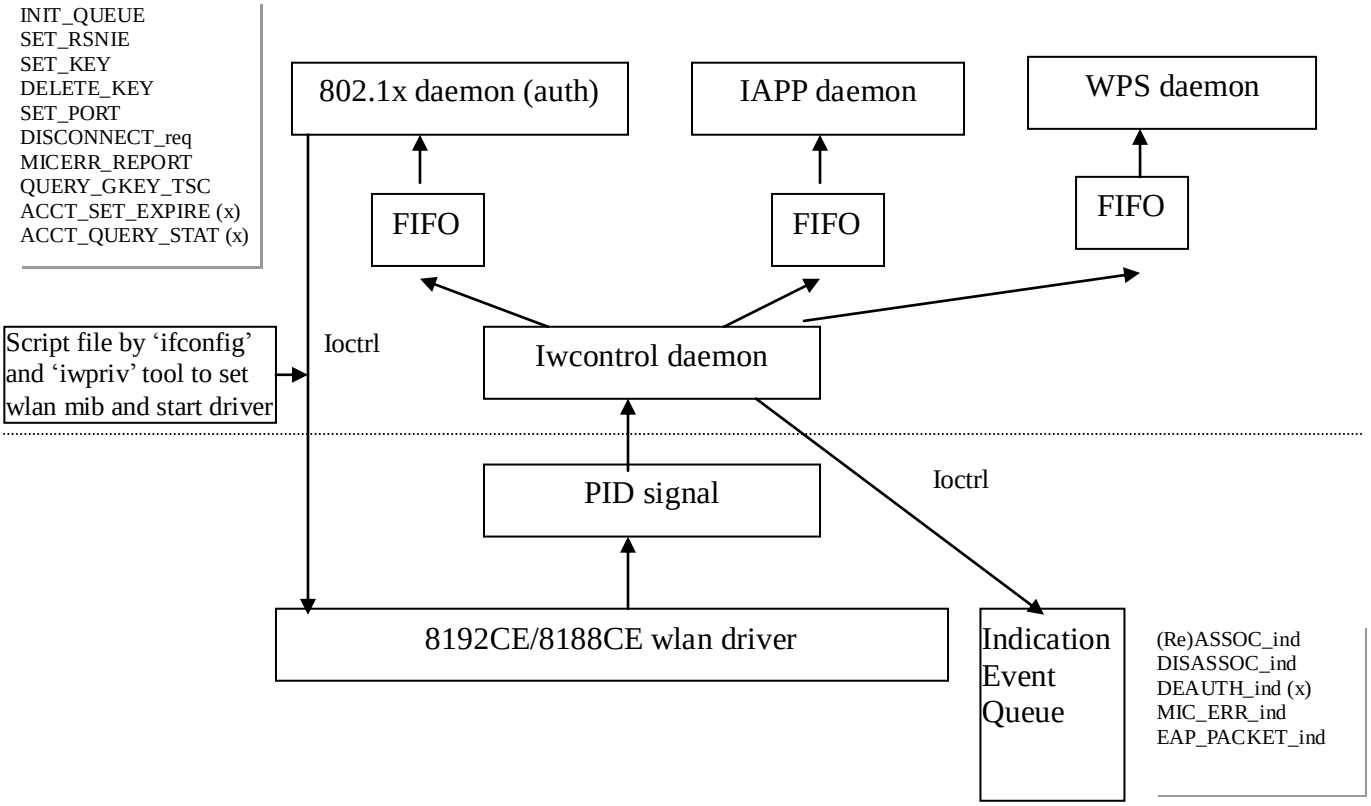
Revision History

Revision	Release date	comment
1.0	2009/11/17	First issue
1.1	2010/1/14	Add comments
1.2	2010/1/29	Add mib of WAPI
1.3	2010/2/24	Add new configuration API support

