET only, no "=" symbol.
MAC_OFFSET:
0000 ~ FFFF ; hexdecimal, 16-bit

12.5.2 MAC_OFFSET=Value

Write MAC register.
MAC_OFFSET:
0000 ~ FFFF ; hexdecimal, 16-bit

Value:

0000 ~FFFF ; hexdecimal, 32-bit

12.6 iwpriv ra0 e2p [parameters]=[val]

Read/Write EEPROM content by address.

12.6.1 EEP_ADDR

Read EEPROM content, EEP_ADDR only, no "=" symbol.
EEP_ADDR:
00 ~ FF ; hexdecimal, 16-bit alignment (0, 2, 4, 6, 8, A, C, …)

12.6.2 EEP_ADDR=Value

Write EEPROM content.
EEP_ADDR:
00 ~ FF ; hexdecimal, 16-bit alignment (0, 2, 4, 6, 8, A, C, …)

Value:

0000 ~FFFF ; hexdecimal, 16-bit

12.7 Example

12.7.1 Hardware access

```
iwpriv ra0 bbp 0                # read BBP register 0
iwpriv ra0 bbp 0=12             # write BBP register 0 as 0x12
iwpriv ra0 mac 0                # read MAC register 0
iwpriv ra0 mac 0=1234abcd       # write MAC register 0 as 0x1234abcd
iwpriv ra0 e2p 0                # read E2PROM 0
iwpriv ra0 e2p c=12ab           # write E2PROM 0xc as 0x12ab
```

12.7.2 Statistic counter operation

```
iwpriv ra0 stat                  # read statistic counter
iwpriv ra0 set ResetCounter=0    # reset statistic counter
```

12.7.3 Suggestion:

1. To turn on ATE functionality, you have to add compile flag "RALINK_ATE" to Makefile
2. Before doing ATE testing, please stop AP function
3. If you want to test another ATE action, prefer to stop AP & ATE function
4. All ATE function settings will lose efficacy after reboot.
5. Before hardware register access, please reference hardware spec.

Note.

In ATE mode, the channel must set via "ATECHANNEL"

12.8 ated

This is the README file for the RT28xx ATE daemon - ated, which comes with RT28xx linux driver.
This README explains the relationship between the linux driver, Windows GUI and RT28xx ATE daemon.
In addtion, this will teach you how to use this ATE daemon.

12.8.1 Introduction

The ated is an optional user space component for RT28xx Linux driver.
When Windows GUI starts, AP enters ATE mode (i.e.,ATESTART) immediately.
It behaves as a proxy between Windows GUI and RT28xx Linux driver when ATE process proceeds.
The ated will be killed automatically when Windows GUI is closed.
You can kill it manually, too(for example, type '$killall ated').
RT28xx linux driver will leave ATE mode either ated is killed or Windows GUI is closed

12.8.2 Environment setup

1. Connect the platform you want to test directly with a Windows host by ether network line.
2. In the Windows host, run WinPcap_4_0.exe for the QA GUI
   or ./RT2880_ATE/RaUI.exe(please unrar "RT2880_ATE.rar" to get it).

12.8.3 How to use ated for ATE purpose

1. First you should set both "HAS_ATE=y" and "HAS_2860_QA=y"   in the file ~/Module/os/Linux/config.mk and compile the driver.
2. Modify the Makefile according to our target "PLATFORM".
3. Change the path of "CROSS_COMPILE" if needed.
4. Then type 'make' command to compile the source code of the daemon.
5. After the driver interface has started up, attach both of the wireless interface and the ethernet interface to the bridge interface.
6. After the interfaces have entered forwarding states, manually start ated, type '$ated -bbrX -iraX'In the Windows host, run RT2860QA_ATE.exe.
7. If your WLAN interface and Bridge interface is "ra0" and "br0" respectively, just type $./ated. (For further usage of options, type $ated -h).
8. In the Windows host, run RT28xxQA_ATE.exe or ./RT2880_ATE/RaUI.exe..
9. Select the wired network adapter, then press OK and wait for a moment.
10. If the Windows host cannot tolerate such a broadcast storm from ated,

please run ated with option -u.(for example : '$./ated -ira1 -u')

11. If your target platform concerns its network security, please run RT28xxQA_unicast.exe instead of RT28xxQA_ATE.exe.

Note:

1. The names of WLAN interface(default is "ra0") and Bridge interface(default is "br0") must be specified manually(for example : '$./ated -bbr1 -ira2') if your WLAN interface or Bridge interface is not "ra0" or "br0" respectively !
2. Please make sure no other RaUI is running before you excute ./RT2880_ATE/RaUI.exe.

12.8.4 Change on Path and Command

1. /ap/ap_ate.c is moved to ./os/Linux/rt_ate.c and ./include/ap_ate.h is moved to ./include/rt_ate.h for RT2860STA to reuse the ATE code.
2. Due to the reason above, two ATE actions -

APSTOP is renamed to ATESTART
APSTART is renamed to ATESTOP

13 AP CLIENT

13.1 Introduction

The AP-Client function provides a 1-to-N MAC address mapping mechanism such that multiple stations behind the AP can transparently connect to the other AP even they didn't support WDS. When enable the AP-Client function, RT2800 driver will create two interfaces, one is the AP interface which provide the features of Access Point, the other is the station interface used to connect to the remote AP. Besides, the software bridge function is used to forward packets between these two interfaces.

The figure 1 shows the network topology and operation module of our AP-client function. The AP1 is an AP-Client feature enabled Access Point and have two wireless interfaces, ra0 and cli0, which provide the AP and station functions, respectively. The AP2 is a legacy Access Point that supports normal AP functions. STA1 associated to AP1 and the STA4 associated to AP2. In general, if the STA1 want to communicate with STA4, the AP2 and AP1 must support WDS or a physical network connection between AP1 and AP2. Now, with the support the AP-Client function, the AP1 can use build-in station interface cli0 connect to AP2, and then STA1 can communicate with STA4 transparently and didn't do any modifications. Also, the stations connect to the AP1 through the Ethernet line also can communicate with STA4 or access the Internet through AP2 transparently.

Figure 1. The network topology and operation module of AP-Client

Before enable the AP-Client feature, there are some restrictions need to remind

(1). Due to the limitation of 1-to-N MAC address mapping, our AP-Client function currently support following protocols:

- (a). All IP-based network applications
- (b). ARP
- (c). DHCP
- (d). PPPoE

(2). The last 2 hexadecimal number of the Mac address of our device must be the multiple of 4.

(3). The OS must provide a software bridge function can bridge multiple network interfaces.

It’s simple to enable the feature of AP-Client; you just need to set the flag “HAS_APCLIENT” as “y” in the driver Makefile and got it.

13.2 Setup AP Client

1. Edit file in /etc/Wireless/RT2800AP/RT2800AP.dat to add
   a) ApCliEnable=1
   b) ApCliSsid=AP2
   c) ApCliBssid=00:10:20:30:40:50 (optional)
   d) ApCliAuthMode=WPAPSK
   e) ApCliEncrypType=TKIP
   f) ApCliWPAPSK=12345678
2. Like the procedure of bringing up main BSSID (ra0), it also must to add "/sbin/ifconfig apcli0 up" and "/usr/sbin/brctl addif br0 apcli0".
3. The AP-client's security policy only supports NONE, WEP (OPEN, SHARED), WPAPSK and WPA2PSK (TKIP, AES).
4. Set the "HAS_APCLI" flag as "y" in config.mk to enable or disable this function.
5. If enable AP client function, the maximum multiple BSSID number would be 7 and the field 'BssidNum' shall larger than 1 and less than 7.
6. Users can also configure AP Client by iwpriv command.

13.3 Supported Parameters in RT2800AP.dat

13.3.1 ApCliEnable=value

Description:
Value Type:
Valid Range:

Value:

0: Disable
1: Enable

13.3.2 ApCliSsid=value

Description:
Value Type: ASCII characters
Valid Range: 1-32 ASCII characters

13.3.3 ApCliBssid=value

Value:

[Mac Address]
eg: 00:10:20:30:40:50
Description:
Value Type:
Valid Range:

13.3.4 ApCliWPAPSK=value

Value:

Description:
Value Type:
Valid Range:
8 ~ 63 ASCII characters or
64 hexadecimal characters

13.3.5 ApCliAuthMode=value

Description:
Value Type: Text
Valid Range:
OPEN
SHARED
WPAPSK
WPA2PSK

13.3.6 ApCliEncrypType=value

Description:
Value Type:
Valid Range:

NONE:	ApCliAuthMode =OPEN
WEP:	ApCliAuthMode =OPEN or SHARED
TKIP:	ApCliAuthMode =WPAPSK or WPA2PSK
AES:	ApCliAuthMode =WPAPSK or WPA2PSK

13.3.7 ApCliDefaultKeyID=value

Value:

1 ~ 4

13.3.8 ApCliKey1Type=value

Description:
Value Type:
Valid Range:

0:	Hexadecimal
1:	ASCII

13.3.9 ApCliKey1Str=value

Description:
Value Type:
Valid Range:
10 or 26 hexadecimal characters eg: 012345678

5 or 13 ASCII characters eg: passd

13.3.10 ApCliKey2Type=value

Description:
Value Type:
Valid Range:
0: Hexadecimal
1: ASCII

13.3.11 ApCliKey2Str=value

Description:
Value Type:
Valid Range:
10 or 26 hexadecimal characters eg: 012345678
5 or 13 ASCII characters eg: passd

13.3.12 ApCliKey3Type=value

Description:
Value Type:
Valid Range:
0: Hexadecimal
1: ASCII

13.3.13 ApCliKey3Str=value

Value
10 or 26 hexadecimal characters eg: 012345678
5 or 13 ASCII characters eg: passd

13.3.14 ApCliKey4Type=value

Value
0: Hexadecimal
1: ASCII

13.3.15 ApCliKey4Str=value

Description:
Value Type: Hexadecimal or ASCII characters
Valid Range: 10 or 26 hexadecimal characters, or 5 or 13 ASCII characters

13.4 iwpriv apcli0 set [parameter]=[Val]

Syntax:	Example
Section# parameters Explanation Value: 0: … 1: … .: …	12.4.1 ApCliEnable Enable or disable the AP-Client Value: 0: Disable 1: Enrollee

13.4.1 ApCliEnable

Enable or disable the AP-Client
Description:
Value Type:
Valid Range:

Value:

0: Disable
1: Enable

13.4.2 ApCliSsid
Set SSID which AP client wants to join
Description:
Value Type:
Valid Range:

Value:

0~z, less than 32 characters

13.4.3 ApCliBssid
Set BSSID which AP Client wants to join
Description:
Value Type:
Valid Range: [MAC address]

Note:

It is an optional command. Users can indicate the desired BSSID by this command. Otherwise, AP Client can also get appropriate BSSID according to SSID automatically.

13.4.4 ApCliWPAPSK
AP Client WPA Pre-Shared Key
Description:
Value Type:
Valid Range: 8~63 ASCII or 64 HEX characters

13.4.5 ApCliAuthMode
Set AP Client Authentication mode
Description:
Value Type:
Valid Range: OPEN, SHARED, WPAPSK, WPA2PSK

13.4.6 ApCliEncrypType
Set AP Client Encryption Type
Description:
Value Type:
Valid Range: NONE, WEP, TKIP, AES

13.4.7 ApCliDefaultKeyID
Set AP Client Default Key ID

Description:
Value Type:
Valid Range: 1~4

13.4.8 ApCliKey1

Set AP Client Key1 String
Description:
Value Type:
Valid Range: 5 ASCII characters or 10 hex numbers, or 13 ASCII characters or 26 hex numbers

13.4.9 ApCliKey2

Set AP Client Key2 String
Description:
Value Type:
Valid Range: 5 ASCII characters or 10 hex numbers, or 13 ASCII characters or 26 hex numbers

13.4.10 ApCliKey3

Set AP Client Key3 String
Description:
Value Type:
Valid Range:
5 ASCII characters or 10 hex numbers, or 13 ASCII characters or 26 hex numbers.

13.4.11 ApCliKey4

Set AP Client Key4 String
Description:
Value Type:
Valid Range: 5 ASCII characters or 10 hex numbers, or 13 ASCII characters or 26 hex numbers

13.4.12 ApCliWscSsid

Set AP Client Key4 String
Description:
Value Type:
Valid Range: 5 ASCII characters or 10 hex numbers, or 13 ASCII characters or 26 hex numbers.

13.5 Example

13.5.1 Example I: Enable AP Client with NONE data security

1. iwpriv apcli0 set ApCliEnable=0
2. iwpriv apcli0 set ApCliAuthMode=OPEN
3. iwpriv apcli0 set ApCliEncrypType=NONE
4. iwpriv apcli0 set ApCliSsid=AP2
5. iwpriv apcli0 set ApCliEnable=1

13.5.2 Example II: OPEN WEP setting

1. iwpriv apcli0 set ApCliEnable=0
2. iwpriv apcli0 set ApCliAuthMode=OPEN
3. iwpriv apcli0 set ApCliEncrypType=WEP

4. iwpriv apcli0 set ApCliDefaultKeyID=1
5. iwpriv apcli0 set ApCliKey1=1234567890
6. iwpriv apcli0 set ApCliSsid=AP2
7. iwpriv apcli0 set ApCliEnable=1

13.5.3 Example III: Shared WEP setting

1. iwpriv apcli0 set ApCliEnable=0
2. iwpriv apcli0 set ApCliAuthMode=SHARED
3. iwpriv apcli0 set ApCliEncrypType=WEP
4. iwpriv apcli0 set ApCliDefaultKeyID=2
5. iwpriv apcli0 set ApCliKey2=2345678901
6. iwpriv apcli0 set ApCliSsid=AP2
7. iwpriv apcli0 set ApCliEnable=1

13.5.4 Example IV: WPAPSK-TKIP setting

1. iwpriv apcli0 set ApCliEnable=0
2. iwpriv apcli0 set ApCliAuthMode=WPAPSK
3. iwpriv apcli0 set ApCliEncrypType=TKIP
4. iwpriv apcli0 set ApCliSsid=AP2
5. iwpriv apcli0 set ApCliWPAPSK=12345678
6. iwpriv apcli0 set ApCliEnable=1

13.5.5 Example V: WPA2PSK-AES setting

1. iwpriv apcli0 set ApCliEnable=0
2. iwpriv apcli0 set ApCliAuthMode=WPA2PSK
3. iwpriv apcli0 set ApCliEncrypType=AES
4. iwpriv apcli0 set ApCliSsid=AP2
5. iwpriv apcli0 set ApCliWPAPSK=12345678
6. iwpriv apcli0 set ApCliEnable=1

14 IGMP SNOOPING

14.1 IGMP Table Learning:

An IGMP table entry consists of Group-Id (Multicast MAC Address), Net-Interface and Member-List. For example, in the picture above we see the “Multicast Filter Table” of AP1 have two IGMP entries. One is “01:00:5e:02:02:03” with two members and another is “01:00:5e:02:02:04 with empty member list”. AP will automatically insert or remove the entry from table by snooping the IGMP-Membership report packet from Station behind AP. And it also could be manual add and del by iwpriv command.

