Qualcomm Atheros, Inc.

AP 10.4 Command Line Interface (CLI)

User Guide

80-Y8052-1 Rev. YC

July 1, 2016

Confidential and Proprietary – Qualcomm Atheros, Inc.

Qualcomm Atheros, Inc.

1700 Technology Drive

San Jose, CA 95110

U.S.A.

Revision history

Revision	Date	Description
A	May 2014	Initial release
B	June 2014	■ Updated Table 1-2; Table 1-3; Table 1-5 and Table 1-16 ■ Updated Section 1.3.20
C	August 2014	■ Added Section 1.3.32.4 ■ Added Section 1.3.37 ■ Added Section ■ Added Section 1.5.8 ■ Updated Table 1-21
D	September 2014	■ Added Section 1.3.32.5 ■ Added Section 1.3.32.6 ■ Added Section 1.3.32.7 ■ Added Section 1.3.38 ■ Updated Section 1.4.4.1 ■ Added Section ■ Added Section ■ UpdatedTable 1-27; Table 1-55; Table 1-57; and Table 1-61
E	October 2014	■ Added Chapter UCI Wireless configuration (QCA- Wi-Fi)
F	November 2014	■ UpdatedTable 1-42 andTable 1-64
G	January 2015	■ Updated Table 1-6 and Table 1-8 ■ Updated Section 1.3.21.2
H	March 2015	■ Updated Table 1-7; values of chwidth ■ Added Section 1.3.21 with `txrx_fw_stats` command argument stats for 1 to 19.
J	March 2015	■ Updated Table 1-45 ■ Added Section 1.3.21
K	June 2015	■ UpdatedSection 1.3.5, Section 1.4.2.3, Section 1.5.2, and Section 2.2
L	July 2015	■ Updated Section 1.5.4, Section , Section 1.5.9, Section 1.5.9 and Section 2.6.
M	July 2015	■ New Section 2.6 added for enabling NSS wifi offload
N	September 2015	■ Added Section 1.3.40 ■ Updated Table 1-61 ■ Added Section 2.7
P	September 2015	■ Updated Section 1.3.38 ■ Updated Section 1.3.40 ■ Updated Section 1.5.4 ■ Added Section 2.8

Revision	Date	Description
R	October 2015	■ Updated Table 1-62
T	November 2015	■ Updated numerous tables in the following sections: □ Section 1.3 □ Section 1.4.2 □ Section 1.5.6.3 ■ Added Section 1.3.32.8 ■ Added Section 1.4.2.6
U	December 2015	■ Updated the following sections □ Section 1.3.38 □ Section 1.4.6 □ Section 2.7 ■ Added Section 2.9
V	January 2016	■ Updated Section 2.4.2
W	January 2016	■ Added the following sections: □ Section 1.3.30 □ Section 2.10 □ Section 2.11 □ Section 2.12 ■ Updated the following sections: □ Section 1.3.7
Y	February 2016	■ Added the following sections: □ Section 1.3.41 □ Section 1.3.42 □ Section 1.3.44 □ Section 1.3.45 □ Section 1.3.46 □ Section 1.3.47 □ Section 2.3.4 □ Section 2.4.3 □ Section 2.5 ■ Updated the following sections: □ Section 1.3.38 □ Section 2.4.1 □ Section 2.4.2 □ Section 2.6 ■ Updated the following tables: □ Table 1-18 □ Table 1-32 □ Table 1-38 □ Table 1-45 □ Table 1-59 □ Table 2-1

Revision	Date	Description
YA	April 2016	■ Added the following sections: □ Section 1.3.48 □ Section 1.3.49 □ Section 1.3.50 ■ Updated the following sections: □ Section 2.3.2 □ Section 2.3.4 □ Table 1-6 □ Table 1-18 □ Table 1-28 □ Table 1-32 □ Table 1-59 □ Table 1-64 □ Table 2-1 □ Table 2-4
YB	May 2016	■ Added the following sections: □ Table 1-48 □ Section 1.3.51 □ Section 1.5.10 □ Section 2.13 ■ Updated the following sections: □ Section 1.3.21 □ Section 1.3.48 □ Table 1-5 □ Table 1-18 □ Table 1-21 □ Table 1-22 □ Table 1-28 □ Table 1-41 □ Table 1-45 □ Table 2-2
YC	June 2016	■ Added the following sections: □ Section 1.3.43 □ Section 1.6 □ Section 2.14 □ Table 2-5 ■ Updated the following sections: □ Table 1-16 □ Table 1-48 □ Table 2-4

Contents

Tables

Preface

This user guide provides information on the Qualcomm Atheros AP Driver Command Line Interface, which is a part of the Qualcomm Atheros AP system. This system consists of the OS kernel, utility functions, and the Qualcomm Atheros AP driver.

