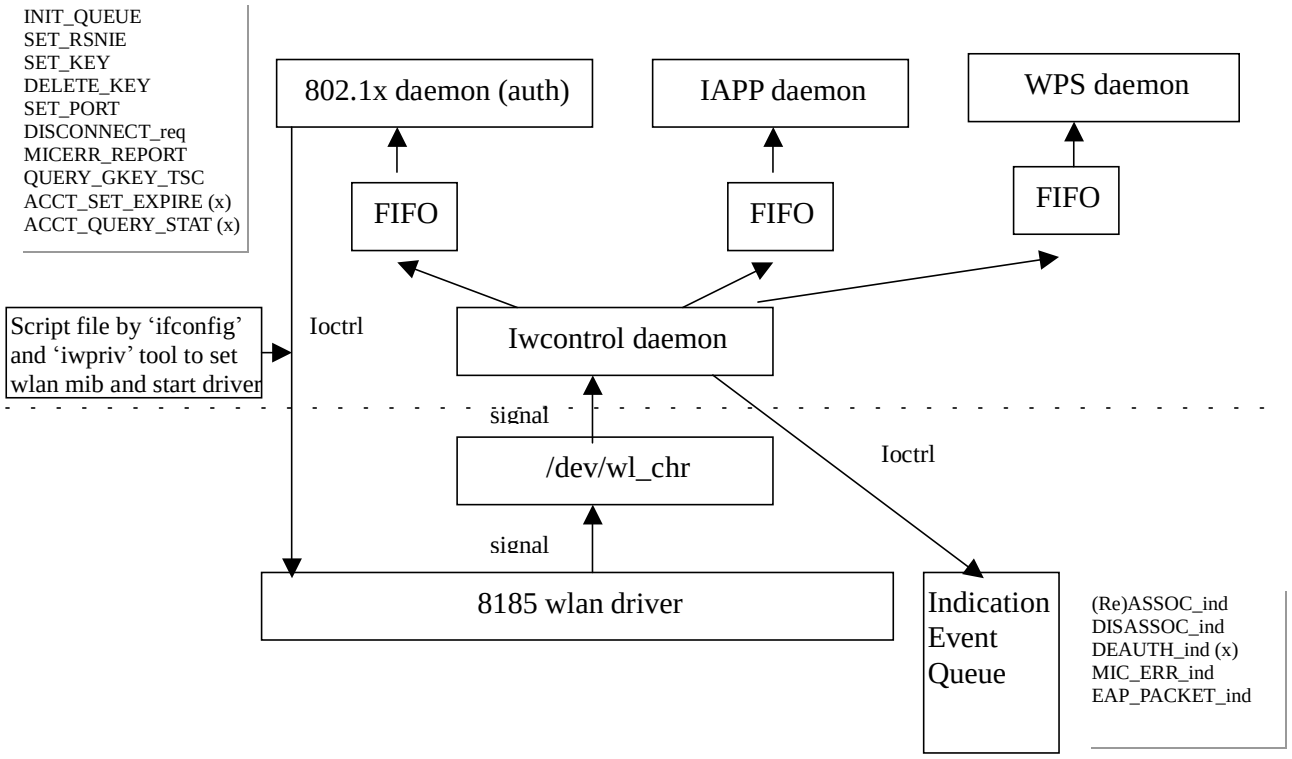
Revision History

Revision	Release date	comment
1.2	5/17/2004	First issue
1.3	6/04/2004	Add ioctl for WDS
1.4	8/09/2004	Add client mode support
1.5	10/22/2004	Add WPA2 support
1.6	1/03/2005	Add EEPROM access interface and MP mode
1.7	4/07/2005	Add ioctl and revise “encryption” entry of table of wpa parameters Modify MP functions
1.8	7/08/2005	Add ioctl for new feature
1.9	9/28/2005	Add ioctl for new feature
1.10	1/20/2006	Add ioctl for new feature
1.11	6/19/2006	Add ioctl for new feature
1.12	10/31/2006	Add ioctl for new feature
1.13	11/6/2006	Add ioctl for new feature Revise features
1.14	11/8/2006	Revise ioctl “qos_enable” default setting
1.15	11/9/2006	Add ioctl for new feature
1.16	2/9/2007	Add ioctl for new feature
1.17	3/16/2007	Add ioctl for new feature