14.2 Multicast Packet Process:

Once a multicast packet whether it comes from portal, WDS or AP-Client. AP will go through the Multicast-filter table to find a match rule for the incoming packet. If have no any match rule in the table then AP will simply drops it. If it does then there are two cases how AP handles a multicast packet. The first cast is the match entry has no member then AP just forwards it to all stations behind the net-interface. If the match entry has members then AP will do unicast clone for all members.

For example, AP1 receive a multicast packet with group-Id, “01:00:5e:02:02:03”, comes from Ethernet then AP1 check the multicast table using group-Id and fount it match the entry with 2 members. So AP1 clone the multicast packet and sent them to Station 1 and Station 2. Another case a multicast packet with group-id (01:00:5e:02:02:04) be sent to AP1 then AP1 just forward it to all Stations behind interface, ra0 since the match entry have no member.

14.3 Iwpriv command for IGMP-Snooping:

Syntax: Section# parameters Explanation Value: 0: … 1: … .: …	Example 13.3.1 IgmpSnEnable Enable IGMP snooping Value: 0x0: Disable 0x1: Enrollee

14.3.1 IgmpSnEnable

The IGMP snooping function and multicast packet filter can be enabled or disabled at running time by iwpriv command “set IgmpSnEnable=<0|1>”.

For e.g.

```
iwpriv ra0 set IgmpSnEnable=1
iwpriv ra0 set IgmpSnEnable=0
```

14.3.2 IgmpAdd :: Group-ID

It also provide a command let user add a entry by iwpriv command “set IgmpAdd=<Group-ID>”, Group-ID could be a MAC address or a IP address.

For e.g.

```
iwpriv ra0 set IgmpAdd=226.2.2.3
iwpriv ra0 set IgmpAdd=01:00:5e:02:02:03
```

14.3.3 IgmpAdd :: Group-Member

Or just add members into a Group by command “set IgmpAdd=<Group-ID-[Member]-... >”, Group-ID could be a MAC address or a IP address.

For e.g.

```
iwpriv ra0 set IgmpAdd=226.2.2.3-00:0c:43:26:61:27-00:0c:43:26:61:28
iwpriv ra0 set IgmpAdd=01:00:5e:02:02:03-00:0c:43:26:61:27-00:0c:43:26:61:28
```

14.3.4 IgmpDel::Group-ID

Also the entry can be deleted by command “set IgmpDelEntry=<Group-ID>”.

For e.g.

```
iwpriv ra0 set IgmpDel=226.2.2.3
iwpriv ra0 set IgmpDel=01:00:5e:02:02:03
```

14.3.5 IgmpDel::Group-Member

Or just delete a member from a Group by command “set IgmpDel=<Group-ID-[Member]-... >”, Group-ID could be a MAC address or a IP address.

For e.g.

```
iwpriv ra0 set IgmpDel=226.2.2.3-00:0c:43:26:61:27-00:0c:43:26:61:28
iwpriv ra0 set IgmpDel=01:00:5e:02:02:03-00:0c:43:26:61:27-00:0c:43:26:61:28
```

15 IOCTL – I/O CONTROL INTERFACE

15.1 Parameters for iwconfig’s IOCTL

Access	Description	ID	Parameters
Get	BSSID, MAC Address	SIOCGIFHWADDR	wrq->u.name, (length = 6)
	WLAN Name	SIOCGIWNAME	wrq->u.name = "RT2800 SoftAP", length = strlen(wrq->u.name)

	SSID	SIOCGIWESSID	struct iw_point *erq = &wrq->u.essid;
			erq->flags=1;
			erq->length = pAd->PortCfg.MBSSID[pAd->IoctlIF].SsidLen;
			if(erq->pointer)
			{
			  if(copy_to_user(erq->pointer,
			            pAd->PortCfg.MBSSID[pAd->IoctlIF].Ssid,
			            erq->length))
			  {
			    Status = -EFAULT;
			    break;
			  }
			}

	Channel / Frequency (Hz)	SIOCGIWFREQ	wrq->u.freq.m = pAd->PortCfg.Channel;
			wrq->u.freq.e = 0;
			wrq->u.freq.i = 0;

	Bit Rate (bps)	SIOCGIWRATE	wrq->u.bitrate.value =
			     RateIdTo500Kbps[pAd->PortCfg.MBSSID[pAd->IoctlIF].TxRate]
			* 500000;
			wrq->u.bitrate.disabled = 0;

	AP’s MAC address	SIOCGIWAP	wrq->u.ap_addr.sa_family = ARPHRD_ETHER;
			memcpy(wrq->u.ap_addr.
			        sa_data,
			        &pAd->PortCfg.MBSSID[pAd->IoctlIF].Bssid, ETH_ALEN);

	Operation Mode	SIOCGIWMODE	wrq->u.mode = IW_MODE_INFRA;

	Range of Parameters	SIOCGIWRANGE	range.we_version_compiled = WIRELESS_EXT;
			range.we_version_source = 14;

	Scanning Results	SIOCGIWSCAN	typedef struct _NDIS_802_11_SITE_SURVEY_TABLE
			{
			  LONG Channel;
			  LONG Rssi;
			  UCHAR Ssid[33];
			  UCHAR Bssid[18];
			  UCHAR EncrypT[8];
			} NDIS_802_11_SITE_SURVEY_TABLE,
			*PNDIS_802_11_SITE_SURVEY_TABLE;
			
			wrq->u.data.length = N* sizeof(NDIS_802_11_SITE_SURVEY_TABLE);
			copy_to_user(wrq->u.data.pointer, site_survey_table,
			wrq->u.data.length);

	Client Association List	SIOCGIWAPLIST	typedef struct _NDIS_802_11_STATION_TABLE
			{
			  UCHAR MacAddr[18];
			  ULONG Aid;
			  ULONG PsMode;
			  ULONG LastDataPacketTime;
			  ULONG RxByteCount;
			  ULONG TxByteCount;
			  ULONG CurrTxRate;
			  ULONG LastTxRate;

			} NDIS_802_11_STATION_TABLE, *PNDIS_802_11_STATION_TABLE;
			
			wrq->u.data.length = i * sizeof(NDIS_802_11_STATION_TABLE);
			copy_to_user(wrq->u.data.pointer, sta_list_table, wrq->u.data.length);

Set	Trigger Scanning	SIOCSIWSCAN	ApSiteSurvey(pAd);

15.2 Parameters for iwpriv's IOCTL

Please refer section 4 and 5 to have iwpriv parameters and values.

Parameters:

```
int           socket_id;
char  name[25];                    // interface name
char  data[255];                   // command string
struct        iwreq wrq;
```

Default setting:

```
wrq.ifr_name = name = "ra0";       // interface name
wrq.u.data.pointer = data;         // data buffer of command string
wrq.u.data.length = strlen(data);  // length of command string
wrq.u.data.flags = 0;
```

15.2.1 Set Data

THESE PARAMETERS ARE THE SAME AS IWPRIV

Command and IOCTL Function		
Set Data		
Function Type	Command	IOCTL

RTPRIV_IOCTL_SET	iwpriv ra0 set SSID=RT2800AP	sprintf(name, "ra0");
		strcpy(data, "SSID=RT2800AP");
		strcpy(wrq.ifr_name, name);
		wrq.u.data.length = strlen(data);
		wrq.u.data.pointer = data;
		wrq.u.data.flags = 0;
		ioctl(socket_id, RTPRIV_IOCTL_SET, &wrq);

15.2.2 Get Data

THESE PARAMETERS ARE THE SAME AS IWPRIV

Command and IOCTL Function		
Get Data		
Function Type	Command	IOCTL

RTPRIV_IOCTL_STATISTICS	Iwpriv ra0 stat	sprintf(name, "ra0");
		strcpy(data, "stat");
		strcpy(wrq.ifr_name, name);
		wrq.u.data.length = strlen(data);
		wrq.u.data.pointer = data;
		wrq.u.data.flags = 0;
		ioctl(socket_id, RTPRIV_IOCTL_STATISTICS, &wrq);

RTPRIV_IOCTL_GSITESURVEY	Iwpriv ra0	sprintf(name, "ra0");

	get_site_survey	strcpy(data, "get_site_survey"); strcpy(wrq.ifr_name, name); wrq.u.data.length = strlen(data); wrq.u.data.pointer = data; wrq.u.data.flags = 0; ioctl(socket_id, RTPRIV_IOCTL_GSITESURVEY, &wrq);
RTPRIV_IOCTL_GET_MAC_TABLE	Iwpriv ra0 get_mac_table	sprintf(name, "ra0"); strcpy(data, "get_mac_table"); strcpy(wrq.ifr_name, name); wrq.u.data.length = strlen(data); wrq.u.data.pointer = data; wrq.u.data.flags = 0; ioctl(socket_id, RTPRIV_IOCTL_GET_MAC_TABLE, &wrq);
RTPRIV_IOCTL_SHOW	Iwpriv ra0 show	sprintf(name, "ra0"); strcpy(data, "get_mac_table"); strcpy(wrq.ifr_name, name); wrq.u.data.length = strlen(data); wrq.u.data.pointer = data; wrq.u.data.flags = 0; ioctl(socket_id, RTPRIV_IOCTL_SHOW, &wrq);
RTPRIV_IOCTL_WSC_PROFILE	Iwpriv ra0 get_wsc_profile	sprintf(name, "ra0"); strcpy(data, "get_mac_table"); strcpy(wrq.ifr_name, name); wrq.u.data.length = strlen(data); wrq.u.data.pointer = data; wrq.u.data.flags = 0; ioctl(socket_id, RTPRIV_IOCTL_WSC_PROFILE, &wrq);
RTPRIV_IOCTL_QUERY_BATABLE	Iwpriv ra0 get_ba_table	sprintf(name, "ra0"); strcpy(data, "get_mac_table"); strcpy(wrq.ifr_name, name); wrq.u.data.length = strlen(data); wrq.u.data.pointer = data; wrq.u.data.flags = 0; ioctl(socket_id, RTPRIV_IOCTL_QUERY_BATABLE, &wrq);

15.2.3 Set Data: BBP, MAC and EEPROM

Command and IOCTL Function		
Set Data: BBP, MAC and EEPROM, Parameters is Same as iwpriv		
Type	Command	IOCTL
RTPRIV_IOCTL_BBP (Set BBP Register Value)	Iwpriv ra0 bbp 17=32	sprintf(name, "ra0"); strcpy(data, " bbp 17=32"); strcpy(wrq.ifr_name, name); wrq.u.data.length = strlen(data); wrq.u.data.pointer = data; wrq.u.data.flags = 0; ioctl(socket_id, RTPRIV_IOCTL_BBP, &wrq);
RTPRIV_IOCTL_MAC (Set MAC Register Value)	Iwpriv ra0 mac 3000=12345678	sprintf(name, "ra0"); strcpy(data, " mac 3000=12345678"); strcpy(wrq.ifr_name, name); wrq.u.data.length = strlen(data); wrq.u.data.pointer = data; wrq.u.data.flags = 0; ioctl(socket_id, RTPRIV_IOCTL_MAC, &wrq);
RTPRIV_IOCTL_E2P	Iwpriv ra0 e2p 40=1234	sprintf(name, "ra0");

(Set EEPROM Value)		strcpy(data, " e2p 40=1234"); strcpy(wrq.ifr_name, name); wrq.u.data.length = strlen(data); wrq.u.data.pointer = data; wrq.u.data.flags = 0; ioctl(socket_id, RTPRIV_IOCTL_E2P, &wrq);

15.2.4 Get Data: BBP, MAC and EEPROM

Command and IOCTL Function		
Get Data: BBP, MAC and EEPROM , Parameters is Same as iwpriv		
Type	Command	IOCTL
RTPRIV_IOCTL_BBP (Get BBP Register Value)	Iwpriv ra0 bbp 17	sprintf(name, "ra0"); strcpy(data, " bbp 17"); strcpy(wrq.ifr_name, name); wrq.u.data.length = strlen(data); wrq.u.data.pointer = data; wrq.u.data.flags = 0; ioctl(socket_id, RTPRIV_IOCTL_BBP, &wrq);
RTPRIV_IOCTL_MAC (Get MAC Register Value)	Iwpriv ra0 mac 3000	sprintf(name, "ra0"); strcpy(data, " mac 3000"); strcpy(wrq.ifr_name, name); wrq.u.data.length = strlen(data); wrq.u.data.pointer = data; wrq.u.data.flags = 0; ioctl(socket_id, RTPRIV_IOCTL_MAC, &wrq);
RTPRIV_IOCTL_E2P (Get EEPROM Value)	Iwpriv ra0 e2p 40	sprintf(name, "ra0"); strcpy(data, " e2p 40"); strcpy(wrq.ifr_name, name); wrq.u.data.length = strlen(data); wrq.u.data.pointer = data; wrq.u.data.flags = 0; ioctl(socket_id, RTPRIV_IOCTL_E2P, &wrq);

15.2.5 Set Raw Data

IOCTL Function	
Set Raw Data by I/O Control Interface	
Function Type	IOCTL
RTPRIV_IOCTL_RADIUS_DATA	sprintf(name, "ra0"); strcpy(wrq.ifr_name, name); memset(data, 0x55, 100); wrq.u.data.length = 100; wrq.u.data.pointer = data; wrq.u.data.flags = 0; ioctl(socket_id, RTPRIV_IOCTL_RADIUS_DATA, &wrq);
RTPRIV_IOCTL_ADD_WPA_KEY	NDIS_802_11_KEY *vp; sprintf(name, "ra0"); strcpy(wrq.ifr_name, name); memset(data, 0, sizeof(NDIS_802_11_KEY)); vp = (NDIS_802_11_KEY *)&data; vp->Length = sizeof(NDIS_802_11_KEY); memset(vp->addr, 0x11, 6); vp->KeyIndex = 2;

	vp->KeyLength = 32;
	memset(vp->KeyMaterial, 0xAA, 32);
	wrq.u.data.length = sizeof(NDIS_802_11_KEY);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = 0;
	ioctl(socket_id, RTPRIV_IOCTL_ADD_WPA_KEY, &wrq);