Features

- 802.11 b/g/n 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
- WDS function support
- Universal repeater mode support
- WMM supported for AP mode
- Support for 8192CE and 8188CE ASIC
- WPS function support
- WAPI 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 1: value can be a single field or multiple fields separated by ',' without any space between fields. Detail parameter may be referred the following table.
Note 2: if the value is the type of byte array, the format of value will be a string of ASCII of 0~f, which using 2 ASCII standing for one byte. For example, when set Tx power of CCK, it will be
"iwpriv wlan0 set_mib TxPowerCCK=08080909090a0a0a0a0b0b0b0c0c"

Up driver:
"ifconfig wlan0 up"

Close driver:
"ifconfig wlan0 down"

MIB command table:

Name	Meaning	Value	Default	Comment
channel	Operation frequency used	0 for auto channel, 1-14 for 11b/11g		
ch_low	The lowest channel to scan and use	1-14 for 11b/11g		
ch_hi	The highest channel to scan and use	1-14 for 11b/11g		
pwrlevelCCK_A	CCK Tx power level for 14 channels (28 hex digits) for path A	RF module dependent		Type of byte array
pwrlevelCCK_B	CCK Tx power level for 14 channels (28 hex digits) for path B	RF module dependent		Type of byte array
pwrlevelHT40_1S_A	40MHz mode HT OFDM 1 spatial stream Tx power level for 14 channels (28 hex digits) for path A	RF module dependent		Type of byte array
pwrlevelHT40_1S_B	40MHz mode HT OFDM 1 spatial stream Tx power level for 14 channels (28 hex digits) for path B	RF module dependent		Type of byte array
pwrdiffHT40_2S	40MHz mode HT OFDM 2 spatial stream Tx power difference between HT40_1S for 14 channels (28 hex digits). Bit[3:0] for path A. Bit[7:4] for path B.	RF module dependent		Type of byte array
pwrdiffHT20	20MHz mode HT OFDM Tx power difference between HT40_1S for 14 channels (28 hex digits). Bit[3:0] for path A. Bit[7:4] for path B.	RF module dependent		Type of byte array
pwrdiffOFDM	Legacy OFDM Tx power difference between HT40_1S for 14 channels (28 hex digits). Bit[3:0] for path A. Bit[7:4] for path B.	RF module dependent		Type of byte array
preamble	CCK preamble type	0 – long preamble, 1 – short preamble		
disable_ch14_ofdm	Disable OFDM sending and receiving in channel 14	0 – enable, 1 – disable		
xcap	Crystal Capacitor value	0 – 255		0 stands the value is not calibrated yet.
tssi1	Tx signal strength value of path A	0 – 255		0 stands the value is

				not calibrated yet.
tssi2	Tx signal strength value of path B	0 – 255		0 stands the value is not calibrated yet.
ther	Thermal value	0 – 255		0 stands the value is not calibrated yet.
MIMO_TR_mode	MIMO mode assignment	1 – 1T2R, 3 – 2T2R, 4 – 1T1R	3	
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 start an IBSS with SSID given here.	"string_value", SSID with 32 chars in max	"defaultSSID"	
bssid2join	Besides SSID, designate target BSSID to join	xxxxxxxxxxxx (12 digits mac address)		Type of byte array
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 rates	Bit0-bit11 for 1,2,5.5,11,6,9,12,18,24,,36,48,54M	0xfff	
basicrates	Basic rates	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 Bit12-Bit27 for MCS0,MCS1,…,MCS15		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. When 11n is used, driver will auto detect if legacy (11b/g) 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. If enabled in N mode, AP will deny the association