Features

- 802.11 a/b/g compatible
- AP mode and client mode support
- Security support 64/128 bits WEP, WPA, and WPA2 (TKIP and AES-CCMP)
- Auto rate adaptive
- Wireless MAC address filter
- Broadcast SSID control
- IAPP (802.11f) support
- Auto channel selection
- Driver based MP functions
- Support for 8255, 8255b (11a/g) RF module
- WDS function support
- Universal repeater mode support
- WMM supported for AP mode
- Support for 8185B ASIC
- WPS function support

System Architecture



WLAN Driver Configuration, IOCTL and PROC

Set mac address:
"ifconfig wlan0 hw ether xxxxxxxxxxxx"

Set wlan MIB:
"iwpriv wlan0 set_mib name=value1[,value2,value3…]"
Note: value can be a single field or multiple fields separated by ',' without any space between fields. Detail parameter may be referred the following table.

Up driver:
"ifconfig wlan0 up"

Close driver:
"ifconfig wlan0 down"

MIB command table:

Name	Meaning	Value	Default	Comment
RFChipID	RF module type	7 – Zebra, 8 – OMC8255, 9 – OMC8255B	7	
channel	Operation frequency used	0 for auto channel, 1-14 for 11b/11g, 34-216 for 11a		
ch_low	The lowest channel to scan and use	1-14 for 11b/11g, 34-216 for 11a		
ch_hi	The highest channel to scan and use	1-14 for 11b/11g, 34-216 for 11a		
TxPowerCCK	CCK Tx power level for 14 channels (28 hex digits)	RF module dependent		
TxPowerOFDM	OFDM Tx power level for 216 channels (432 hex digits)	RF module dependent		Ch163 ~ Ch181 mapped from pwr[14] ~ pwr[32]. Ch182 ~ Ch216 mapped from pwr[64] ~ pwr[98].
DefaultAnt	Select which antenna is used when diversity is off	0 – antenna A, 1 – antenna B		
preamble	CCK preamble type	0 – long preamble, 1 – short preamble		
DIG_enable	Flag to enable Dynamic Initial Gain	0 – disable, 1 – enable		
initialGain	Initial gain value	1-7	4	
HighPowerChk	Flag to enable high power check	0 – disable, 1 – enable	1	
AntDvrsty	Flag to enable dynamic antenna diversity for dead angle	0 – disable, 1 – enable		
txChargePump	Charge pump value for Tx	0-7	6	
rxChargePump	Charge pump value for Rx	0-7	0	
DRSA_disable	Dynamic RF sensitivity adjustment	0 – auto, 1 – disable		Only effective with Zebra RF
disable_ch14_ofdm	Disable OFDM sending and receiving in channel 14	0 – enable, 1 – disable		
ExtAntDvry	External antenna diversity	0 – disable, 1 – enable		
LNA_enable	Support LNA	0 – disable, 1 – enable		
EhTxPower	Enhanced Tx Power	0 – disable, 1 – enable	1	Only effective with OMC RF
ssid	SSID	"string_value", SSID with 32 characters in max		
defssid	If don't give SSID in Ad-hoc client mode and no IBSS available, it will	"string_value", SSID with 32 chars in max	"defaultSSID"	