RTPRIV_IOCTL_ADD_PMKID_CACHE	NDIS_802_11_KEY *vp;
	
	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(NDIS_802_11_KEY));
	vp = (NDIS_802_11_KEY *)&data;
	vp->Length = sizeof(NDIS_802_11_KEY);
	memset(vp->addr, 0x11, 6);
	vp->KeyIndex = 2;
	vp->KeyLength = 32;
	memset(vp->KeyMaterial, 0xBB, 32);
	wrq.u.data.length = sizeof(NDIS_802_11_KEY);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = 0;
	ioctl(socket_id, RTPRIV_IOCTL_ADD_PMKID_CACHE, &wrq);

15.2.6 Set Raw Data with Flags

IOCTL Function	
Set Raw Data by I/O Control Interface with Flags	
Function Type	IOCTL

RT_SET_APD_PID	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, 4);
	data[0] = 12;
	wrq.u.data.length = 4;
	wrq.u.data.pointer = data;
	wrq.u.data.flags = RT_SET_APD_PID;
	ioctl(socket_id, RT_PRIV_IOCTL, &wrq);

RT_SET_DEL_MAC_ENTRY	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0xdd, 6);
	strcpy(wrq.ifr_name, name);
	wrq.u.data.length = 6;
	wrq.u.data.pointer = data;
	wrq.u.data.flags = RT_SET_DEL_MAC_ENTRY;
	ioctl(socket_id, RT_PRIV_IOCTL, &wrq);

RT_OID_WSC_SET_SELECTED_REGISTRAR	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, decodeStr, decodeLen);
	strcpy(wrq.ifr_name, name);
	wrq.u.data.length = decodeLen;
	wrq.u.data.pointer = data;
	wrq.u.data.flags = RT_OID_WSC_SET_SELECTED_REGISTRAR;
	ioctl(socket_id, RT_PRIV_IOCTL, &wrq);

RT_OID_WSC_EAPMSG	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, wscU2KMsg, wscU2KMsgLen);
	strcpy(wrq.ifr_name, name);
	wrq.u.data.length = wscU2KMsgLen;
	wrq.u.data.pointer = data;

	wrq.u.data.flags = RT_OID_WSC_EAPMSG;
	ioctl(socket_id, RT_PRIV_IOCTL, &wrq);

15.2.7 Get Raw Data with Flags

IOCTL Function	
Get Raw Data by I/O Control Interface with Flags	
Function Type	IOCTL

RT_QUERY_ATE_TXDONE_COUNT	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(ULONG));
	wrq.u.data.length = sizeof(ULONG);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = RT_QUERY_ATE_TXDONE_COUNT;
	ioctl(socket_id, RT_PRIV_IOCTL, &wrq);

RT_QUERY_SIGNAL_CONTEXT	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(RT_SIGNAL_STRUC));
	strcpy(wrq.ifr_name, name);
	wrq.u.data.length = sizeof(RT_SIGNAL_STRUC);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = RT_QUERY_SIGNAL_CONTEXT;
	ioctl(socket_id, RT_PRIV_IOCTL, &wrq);

RT_OID_WSC_QUERY_STATUS	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(INT));
	strcpy(wrq.ifr_name, name);
	wrq.u.data.length = sizeof(INT);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = RT_OID_WSC_QUERY_STATUS;
	ioctl(socket_id, RT_PRIV_IOCTL, &wrq);

RT_OID_WSC_PIN_CODE	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(ULONG));
	strcpy(wrq.ifr_name, name);
	wrq.u.data.length = sizeof(ULONG);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = RT_OID_WSC_PIN_CODE;
	ioctl(socket_id, RT_PRIV_IOCTL, &wrq);

RT_OID_WSC_UUID	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(UCHAR));
	strcpy(wrq.ifr_name, name);
	wrq.u.data.length = sizeof(UCHAR);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = RT_OID_WSC_UUID;
	ioctl(socket_id, RT_PRIV_IOCTL, &wrq);

RT_OID_WSC_MAC_ADDRESS	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, MAC_ADDR_LEN);
	strcpy(wrq.ifr_name, name);
	wrq.u.data.length = MAC_ADDR_LEN;
	wrq.u.data.pointer = data;
	wrq.u.data.flags = RT_OID_WSC_MAC_ADDRESS;
	ioctl(socket_id, RT_PRIV_IOCTL, &wrq);

RT_OID_GET_PHY_MODE	sprintf(name, "ra0");

	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(ULONG));
	strcpy(wrq.ifr_name, name);
	wrq.u.data.length = sizeof(ULONG);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = RT_OID_GET_PHY_MODE;
	ioctl(socket_id, RT_PRIV_IOCTL, &wrq);

RT_OID_GET_LLTD_ASSO_TANLE	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(RT_LLTD_ASSOICATION_TABLE));
	strcpy(wrq.ifr_name, name);
	wrq.u.data.length = sizeof(RT_LLTD_ASSOICATION_TABLE);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = RT_OID_GET_LLTD_ASSO_TANLE;
	ioctl(socket_id, RT_PRIV_IOCTL, &wrq);

15.3 Sample User Space Application

```
//==========================================================================
//
// rtuser:
//      1. User space application to demo how to use IOCTL function.
//      2. Most of the IOCTL function is defined as "CHAR" type and return with string message.
//      3. Use sscanf to get the raw data back from string message.
//      4. The command format "parameter=value" is same as iwpriv command format.
//      5. Remember to insert driver module and bring interface up prior execute rtuser.
//              change folder path to driver "Module"
//              dos2unix *          ; in case the files are modified from other OS environment
//              chmod 644 *
//              chmod 755 Configure
//              make config
//              make
//              insmod RT2800ap.o
//              ifconfig ra0 up
//
// Refer Linux/if.h to have
//              #define ifr_name    ifr_ifrn.ifrn_name              /* interface name  */
//
// Make:
//              cc -Wall -ortuser rtuser.c
//
// Run:
```

```
//                  ./rtuser
//
//=====================================================================

#include <stdio.h>
#include <string.h>
#include <sys/socket.h>
#include <sys/ioctl.h>
#include <unistd.h>                              /* for close */
#include <Linux/wireless.h>

//=============================================================================

#if WIRELESS_EXT <= 11
#ifndef SIOCDEVPRIVATE
#define SIOCDEVPRIVATE                           0x8BE0
#endif
#define SIOCIWFIRSTPRIV                          SIOCDEVPRIVATE
#endif

//
//SET/GET CONVENTION :
// * ------------------
// * Simplistic summary :
// *      o even numbered ioctls are SET, restricted to root, and should not
// *      return arguments (get_args = 0).
// *      o odd numbered ioctls are GET, authorised to anybody, and should
// *      not expect any arguments (set_args = 0).
//
#define RT_PRIV_IOCTL                  (SIOCIWFIRSTPRIV + 0x01)
#define RTPRIV_IOCTL_SET                            (SIOCIWFIRSTPRIV + 0x02)
#define RTPRIV_IOCTL_BBP                            (SIOCIWFIRSTPRIV + 0x03)
#define RTPRIV_IOCTL_MAC                            (SIOCIWFIRSTPRIV + 0x05)
#define RTPRIV_IOCTL_E2P                            (SIOCIWFIRSTPRIV + 0x07)
#define RTPRIV_IOCTL_STATISTICS                     (SIOCIWFIRSTPRIV + 0x09)
```

```
#define RTPRIV_IOCTL_ADD_PMKID_CACHE        (SIOCIWFIRSTPRIV + 0x0A)

#define RTPRIV_IOCTL_RADIUS_DATA            (SIOCIWFIRSTPRIV + 0x0C)

#define RTPRIV_IOCTL_GSITESURVEY            (SIOCIWFIRSTPRIV + 0x0D)

#define RTPRIV_IOCTL_ADD_WPA_KEY            (SIOCIWFIRSTPRIV + 0x0E)

#define RTPRIV_IOCTL_GET_MAC_TABLE          (SIOCIWFIRSTPRIV + 0x0F)



#define OID_GET_SET_TOGGLE                  0x8000


#define RT_QUERY_ATE_TXDONE_COUNT           0x0401

#define RT_QUERY_SIGNAL_CONTEXT             0x0402

#define RT_SET_APD_PID                      (OID_GET_SET_TOGGLE + 0x0405)

#define RT_SET_DEL_MAC_ENTRY                (OID_GET_SET_TOGGLE + 0x0406)

//---------------------------------------------------------


#ifndef     TRUE

#define     TRUE            1

#endif


#ifndef     FALSE

#define     FALSE       0

#endif


#define MAC_ADDR_LEN                    6

#define ETH_LENGTH_OF_ADDRESS               6

#define MAX_LEN_OF_MAC_TABLE                64

//---------------------------------------------------------


typedef struct _COUNTERS

{

    unsigned long       TxSuccessTotal;;

    unsigned long       TxSuccessWithRetry;

    unsigned long       TxFailWithRetry;
```

```
          unsigned long        RtsSuccess;
          unsigned long        RtsFail;
          unsigned long        RxSuccess;
          unsigned long        RxWithCRC;
          unsigned long        RxDropNoBuffer;
          unsigned long        RxDuplicateFrame;
          unsigned long        FalseCCA;
          unsigned long        RssiA;
          unsigned long        RssiB;
}         COUNTERS;
```

PS. User can check with “iwpriv ra0 stat” to make sure the TXRX status is correct when porting the ATE related test program.

```
//---------------------------------------------------------

typedef        struct _SITE_SURVEY
{
          unsigned char             channel;
          unsigned short            rssi;
          unsigned char             ssid[33];
          unsigned char             bssid[6];
          unsigned char             security[9];
}         SITE_SURVEY;

//---------------------------------------------------------

typedef union  _MACHTTRANSMIT_SETTING {
          struct {
          unsigned short            MCS:7;            // MCS
          unsigned short            BW:1;             //channel bandwidth 20MHz or 40 MHz
          unsigned short            ShortGI:1;
          unsigned short            STBC:2;           //SPACE
          unsigned short            rsv:3;
          unsigned short            MODE:2;           // Use definition MODE_xxx.
          }         field;
          unsigned short            word;
```

```
 } MACHTTRANSMIT_SETTING, *PMACHTTRANSMIT_SETTING;

typedef struct _RT_802_11_MAC_ENTRY {
    unsigned char              Addr[6];
    unsigned char              Aid;
    unsigned char              Psm;          // 0:PWR_ACTIVE, 1:PWR_SAVE
    unsigned char              MimoPs;       // 0:MMPS_STATIC, 1:MMPS_DYNAMIC, 3:MMPS_Enabled
    MACHTTRANSMIT_SETTING          TxRate;
} RT_802_11_MAC_ENTRY, *PRT_802_11_MAC_ENTRY;

typedef struct _RT_802_11_MAC_TABLE {
    unsigned long              Num;
    RT_802_11_MAC_ENTRY Entry[MAX_LEN_OF_MAC_TABLE];
} RT_802_11_MAC_TABLE, *PRT_802_11_MAC_TABLE;

// Key mapping keys require a BSSID
typedef struct _NDIS_802_11_KEY
{
        unsigned long          Length;       // Length of this structure
        unsigned char          addr[6];
        unsigned long          KeyIndex;
        unsigned long          KeyLength;      // length of key in bytes
        unsigned char          KeyMaterial[32];  // variable length depending on above field
} NDIS_802_11_KEY, *PNDIS_802_11_KEY;

typedef struct _RT_SIGNAL_STRUC {
        unsigned short         Sequence;
        unsigned char          MacAddr[MAC_ADDR_LEN];
        unsigned char          CurrAPAddr[MAC_ADDR_LEN];
        unsigned char          Sig;
} RT_SIGNAL_STRUC, *PRT_SIGNAL_STRUC;

//---------------------------------------------------------

COUNTERS         counter;
```

```
SITE_SURVEY     SiteSurvey[100];
char            data[4096];

//==============================================================================

int main( int argc, char ** argv )
{
        char        name[25];
        int         socket_id;
        struct  iwreq wrq;
        int         ret;

        // open socket based on address family: AF_NET ----------------------------
        socket_id = socket(AF_INET, SOCK_DGRAM, 0);
        if(socket_id < 0)
        {
                printf("\nrtuser::error::Open socket error!\n\n");
                return -1;
        }

        // set interface name as "ra0" --------------------------------------------
        sprintf(name, "ra0");
        memset(data, 0x00, 255);
//
//example of iwconfig ioctl function ==========================================
//
        // get wireless name ----------------------------------------------------
        strcpy(wrq.ifr_name, name);
        wrq.u.data.length = 255;
        wrq.u.data.pointer = data;
        wrq.u.data.flags = 0;
        ret = ioctl(socket_id, SIOCGIWNAME, &wrq);
        if(ret != 0)
        {
                printf("\nrtuser::error::get wireless name\n\n");
```

```
            goto rtuser_exit;
        }

        printf("\nrtuser[%s]:%s\n", name, wrq.u.name);
//
//example of iwpriv ioctl function ==========================================
//
        //WPAPSK, remove "set" string -------------------------------------------
        memset(data, 0x00, 255);
        strcpy(data, "WPAPSK=11223344");
        strcpy(wrq.ifr_name, name);
        wrq.u.data.length = strlen(data)+1;
        wrq.u.data.pointer = data;
        wrq.u.data.flags = 0;
        ret = ioctl(socket_id, RTPRIV_IOCTL_SET, &wrq);
        if(ret != 0)
        {
            printf("\nrtuser::error::set wpapsk\n\n");
            goto rtuser_exit;
        }
        //set e2p, remove "e2p" string ------------------------------------------
        memset(data, 0x00, 255);
        strcpy(data, "80=1234");
        strcpy(wrq.ifr_name, name);
        wrq.u.data.length = strlen(data)+1;
        wrq.u.data.pointer = data;
        wrq.u.data.flags = 0;
        ret = ioctl(socket_id, RTPRIV_IOCTL_E2P, &wrq);
        if(ret != 0)
        {
            printf("\nrtuser::error::set eeprom\n\n");
            goto rtuser_exit;
        }

        //printf("\n%s\n", wrq.u.data.pointer);
```

```
{
        int addr, value, p1;

        // string format: "\n[0x%02X]:0x%04X  " ==> "[0x20]:0x0C02"
        sscanf(wrq.u.data.pointer, "\n[%dx%02X]:%04X  ", &p1, &addr, &value);
        printf("\nSet EEP[0x%02X]:0x%04X\n", addr, value);
}

//get e2p, remove "e2p" string -------------------------------------------
memset(data, 0x00, 255);
strcpy(data, "80");
strcpy(wrq.ifr_name, name);
wrq.u.data.length = strlen(data)+1;
wrq.u.data.pointer = data;
wrq.u.data.flags = 0;
ret = ioctl(socket_id, RTPRIV_IOCTL_E2P, &wrq);
if(ret != 0)
{
        printf("\nrtuser::error::get eeprom\n\n");
        goto rtuser_exit;
}

//printf("\n%s\n", wrq.u.data.pointer);
{
        int addr, value, p1, p2;

        // string format: "\n[0x%02X]:0x%04X  " ==> "[0x20]:0x0C02"
        sscanf(wrq.u.data.pointer, "\n[%dx%04X]:%dx%X  ", &p1, &addr, &p2, &value);
        printf("\nGet EEP[0x%02X]:0x%04X\n", addr, value);
}

//set mac, remove "mac" string -------------------------------------------
memset(data, 0x00, 255);
strcpy(data, "2b4f=1");
strcpy(wrq.ifr_name, name);
```

```
wrq.u.data.length = strlen(data)+1;
wrq.u.data.pointer = data;
wrq.u.data.flags = 0;
ret = ioctl(socket_id, RTPRIV_IOCTL_MAC, &wrq);
if(ret != 0)
{
        printf("\nrtuser::error::set mac register\n\n");
        goto rtuser_exit;
}

//printf("\n%s\n", wrq.u.data.pointer);
{
        int addr, value, p1;

        // string format: "\n[0x%02X]:0x%04X  " ==> "[0x20]:0x0C02"
        sscanf(wrq.u.data.pointer, "\n[%dx%08X]:%08X  ", &p1, &addr, &value);
        printf("\nSet MAC[0x%08X]:0x%08X\n", addr, value);
}
```