<table>
<tr><td></td><td></td><td></td><td></td><td>from 11B/G STA.</td></tr>
<tr><td>fast_roaming</td><td>Client mode fast roaming</td><td>0 – disable, 1 – enable</td><td></td><td></td></tr>
<tr><td>lowestMlcstRate</td><td>Use lowest basic rate to send multicast and broadcast</td><td>0 – disable, 1 – enable</td><td></td><td></td></tr>
<tr><td>stanum</td><td>Limit max associated sta number</td><td>0-32. 0 – disable (not limit).</td><td></td><td></td></tr>
<tr><td>authtype</td><td>802.11 Authentication type</td><td>0 – open system, 1 – shared key, 2 – auto</td><td>2</td><td></td></tr>
<tr><td>encmode</td><td>Encryption mode</td><td>0 – disabled, 1 – WEP64, 2 – TKIP, 4 – AES(CCMP), 5 – WEP128</td><td></td><td>Set to 2 always under WPA/WPA2 mode</td></tr>
<tr><td>wepdkeyid</td><td>WEP default Tx key</td><td>0-3</td><td></td><td></td></tr>
<tr><td>psk_enable</td><td>PSK mode</td><td>0 – disable, 1 – WPA, 2 – WPA2, 3 – WPA/WPA2 mixed</td><td></td><td></td></tr>
<tr><td>wpa_cipher</td><td>WPA PSK cipher suite</td><td>2 –TKIP, 8 – AES(CCMP), 10 – TKIP/AES mixed</td><td></td><td></td></tr>
<tr><td>wpa2_cipher</td><td>WPA2 PSK cipher suite</td><td>2 –TKIP, 8 – AES(CCMP), 10 – TKIP/AES mixed</td><td></td><td></td></tr>
<tr><td>passphrase</td><td>PSK key</td><td>32 characters or 64 hex digits</td><td></td><td></td></tr>
<tr><td>gk_rekey</td><td>Group key update time</td><td>0 – disable, >1 – enable</td><td></td><td>Time unit is second</td></tr>
<tr><td>802_1x</td><td>Flag of using 802.1x</td><td>0 – disable, 1 – enable</td><td></td><td>When 802.1x is enabled, the Auth daemon must be invoked</td></tr>
<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>Type of byte array</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>Type of byte array</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>Type of byte array</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>Type of byte array</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>30000</td><td>Time unit is 10 ms.</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>
</table>

iapp_enable	IAPP enable/disable	0 – disable, 1 - enable		
block_relay	Block packet relaying between associated clients	0 – relay, 1 – block relay and drop, 2 – block relay and indicate to bridge		
deny_any	Deny the association SSID of “any” including upper and lower cases	0 – disable, 1 – enable		
crc_log	Calculate CRC error packets	0 – disable, 1 – enable		
wifi_specific	Do WiFi specific check	0 – disable, 1 – enable	2	
disable_txsc	Tx shortcut enable/disable	0 – enable, 1 – enable		
disable_rxsc	Rx shortcut enable/disable	0 – enable, 1 – enable		
disable_brsc	Bridge shortcut enable/disable	0 – enable, 1 – enable		
keep_rsnie	Don’t clean RSN IE while reinitialize the interface	0 – erase, 1 – keep		
band	Band selection	1 – 11b, 2 – 11g, 4 – 11a, 8 – 11n	3	
cts2self	Use cts2Self for protection mode	0 – no, 1 – yes	1	
wds_enable	WDS enable/disable	0 – disable, 1 – enable		
wds_pure	Flag to enable pure WDS mode that don’t broadcast beacon and don’t accept any station	0 – disable, 1 – enable		
wds_priority	Give WDS packets higher priority	0 – disable, 1 – enable		
wds_num	Set number of WDS	Suggest set ‘0’ whenever driver is 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 key	10 hex digits for WEP64, 26 hex digits for WEP128		Type of byte array
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.
clone_mac_addr	Assign the target MAC to clone	xxxxxxxxxxxx (12 digits mac address)		Type of byte array
nat25sc_disable	NAT2.5 shortcut enable/disable	0 – enable, 1 – disable		
show_hidden_bss	Show hidden BSS in site survey	0 – disable, 1 – enable		
ack_timeout	Set ACK timeout value	0-255		0 means using standard value. In unit of us.
private_ie	Send and get private IE	At most 64 hex digits byte array		
groupID	Group ID of virtual AP (multiple SSID)	0-65535		When AP (including root and virtual) set the same group ID, the wlan traffics could be relayed. Root interface: wlan0 Virtual interface: wlan0-va0~wlan0-va3