About this document

The document consists of the following chapters and appendixes:

Chapter 1	AP Driver Command Line Interface
Chapter 2	UCI Wireless Configuration (QCA-Wi-Fi)
Appendix A	Country Code Definitions

NOTE All 160/80+80MHz related information is applicable only for QCA9994 platform.

Additional resources

Qualcomm Atheros reference design hardware, software, and documentation contain proprietary information of Qualcomm Atheros, Inc., and are provided under a license agreement containing restrictions on use and disclosure, and are also protected by copyright law. Reverse engineering of this hardware, software, or documentation is prohibited.

For additional information, refer to the following documents:

- *AP 10.4 Programmer’s Guide, 80-Y8053-1*

Document conventions

Text conventions

bold

Bold type within paragraph text indicates commands, file names, directory names, paths, or returned values.

Example: The DK_Client package will not function unless you use the **wdreg_install** batchfile.

italic

Within commands, italics indicate a variable that the user must specify.

Example: **mem_alloc** *size_in_bytes*

Titles of manuals or other published documents are also set in italics.

`Courier`

The Courier font indicates output or display.

Example:

```
Error:Unable to allocate memory for transfer!
```

[ ]

Within commands, items enclosed in square brackets are optional parameters or values that the user can choose to specify or omit.

{ }

Within commands, items enclosed in braces are options from which the user must choose.

|

Within commands, the vertical bar separates options.

…

An ellipsis indicates a repetition of the preceding parameter.

>

The right angle bracket separates successive menu selections.

Example: Start > Programs > DK > wdreg_install.

Change bars indicates content that has been added or changed in this revision of the document.

1 AP Driver Command Line Interface

The AP driver command line interface consists of wireless tools presented in this document to view and modify AP driver environment variables. Disable selected MCS for given SSID iwpriv options.

1.1 Wireless tools

The wireless tools interface is the primary interface used in Linux for configuring and operating the WLAN interface. The tools themselves are open source, and require specific support through the IOCTL interface for the driver. The Qualcomm Atheros WLAN driver supports these tools out of the box without modification. Any version of wireless tools after the version 28 can be used with the Qualcomm Atheros WLAN driver system.

The wireless tools use device names to determine the device to configure. In the Qualcomm Atheros driver, two device types are created when AP is brought up. The radio layer, also known as the ATH/HAL layer, is instantiated as a **wifi***N* device, where N is the specific instance starting with zero. The first radio instance, for example, would be called **wifi0**. The protocol, or 802.11 layer, is instantiated as an **ath***N* device. These devices are also known as virtual APs (VAPs). Multiple VAPs can be associated with a single radio, but only one radio may be associated with any particular VAP (one-to-many relationship). Each layer controls specific aspects of system operation, and therefore each layer has specific commands that apply to it.

The two main programs of the wireless tools suite are **iwconfig** and **iwpriv**. These commands are used to get or change specific configuration or system operating parameters. Many commands are only effective before the AP interface is in the up state, so these commands must be performed prior to issuing the **ifconfig up** command to the interface. This document defines the valid parameters and commands for each command.

NOTE The radio layer does not support the **iwconfig** command, which is used exclusively in the protocol layer. Also, any **ifconfig** commands used on the AP must be applied to the protocol layer (ATH) device and has no effect on the radio layer.

There are two WLAN driver models available in the 10.4 AP software — Direct Attach (DA) and Offload (OL).

In the Direct Attach (DA) model, the entire WLAN driver runs on the host platform and interfaces with the WLAN hardware through the host bus interface (that is, PCI, PCIe, AHB, and so on). Examples of Direct Attach chipsets are AR928x, AR938x, AR939x, AR958x, AR959x, AR934x, AR935x, QCA953x, QCA955x and IPQ4019.

In the Offload (OL) model, the WLAN driver component runs on the target and thin interface layer software is added on both the host and target for the host-target communications. Examples of Offload chipsets are QCA988x, QCA989x, QCA9990 and IPQ4019.

Throughout this document, each CLI command table will consist two columns labeled “DA” and OL” to indicate the WLAN driver model which a CLI command supports.