//get mac, remove "mac" string --

```
memset(data, 0x00, 255);
strcpy(data, "2b4f");
strcpy(wrq.ifr_name, name);
wrq.u.data.length = strlen(data)+1;
wrq.u.data.pointer = data;
wrq.u.data.flags = 0;
ret = ioctl(socket_id, RTPRIV_IOCTL_MAC, &wrq);
if(ret != 0)
{
        printf("\nrtuser::error::get mac register\n\n");
        goto rtuser_exit;
}

//printf("\n%s\n", wrq.u.data.pointer);
{
	int addr, value, p1;

	// string format: "\n[0x%02X]:0x%04X  " ==> "[0x20]:0x0C02"
	sscanf(wrq.u.data.pointer, "\n[%dx%08X]:%08X  ", &p1, &addr, &value);
	printf("\nGet MAC[0x%08X]:0x%08X\n", addr, value);
}
```

//set bbp, remove "bbp" string ---

```
memset(data, 0x00, 255);
strcpy(data, "17=32");
strcpy(wrq.ifr_name, name);
wrq.u.data.length = strlen(data)+1;
wrq.u.data.pointer = data;
wrq.u.data.flags = 0;
ret = ioctl(socket_id, RTPRIV_IOCTL_BBP, &wrq);
if(ret != 0)
{
	printf("\nrtuser::error::set bbp register\n\n");
	goto rtuser_exit;
}

//printf("\n%s\n", wrq.u.data.pointer);
{
	int id, addr, value, p1;

	// string format: "\n[0x%02X]:0x%04X  " ==> "[0x20]:0x0C02"
	sscanf(wrq.u.data.pointer, "\nR%02d[%dx%02X]:%02X\n", &id, &p1, &addr, &value);
	printf("\nSet BBP R%02d[0x%02X]:0x%02X\n", id, addr, value);
}
```

//get bbp, remove "bbp" string ---

```
memset(data, 0x00, 255);
strcpy(data, "17");
strcpy(wrq.ifr_name, name);
wrq.u.data.length = strlen(data)+1;
wrq.u.data.pointer = data;
wrq.u.data.flags = 0;
ret = ioctl(socket_id, RTPRIV_IOCTL_BBP, &wrq);
if(ret != 0)
{
        printf("\nrtuser::error::get bbp register\n\n");
        goto rtuser_exit;
}

//printf("\n%s\n", wrq.u.data.pointer);
{
        int id, addr, value, p1;

        // string format: "\n[0x%02X]:0x%04X  " ==> "[0x20]:0x0C02"
        sscanf(wrq.u.data.pointer, "\nR%02d[%dx%02X]:%02X  ", &id, &p1, &addr, &value);
        printf("\nGet BBP R%02d[0x%02X]:0x%02X\n", id, addr, value);
}

//get statistics, remove "stat" string -----------------------------------
memset(data, 0x00, 2048);
strcpy(data, "");
strcpy(wrq.ifr_name, name);
wrq.u.data.length = 0;
wrq.u.data.pointer = data;
wrq.u.data.flags = 0;
ret = ioctl(socket_id, RTPRIV_IOCTL_STATISTICS, &wrq);
if(ret != 0)
{
        printf("\nrtuser::error::get statistics\n\n");
        goto rtuser_exit;
}

printf("\n========= Get AP Statistics ==========\n");
{
        int i;
```

```
            char *sp = wrq.u.data.pointer;
            unsigned long *cp = (unsigned long *)&counter;

            for (i = 0 ; i < 13 ; i++)
            {
                    sp = strstr(sp, "= ");
                    sp = sp+2;
                    sscanf(sp, "%ul", (unsigned int *)&cp[i]);
            }
        printf("Tx success                          = %u\n", (unsigned int)counter.TxSuccessTotal);
        printf("Tx success without retry        = %u\n", (unsigned int)
                                                                  counter.TxSuccessWithoutRetry);
        printf("Tx success after retry          = %u\n", (unsigned int)counter.TxSuccessWithRetry);
        printf("Tx fail to Rcv ACK after retry  = %u\n", (unsigned int)counter.TxFailWithRetry);
        printf("RTS Success Rcv CTS                 = %u\n", (unsigned int)counter.RtsSuccess);
        printf("RTS Fail Rcv CTS                    = %u\n", (unsigned int)counter.RtsFail);
        printf("Rx success                              = %u\n", (unsigned int)counter.RxSuccess);
        printf("Rx with CRC                         = %u\n", (unsigned int)counter.RxWithCRC);
        printf("Rx drop due to out of resource= %u\n", (unsigned int)counter.RxDropNoBuffer);
        printf("Rx duplicate frame                  = %u\n", (unsigned int)counter.RxDuplicateFrame);
        printf("False CCA (one second)          = %u\n", (unsigned int)counter.FalseCCA);
        printf("RSSI-A                              = %d\n", (  signed int)counter.RssiA);
        printf("RSSI-B (if available)           = %d\n", (  signed int)counter.RssiB);
       }

#if 0
        //set AP to do site survey, remove "set" string --------------------------
        memset(data, 0x00, 255);
        strcpy(data, "SiteSurvey=1");
        strcpy(wrq.ifr_name, name);
        wrq.u.data.length = strlen(data)+1;
        wrq.u.data.pointer = data;
        wrq.u.data.flags = 0;
        ret = ioctl(socket_id, RTPRIV_IOCTL_SET, &wrq);
#endif
```

```
//get AP's site survey, remove "get_site_survey" string -------------------
memset(data, 0x00, 2048);
strcpy(data, "");
strcpy(wrq.ifr_name, name);
wrq.u.data.length = 4096;
wrq.u.data.pointer = data;
wrq.u.data.flags = 0;
ret = ioctl(socket_id, RTPRIV_IOCTL_GSITESURVEY, &wrq);
if(ret != 0)
{
        printf("\nrtuser::error::get site survey\n\n");
        goto rtuser_exit;
}

//printf("\n%s\n", wrq.u.data.pointer);
printf("\n========== Get Site Survey AP List ==========");
if(wrq.u.data.length > 0)
{
        int     i, apCount;
        char *sp, *op;
        int     len = wrq.u.data.length;

        op = sp = wrq.u.data.pointer;
        sp = sp+1+8+8+35+19+8+1;
        i = 0;
        // santy check
        //      1. valid char data
        //      2. rest length is larger than per line length ==> (1+8+8+35+19+8+1)
        while(*sp && ((len - (sp-op)) > (1+8+8+35+19+8)))
        {
                //if(*sp++ == '\n')
                //      continue;
                //printf("\n\nAP Count: %d\n", i);
```

```
          sscanf(sp, "%d", (int *)&SiteSurvey[i].channel);
          //printf("channel: %d\n", SiteSurvey[i].channel);

          sp = strstr(sp, "-");
          sscanf(sp, "-%d", (int *)&SiteSurvey[i].rssi);
          //printf("rssi: -%d\n", SiteSurvey[i].rssi);

          sp = sp+8;
          strncpy((char *)&SiteSurvey[i].ssid, sp, 32);
          SiteSurvey[i].ssid[32] = '\0';
          //printf("ssid: %s\n", SiteSurvey[i].ssid);

          sp = sp+35;
          sscanf(sp, "%02x:%02x:%02x:%02x:%02x:%02x",
                    (int *)&SiteSurvey[i].bssid[0], (int *)&SiteSurvey[i].bssid[1],
                    (int *)&SiteSurvey[i].bssid[2], (int *)&SiteSurvey[i].bssid[3],
                    (int *)&SiteSurvey[i].bssid[4], (int *)&SiteSurvey[i].bssid[5]);
          //printf("bssid: %02x:%02x:%02x:%02x:%02x:%02x\n",
          //        SiteSurvey[i].bssid[0], SiteSurvey[i].bssid[1],
          //        SiteSurvey[i].bssid[2], SiteSurvey[i].bssid[3],
          //        SiteSurvey[i].bssid[4], SiteSurvey[i].bssid[5]);

          sp = sp+19;
          strncpy((char *)&SiteSurvey[i].security, sp, 8);
          SiteSurvey[i].security[8] = '\0';
          //printf("security: %s\n", SiteSurvey[i].security);

          sp = sp+8+1;
          i = i+1;
     }

     apCount = i;
     printf("\n%-4s%-8s%-8s%-35s%-20s%-8s\n",
          "AP", "Channel", "RSSI", "SSID", "BSSID", "Security");
     for(i = 0 ; i < apCount ; i++)
```

```
	{//4+8+8+35+20+8
		printf("%-4d", i+1);
		printf("%-8d", SiteSurvey[i].channel);
		printf("-%-7d", SiteSurvey[i].rssi);
		printf("%-35s", SiteSurvey[i].ssid);
		printf("%02X:%02X:%02X:%02X:%02X:%02X   ",
				SiteSurvey[i].bssid[0], SiteSurvey[i].bssid[1],
				SiteSurvey[i].bssid[2], SiteSurvey[i].bssid[3],
				SiteSurvey[i].bssid[4], SiteSurvey[i].bssid[5]);
		printf("%-8s\n", SiteSurvey[i].security);
	}
}
```