	start an IBSS with SSID given here.			
bssid2join	Besides SSID, designate target BSSID to join	xxxxxxxxxxxx (12 digits mac address)		
bcnint	Beacon interval in ms	20-1024	100	
dtimperiod	DTIM period	1-255	1	Suggest to set 1 because patent issue
swcrypto	S/w encryption enabled/disabled	0 – disable, 1 – enable		
aclmode	Access control mode	0 – disable, 1 – accept, 2 – deny		
aclnum	Set number of ACL	Suggest set '0' whenever driver is re-initialized		
acladdr	Set access control address	xxxxxxxxxxxx (12 digits mac address)		When acl is added, the aclnum will be increased automatically.
oprates	Operational rate	Bit0-bit11 for 1,2,5.5,11,6,9,12,18,24,,36,48,54M	0xfff	
basicrates	Basic rate	Bit0-bit11 for 1,2,5.5,11,6,9,12,18,24,,36,48,54M	0xf	
regdomain	Regulation domain	1-10 (FCC, IC, ETSI, SPAIN, FRANCE, MKK, ISREAL, MKK1, MKK2, MKK3)	1	
autorate	Auto rate adaptive	0 – disable, 1 – enable	1	
fixrate	Fixed Tx rate	Bit0-bit11 for 1,2,5.5,11,6,9,12,18,24,,36,48,54M		Refer when auto rate is disabled
disable_protection	Forcedly disable protection mode	0 – auto, 1 – disable protection		Normally when 11g is used, driver will auto detect if legacy (11b) device is existed. If yes, it will enable protection mode automatically.
disable_olbc	Forcedly OLBC detection	0 – auto, 1 – disable protection		Normally 11g AP should detect OLBC. If disabled, AP will enter protection mode only when legacy device associated.
deny_legacy	Deny the association from legacy STA	0 – disable, 1 – deny		If enabled in B+G mode, AP will deny the association from 11B STA.
fast_roaming	Client mode fast roaming	0 – disable, 1 – enable		
lowestMlcstRate	Use lowest basic rate to send multicast and broadcast	0 – disable, 1 – enable		
authtype	802.11 Authentication type	0 – open system, 1 – shared key, 2 – auto	2	
encmode	Encryption mode	0 – disabled, 1 – WEP64, 2 – TKIP, 4 – AES(CCMP), 5 – WEP128		
wepdkeyid	WEP default Tx key	0-3		
psk_enable	PSK mode	0 – disable, 1 – WPA, 2 – WPA2		
wpa_cipher	WPA PSK cipher suite	2 –TKIP, 8 – AES(CCMP)		
wpa2_cipher	WPA2 PSK cipher suite	2 –TKIP, 8 – AES(CCMP)		
passphrase	PSK key	32 characters or 64 hex digits		
gk_rekey	Group key update time	0 – disable, >1 – enable		Time unit is second
802_1x	Flag of using 802.1x	0 – disable, 1 – enable		When 802.1x is enabled, the Auth daemon must be invoked