				.
vap_enable	Tell driver if multiple AP function is enabled or disabled	0 – disable, 1 – enable		If multiple AP is enabled, this mib must be set to 1.
func_off	Temporary disable wlan function	0 – normal, 1 – wlan off		
qos_enable	Support WMM and QoS	0 – disable, 1 – enable		
apsd_enable	Support WMM APSD function	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		
supportedmcs	Supported MCS rates	Bit 0-15 for MCS0, …, MCS15	0xffff	
basicmcs	Basic MCS rates	Bit 0-15 for MCS0, …, MCS15		
use40M	Support 40M bandwidth in 11n mode	0 – disable, 1 – enable		
2ndchoffset	Control sideband offset	1 – secondary channel is below the primary channel, 2 – secondary channel is above the primary channel	1	
shortGI20M	Support short GI in 20M bandwidth	0 – disable, 1 – enable		
shortGI40M	Support short GI in 40M bandwidth	0 – disable, 1 – enable		
amsdu	Support packet aggratation	0 – disable, 1 – enable		
lgyEncRstrct	Restrict legacy encryption in N mode	Bit 0: WEP, Bit 1: TKIP		
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	
ledBlinkingFreq	Multiple of wlan LED blinking frequency.	1~100	1	This value will be referred only when mib value of 'led_type' is greater than 1.
wapiType	WAPI mode	0 - Disable 1 - Certificate 2 – PSK	0	
wapiPsk	WAPI PSK	Up to 32 characters		
wapiPsklen	WAPI PSK length	0~32		
wapiUCastKeyType	Unicast key update mode	1 – Disable 2 – Time based 3 – Packet based 4 – Mix mode(Rekey when time or packet number exceeds threshold)		This object selects a mechanism for rekeying the unicast key.
wapiUCastKeyTimeout	Timeout threshold of time-based unicast key update mechanism	Unit: sec.		
wapiUCastKeyPktNum	Packet number threshold of packet based unicast key update mechanism			
wapiMCastKeyType	Multicast key update mode	1 – Disable 2 – Time based 3 – Packet based 4 – Mix mode(Rekey when time or packet number exceeds threshold)		This object selects a mechanism for rekeying the multicast key.
wapiMCastKeyTimeout	Timeout threshold of time-based multicast key update mechanism	Unit: sec.		

wapiMCastKeyPkt Num	Packet number threshold of packet based multicast key update mechanism			

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

Driver based MP function:
We supported Driver based MP functions controlled by "iwpriv" utility. Please refer to *"8192C 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     network;
        unsigned char     ht_info;
        unsigned char     resv[6];
} 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 char     meshid[32];
        unsigned char     *meshidptr;
        unsigned short    meshidlen;
        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

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
- ***txdesc0, …, txdesc5*** – show tx descriptor contents for queue 0 to queue 5
- ***rxdesc*** – show rx descriptor contents
- ***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
- ****.txt*** – MAC and PHY parameter files

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.

WAPI Configuration

The driver has already supported WAPI function. Please refer to *"WAPI Porting Guide.doc"* for detail explanation and usages.

Configuration File support

The driver can be configured via a ***configuration file*** each time an interface is up.

Kernel configuration:
Select *"Network device support ---> Wireless LAN (non-hamradio) ---> Config File support"*; then rebuild kernel image.

Configuration file:
- *Path*: /etc/Wireless/RTL8192CD.dat
- *Sytax:* 'wlan_interface'_'mib_command' , e.g. wlan0_ssid=xxxx.