1.2 iwconfig parameters

The **iwconfig** command encompasses a fixed set of parameters used to set up and operate the WLAN interface. They are used in much the same way as the **ifconfig** command and its parameters, but are specific to 802.11 device operations. Thus they interface to the particular VAP interface.

NOTE The radio layer does not support **iwconfig**.

Table 1-1 iwconfig parameters

Parameter	Format	DA	OL	Description
ap	iwconfig ath*N* ap *macaddr*	N	N	Selects the specific AP with which a client will associate; used only for WDS client modes in the AP environment. The only valid argument is the MAC address of the desired AP. The help text also indicates *off* and *auto* choices, but these only disable the selection of a specific MAC address. Disabled by default. The AP command is not currently supported. `#iwconfig ath0 ap 00:03:7f:01:23:45`
channel	iwconfig athN channel *opchannel*	Y	Y	Selects the operation channel. In AP mode, it is the channel the AP operates in. For STA operations, the STA associates to the appropriate AP based on the MAC address setting and the ESSID, so the channel is not important. **The channel argument only takes the channel number. See the freq command for setting the specific frequency. If an invalid channel is selected, this command returns an error status. The VAP for this interface should be destroyed at this point, as it will not be properly configured with a channel. It has no default value. The provided scripts bring up the first interface on channel 6 by default, and the second on channel 40 by default (for dual concurrent operations).** **NOTE** Issue the “ifconfig athN down” command before issuing the channel change command and “ifconfig athN up” after making the channel change. `#iwconfig ath0 channel 11`

Table 1-1 iwconfig parameters (cont.)

Parameter	Format	DA	OL	Description

enc
key
	iwconfig ath*N*
	key [index] *keyvalue*
		Y	Y	The commands **enc** and **key** are synonyms for the same command to set and manage WEP keys. The hardware will support up to four WEP keys per radio module. The optional index value indicates which key is being set/activated. The index value can be from 1 to 4.
				■ The *keyvalue* parameter can be specified in either hex mode or as an ASCII string.
				■ Key values can be specified for either WEP 64 (40) bit mode, requiring 5 bytes, or WEP 128 (108) bit mode, requiring 13 bytes.
				■ In hexadecimal mode, this comes out to 10 or 26 hex digits, respectively.
				■ Hex digits are separated in groups of 4 by hyphens.
				■ When specifying ASCII keys, the keys will require 5 or 13 characters, respectively.
				■ All ASCII key strings are preceded by the **s:** indicator.
				To turn WEP off, use the off command without index. WEP is automatically turned on when a key is specified. Specifying a key index without a key value will select that key as the active key.
				`#iwconfig ath0 key [2] DEAD-BEEF-EA`
				`#iwconfig ath0 key [1] s:AnASCIIkeyVal`
				`#iwconfig ath0 key off`

essid	iwconfig ath*N* essid *"Name of Network"*	Y	Y	Sets the name of the BSS as it is provided in the beacon message. While no official definition exists for ESSID in the 802.11 specification; this term is commonly used for a BSS network name in the Linux environment. The network name can be up to 32 characters in length and can contain spaces. When running in AP mode, it is the name of the network as advertised in the beacon message. In STA mode, it is the network name that the STA associates with. The name can be quoted ("") or not, *but must be quoted when including spaces*. The ESSID is blank by default. The provided scripts set the ESSID to **Atheros_Xspan_2G** for the first and **Atheros_Xspan_5G** for the second interface in a dual concurrent configuration. `#iwconfig ath0 essid AP50_Test`
frag	iwconfig ath*N* frag *maxfragsize*			Sets the fragmentation threshold, which is the maximum fragment size. The fragmentation threshold must be an even number. If the input value is odd, the threshold value is set to the even number that precedes it; that is, "input-1". Note that this is not valid for 802.11n aggregation operations. In addition, this parameter is not supported for QCA988x/989x, and QCA999x/998x radios. The argument level indicates the maximum fragment size, or setting to off disables fragmentation. Fragmentation is off by default. `#iwconfig ath0 frag 512`
		Y	Y	Non-HT mode
		Y	N	HT mode

Table 1-1 iwconfig parameters (cont.)