<table>
<tr><td>default_port</td><td>Default state of 802.1x control port</td><td>0 – data packet is not allowed to pass through until 802.1x authentication is ok
1 – data packet is allowed pass through even 802.1x authentication is not ok</td><td></td><td>Refer when 802_1x is set to 1</td></tr>
<tr><td>wepkey1</td><td>WEP key1</td><td>10 hex digits for WEP64, 26 hex digits for WEP128</td><td></td><td></td></tr>
<tr><td>wepkey2</td><td>WEP key2</td><td>10 hex digits for WEP64, 26 hex digits for WEP128</td><td></td><td></td></tr>
<tr><td>wepkey3</td><td>WEP key3</td><td>10 hex digits for WEP64, 26 hex digits for WEP128</td><td></td><td></td></tr>
<tr><td>wepkey4</td><td>WEP key4</td><td>10 hex digits for WEP64, 26 hex digits for WEP128</td><td></td><td></td></tr>
<tr><td>opmode</td><td>Operation mode (AP or client)</td><td>16 – AP, 8 – Infrastructure client, 32 – Ad-hoc client</td><td>16</td><td></td></tr>
<tr><td>hiddenAP</td><td>Hidden AP enable/disable</td><td>0 – disabled, 1 – enabled</td><td></td><td></td></tr>
<tr><td>rtsthres</td><td>RTS threshold</td><td>0-2347</td><td>2347</td><td></td></tr>
<tr><td>fragthres</td><td>Fragment threshold</td><td>256-2346</td><td>2346</td><td></td></tr>
<tr><td>shortretry</td><td>Short retry limit</td><td>1-255</td><td>3</td><td></td></tr>
<tr><td>longretry</td><td>Long retry limit</td><td>1-255</td><td>3</td><td></td></tr>
<tr><td>expired_time</td><td>Client inactivity time in 10ms</td><td>>100</td><td>3000</td><td></td></tr>
<tr><td>led_type</td><td>WLAN LED type</td><td><table><tr><th></th><th>LED0</th><th>LED1</th></tr><tr><td>0</td><td>tx</td><td>rx</td></tr><tr><td>1</td><td>enable/tx/rx</td><td>n/a</td></tr><tr><td>2</td><td>link</td><td>tx/rx (d,m)</td></tr><tr><td>3</td><td>link/rx/tx (d,m)</td><td>n/a</td></tr><tr><td>4</td><td>link</td><td>tx/rx (d)</td></tr><tr><td>5</td><td>link/tx/rx (d)</td><td>n/a</td></tr><tr><td>6</td><td>enable</td><td>tx/rx (d)</td></tr><tr><td>7</td><td>enable/tx/rx (d)</td><td>n/a</td></tr><tr><td>8</td><td>11a tx/rx (d)</td><td>11g tx/rx (d)</td></tr></table>0-1 – hw control
2-8 – sw control
d – count data frames
m – count management frames</td><td></td><td></td></tr>
<tr><td>iapp_enable</td><td>IAPP enable/disable</td><td>0 – disable, 1 - enable</td><td></td><td></td></tr>
<tr><td>block_relay</td><td>Block packet relaying between associated clients</td><td>0 – relay, 1 – block relay and drop, 2 – block relay and indicate to bridge</td><td></td><td></td></tr>
<tr><td>deny_any</td><td>Deny the association SSID of “any” including upper and lower cases</td><td>0 – disable, 1 – enable</td><td></td><td></td></tr>
<tr><td>crc_log</td><td>Calculate CRC error packets</td><td>0 – disable, 1 – enable</td><td></td><td></td></tr>
<tr><td>wifi_specific</td><td>Do WiFi specific check</td><td>0 – disable, 1 – enable</td><td></td><td></td></tr>
<tr><td>disable_txsc</td><td>Tx shortcut enable/disable</td><td>0 – enable, 1 – enable</td><td></td><td></td></tr>
<tr><td>disable_rxsc</td><td>Rx shortcut enable/disable</td><td>0 – enable, 1 – enable</td><td></td><td></td></tr>
<tr><td>disable_brsc</td><td>Bridge shortcut enable/disable</td><td>0 – enable, 1 – enable</td><td></td><td></td></tr>
<tr><td>keep_rsnie</td><td>Don’t clean RSN IE while reinitialize the interface</td><td>0 – erase, 1 – keep</td><td></td><td></td></tr>
<tr><td>band</td><td>Band selection</td><td>1 – 11b, 2 – 11g, 3 – 11b+g, 4 – 11a</td><td>3</td><td></td></tr>
<tr><td>cts2self</td><td>Use cts2Self for protection mode</td><td>0 – no, 1 – yes</td><td>1</td><td></td></tr>
<tr><td>wds_enable</td><td>WDS enable/disable</td><td>0 – disable, 1 – enable</td><td></td><td></td></tr>
<tr><td>wds_pure</td><td>Flag to enable pure WDS mode that don’t broadcast beacon and don’t accept any station</td><td>0 – disable, 1 – enable</td><td></td><td></td></tr>
<tr><td>wds_priority</td><td>Give WDS packets higher priority</td><td>0 – disable, 1 – enable</td><td></td><td></td></tr>
<tr><td>wds_num</td><td>Set number of WDS</td><td>Suggest set ‘0’ whenever driver is</td><td></td><td></td></tr>
</table>