//get AP's mac table, remove "get_mac_table" string ----------------------

```
memset(data, 0x00, 2048);
strcpy(data, "");
strcpy(wrq.ifr_name, name);
wrq.u.data.length = 2048;
wrq.u.data.pointer = data;
wrq.u.data.flags = 0;
ret = ioctl(socket_id, RTPRIV_IOCTL_GET_MAC_TABLE, &wrq);
if(ret != 0)
{
	printf("\nrtuser::error::get mac table\n\n");
	goto rtuser_exit;
}

printf("\n========== Get Associated MAC Table ==========");
{
	RT_802_11_MAC_TABLE		*mp;
	int			i;

	mp = (RT_802_11_MAC_TABLE *)wrq.u.data.pointer;
	printf("\n%-4s%-20s%-4s%-10s%-10s%-10s\n",
		"AID", "MAC_Address", "PSM", "LastTime", "RxByte", "TxByte");
```

```
        for(i = 0 ; i < mp->Num ; i++)
        {
            printf("%-4d", mp->Entry[i].Aid);
            printf("%02X:%02X:%02X:%02X:%02X:%02X   ",
                    mp->Entry[i].Addr[0], mp->Entry[i].Addr[1],
                    mp->Entry[i].Addr[2], mp->Entry[i].Addr[3],
                    mp->Entry[i].Addr[4], mp->Entry[i].Addr[5]);
            printf("%-4d", mp->Entry[i].Psm);
            printf("%-10u", (unsigned int)mp->Entry[i].HSCounter.LastDataPacketTime);
            printf("%-10u", (unsigned int)mp->Entry[i].HSCounter.TotalRxByteCount);
            printf("%-10u", (unsigned int)mp->Entry[i].HSCounter.TotalTxByteCount);
            printf("\n");
        }
        printf("\n");
    }

//set: raw data
//      RTPRIV_IOCTL_RADIUS_DATA
//      RTPRIV_IOCTL_ADD_WPA_KEY
//      RTPRIV_IOCTL_ADD_PMKID_CACHE

    //set RADIUS Data --------------------------------------------------------
    printf("\nrtuser::set radius data\n\n");
    memset(data, 0x55, 100);
    strcpy(wrq.ifr_name, name);
    wrq.u.data.length = 100;
    wrq.u.data.pointer = data;
    wrq.u.data.flags = 0;
    ret = ioctl(socket_id, RTPRIV_IOCTL_RADIUS_DATA, &wrq);
    if(ret != 0)
    {
        printf("\nrtuser::error::set radius data\n\n");
        goto rtuser_exit;
    }
```

```
//add WPA Key --------------------------------------------------------------
printf("\nrtuser::add wpa key\n\n");
{
        NDIS_802_11_KEY         *vp;

        memset(data, 0, sizeof(NDIS_802_11_KEY));
        vp = (NDIS_802_11_KEY *)&data;

        vp->Length = sizeof(NDIS_802_11_KEY);
        memset(vp->addr, 0x11, 6);
        vp->KeyIndex = 2;
        vp->KeyLength = 32;
        memset(vp->KeyMaterial, 0xAA, 32);

        strcpy(wrq.ifr_name, name);
        wrq.u.data.length = sizeof(NDIS_802_11_KEY);
        wrq.u.data.pointer = data;
        wrq.u.data.flags = 0;
        ret = ioctl(socket_id, RTPRIV_IOCTL_ADD_WPA_KEY, &wrq);
        if(ret != 0)
        {
                printf("\nrtuser::error::add wpa key\n\n");
                goto rtuser_exit;
        }
}

//add PMKID_CACHE ----------------------------------------------------------
printf("\nrtuser::add PMKID_CACHE\n\n");
{
        NDIS_802_11_KEY         *vp;

        memset(data, 0, sizeof(NDIS_802_11_KEY));
        vp = (NDIS_802_11_KEY *)&data;
```

```
		vp->Length = sizeof(NDIS_802_11_KEY);
		memset(vp->addr, 0x11, 6);
		vp->KeyIndex = 2;
		vp->KeyLength = 32;
		memset(vp->KeyMaterial, 0xBB, 32);

		strcpy(wrq.ifr_name, name);
		wrq.u.data.length = sizeof(NDIS_802_11_KEY);
		wrq.u.data.pointer = data;
		wrq.u.data.flags = 0;
		ret = ioctl(socket_id, RTPRIV_IOCTL_ADD_PMKID_CACHE, &wrq);
		if(ret != 0)
		{
			printf("\nrtuser::error::add PMKID_CACHE\n\n");
			goto rtuser_exit;
		}
	}

//set: raw data
//	RT_SET_APD_PID
//	RT_SET_DEL_MAC_ENTRY

	//set APD_PID ------------------------------------------------------------
	printf("\nrtuser::set APD_PID\n\n");
	memset(data, 0, 4);
	data[0] = 12;
	strcpy(wrq.ifr_name, name);
	wrq.u.data.length = 4;
	wrq.u.data.pointer = data;
	wrq.u.data.flags = RT_SET_APD_PID;
	ret = ioctl(socket_id, RT_PRIV_IOCTL, &wrq);
	if(ret != 0)
	{
		printf("\nrtuser::error::set APD_PID\n\n");
		goto rtuser_exit;
```

```
        }

        //set DEL_MAC_ENTRY -------------------------------------------------------
        printf("\nrtuser::set DEL_MAC_ENTRY\n\n");
        memset(data, 0xdd, 6);
        strcpy(wrq.ifr_name, name);
        wrq.u.data.length = 6;
        wrq.u.data.pointer = data;
        wrq.u.data.flags = RT_SET_DEL_MAC_ENTRY;
        ret = ioctl(socket_id, RT_PRIV_IOCTL, &wrq);
        if(ret != 0)
        {
                printf("\nrtuser::error::set DEL_MAC_ENTRY\n\n");
                goto rtuser_exit;
        }

//get: raw data
//      RT_QUERY_ATE_TXDONE_COUNT
//      RT_QUERY_SIGNAL_CONTEXT

        //get ATE_TXDONE_COUNT ----------------------------------------------------
        printf("\nrtuser::get ATE_TXDONE_COUNT\n\n");
        memset(data, 0, 4);
        strcpy(wrq.ifr_name, name);
        wrq.u.data.length = 4;
        wrq.u.data.pointer = data;
        wrq.u.data.flags = RT_QUERY_ATE_TXDONE_COUNT;
        ret = ioctl(socket_id, RT_PRIV_IOCTL, &wrq);
        if(ret != 0)
        {
                printf("\nrtuser::error::get ATE_TXDONE_COUNT\n\n");
                goto rtuser_exit;
        }
        printf("\nATE_TXDONE_COUNT:: %08lx\n\n", (unsigned long)*wrq.u.data.pointer);
```

```
//get SIGNAL_CONTEXT ------------------------------------------------------
printf("\nrtuser::get SIGNAL_CONTEXT\n\n");
{
	RT_SIGNAL_STRUC			*sp;

	memset(data, 0, sizeof(RT_SIGNAL_STRUC));
	strcpy(wrq.ifr_name, name);
	wrq.u.data.length = sizeof(RT_SIGNAL_STRUC);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = RT_QUERY_SIGNAL_CONTEXT;
	ret = ioctl(socket_id, RT_PRIV_IOCTL, &wrq);
	if(ret != 0)
	{
		printf("\nrtuser::error::get SIGNAL_CONTEXT\n\n");
		goto rtuser_exit;
	}
	sp = (RT_SIGNAL_STRUC *)wrq.u.data.pointer;
	printf("\n===== SIGNAL_CONTEXT =====\n\n");
	printf("Sequence    = 0x%04x\n", sp->Sequence);
	printf("Mac.Addr    = %02x:%02x:%02x:%02x:%02x:%02x\n",
				sp->MacAddr[0], sp->MacAddr[1],
				sp->MacAddr[2], sp->MacAddr[3],
				sp->MacAddr[4], sp->MacAddr[5]);
	printf("CurrAP.Addr = %02x:%02x:%02x:%02x:%02x:%02x\n",
				sp->CurrAPAddr[0], sp->CurrAPAddr[1],
				sp->CurrAPAddr[2], sp->CurrAPAddr[3],
				sp->CurrAPAddr[4], sp->CurrAPAddr[5]);
	printf("Sig         = %d\n\n", sp->Sig);
}

//SSID, remove "set" string ------------------------------------------------
memset(data, 0x00, 255);
strcpy(data, "SSID=rtuser");
strcpy(wrq.ifr_name, name);
wrq.u.data.length = strlen(data)+1;
```

```
	wrq.u.data.pointer = data;
	wrq.u.data.flags = 0;
	ret = ioctl(socket_id, RTPRIV_IOCTL_SET, &wrq);
	if(ret != 0)
	{
		printf("\nrtuser::error::set SSID\n\n");
		goto rtuser_exit;
	}

rtuser_exit:
	if (socket_id >= 0)
		close(socket_id);

	if(ret)
		return ret;
	else
		return 0;
}
```

16 PORTING GUIDE

This source code package can be use with Linux versions after RedHat Linux 7.3

16.1 Source code package file path and description

./Module/ap/	# ap specific
./Module/common/	# common use
./Module/include/	# header file
./Module/os/Linux/	# OS specific
./Module/tools/	# tool to convert firmware
./Module/	
Makefile	
RT2860AP.dat	# initial profile stored in /etc/Wireless/RT2860AP/

16.2 Compile Flags

Add compile flags (CFLAGS) to Makefile to support specific driver codes.

1.	-DDBG	# turn on driver debug message
2.	-DRALINK_ATE	# turn on ATE functionality
3.	-DRALINK_2860_QA	# turn on QA support, refer to Q&A.
4.	-DBIG_ENDIAN	# turn on BigEndian platform's code
5.	-DMBSS_SUPPORT	# turn on multiple BSSID support
6.	-DAGGREGATION_SUPPORT	# turn on packet aggregation support
7.	-DPIGGYBACK_SUPPORT	# turn on piggy back ack support
8.	-DWDS_SUPPORT	# turn on WDS support
9.	-DWMM_SUPPORT	# turn on WMM support
10.	-DUAPSD_AP_SUPPORT	# turn on WMM-PS wupport
11.	-DAPCLI_SUPPORT	# turn on ApClient support
12.	-DMAT_SUPPORT	# turn on ApClient's MAT support
13.	-DIGMP_SNOOP_SUPPORT	# turn on IGMP support
14.	-DWSC_AP_SUPPORT	# turn on WSC support
15.	-DLLTD_SUPPORT	# turn on LLTD support
16.	-DCONFIG_5VT_ENHANCE	# turn on 5VT platform enhancement

16.3 Porting Note

1. In single processor system, macro like NdisAllocateSpinLock, NdisReleaseSpinLock and NdisAcquireSpinLock in rtmp.h can be re-implement as semaphore lock to improve proformance.
2. This module provide several interfaces for user layer process to communicate with module, like iwconfig/iwpriv or proprietary ioctl. You can remove interface-code you don't need to minimize code size.
3. In embedded system, it is prefered to modify the "NdisMoveMemory" routine in rtmp_init.c as kernel's memcpy routine to enhance performance.
4. When performance can not reach to reasonable value, tuning DRAM timing(clock) maybe have some effort.
5. For embedded device application, add “PACKED” to data structure that is related to:
   5.1. Hardware – MAC: PCI device, Little-Endian, 32-bit alignment
   5.2. 802.11 header – Little-Endian

16.4 RT2800 Notes for Embedded Device Applications

1. PCI's byte order is Little-Endian.

2. 802.11's header is Little-Endian.
3. RT2800 is PCI based device:
   3.1. Bus Master
   3.2. DMA Based
   3.3. Physical Memory Access
   3.4. Non-Cacheable(Data-Cache)
   3.5. Effect to Descriptor and Data Buffer
4. Hardware is referred to Fixed Offset, no padding and apply PACKED to
   4.1. Data Structure
   4.2. 802.11 Header
5. Spinlock_xxx:
   5.1. spin_lock_irqsave(&flags)
   5.2. spin_unlock_irqrestore(flags)
6. Big-Endian:
   6.1. Bit is Reverse relative to Little-Endian
   6.2. After data swap to fit data structure
   6.3. If reference only, needn't to write back
   6.4. If modified, need to write back
7. Security Setting:
   7.1. 1st: Set SSID
   7.2. 2nd: Set Pass-Parse
   7.3. 3rd: Set SSID to update capability information.
8. TxRate fixed at 11Mbps
   8.1. Check assoc.c on build association connection, data rate is fixed on each associated station.
   8.2. After data rate changed, station have to de-associate then re-associate to take in effect on rate change.
9. B/G Protection = ON:
   9.1. Would trigger CTS-To-Self mechanism
   9.2. Performance would downgrade around 25% to 33%
   9.3. Check below factors:
      9.3.1. Slot time is short or long ?
      9.3.2. Short retry or long retry ?
      9.3.3. SIF time's setting ?
10. MCU not ready.
    10.1. Do delay loop to wait MCU ready.
11. Clear Beacon's Tx valid bit before setup Beacon frame on AP initial stage.
12. Default 8-bit to load firmware, depends on platform may change to 32-bit and/or have to do byte-swap.

17 MAKE FILES

THE PATH PLACE HERE IS USED AS AN EXAMPLE AND IS FOR REFERENCE ONLY.

PLEASE MODIFY THE PATH TO MATCH TARGET SOURCE AND TOOL CHAINS BY FOLLOW THE INSTRUCTIONS FROM VENDOR'S BSP.

```
#MODE  STA or AP

#RT2860_MODE = STA

RT2860_MODE = AP

#TARTET = LINUX or UCOS

TARGET = LINUX

#RT2860_DIR = home directory of RT2860 source code

RT2860_DIR = $(shell pwd)

#PLATFORM = 5VT

PLATFORM = PC

#PLATFORM = STAR

#PLATFORM = IXP

#PLATFORM = INF_TWINPASS


ifeq ($(PLATFORM),5VT)

LINUX_SRC = /root/CVS_PROJECT/Gemtek_5VT_Ralink_SDK-20070109/Linux-2.6.17.6-11n5

CROSS_COMPILE = /opt/crosstool/uClibc/bin/arm-Linux-

endif


ifeq ($(PLATFORM),STAR)

LINUX_SRC = /opt/star/kernel/Linux-2.4.27-star

CROSS_COMPILE = /opt/star/tools/arm-Linux/bin/arm-Linux-

endif


ifeq ($(PLATFORM),PC)

# Linux 2.6

#LINUX_SRC = /lib/modules/$(shell uname -r)/build

# Linux 2.4 Change to your local setting

LINUX_SRC = /usr/src/Linux-2.4.27

CROSS_COMPILE =

endif
```

```
ifeq ($(PLATFORM),IXP)

LINUX_SRC = /project/stable/Gmtek/snapgear-uclibc/Linux-2.6.x

CROSS_COMPILE = arm-Linux-

endif


ifeq ($(PLATFORM),INF_TWINPASS)

# Linux 2.6

#LINUX_SRC = /lib/modules/$(shell uname -r)/build

# Linux 2.4 Change to your local setting

LINUX_SRC = /project/stable/twinpass/release/2.0.1/source/kernel/opensource/Linux-2.4.31/

CROSS_COMPILE = mips-Linux-

endif



export RT2860_DIR RT2860_MODE LINUX_SRC CROSS_COMPILE PLATFORM


all: build_tools $(TARGET)



build_tools:

    make -C tools

    $(RT2860_DIR)/tools/bin2h


UCOS:

    make -C os/ucos/ MODE=$(RT2860_MODE)

    echo $(RT2860_MODE)


LINUX:

ifneq (,$(findstring 2.4,$(LINUX_SRC)))

    cp -f os/Linux/Makefile.4 os/Linux/Makefile

    make -C os/Linux/

ifeq ($(RT2860_MODE),AP)

    cp -f $(RT2860_DIR)/os/Linux/rt2860ap.o /tftpboot

else
```

```
    cp -f $(RT2860_DIR)/os/Linux/rt2860sta.o /tftpboot
endif
else
    cp -f os/Linux/Makefile.6 os/Linux/Makefile
    make  -C  $(LINUX_SRC) SUBDIRS=$(PWD)/os/Linux modules
ifeq ($(RT2860_MODE),AP)
    cp -f $(RT2860_DIR)/os/Linux/rt2860ap.ko /tftpboot
else
    cp -f $(RT2860_DIR)/os/Linux/rt2860sta.ko /tftpboot
endif
endif

release:
ifeq ($(TARGET), LINUX)
    make -C os/Linux -f Makefile.release release
endif


clean:
ifeq ($(TARGET), LINUX)
ifneq (,$(findstring 2.4,$(LINUX_SRC)))
    cp -f os/Linux/Makefile.4 os/Linux/Makefile
else
    cp -f os/Linux/Makefile.6 os/Linux/Makefile
endif
    make -C os/Linux clean
    rm -rf os/Linux/Makefile
endif
ifeq ($(TARGET), UCOS)
    make -C os/ucos clean MODE=$(RT2860_MODE)
endif
```

18 MISCELLANEOUS

18.1 Multiple BSSID

1. Before turn on multiple BSSID, make sure the byte5 of MAC address in EEPROM is a multiple of 1/2/4/8 and reserve multiple MAC address when manufacturing. example, 00:0A:0B:0C:0D:04; 00:0A:0B:0C:0D:88.

2. When enable multiple BSSID function, the field 'BssidNum' shall larger than 1 and less than 8.

3. BssidNum can only be modified with editing configure file.
   When change the ' BssidNum ' field, the driver must restart, and modify bridge_setup file to group virtual interface.
   Others parameters can pass through iwpriv according to their interface.