Parameter	Format	DA	OL	Description

freq	iwconfig ath*N* freq *opfreq*	Y	Y	Similar to the channel command, this command selects the frequency of operation. Note that the frequency value *opfreq* must be a valid frequency supported by the regulatory requirements table for the device. This command takes both channel numbers and frequency values. For frequency values, the suffix K, M, or G can be appended to the value to specify kHz, MHz, or GHz. The values of 2.412G, 2412M, and 2412000K are all the same value.
				If the value of opfreq is set to 0, Auto Channel Selection is triggered, which will enable automatic selection of the best possible operational channel for the AP in the presence of various types of interference.
				This command also returns an error if the indicated frequency is invalid for the device.
				`# iwconfig ath0 freq 5.2G`
				`#iwconfig ath0 freq 40`

rate	iwconfig ath*N* rate *rateval|auto*	Y	Y	Selects a fixed rate for transmit, or enables the internal rate control logic. When *rateval* is provided, it specifies the bit rate desired. Using the M or k suffix can be used to indicate the rate, such as 36M. Specifying *auto* instead of a fixed rate will enable the rate control logic internal to the driver. This is the default configuration. Setting 802.11n and 802.11ac fixed rates adds more complexity. Selecting MCS rates cannot be accomplished through this command. For 802.11n rates – use iwpriv commands **Set11NRates** and **Set11NRetries** For 802.11ac rates – use iwpriv commands **nss** and **vhtmcs**. Not supported on QCA955x. `#iwconfig ath0 rate 36M`
retry	—	N	N	Software retry is not supported
rts	iwconfig ath*N* rts *minpktsize*	Y	Y	Sets the minimum packet size for which RTS/CTS protection is used. This setting is used to reduce the amount of arbitration that occurs with short packet transmission, improving throughput. The value of *minpktsize* is set to the minimum packet size for which to use the RTS/CTS handshake. Setting *minpktsize* to a value of 0 disables RTS/CTS handshake entirely. The use of RTS/CTS in 802.11n is governed by rate tables and other settings, so this command may not have the desired effect when using 802.11n rates. The threshold should be more than 256 B (as defined by iwconfig). `#iwconfig ath0 rts 256`
sens	—	N	N	Receiver sensitivity control is not supported.
txpower	iwconfig ath*N* txpower *pwrsetting*	Y	Y	Sets the Tx power for all packets on the device. This power is limited by the regulatory limits encoded into the driver, and selected by setting the country code (see the iwpriv command **setCountry**). The value of *pwrsetting* is provided in units of dBm. Setting the power_setting value to off will enable the internal power control logic for setting power level. Default Tx power levels are dependent on information in the selected regulatory table. `#iwconfig ath0 txpower 30`

1.3 iwpriv parameters

This section defines all of the **iwpriv** parameters available for each layer.

NOTE There are some duplicate parameters between the layers. It is recommended to use the radio layer (**wifi***N*) parameters over the protocol layer (**ath***N)* parameters when duplication exists.

The radio layer parameters are provided to configure the radio layer for all VAPs attached to the radio. Common parameters for the radio include the frequency (channel), the channel width mode (HT20/40), and other parameters that apply to radio operations.

NOTE All VAPS attached to the specific radio are affected by the configurations made to the radio layer.

For all parameters having a corresponding "get" parameter, the current value(s) are returned.

1.3.1 Aggregation parameters

Table 1-2 802.11ac interface aggregation parameters

Parameter	Format	DA	OL	Description
AMPDU **getAMPDU**	iwpriv wifi*N* AMPDU {1|0}	Y	N	Enables (1) or disables (0) Tx AMPDU aggregation for the entire interface. Receiving aggregate frames will still be performed, but no aggregate frames will be transmitted if this is disabled. The get parameter returns the current value. Default is 1. Specific to 802.11n. #iwpriv wifi0 AMPDU 1 #iwpriv wifi0 getAMPDU wifi0 getAMPDU:1

ampdu get_ ampdu	iwpriv ath*N* ampdu
	{1...64}
		N	Y	Sets maximum number of mpdus gets aggregated in a single AMPDU. Specific to 802.11ac.
				`#iwpriv ath0 ampdu 1`
				`#iwpriv ath0 get_ampdu ath0 get_ ampdu:1`

amsdu get_ amsdu	iwpriv ath*N* amsdu
	{1...31}
		N	Y	Sets maximum number of AMSDU subframes. Specific to 802.11ac.
				`#iwpriv ath0 amsdu 1`
				`#iwpriv ath0 get_amsdu ath0 get_ amsdu:1`

maxampdu
get_ maxampdu
	iwpriv ath*N* maxampdu
	{0...3}
		N	Y	Set/gets HT capability field, Maximum A-MPDU length exponent. Value range is 0 to 3. Maximum A-MPDU length exponent indicates the maximum length of A- MPDU that the station can receive.
				`#iwpriv ath0 maxampdu 1`
				`#iwpriv ath0 get_maxampdu ath0 get_maxampdu:1`