		re-initialized		
wds_add	Set mac address of WDS AP	xxxxxxxxxxxx (12 digits mac address). The max entry could be added is 8 in default configuration.		When mac address is added, the wds_num will be increased automatically.
wds_encrypt	WDS encryption mode	0 – disabled, 1 – WEP64, 2 – TKIP, 4 – AES (CCMP), 5 – WEP128		
wds_wepkey	WDS WEP default Tx key	0-3		
wds_passphrase	WDS PSK key	32 characters or 64 hex digits		
nat25_disable	Disable NAT2.5 transformation in client mode	0 – enable, 1 – disable		
macclone_enable	Enable MAC clone from the first incoming packet	0 – disable, 1 – enable		
dhcp_bcst_disable	Flag of adding broadcast flag into DHCP request	0 – enable, 1 – disable		
add_pppoe_tag	Add extra tag in PPPoE packets by NAT2.5	0 – disable, 1 – enable	1	When set to 0, NAT2.5 can only support one session buildup at the same time.
nat25sc_disable	NAT2.5 shortcut enable/disable	0 – enable, 1 – disable		
disable_DFS	Disable DFS function	0 – enable, 1 – disable		DFS function will work in W53 band in region domain of MKK and MKK3
show_hidden_bss	Show hidden BSS in site survey	0 – disable, 1 – enable		
turbo_mode	Support turbo mode in Realtek family	0 – auto, 1 – always, 2 – off		
ack_timeout	Set ACK timeout value	0-255		0 means using standard value. In unit of 4 us.
tx_priority	Support high priority Tx	0 – disable, 1 – enable		Send high priority DSCP packet by normal queue.
private_ie	Send and get private IE	At most 64 hex digits byte array		
qos_enable	Support WMM and QoS	0 – disable, 1 – enable		
wsc_enable	Support WiFi Protection Setup	Bit0 for client mode, Bit1 for AP mode		
pin	PIN setting for WPS	"string_value" with 8 characters in max		
debug_err	Flag of DEBUG_ERR() macro	Bit value defined in 8185ag_debug.h (in hex)	ffffffff	
debug_info	Flag of DEBUG_INFO() macro	Bit value defined in 8185ag_debug.h (in hex)	0	
debug_warn	Flag of DEBUG_WARN() macro	Bit value defined in 8185ag_debug.h (in hex)	0	
debug_trace	Flag of DEBUG_TRACE() macro	Bit value defined in 8185ag_debug.h (in hex)	0	

Note1: The default value of MIB will be '0' if it is not specified.

Read wlan register command:

"iwpriv wlan0 read_reg type,offset"

- *type* could be b - for byte, w – for word, dw – for double word
- *offset* indicates the register offset in hex

Write wlan register command:

"iwpriv wlan0 write_reg type,offset,value"

- *type* may be b - for byte, w – for word, dw – for double word
- *offset* indicates the register offset in hex
- *value* for write in hex

Read memory command:

"iwpriv wlan0 read_mem type,start,len"

- *type* may be b - for byte, w – for word, dw – for double word
- *start* indicates the memory start address in hex
- *len* is for read length in hex

Write memory command:

"iwpriv wlan0 write_mem type,start,len,value"

- *type* may be b - for byte, w – for word, dw – for double word
- *start* indicates the memory start address in hex
- *len* is for write length in hex
- *value* for write in hex

Read EEPROM command:

"iwpriv wlan0 read_eeprom name"

- *name* is the name of variable stored in EEPROM

Write EEPROM command:

"iwpriv wlan0 write_eeprom name=value[,value2,value3…]"

- *name* may be b - for byte, w – for word, dw – for double word
- *value* indicates the memory start address in hex

Name table:

Name	**Meaning**	**Value**	**Comment**
RFChipID	RF module type	9 – Zebra, 10 – OMC8255	
Mac	MAC address of the NIC	6 bytes MAC address	
TxPowerCCK	CCK Tx power level for 14 channels	14 bytes array	
TxPowerOFDM	OFDM Tx power level for 162 channels	162 bytes array	

Note1: Only TxPowerCCK and TxPowerOFDM can be written into EERPOM

Driver based MP function:

We supported Driver based MP functions controlled by "iwpriv" utility. Please refer to *"8185 Linux Driver MP.doc"* for detail explanation and usages.