4. The parameter that support multiple BSSID is listed as followed,

SSID	Key2Str	IEEE8021X
AuthMode	Key3Type	TxRate
EncrypType	Key3Str	HideSSID
WPAPSK	Key4Type	PreAuth
DefaultKeyID	Key4Str	WmmCapable
Key1Type	AccessPolicy	* Others are not supported.
Key1Str	AccessControlList	
Key2Type	NoForwarding	

5. Example of notation to represent multiple ssid's parameter:

   1.) BssidNum=4

   2.) SSID=SSID-A;SSID-B;SSID-C;SSID-D

   3.) AuthMode=OPEN;SHARED;WPAPSK;WPA

   4.) EncrypType=NONE;WEP;TKIP;AES

6. The WDS's security policy must be the same as main BSSID and only support NONE, WEP, TKIP, and AES.

7. MBSSID and WDS.
   There 64 security key table in MAC(RT2800).
   Entry 0:   For reserved.
   Entry 1 - 59:   For Associated STA and WDS link.
   Current driver defined WDS number to 4.

18.2 Concurrent A+G with two devices

Below table is brief example for two interface.

For example, Linux HotPlug system found new device would create one driver instance(create new space for driver image) for new device to hold private informations(memory consumed).

RT2800 Interface Bring Up Sequence

NIC#	Sequence	Normal	WDS(Virtual)			
			1	2	3	4
Two	ifconfig ra0 up	ra0	wds0	wds1	wds2	wds3
	ifconfig ra1 up	ra1	wds4	wds5	wds6	wds7

NIC#	Sequence	Normal	MBSSID (Physical)			WDS(Virtual)			
						1	2	3	4
Two	ifconfig ra0 up	ra0	ra2	ra3	ra4	wds0	wds1	wds2	wds3
	ifconfig ra1 up	ra1	ra5	ra6	ra7	wds4	wds5	wds6	wds7

WDS IS A VIRTUAL INTERFACE WITHOUT IOCTL FUNCTIONALITY.

18.3 Site Survey

1. Site survey issue "iwpriv ra0 set SiteSurvey=1"
2. After 4 seconds (wait site survey process complete) then issue "iwpriv ra0 get_site_survey" command to get data.
3. We can use system("iwpriv ra0 get_site_survey > /etc/site_survey.dat") then it will write the site survey data to /etc/site_survey.dat.

18.4 OLBC

DisableOLBC=1	Disable Co-Channel OLBC AP/STA Detection.
DisableOLBC=0	Enable Co-Channel OLBC AP/STA Detection.

Overlapping Legacy BSS Condition (OLBC)

BGProtection	DisableOLBC	
	1 (Disable)	0 (Enable)
AUTO	Condition to Turn ON CTS-To-Self Protection	

	Only Associated 11B Client(STA).	Associated 11B Client(STA) Co-Channel with 11B only mode Other 11B’s AP 11B’s STA that associated to Other 11B’s AP
ON	CTS-To-Self Protection Always ON	CTS-To-Self Protection Always ON
OFF	No CTS-To-Self Protection	No CTS-To-Self Protection

Note:

1. BGProtection only has CTS-To-Self.
2. If the condition of RTS-CTS Threshold be triggerred then RTS-CTS Protection will turn on, no matter what setting of BGProtection.

   ~~Example 1:~~

   Assume:

   - a. RTS Threshold = 500 Bytes.
   - b. Length of Data Packet = 600 bytes

   Result:

   a. Packet#1 RTS

   - a. Packet#2 ← CTS
   - b. Packet#3 Data Packet#1 (500 Bytes)
   - c. Packet#4 ← Ack
   - d. Packet#5 Data Packet#2 (100 Bytes)
   - e. Packet#6 ← Ack

   ~~Example 2:~~

   Assume:

   - a. RTS Threshold = 500 Bytes.
   - b. Length of Data Packet = 490 bytes

   Result:

   - a. Packet#1 Data Packet#1 (490 Bytes)
   - b. Packet#2 ← Ack

3. For OLBC, please refer to section 2.21 of “WiFi-802_11g-TestPlan_V2_2.pdf”.

18.5 Tx Power

RT2800 Tx Power Cross Reference			
EEPROM	RF[R3], Tx1	RF[R4], Tx2	Description

0x00 = 0	0x00 = 0	0x00 = 0	
0x01 = 1	0x01 = 1	0x01 = 1	
0x02 = 2	0x02 = 2	0x02 = 2	
0x03 = 3	0x03 = 3	0x03 = 3	
0x04 = 4	0x04 = 4	0x04 = 4	
0x05 = 5	0x05 = 5	0x05 = 5	
0x06 = 6	0x06 = 6	0x06 = 6	
0x07 = 6	0x07 = 6	0x07 = 6	
0x08 = 8	0x08 = 8	0x08 = 8	
0x09 = 9	0x09 = 9	0x09 = 9	
0x0A = 10	0x0A = 10	0x0A = 10	
0x0B = 11	0x0B = 11	0x0B = 11	
0x0C = 12	0x0C = 12	0x0C = 12	
0x0D = 13	0x0D = 13	0x0D = 13	
0x0E = 14	0x0E = 14	0x0E = 14	
0x0F = 15	0x0F = 15	0x0F = 15	In normal BBP range
0x10 = 16	0x10 = 16	0x10 = 16	Per Step = 1 = 0.5dB
0x11 = 17	0x11 = 17	0x11 = 17	
0x12 = 18	0x12 = 18	0x12 = 18	
0x13 = 19	0x13 = 19	0x13 = 19	
0x14 = 20	0x14 = 20	0x14 = 20	
0x15 = 21	0x15 = 21	0x15 = 21	
0x16 = 22	0x16 = 22	0x16 = 22	
0x17 = 23	0x17 = 23	0x17 = 23	
0x18 = 24	0x18 = 24	0x18 = 24	
0x19 = 25	0x19 = 25	0x19 = 25	
0x1A = 26	0x1A = 26	0x1A = 26	
0x1B = 27	0x1B = 27	0x1B = 27	
0x1C = 28	0x1C = 28	0x1C = 28	
0x1D = 29	0x1D = 29	0x1D = 29	
0x1E = 30	0x1E = 30	0x1E = 30	
0x1F = 31	0x1F = 31	0x1F = 31	

TxPower=value

parameter :: TxPower

Value

100 ~ 90 use value in E2PROM as default

90 ~ 60 default value -2

60 ~ 30 default value -6

30 ~ 15 default value -12

15 ~ 9default value -18

9 ~ 0 default value -24

Note:

1. Range: 1 ~ 100 (unit in percentage)

2. This value restricted by HW characteristic.

TxPower		
percentage		
100 ~ 90	Default value from E2PROM	

90 ~ 60	default value -2	-1dB
60 ~ 30	default value -6	-3dB
30 ~ 15	default value -12	-6dB
15 ~ 9	default value -18	-9dB
9 ~ 0	default value -24	-12dB

18.6 Auto Channel Selection

18.6.1 Rules

RT2800AP driver will traverse all supported channels when system bootup.

Driver will stay 0.5 sec in each channel and collect necessary information - Max RSSI.

Driver implements a dirty rate for each channel to qualify which channel is suitable for selecting.

If the Max RSSI is not equal to zero, the channel's dirty rate will plus 10.

The upper and the lower 4 channel's dirty rate will plus one.

Finally,

RULE 1. pick up a good channel that no one used (dirtyness=0)

RULE 2. if not available, then co-use a channel that's no interference (dirtyness=10)

RULE 3. if not available, then co-use a channel that has minimum interferenc (dirtyness=11,12)

RULE 4. still not available, pick up the first channel

When AP scan through each channel (stay 0.5 sec) upon bootup. It'll maintain a max_rx_rssi for each channel, which value is actually acquired from each correctly received BEACON frames.

max_rx_rssi[ch] is used ony when this AP can't find a 100% clean channel (no neighbor AP within 5 channel apart) and there're more than 1 equal-dirty channels to choose from. In this case, this AP would choose the channel with smallest max_rx_rssi[ch] because this means the neighbor AP is more far away than the one in other channel.

The fundamental problem is -

Auto Channel Selection function decide channel dirtyness solely base on correcty received 802.11 BEACONs. All other signal/frame are not used (or not able to use) as an indication.

18.6.2 Practice

1. In the shielding room, the client can see 4 out side APs with very low power level. Channel_2 -91dB, Channel_3 -92dB, Channel_4 -91dB, Channel_6 -91dB. Set the channel to Auto and power on 5 times, the RT2800AP goes to CH 1,1,1,1,1.

If there are several outside APs and the signal are too weak and are actually invisible (no CRC-ok BEACON seen) at least during the RT2800AP power-on period (e.g. theRSSI is -91dB). Therefore all 11 channels(assume country region is FCC) are clean, thus RT2800AP just pickup the first clean channel which is channel 1.

2. In the shielding room, set one AP to Channel_1, and power on RT2800AP 5 times, it goes to Channel 6, 6, 6, 6, 6.

   Now channel 1 is occupied, so does channel 2,3,4,5 become a little dirty (to avoid interference from AP_Channel_1), channel 6 is chosen because it's the first clean channel.

3. As item 2, now add another AP to Channel_6, and power on RT2800AP 5 times, it goes to Channel 11, 11, 11, 11, 11.

   Then channel 6 also occupied, and channel 2,3,4,5,7,8,9,10 all dirty. Channel 11 is a correct decision.

4. As item 3, now add another AP to Channel_11, and power on RT2800AP 5 times, it goes to Channel 1, 6, 6, 6, 6.

   Now channel 11 is occupied, and no clean channel at all. RT2800AP decide to co-channel with other AP, but prefer that co-channel AP to be as far away as possible so it may choose channel 1, 6, or 11 depending which co-channel AP has smallest RSSI.

   Since all devices stay in shielding room, the RSSI may be very close. This explains why RT2800AP sometimes choose channel 1, sometimes choose channel 6. You can check the distance of each AP to confirm that AP_Channel_1 and AP_Channel_6 is about the same distance to RT2800AP, while AP_Channel_11 is closer.

5. Add 16M(Tx+Rx) traffic to AP in Channel_6, and power on RT2800AP 5 times, it goes to Channel 1, 6, 6, 1, 6.

   Since RT2800AP only count max_rx_rssi[ch] from correctly received BEACON. The extra traffic load won't affect the election result. RT2800AP still picks up eiher Channel 1 or Channel 6 depends on the max_rx_rssi.

Maybe this algorithm is not perfect. But think about that data traffic is bursty by nature. So put weighing on this 0.5sec bootup-time traffic doesn't mean that much. AP_Channel_1 and AP_Channel_11 still may generate heavy loading later on.

As for

a. Channel 2,3,4,5, will interfere both AP_Channel_1 and AP_Channel_6, and

b. Channel 7,8,9,10 will intefere both AP_Channel_6 and AP_Channel_11.

So why picking up channel 3 or 8 is not a good choice.

18.7 The Difference of WPA1 and WPA2

18.7.1 WPA1

18.7.1.1 WI-FI WPA

Refer to "~~Wi-Fi 802.11g Interoperability Test Plan Version 2.4, Page 7~~":

"The WPA protocol is defined by Wi-Fi document 'WPA for 802.11 Specification – Version 2.0, April 29, 2003'. The WPA Specification captures those clauses of the ~~IEEE 802.11i Draft 3.0~~ that define Wi-Fi Protected Access."

18.7.1.2 IEEE 802.11I/D3.0 WPA

1. Pairwise key would be installed after 4-way handshake.
2. Group key would be installed before 2-way handshake.
3. Refer to "~~P802.11i/D3.0, November 2002, Page 80, Section 8.4.5 MPDU filtering, Figure 45—Sequence of Filtering-related Events~~" for detail information.

18.7.1.3 WPA1 PRACTICE

```
*RT2800*<7>AUTH_RSP-Rcv AUTH seq#1,Alg=0,Status=0 from 00:0c:43:26:61:25 to IF(ra0)

*RT2800*<7>MacTableInsertEntry -IF(ra0) allocate entry #1, Total= 1

*RT2800*<7>AUTH_RSP - IF(0) Send AUTH response (SUCCESS)...

*RT2800*<7>ASSOC - receive ASSOC request from 00:0c:43:26:61:25

*RT2800*<7>AssignAid (AID=1)

*RT2800*<7>BuildAssoc-IF(0):AuthMode=4,WepStatus=6,GroupWepStatus=6,WpaState=7,AGGRE=1,PiggyBack=1,
APSD=0

*RT2800*<7>LOG#6 00:0c:43:26:61:25 successfully associated

*RT2800*<7>Init entry init retry timer

*RT2800*<7>assign AID=1 to 00:0c:43:26:61:25,MaxSupportedRate=54Mbps,CurrTxRate=54Mbps

*RT2800*<7>RSNIE_Len=0x16,pEntry->RSNIE_Len=22,pEntry->PrivacyFilter=1

*RT2800*<7>ASSOC - Send ASSOC response (Status=0) from IF(ra0)...

WpaEAPOLStartAction ====>>

==>WPAStart4WayHS

STA from 00:0c:43:26:61:25

PMK = 99:61:62:c4-86:a8:8d:bf

pEntry->AuthMode == Ndis802_11AuthModeWPA/WPAPSK

WPA - RTMPToWirelessSta ====>> to IF(ra0)

<== WPAStart4WayHS:pEntry->WpaState=8, FrameLen=113

Receive EAPOL-Key frame, TYPE = 3, Length =0

WPAMsgTypeSubst (EAPType=3)

WpaEAPOLKeyAction ===>

PeerPairMsg2Action ===>

PTK-ed 32 1f e3 2a 6f c4 e9
```

```
ANonce1-d5 1c 3c 54 7b 91 cb fd

ANonce2-dc 39 f1 bc cc 2 5e 77

MIC VALID in Msg 2 of 4-way handshake!!

RSN_IE VALID in Msg 2 of 4-way handshake!!

RTMPToWirelessSta : ETHTYPE = 88 8e FrameLen = 137!

WPA - RTMPToWirelessSta ====>> to IF(ra0)

Send Msg3 and setup timeout timer

Receive EAPOL-Key frame, TYPE = 3, Length =0

WPAMsgTypeSubst (EAPType=3)

WpaEAPOLKeyAction ===>

WpaEAPOL Peer Pair Msg4 Action ===>

MIC valid in Msg 4 of 4-way handshake!!

WPA1(PairwiseKey) = 63:c5:5d:75-7e:8c:b6:08

WPA1(RxMic) = fc:7a:1c:5f-95:72:62:e2

WPA1(TxMic) = 83:35:1f:67-54:fe:a5:67

*RT2800*<7>AsicAddPairwiseKeyEntry: #1 Alg=AES mac=00:0c:43:26:61:25 key=63-c5-5d-..

IF(ra0) WPA Group Key ID  = 1

c 37 cf 69 cd 7c 85 49

83 f9 e2 2c ad a8 cc e7

f0 7 d2 b9 62 9a bd 3e

e9 b5 c0 a2 1 f9 d6 17

*RT2800*<7>AsicAddSharedKeyEntry(BssIndex=0): AES key #1

*RT2800*<7>     Key =0c:37:cf:69:cd:7c:85:49:83:f9:e2:2c:ad:a8:cc:e7:

*RT2800*<7>     Rx MIC Key = e9:b5:c0:a2:01:f9:d6:17:

*RT2800*<7>     Tx MIC Key = f0:07:d2:b9:62:9a:bd:3e:

<== IF(ra0) WPAHardTransmit - FrameLen = 137

WPA - RTMPToWirelessSta ====>> to IF(ra0)

IF(ra0) recv WpaEAPOL Peer PAIR Msg4 Action and send GROUP Msg1

Receive EAPOL-Key frame, TYPE = 3, Length =0

WPAMsgTypeSubst (EAPType=3)

WpaEAPOLKeyAction ===>

PeerGroupMsg2Action ===> from MAC(00:0c:43:26:61:25)

Replay Counter VALID in Msg 2 of GROUP 2-way handshake!!!

MIC Valid in Msg 2 of GROUP 2-way handshake.
```

```
===> AP SETKEYS DONE - (ra0) WPA1, AuthMode=4, WepStatus=6
```

18.7.2 WPA2

18.7.2.1 WI-FI WPA2

Wi-Fi 802.11 WPA2 Interoperability Test Plan Version 2.4.2, Page 7:

“The WPA2 protocol is based upon the IEEE 802.11i specification.”