Notes:
1. *Add '#' in front of comment lines.*
2. *Space is NOT allowed between 'wlan_interface' and 'mib_command'.*
3. *If the user needs to configure MIB values with special characters, e.g. '#', the value of 'mib_command' MUST be* ***quoted*** *E.g. wlan0_ssid="#XXXXX@##$$%%"*
4. *'wlan_interface': wlan interface, e.g., wlan0, wlan0-va0. However, please* ***DO NOT*** *configure* ***WDS*** *interfaces because WDS is configured in wlan0 interface.*
5. *'mib_command': MIB commands, e.g., ssid=xxxx, please refer to table "MIB command table" and following "Extended MIB command table"*
6. *MIB value should be also configured for each virtual interface separately.*
7. *Each time an interface is up, the configuration file will be loaded.*

Extended MIB command table (available only if Config File support is turned on):

Name	Meaning	Value	Default	Comment
hwaddr	MAC address of WLAN interface	12 hex digits, e.g. 00e04c8192a1	0	
meshSilence	In AP+Mesh mode but not enable mesh function	0 – mesh enabled, 1 – mesh disabled	0	Available if mesh is built with kernel image

iwconfig/iwlist support

The driver has already supported iwconfig and iwlist (Wireless Tools v29) for getting or setting some configurations.

Kernel configuration:
Select *"Network device support ---> Wireless LAN (non-hamradio) ---> Wireless Extensions v18 support"* and *"Network device support ---> Wireless LAN (non-hamradio) ---> Wireless Tools v29 support"*; then rebuild kernel image.

iwconfig – configure a wireless network interface.
Notes: Because *'iwconfig'* cannot fully cover all the configurations of the AP, we suggest the users using *'iwpriv'* to setup the AP.

Synopsis of **iwconfig**:
- iwconfig [*interface*]
- iwconfig *interface* [essid *X*] [mode *M*] [freq *F*] [channel *C*] [ap *A*] [rate *R*] [rts *RT*] [frag *FT*] [enc *E*] [key *K*] [retry *R*]
- iwconfig --help
- iwconfig --version

Parameters of **iwconfig**

Name	Meaning	Value	Access	Comment
essid	ESSID	any string, e.g. iwconfig essid "My SSID"	GET/SET	
mode	operating mode of the device	*Ad-Hoc*, *Managed* (client mode), *Master* (AP mode), *Repeater*, *Monitor*	GET	
freq	operating frequency	frequency in GHz	GET/SET	
channel	operating channel value	channel value	GET/SET	
ap	MAC address	e.g. *00:e0:4c:01:23:45*	GET	
rate/bit[rate]	maximum available bit rate	bit rate in Mb/s	GET	
rts[_threshold]	RTS threshold	packet size or off	GET/SET	
frag[mentation_threshold]	fragmentation threshold	packet size; off: based on driver setting	GET/SET	
key/enc[ryption]	WEP key settings	mode: open/restricted; keys in 10 or 32 hex-digit	GET	
retry	retry limits	number of retrys	GET	

Notes: for more detailed information, please refer to the manual of iwconfig.

iwlist – Get more detailed wireless information from a wireless interface
Notes: Because *'iwlist'* cannot fully cover all the configurations of the AP, we suggest the users using *'iwpriv'* to access settings of the AP.

Synopsis of **iwlist**:
- iwlist [*interface*] [*keyword*]
- iwlist --help
- iwlist --version

keywords of **iwlist**

Name	**Meaning**	**Value**	**Comment**
scanning	site survey of neighboring WLAN devices	list of Access Points and Ad-Hoc cells in range.	
channel/frequency	supported channel and frequency	frequencies in GHz corresponding to the channels	varied as domain region changed
bitrate/rate	supported rate and extended supported rate announced in beacon	supported bit-rates in Mb/s	HT rates are not listed by iwlist
keys/encryption	WEP encryption information	key sizes, list of available keys and current transmit key	
ap/accsspoints/peers	Associated peer list	list of associated peers	
auth	Authentication capabilities	WPA, WPA2, CIPHER-TKIP, CIPHER-CCMP	

Notes: for more detailed information, please refer to the manual of iwlist.

Limitation

- H/W encryption CAM size is 32
- Multiple BSSID CAM size is 8
- Tx SKB buffer must have 8 bytes space in tail for TKIP MIC
- Support 32 wlan clients in current configuration
- Support 8 WDS number in current configuration