Additional IOCTL commands (for web display):

id	**meaning**	**Input**	**output**	**comment**
0x8b30	Get station info	None	64 array of sta_info_2_web (note1)	
0x8b31	Get associated station number	None	1 word (2 bytes)	
0x8b32	Get version information	None	2 byte of version infomation	
0x8b33	Issue scan request	None	1 byte of result (-1:fail, 0: success)	
0x8b34	Get scan result and scanned BSS database	1 byte flag (get BSS database or not)	4 bytes of number of entries and array of bss_desc (note4) with flag set to 0	
0x8b35	Issue join request	bss_desc to join	1 byte of result (0: success, 1: scanning, 2: fail)	
0x8b36	Get join result	None	1 byte of result (note5)	

0x8b37	Get BSS info	None	bss_info_2_web structure (note2)	This is used typically in client mode
0x8b38	Get WDS info	None	8 array of wds_info (note3)	

Note1:

```
typedef struct _sta_info_2_web {
        unsigned short    aid;
        unsigned char     addr[6];
        unsigned long     tx_packets;
        unsigned long     rx_packets;
        unsigned long     expired_time;
        unsigned short    flags; // bit2 indicate whether this entry is valid, bit3 indicates if sta is in sleeping
        unsigned char     TxOperaRate; // current used tx rate in 500 k bps (e.g., 108 for 55M)
        unsigned char     rssi; // received signal strength indication
        unsigned long     link_time; // 1 sec unit
        unsigned long     tx_fail;
        unsigned long     tx_bytes;
        unsigned long     rx_bytes;
        unsigned char     resv[8];
} sta_info_2_web;
```

Note2:

```
typedef enum _wlan_mac_state {
      STATE_DISABLED=0, STATE_IDLE, STATE_SCANNING, STATE_STARTED, STATE_CONNECTED,
STATE_WAITFORKEY
} wlan_mac_state;

typedef struct _bss_info_2_web {
      unsigned char state;       // defined in wlan_mac_state
      unsigned char channel;
      unsigned char txRate;
      unsigned char bssid[6];
      unsigned char rssi, sq;
      unsigned char ssid[33];
} bss_info_2_web;
```

Note3:

```
typedef struct _wds_info {
        unsigned char     state;
        unsigned char     addr[6];
        unsigned long     tx_packets;
        unsigned long     rx_packets;
        unsigned long     tx_errors;
        unsigned char     TxOperaRate;
} wds_info;
```

Note4:

```
struct ibss_priv {
        unsigned short    atim_win;    };
struct bss_desc {
        unsigned char     bssid[6];
        unsigned char     ssid[32];
        unsigned char     *ssidptr;
        unsigned short    ssidlen;
        unsigned int      bsstype;
        unsigned short    beacon_prd;
        unsigned char     dtim_prd;
        unsigned long     t_stamp[2];
        struct ibss_priv  ibss_par;
        unsigned short    capability;
```

```
    unsigned char    channel;
    unsigned long    basicrate;
    unsigned long    supportrate;
    unsigned char    bdsa[6];
    unsigned char    rssi;
    unsigned char    sq;
    unsigned char    network;
};
```

Note5:
0xff: pending
2-4: success
others: fail

Create a character device "*wl_char"* in file system before starting any application daemons:
"mknod -m666 /mnt/dev/wl_chr c 13 0"

Files under '/proc/wlan0':
- ***cam_info*** – dump h/w encryption cam content
- ***mib_xxx*** – show mib info
- ***sta_info*** – show all associated station info
- ***sta_keyinfo*** – show the encryption keys of all associated station info
- ***txdescH, txdescN, txdescL*** – show tx descriptor contents for high, normal, and low queue
- ***buf_info*** – show the internal buffer pointers and counts
- ***desc_info*** – show tx and rx descriptor pointers, indexes, and register contents
- ***stats*** – show Tx, Rx, and beacon statistics

iwcontrol Daemon Configuration

Need start daemon when:
- 802.1x daemon is used
- IAPP daemon is used
- WPS daemon is used

Note: iwcontrol daemon should be started after 802.1x, IAPP, or WPS daemon is running

Start daemon:
"iwcontrol wlan_interface ...."
- *wlan_interface:* wlan interface, e.g., wlan0