18.7.2.2 IEEE 802.11I WPA

1. Group key would be installed after AP received message 2 before send message 3.
2. Pairwise key would be installed after AP received message 4.
3. Refer to “~~IEEE Std 802.11i-2004, Page 87, Section 8.5.3.3 4-Way Handshake Message 3~~” for detail information.

18.7.2.3 WPA2 PRACTICE

```
*RT2800*<7>ASSOC - receive DIS-ASSOC request from 00:0c:43:26:61:25

*RT2800*<7>AUTH_RSP-Rcv AUTH seq#1,Alg=0,Status=0 from 00:0c:43:26:61:25 to IF(ra0)

*RT2800*<7>MacTableInsertEntry -IF(ra0) allocate entry #1, Total= 1

*RT2800*<7>AUTH_RSP - IF(0) Send AUTH response (SUCCESS)...

*RT2800*<7>ASSOC - receive ASSOC request from 00:0c:43:26:61:25

*RT2800*<7>AssignAid (AID=1)

*RT2800*<7>BuildAssoc-IF(0):AuthMode=7,WepStatus=6,GroupWepStatus=6,WpaState=7,AGGRE=1,PiggyBack=1,
APSD=0

*RT2800*<7>LOG#8 00:0c:43:26:61:25 successfully associated

*RT2800*<7>Init entry init retry timer

*RT2800*<7>assign AID=1 to 00:0c:43:26:61:25,MaxSupportedRate=54Mbps,CurrTxRate=54Mbps

*RT2800*<7>RSNIE_Len=0x14,pEntry->RSNIE_Len=20,pEntry->PrivacyFilter=1

*RT2800*<7>ASSOC - Send ASSOC response (Status=0) from IF(ra0)...


WpaEAPOLStartAction ====>>

==>WPAStart4WayHS

STA from 00:0c:43:26:61:25

PMK = 99:61:62:c4-86:a8:8d:bf

pEntry->AuthMode == Ndis802_11AuthModeWPA2/WPA2PSK

WPA - RTMPToWirelessSta ====>> to IF(ra0)

<== WPAStart4WayHS:pEntry->WpaState=8, FrameLen=113
```

```
Receive EAPOL-Key frame, TYPE = 3, Length =0

WPAMsgTypeSubst (EAPType=3)

WpaEAPOLKeyAction ===>

PeerPairMsg2Action ===>

PTK-20 75 9f 5c 42 ac 7 cd

ANonce1-15 5c 19 72 8e 78 74 3

ANonce2-5a 7f c2 ef 86 c8 ee 6c

MIC VALID in Msg 2 of 4-way handshake!!

RSN_IE VALID in Msg 2 of 4-way handshake!!

WPA2 Group Key ID  = 1

G_Key :c 37 cf 69 cd 7c 85 49

   83 f9 e2 2c ad a8 cc e7

TX Mic:f0 7 d2 b9 62 9a bd 3e

RX Mic:e9 b5 c0 a2 1 f9 d6 17

*RT2800*<7>AsicAddSharedKeyEntry(BssIndex=0): AES key #1

*RT2800*<7>     Key =0c:37:cf:69:cd:7c:85:49:83:f9:e2:2c:ad:a8:cc:e7:

*RT2800*<7>     Rx MIC Key = e9:b5:c0:a2:01:f9:d6:17:

*RT2800*<7>     Tx MIC Key = f0:07:d2:b9:62:9a:bd:3e:

RTMPToWirelessSta : ETHTYPE = 88 8e FrameLen = 169!

WPA - RTMPToWirelessSta ====>> to IF(ra0)

Send Msg3 and setup timeout timer

Receive EAPOL-Key frame, TYPE = 3, Length =0

WPAMsgTypeSubst (EAPType=3)

WpaEAPOLKeyAction ===>

Wpa2PeerPairMsg4Action ===> from MAC:00:0c:43:26:61:25

Replay Counter VALID in Msg 4 of 4-way handshake!

MIC Valid in Msg 4 of 4-way handshake!!

*RT2800*<7>AsicAddPairwiseKeyEntry: #1 Alg=AES mac=00:0c:43:26:61:25 key=df-53-f5-..

===> AP SETKEYS DONE (ra0) - WPA2, AuthMode=7, WepStatus=6
```

18.8 SNMP MIBs

18.8.1 RT61AP Supported v.s. IEEE802dot11-MIB

IEEE802dot11-MIB	Access	Support	OID	RT61AP.dat
ieee802dot11				

dot11smt		-		
dot11StationConfigTable	not-accessible	-		
dot11StationConfigEntry	not-accessible	-		
dot11StationID	read-write	Y	OID_802_3_CURRENT_ADDRESS	N
dot11MediumOccupancyLimit	read-write	N		N
dot11CFPollable	read-only	N		N
dot11CFPPeriod	read-write	N		N
dot11CFPMaxDuration	read-write	N		N
dot11AuthenticationResponseTimeOut	read-write	N		N
dot11PrivacyOptionImplemented	read-only	Y	RT_OID_802_11_PRIVACYOPTIONIMPLEMENTED	N
dot11PowerManagementMode	read-write	Y	RT_OID_802_11_POWERMANAGEMENTMODE	N
dot11DesiredSSID	read-write	N		N
dot11DesiredBSSType	read-write	N		N
dot11OperationalRateSet	read-write	N		N
dot11BeaconPeriod	read-write	N		N
dot11DTIMPeriod	read-write	N		N
dot11AssociationResponseTimeOut	read-write	N		N
dot11DisassociateReason	read-only	N		N
dot11DisassociateStation	read-only	N		N
dot11DeauthenticateReason	read-only	N		N
dot11DeauthenticateStation	read-only	N		N
dot11AuthenticateFailStatus	read-only	N		N
dot11AuthenticateFailStation	read-only	N		N
dot11AuthenticationAlgorithmsTable	not-accessible	-		-
dot11AuthenticationAlgorithmsEntry	not-accessible	-		-
dot11AuthenticationAlgorithmsIndex	not-accessible	Y		N
dot11AuthenticationAlgorithm	read-only	Y		N
dot11AuthenticationAlgorithmsEnable	read-write	Y		N
dot11WEPDefaultKeysTable	not-accessible	-		-
dot11WEPDefaultKeysEntry	not-accessible	-		-
dot11WEPDefaultKeyIndex	not-accessible	Y		N
dot11WEPDefaultKeyValue	read-write	Y	OID_802_11_WEPDEFAULTKEYVALUE	Y
dot11WEPKeyMappingsTable	not-accessible	-		-
dot11WEPKeyMappingsEntry	not-accessible	-		-
dot11WEPKeyMappingIndex	not-accessible	N		N
dot11WEPKeyMappingAddress	read-create	N		N
dot11WEPKeyMappingWEPOn	read-create	N		N
dot11WEPKeyMappingValue	read-create	N		N
dot11WEPKeyMappingStatus	read-create	N		N
dot11PrivacyTable	not-accessible	-		
dot11PrivacyEntry	not-accessi	-		

	ble			
dot11PrivacyInvoked	read-write	Y		N
dot11WEPDefaultKeyID	read-write	Y	OID_802_11_WEPDEFAULTKEYID	Y
dot11WEPKeyMappingLength	read-write	Y	RT_OID_802_11_WEPKEYMAPPINGLEN GTH	N
dot11ExcludeUnencrypted	read-write	N		N
dot11WEPICVErrorCount	read-only	N		N
dot11WEPExcludedCount	read-only	N		N
dot11SMTnotification	-	-		
dot11Disassociate	-	N		N
dot11Deauthenticate	-	N		N
dot11AuthenticateFail	-	N		N
dot11mac				
dot11OperationTable	not-accessi ble	-		
dot11OperationEntry	not-accessi ble	-		
dot11MACAddress	read-only	Y	RT_OID_802_11_MAC_ADDRESS	N
dot11RTSThreshold	read-write	Y	OID_802_11_RTS_THRESHOLD	Y
dot11ShortRetryLimit	read-write	Y	OID_802_11_SHORTRETRYLIMIT	N
dot11LongRetryLimit	read-write	Y	OID_802_11_LONGRETRYLIMIT	N
dot11FragmentationThreshold	read-write	Y	OID_802_11_FRAGMENTATION_THRES HOLD	Y
dot11MaxTransmitMSDULifetim e	read-write	N		N
dot11MaxReceiveLifetime	read-write	N		N
dot11ManufacturerID	read-only	Y	RT_OID_802_11_MANUFACTUREID	N
dot11ProductID	read-only	Y	RT_OID_802_11_PRODUCTID	N
dot11CountersTable	not-accessi ble	-		
dot11CountersEntry	not-accessi ble	-		
dot11TransmittedFragmentCoun t	read-only	Y	OID_802_11_STATISTICS	N
dot11MulticastTransmittedFram eCount	read-only	Y	OID_802_11_STATISTICS	N
dot11FailedCount	read-only	Y	OID_802_11_STATISTICS	N
dot11RetryCount	read-only	Y	OID_802_11_STATISTICS	N
dot11MultipleRetryCount	read-only	Y	OID_802_11_STATISTICS	N
dot11FrameDuplicateCount	read-only	Y	OID_802_11_STATISTICS	N
dot11RTSSuccessCount	read-only	Y	OID_802_11_STATISTICS	N
dot11RTSFailureCount	read-only	Y	OID_802_11_STATISTICS	N
dot11ACKFailureCount	read-only	Y	OID_802_11_STATISTICS	N
dot11ReceivedFragmentCount	read-only	Y	OID_802_11_STATISTICS	N
dot11MulticastReceivedFrameC ount	read-only	Y	OID_802_11_STATISTICS	N
dot11FCSErrorCount	read-only	Y	OID_802_11_STATISTICS	N
dot11TransmittedFrameCount	read-only	N		N
dot11WEPUndecryptableCount	read-only	N		N
dot11GroupAddressesTable	not-accessi ble	-		-
dot11GroupAddressesEntry	not-accessi ble	-		-
dot11GroupAddressesIndex	not-accessi ble	N		N
dot11Address	read-create	N		N
dot11GroupAddressesStatus	read-create	N		N
dot11res				

dot11resAttribute				
dot11ResourceTypeIDName	read-only	-		
dot11ResourceInfoTable	not-accessible	-		
dot11ResourceInfoEntry	not-accessible	-		
dot11manufacturerOUI	read-only	Y	RT_OID_802_11_MANUFACTUREROUI	N
dot11manufacturerName	read-only	Y	RT_OID_802_11_MANUFACTURERNAME	N
dot11manufacturerProductName	read-only	Y	RT_OID_DEVICE_NAME	N
dot11manufacturerProductVersion	read-only	Y	RT_OID_VERSION_INFO	N
dot11phy				
dot11PhyOperationTable	not-accessible	-		
dot11PhyOperationEntry	not-accessible	-		
dot11PHYType	read-only	Y	RT_OID_802_11_PHY_MODE	N
dot11CurrentRegDomain	read-write	Y		Y
dot11TempType	read-only	N		N
dot11PhyAntennaTable	not-accessible	-		
dot11PhyAntennaEntry	not-accessible	-		
dot11CurrentTxAntenna	read-write	Y	OID_802_11_TX_ANTENNA_SELECTED	N
dot11DiversitySupport	read-only	Y	OID_802_11_RX_ANTENNA_SELECTED	N
dot11CurrentRxAntenna	read-write	Y	OID_802_11_RX_ANTENNA_SELECTED	N
dot11PhyTxPowerTable	not-accessible	-		
dot11PhyTxPowerEntry	not-accessible	-		
dot11NumberSupportedPowerLevels	read-only	N		N
dot11TxPowerLevel1	read-only	N		N
dot11TxPowerLevel2	read-only	N		N
dot11TxPowerLevel3	read-only	N		N
dot11TxPowerLevel4	read-only	N		N
dot11TxPowerLevel5	read-only	N		N
dot11TxPowerLevel6	read-only	N		N
dot11TxPowerLevel7	read-only	N		N
dot11TxPowerLevel8	read-only	N		N
dot11CurrentTxPowerLevel	read-write	N		N
dot11PhyFHSSTable	not-accessible	-		
dot11PhyFHSSEntry	not-accessible	-		
dot11HopTime	read-only	N		N
dot11CurrentChannelNumber	read-write	N		N
dot11MaxDwellTime	read-only	N		N
dot11CurrentDwellTime	read-write	N		N
dot11CurrentSet	read-write	N		N
dot11CurrentPattern	read-write	N		N
dot11CurrentIndex	read-write	N		N
dot11PhyDSSSTable	not-accessible	-		
dot11PhyDSSSEntry	not-accessible	-		