Table 1-2 802.11ac interface aggregation parameters

vhtmaxampdu
get_
vhtmaxampdu
	iwpriv ath*N* vhtmaxampdu *{0...7}*	N	Y	Set/gets VHT capability field, Maximum A-MPDU length exponent. Value range is 0 to 7. Maximum A-MPDU length exponent indicates the maximum length of A-MPDU that the station can receive.
				`#iwpriv ath0 vhtmaxampdu 1`
				`#iwpriv ath0 get_vhtmaxampdu`
				`   ath0 get_vhtmaxampdu:1`

aggr_burst get_ aggr_burst	Iwpriv wifiN aggr_ burst 0|1|2|3 duration			Set the aggr burst duration on particular traffic class. 0 – BE 1 – BK 2 – VI 3 – VO Get gets the output `   #iwpriv wifi0 aggr_burst 0 800`

Table 1-3 802.11na and 802.11ac interfaces specific statistics

Parameter	Format	DA	OL	Description

burst
get_burst
	iwpriv wifiN burst
	{1|0}
		N	Y	Enables (1) or disables (0) SIFS bursting for the entire interface. The AMPDU size is dynamically determined based on Rate chosen and burst duration is also dynamically chosen. The get parameter returns the current value. Default is 1 for certain QCA9880 cards. It is 0 for the rest. Specific to 802.11n. Not valid for partial offload.
				`#iwpriv wifi0 burst 1`
				`#iwpriv wifi0 get_burst`
				`   get_burst:1`

txrx_fw_stats	iwpriv ath*N* txrx_fw_stats *{1,2,3,6,13,14,16}*	N	Y	Tx and Rx related statistics from target `#iwpriv ath0 txrx_fw_stats 1`
setaddbaoper	`iwpriv ath`*N* `setaddbaoper` *1|0*			Enables/disables automatic processing for aggregation/block ACK setup frames. To use the manual addba/delba commands, it must be set to 0 (off) to keep the driver from also responding. Has a corresponding get parameter, and its default value is 1 (enabled). `#iwpriv ath0 setaddbaoper 0`

1.3.2 ANI parameters

Table 1-4 ANI parameters

ani_enable
get_**ani_enable**
	iwpriv wifi*N* ani_enable *{1|0}*	Y	Y	Enables (1) or disables (0) ANI functionality. The default is 0. This command is specific to 802.11ac.
				`#iwpriv wifi0 ani_enable 1`
				`#iwpriv wifi0 get_ani_enable`
				`   wifi0 get_ani_enable:0`

ANIEna GetANIEna	iwpriv wifi*N* ANIEna *{1|0}*	Y	N	Enables (1) or disables (0) ANI functionality. The default is 1. `#iwpriv wifi0 ANIEna 0` #iwpriv wifi0 GetANIEna wifi0 GetANIEna:0

1.3.3 Association/ACL parameters

Table 1-5 Association/ACL parameters

Parameter	Format	DA	OL	Description	

addmac delmac getmac
maccmd get_ maccmd
acl_notify
get_acl_notify
	iwpriv ath*N* addmac *macaddr*
	iwpriv ath*N* delmac *macaddr*
	iwpriv ath*N* maccmd *cmd*
	iwpriv athN acl_ notify 1|0
	iwpriv athN get_ acl_notify
		Y	Y	These parameters set up and modify the MAC filtering list. MAC filtering allows users to either limit specific MAC addresses from associating with the AP, or specifically indicates which MAC addresses can associate with the AP.
				addmac adds specified MAC addresses to the access control list (ACL). **delmac** deletes addresses from the ACL. These parameters have no get equivalents. **getmac** displays the list of MAC addresses monitored by the ACL.
				`#iwpriv ath0 addmac 00:03:7f:00:00:20`
				`#iwpriv ath0 delmac 00:03:7f:00:12:34`
				`#iwpriv ath0 getmac`
				`   ath0 getmac:00:03:7f:00:00:20`
				maccmd instructs how the ACL is used to limit access the AP. The default is 0. The get parameter returns the current value.
				Valid *cmd* values:	

				0	Disable ACL checking
				1	Only allow association with MAC addresses on the list
				2	Deny association with any MAC address on the list
				3	Flush the current ACL list
				4	Suspend current ACL policies. Re-enable with a 1 or 2 command.