Note:
1. *iwcontrol daemon will parse the pid files in "/var/run" and create FIFO files to do IPC with WPS, IAPP, and 1x daemon.*
2. *Multiple wireless interfaces can be supported in iwcontrol parameters.*

802.1x Daemon Configuration

Need start daemon when:
- WPA/WPA2 is used
- WEP + 802.1x (authentication with radius server)
- No encryption + 802.1x (authentication with radius server)

Start 802.1x daemon:

"auth wlan_interface lan_interface auth wpa_conf &"

- ***wlan_interface:*** wlan interface, e.g., wlan0
- *lan_interface:* lan interface, which connects to Radius server, e.g., br0
- *auth*: denote to act as authenticator
- *wpa_conf*: path of wpa config file, e.g., /var/wpa-wlan0.conf

Note:

1. *Multiple 802.1x daemons will be created for different wireless interfaces.*
2. *PID file "/var/run/auth-wlanx.pid" will be created for each 1x daemon*

Parameter format in wpa config file:

"keyword = value"

table of wpa parameters

keyword	**value**	**Comment**
encryption	0 – disable, 1 – WEP, 2 – WPA, 4 – WPA2 only, 6 – WPA2 mixed	
ssid	"string_value", 1-32 char	
enable1x	0/1 – disable/enable 1x Radius authentication	Refer when encryption is set to 0, 1
enableMacAuth	0/1 – disable/enable MAC authentication	
SupportNonWpaClient	0/1 – disable/enable none WPA client support when WPA is set	This feature is not supported now
wepKey	1 – WEP64, 2 – WEP128	Refer when encryption is set 1 (wep)
wepGroupKey	set "" as default	No use
authentication	1 – Radius, 2 – PSK (pre-shared key)	
unicastCipher	1 – TKIP, 2 – AES	
wpa2UnicastCipher	1 – TKIP, 2 – AES	
usePassphrase	0 – use psk value as key in raw data, 1 – use passphrase algorithm to convert psk value	
psk	"string_value", if usePassphrase=0 (raw data), it should be 64 hex digits. If usePassphrase=1, the string length should be >=8 and <=64.	
groupRekeyTime	Group key re-key time	No use
rsPort	UDP Port number of radius server	Normally 1812 is used
rsIP	IP address of radius server (e.g., 192.168.1.1)	
rsPassword	"string_value", password of radius server with 31 char in max	
rs2Port	UDP Port number of radius server set 2	Normally 1812 is used
rs2IP	IP address of radius server (e.g., 192.168.1.1) set 2	
rs2Password	"string_value", password of radius server with 31 char in max set 2	
rsMaxReq	Max retry number of request packet with radius server	Set 3 as default
rsAWhile	Timeout time (in second) of waiting rsp packet of radius server	Set 5 as default
accountRsEnabled	0/1 – disable/enable accounting radius server	
accountRsPort	UDP Port number of accounting radius server	
accountRsIP	IP address of accounting radius server	
accountRsPassword	"string_value", password of accounting radius server with 31 char in max	
accountRsUpdateEnabled	0/1 – disable/enable the feature of statistic update with accounting server	
accountRsUpdateTime	Update time in seconds	
accountMaxReq	Max retry number of request packet with accounting radius server	

accountAWhile	Timeout time (in second)of waiting rsp packet of accounting radius server	

IAPP Configuration

Start IAPP daemon:
"iapp lan_interface wlan_interface …&"
- ***lan_interface:*** interface name which IAPP daemon use to send IAPP packet (e.g., br0)
- ***wlan_interface:*** wlan interface, e.g., wlan0

Notes:
1. *IAPP can support multiple wireless interfaces.*
2. *PID file "/var/run/iapp.pid" will be created for iapp deamon.*

WPS Configuration

The driver has already supported WPS function, but it needs to cooperate with WPS daemon in user level. Please refer to *"Realtek_WPS_user_guide.doc"* for detail explanation and usages.

Limitation

- H/W encryption CAM size is 16
- TKIP MIC should be calculated by S/W
- Tx SKB buffer must have 8 bytes space in tail for TKIP MIC
- Support 64 wlan clients in current configuration
- Support 8 WDS number in current configuration