dot11CurrentChannel	read-write	Y	OID_802_11_CURRENTCHANNEL	Y
dot11CCAModeSupported	read-only	N		N
dot11CurrentCCAMode	read-write	N		N
dot11EDThreshold	read-write	N		N
dot11PhyIRTable	not-accessible	-		
dot11PhyIREntry	not-accessible	-		
dot11CCAWatchdogTimerMax	read-write	N		N
dot11CCAWatchdogCountMax	read-write	N		N
dot11CCAWatchdogTimerMin	read-write	N		N
dot11CCAWatchdogCountMin	read-write	N		N
dot11RegDomainsSupportedTable	not-accessible	-		
dot11RegDomainsSupportEntry	not-accessible	-		
dot11RegDomainsSupportIndex	not-accessible	Y		N
dot11RegDomainsSupportValue	read-only	Y		N
dot11AntennasListTable	not-accessible	-		
dot11AntennasListEntry	not-accessible	-		
dot11AntennaListIndex	not-accessible	Y		N
dot11SupportedTxAntenna	read-write	Y	OID_802_11_TX_ANTENNA_SELECTED	N
dot11SupportedRxAntenna	read-write	Y	OID_802_11_RX_ANTENNA_SELECTED	N
dot11DiversitySelectionRx	read-write	Y	OID_802_11_RX_ANTENNA_SELECTED	N
dot11SupportedDataRatesTxTable	not-accessible	-		
dot11SupportedDataRatesTxEntry	not-accessible	-		
dot11SupportedDataRatesTxIndex	not-accessible	Y		N
dot11SupportedDataRatesTxValue	read-only	Y	OID_802_11_DESIRED_RATES	N
dot11SupportedDataRatesRxTable	not-accessible	-		
dot11SupportedDataRatesRxEntry	not-accessible	-		
dot11SupportedDataRatesRxIndex	not-accessible	Y	OID_802_11_DESIRED_RATES	
dot11SupportedDataRatesRxValue	read-only	Y		
dot11PhyOFDMTable	not-accessible	-		
dot11PhyOFDMEntry	not-accessible	-		
dot11CurrentFrequency	read-write	N	OID_802_11_CURRENTCHANNEL	Y
dot11TIThreshold	read-write	N		N
dot11FrequencyBandsSupported	read-only	N		N

18.8.2 RALINK OID for SNMP MIB

RALINK OID for SNMP		
Value	Name	Structure
0x010B	OID_802_11_NUMBER_OF_ANTENNAS	USHORT numant;
0x010C	OID_802_11_RX_ANTENNA_SELECTED	USHORT whichant;

0x010D	OID_802_11_TX_ANTENNA_SELECTED	USHORT whichant;
0x050C	RT_OID_802_11_PHY_MODE	ULONG linfo;
0x050E	OID_802_11_DESIRED_RATES	typedef UCHAR NDIS_802_11_RATES[NDIS_802_11_LENGTH_RATES]; #define NDIS_802_11_LENGTH_RATES 8
0x0514	OID_802_11_RTS_THRESHOLD	ULONG linfo;
0x0515	OID_802_11_FRAGMENTATION_THRESHOLD	ULONG linfo;
0x0607	RT_OID_DEVICE_NAME	char name[128];

0x0608	RT_OID_VERSION_INFO	typedef struct PACKED _RT_VERSION_INFO{
		UCHAR DriverVersionW;
		UCHAR DriverVersionX;
		UCHAR DriverVersionY;
		UCHAR DriverVersionZ;
		UINT DriverBuildYear;
		UINT DriverBuildMonth;
		UINT DriverBuildDay;
		} RT_VERSION_INFO, *PRT_VERSION_INFO;

0x060A	OID_802_3_CURRENT_ADDRESS	char addr[128];

0x060E	OID_802_11_STATISTICS	typedef struct _NDIS_802_11_STATISTICS
		{
		ULONG Length; // Length of structure
		ULONG TransmittedFragmentCount;
		ULONG MulticastTransmittedFrameCount;
		ULONG FailedCount;
		ULONG RetryCount;
		ULONG MultipleRetryCount;
		ULONG RTSSuccessCount;
		ULONG RTSFailureCount;
		ULONG ACKFailureCount;
		ULONG FrameDuplicateCount;
		ULONG ReceivedFragmentCount;
		ULONG MulticastReceivedFrameCount;
		ULONG FCSErrorCount;
		} NDIS_802_11_STATISTICS, PNDIS_802_11_STATISTICS;

0x0700	RT_OID_802_11_MANUFACTUREROUI	char oui[128];
0x0701	RT_OID_802_11_MANUFACTURERNAME	char name[128];
0x0702	RT_OID_802_11_RESOURCETYPEIDNAME	char name[128];
0x0703	RT_OID_802_11_PRIVACYOPTIONIMPLEMENTED	ULONG linfo;
0x0704	RT_OID_802_11_POWERMANAGEMENTMODE	ULONG linfo;

0x0705	OID_802_11_WEPDEFAULTKEYVALUE	typedef struct _DefaultKeyIdxValue
		{
		UCHAR KeyIdx;

		UCHAR Value[16];
		}DefaultKeyIdxValue;

0x0706	OID_802_11_WEPDEFAULTKEYID	UCHAR keyid;
0x0707	RT_OID_802_11_WEPKEYMAPPINGLENGTH	UCHAR len;
0x0708	OID_802_11_SHORTRETRYLIMIT	ULONG linfo;
0x0709	OID_802_11_LONGRETRYLIMIT	ULONG linfo;
0x0710	RT_OID_802_11_PRODUCTID	char id[128];
0x0711	RT_OID_802_11_MANUFACTUREID	char id[128];
0x0712	OID_802_11_CURRENTCHANNEL	UCHAR channel
0x0713	RT_OID_802_11_MAC_ADDRESS	char macaddress[128]

19 Q&A

1. Why WPAPSK can not work?

   Ans:

   i. Please make sure the parameter "DefaultKeyID" is set to 2 in configuration file.

2. How to switch driver to operate in A band?

   Ans:

   i. Make sure RFIC support A band.

   ii. Check parameter "WirelessMode" is set to support A band.

   iii. Channel set to 36, 40…..

3. When I set channel as 1, but it will appear in channel 3. Why?

   Ans:

   i. Make sure the channel is match with CountryRegion or CountryRegionABand.

4. How can I know the version of package?

   Ans:

   i. can see the definition of DRIVER_VERSION in rt_config.h.

   ii. use command "iwpriv ra0 set DriverVersion=0", it will export to debug console.

5. Linux SoftAP Driver does not support antenna diversity.

   If the setting in EEPROM turns on antenna diversity, you can set "TxAntenna" in config file as 1(Antenna A) or 2(Antenna B) to fix antenna.

6. FixedTxMode =[1, 2] denotes setting Tx mode to [CCK, OFDM] respectively.

Applied with HT_MCS, FixedTxMode can be used to fix Tx rate in legacy mode manually:

1) FixedTxMode = 1 , HT_MCS = 0 ~ 11 set to the CCK Tx rate, other HT_MCS values will be taken the same as max. CCK rate, ie: (MCS=11)
2) FixedTxMode = 2, HT_MCS = 0 ~ 7 set to the OFDM Tx rate, other HT_MCS values will be taken the same as max. OFDM rate, ie: (MCS=7)
3) Other values of FixedTxMode will prevent this parameter from working. (not used)
4) Note that this parameter will override the setting of HT_OpMode if HT_MCS != 33 (AUTO mode), ie:

   If HT_OpMode and FixedTxMode is set at the same time, HT_MCS will be taken as legacy rate, instead of HT:

   (1) HT_OpMode = 1
   FixedTxMode = 1
   HT_MCS = 11
   Tx rate will be 11 Mbps (CCK, MCS=11), instead of 52 Mbps (HT, MCS=11)

   (2) HT_OpMode = 0

FixedTxMode = 2
HT_MCS = 7
Tx rate will be 54 Mbps (OFDM, MCS=7), instead of 65 Mbps (HT, MCS=7)

(3) HT_OpMode = 0
FixedTxMode = 0
HT_MCS = 7
Tx rate will be 65 Mbps (HT) , because the FixedTxMode is not used. (invalid value)

(4) HT_OpMode = 1
FixedTxMode = 2
HT_MCS = 33
Tx rate will be set by HT - Auto Switch, the FixedTxMode doesn't work in AUTO mode.

7. New format of the profile in RT2860AP

For dissection issues about the delimiter ';' in MBSS support, the content of RT2860AP.dat is modified to below format:

1) RT2860AP.dat

```
#The word of "Default" must not be removed
Default
CountryRegion=5
CountryRegionABand=7
CountryCode=TW
BssidNum=1
SSID1=RT2860AP
SSID2=
SSID3=
SSID4=
……
WPAPSK1=
WPAPSK2=
WPAPSK3=
WPAPSK4=
DefaultKeyID=1
Key1Type=0
Key1Str1=
Key1Str2=
Key1Str3=
Key1Str4=
Key2Type=0
Key2Str1=
Key2Str2=
Key2Str3=
Key2Str4=
Key3Type=0
Key3Str1=
Key3Str2=
Key3Str3=
Key3Str4=
Key4Type=0
Key4Str1=
Key4Str2=
Key4Str3=
Key4Str4=
……
HT_GI=1
HT_STBC=1
HT_MCS=33
```

2) Contains Any Delimiter:

If your individual SSID name, WPAPSK passphrase, or KeyStr contains any delimiter(i.e., semicolon ';'), you MUST use the new-added fields in RT2860AP.dat. For e.g.

```
……

BssidNum=4

SSID1=RT2860;AP1

SSID2=RT2860;AP2

SSID3=RT2860;AP3

SSID4=RT2860;AP4

……

AuthMode=OPEN;SHARED;WPAPSK;WPAPSK2

EncrypType=WEP;WEP;TKIP;AES

……

WPAPSK1=

WPAPSK2=

WPAPSK3=12;34;56

WPAPSK4=W;X;Y;Z;

DefaultKeyID=1;2

Key1Type=1;0;0;0

#Key1 of BSS0(WEP128)

Key1Str1=RalinkSuccess

#Key1 of BSS1

Key1Str2=

#Key1 of BSS2

Key1Str3=

#Key1 of BSS3

Key1Str4=
Key2Type=0;1;0;0

#Key2 of BSS0

Key2Str1=

#Key2 of BSS1(WEP64)

Key2Str2=f;g;h

#Key2 of BSS2

Key2Str3=

#Key2 of BSS3

Key2Str4=

Key3Type=0

#Key3 of BSS0

Key3Str1=

#Key3 of BSS1

Key3Str2=

#Key3 of BSS2

Key3Str3=

#Key3 of BSS3

Key3Str4=

Key4Type=0

#Key4 of BSS0

Key4Str1=

#Key4 of BSS1

Key4Str2=

#Key4 of BSS2

Key4Str3=

#Key4 of BSS3

Key4Str4=
```

3) Contains No Delimiter:

If no delimiter (semicolon ';') exists in the strings of individual SSID, WPAPSK, or KeyStr, you could use both the legacy format or the new one.

For example illustrating usage of KeyStr:

```
---Legacy format---

DefaultKeyID=1;1;1

Key1Type=1;1;1

Key1Str= abcde;fghij;klmno

Key2Type=


---New format---

DefaultKeyID=1;1;1

Key1Type=1;1;1

Key1Str1=abcde

Key1Str2=fghij

Key1Str3=klmno

Key1Str4=
```

4) Feel free to use ';' or not in SSID, WPAPSK, and KeyStr if your BssidNum=1.

Note:

(1) Please make sure your WPAPSK passphrase length or each KeyStr length is legal !
(2) When the old-format fields and the new-format fields coexist in the profile, the new one will take effect, not the old one, no matter the new fields have values assigned to them or not. For example illustrating usage of SSID :

```
BssidNum=4

SSID=Intel;Broadcom;Atheros;Marvell

SSID1=Ralink_no1!

SSID2=

SSID3=

SSID4=

……
```

Your SSID name of BSS0 will be Ralink_no1!.

8. 11n Bit Rate Derivation

1. The BitRate of 11n need below information on MAC driver and the real rates will be triggerred by PHY layer depends on below three factors.

   a. MCS

   b. BW

c. GI

2. Bandwidth:
Data subcarriers on different bandwidth, 20MHz and 40MHz.

a. N_{SD}: Number of data subcarriers.

N_{SD}[40Mhz] = 108

N_{SD}[20Mhz] = 52

N_{SD}[40Mhz]/N_{SD}[20MHz] = 108/52

= 2.0769230769230769230769230769231

E.g.

MCS=15, GI=800ns, BW=20MHz, DataRate = 130Mbps

MCS=15, GI=800ns, BW=40MHz, DataRate = 130 * [$N_{sd(40Mhz)}$ / $N_{sd(20Mhz)}$]

= 130 * [108 / 52]

= 270Mbps

b. Please refer to “IEEE P802.11n/D2.04, June 2007” on page 314 for below table.

3. Guard Interval.

a. Definition:
T_{sym}: 4us , Symbol Interval
T_{syms}: 3.6us , Symbol interval of Short GI.

b. Ratio of symbol interval on GI, refer to below EWC PHY Sepc.
Tsym / Tsyms = 4usec / 3.6usec

= 10/9

E.g.

MCS=15, 40MHz Bandwidth, and 400ns Short Guard Interval.
270.0 * (10/9) = 300.0 for Short GI.

c. Reference:

1) IEEE 802.11n draft 2.04, page 316 and

Table 211—MCS parameters for optional 40 MHz, N SS = 2, NES = 1, EQM (#665)

MCS Index	Modulation	R	NBPSCS(iSS)	NSD	NSP	NCBPS	NDBPS	Data rate (Mb/s)	
								800 ns GI	400 ns GI
8	BPSK	1/2	1	108	6	216	108	27.0	30.0
9	QPSK	1/2	2	108	6	432	216	54.0	60.0
10	QPSK	3/4	2	108	6	432	324	81.0	90.0
11	16-QAM	1/2	4	108	6	864	432	108.0	120.0
12	16-QAM	3/4	4	108	6	864	648	162.0	180.0
13	64-QAM	2/3	6	108	6	1296	864	216.0	240.0
14	64-QAM	3/4	6	108	6	1296	972	243.0	270.0
15	64-QAM	5/6	6	108	6	1296	1080	270.0	300.0

2) EWC PHY spec. page 13.

3) EWC PHY spec. page 13.