				`#iwpriv ath0 maccmd 1`
				`#iwpriv ath0 get_maccmd`
				`   ath0 get_maccmd:1`
				acl_notify configures if denial of association should be notified on the interface. Enable: 1 Disable: 0
				The default value is 1. The get parameter **get_acl_notify** returns the current configuration
				`#iwpriv ath0 acl_notify 1`
				`#iwpriv ath0 get_acl_notify`
				`   ath0 get_acl_notify:1`	

Table 1-5 Association/ACL parameters (cont.)

ap_bridge
get_ap_bridge
	iwpriv ath*N* ap_ bridge *mode*	Y	Y	Enables(0) or disables(1) bridging within the AP driver; has the effect of allowing a STA associated to the AP to access any other STA associated to the AP. This command eliminates bridging between clients. Its default value is 1. The get parameter returns the current value.
				`#iwpriv ath0 ap_bridge 0`
				`#iwpriv ath0 get_ap_bridge`
				`   ath0 get_ap_bridge:0`

kickmac	iwpriv ath*N* kickmac *macaddr*	Y	Y	Forces the AP to disassociate the specified STA. `#iwpriv ath0 kickmac 00:18:41:9b:c8:87`

sko
get_sko
	iwpriv ath*N* sko *max_retries*	N	Y	Sets STA quick kickout maximum consecutive retries value. If the node is not a NAWDS repeater and failed count reaches this value, it kicks out the node. The default value is 50. The get parameter returns the current value.
				`#iwpriv ath0 sko 50`
				`#iwpriv ath0 get_sko`
				`   ath0 get_sko:50`
				NOTE wnm needs to be ‘0’ to enable this command.

block_interbss	Iwpriv wifiN block_ interbss 0|1	N	Y	Allow or disallow forwarding the traffic between stations of two different VAPs. 0 – Allow traffic to switch between stations of two vaps 1 – Disallow the traffic to switch between stations

1.3.4 Beacon configuration parameters

Table 1-6 Beacon configuration parameters

Parameter	Format	DA	OL	Description	

enable_amsdu
get_amsdu
	iwpriv wifi*N* amsdu *0/1*	Y	Y	Sets the aggregated MSDUs (AMSDUs) transmission setting:
				`#iwpriv wifi0 amsdu 0`
				`#iwpriv wifi0`
				`   get_amsdu wifi0`
				`   get_amsdu:0`	

				0	Disable AMSDU transmission
				1	Enable AMSDU transmission

Table 1-6 Beacon configuration parameters (cont.)

Parameter	Format	DA	OL	Description

bintval
get_bintval
	iwpriv ath*N* bintval *beaconinterval*	Y	Y	Sets the AP's beacon interval value, in ms. The value determines the number of ms between beacon transmissions. For the multiple VAP case, the beacons are transmitted evenly within this interval. Thus, if four VAPs are created and if the beacon interval is 200 ms, a beacon will be transmitted from the radio portion every
				50 ms, from each VAP in a round-robin fashion. The default value of the interval is 100 ms. The get parameter returns the current value.
				The minimum beacon interval can be set as follows:
				Number of VAPS upto 2 – bintval can be >= 40 ms
				Number of VAPS upto 8 – bintval can be >= 100 ms
				Number of VAPS upto 16 – bintval can be >= 200 ms
				`#iwpriv ath0 bintval 400`
				`#iwpriv ath0 get_bintval`
				`   ath0 get_bintval:200`
				The maximum beacon interval can be set to 3500 ms

blockdfschan	iwpriv ath*N* blockdfschan *{1|0}*	Y	Y	Disables the selection of DFS channels when the 802.11h channel switch processing is selecting a new channel. Typically, when a radar signal is detected on a channel, a new channel is picked randomly from the list. DFS channels are normally included in the list, so if there are several radars in the area, another hit is possible. Setting this selection to 0 enables the use of DFS channels in the selection process, while a value of 1 disables DFS channels. The default value is 1. This limits the number of available channels. No get parameter available. `#iwpriv ath0 blockdfschan 1`

countryie
get_countryie
	iwpriv ath*N* countryie *{1|0}*	Y	Y	An enable/disable control that determines if the country IE is to be sent out as part of the beacon. The country IE is used by 802.11h processing to allow STAs to self-configure regulatory tables to the country. Sending this IE configures all such STAs to the country the AP is configured to. The default value is 1 (enabled). The get parameter returns the current value. Result is correct; ignore error message on the console.
				`#iwpriv ath0 countryie 1`
				`#iwpriv ath0 get_countryie`
				`   ath0 get_countryie:1`

doth
get_doth
	iwpriv ath*N* doth *{1|0}*	Y	Y	Enables or disables support for 802.11h regulatory information selection. For the AP, this enables or disables transmission of country IE information in the beacon. STAs supporting 802.11h configures regulatory information according to the information in the country IE. The default value is 1 (enabled). The get parameter returns the current value. Result is correct; ignore error message on the console.
				`#iwpriv ath0 doth 1`
				`#iwpriv ath0 get_doth`
				`   ath0 get_doth:1`

Table 1-6 Beacon configuration parameters (cont.)

Parameter	Format	DA	OL	Description	
doth_ chanswitch	`iwpriv ath`*N* `doth_ chanswitch` *channel tbtt*			Forces the AP to perform a channel change, and forces a channel change announcement message. Used to test the 802.11h channel switch mechanism. Has no corresponding get parameter. `#iwpriv ath0 doth_chanswitch 3 5`	
				channel	Specifies channel to which AP will switch
				tbtt	Number of beacons to wait before doing the switch

dtim_period
get_dtim_period
	iwpriv ath*N* dtim_ period *deliveryperiod*	Y	Y	Used to set the DTIM period. The DTIM is an interval specified by the AP to the STA indicating when multicast traffic may be available for the STA, requiring the STA to be awake to receive the messages. This parameter will set the AP DTIM period, in ms. A longer DTIM will provide for a greater power savings, but will increase multicast latency. This parameter has a default value of 1 ms (min) and 255 ms Max. The get parameter returns the current value.
				`#iwpriv ath0 dtim_period 5`
				`#iwpriv ath0 get_dtim_period`
				`wifi0 get_dtim_period:1`	

hide_ssid
get_hide_ssid
	iwpriv ath*N* hide_ ssid *{1|0}*	Y	Y	Hides the SSID, disabling it in the transmitted beacon, when enabled. Used for secure situations where the AP does not want to advertise the SSID name. A value of 0 will enable the SSID in the transmitted beacon. The get parameter returns the current value. The default value is 0.
				`#iwpriv ath0 hide_ssid 1`
				`#iwpriv ath0 get_hide_ssid`
				`ath0 get_hide_ssid:1`	

pureg
get_pureg
	iwpriv ath*N* pureg *{1|0}*	Y	Y	Enables or disables pure G mode. This mode does not allow 802.11b rates, and only uses OFDM modulation. The get parameter returns the current value. The default value is 0. Result is correct; ignore error message on the console.
				`#iwpriv ath0 pureg 1`
				`#iwpriv ath0 get_pureg`
				`ath0 get_pureg:1`	

puren
get_puren
	iwpriv ath*N* puren *{1|0}*	Y	Y	Enables/disables pure 11N mode, which does not accept STAs that do not have HT caps in AP mode. Result is correct; ignore error message on the console.
				`#iwpriv ath0 puren 1`
				`#iwpriv ath0 get_puren`
				`ath0 get_puren:1`	

				1	Enable pure 11N mode
				0	Disable pure 11N mode

set_bcnburst
get_bcnburst
	iwpriv wifi*N* set_ bcnburst *{1|0}*	Y	Y	Set the beaconing scheme. to burst or staggered mode. The default is staggered mode.
				`#iwpriv wifi0 set_bcnburst 0`
				`#iwpriv wifi0 get_bcnburst`
				wifi0 get_bcnburst: 0	

				1	burst mode
				0	staggered mode

Table 1-6 Beacon configuration parameters (cont.)

Parameter	Format	DA	OL	Description

setoptie
getoptie
	iwpriv athN setoptie
	iwpriv athN getoptie
				Sets/gets application specific optional IE buffer.
				`#iwpriv ath0 setoptie`
				`#iwpriv ath0 getoptie`
				`   ath0 getoptie:`

shortgi
get_shortgi
	iwpriv ath*N* shortgi *{1|0}*	Y	Y	Enables/disables the short gating interval (shortgi) when transmitting HT40 frames. This effectively increases the PHY rate by 25%. This is a manual control typically used for testing. The get parameter returns the current value. The default value is 1.
				`#iwpriv ath0 shortgi 1`
				`#iwpriv ath0 get_shortgi`
				`   ath0 get_shortgi:1`

vap_contryie
get_vapcontryie
	iwpriv ath*N* vap_contryie *1/0*	Y	N	Enables/disables Country IE support of the specified VAP ath*N* in nBSSID mode. Default value is 1. Not supported.
				`#iwpriv ath0 vap_contryie 1`
				`#iwpriv ath0 get_vapcontryie`
				`   ath0 get_vapcontryie:1`
				1 Enable Country IE support
				0 Disable Country IE support

vap_doth
get_vapdoth
	iwpriv ath*N* vap_doth *1/0*	Y	Y	Enables (1) or disables (0) 802.11h support of the specified VAP in mBSSID mode. Default value is 1. Result is correct; ignore error message on the console.
				`#iwpriv ath0 vap_doth 1`
				`#iwpriv ath0 get_vap_doth`
				`   ath0 get_vap_doth:1`
				1 Enable 802.11h support
				0 Disable 802.11h support

1.3.5 Channel width parameters

Table 1-7 Channel width parameters

Parameter	Format	DA	OL	Description

chextoffset
get_ chextoffset
	iwpriv ath*N* chextoffset *channeloffset*	Y	Y	Sets the extension (Secondary) channel offset field in the AP beacon High Throughput Information Element (HT IE). If this parameter is not executed, then the extension channel offset is taken from the device settings. This parameter has a corresponding get parameter. The default value is 0.
				`#iwpriv ath0 chextoffset 0`
				`#iwpriv ath0 get_chextoffset`
				`   ath0 get_chextoffset:0`
				0 Use the device settings
				1 None
				2 Extension (Secondary) channel is above the control (Primary) channel
				3 Extension (Secondary) channel is below the control (Primary) channel

Table 1-7 Channel width parameters (cont.)

<table>
<tr><th>Parameter</th><th>Format</th><th>DA</th><th>OL</th><th colspan="2">Description</th></tr>
<tr><td rowspan="5"><b>chwidth</b>
<b>get_chwidth</b></td><td rowspan="5">iwpriv ath<i>N</i> chwidth <i>channelwidth</i></td><td rowspan="5">Y</td><td rowspan="5">Y</td><td colspan="2">Sets the channel width field in the AP beacon High Throughput Information Element (HTIE). If this command is not executed, then the channel width is taken from the device settings. The get parameter returns the current value. The default value is 0.
Sets the current channel width setting. Not necessarily the value set by <b>cwmode</b>, because it can be automatically overridden.
<code>#iwpriv ath0 chwidth 0</code>
<code>#iwpriv ath0 get_chwidth</code>
<code>   ath0 get_chwidth:0</code></td></tr>
<tr><td>0</td><td>(HT)20 MHz</td></tr>
<tr><td>1</td><td>20 MHz</td></tr>
<tr><td>2</td><td>20/40 MHz</td></tr>
<tr><td>≥3</td><td>20/40/80 MHz</td></tr>
<tr><td><b>cwmenable</b>
<b>get_ cwmenable</b></td><td>iwpriv ath<i>N</i> cwmenable <i>{1|0}</i></td><td>Y</td><td>Y</td><td colspan="2">Enables or disables automatic channel width management. If set to 0, the CWM state machine is disabled (1 enables the state machine). Used when static rates and channel widths are desired. The default is 1. The get parameter returns the current value.
<code>#iwpriv ath0 cwmenable 1</code>
<code>#iwpriv ath0 get_cwmenable</code>
<code>   ath0 get_cwmenable:1</code></td></tr>
</table>

Table 1-7 Channel width parameters (cont.)

Parameter	Format	DA	OL	Description	

mode
get_mode
	iwpriv ath*N* mode *operatingmode*	Y	Y	Sets the current operating mode of the interface. The argument is a string that defines the desired mode of operation. The mode also affects the configuration of the radio layer. The argument for mode is provided as a string. The default value is AUTO. The get parameter returns the mode as a string value.
				`#iwpriv ath0 mode 11NAHT20`
				`# iwpriv ath0 get_mode`
				`   ath0 get_mode:11ng20`
				The operating modes include:	

				AUTO	Mode is set automatically
				11A	Legacy operation in 802.11a (5 GHz)
				11B	Legacy operation in 802.11b (2.4 GHz)
				11G	802.11g
				11NAHT20	802.11n A-band 20 MHz channels
				11NGHT20	802.11n G-band 20 MHz channels
				11NAHT40PLUS	802.11n A-band 40 MHz channels. Select frequency channels higher than the primary control channel as the extension channel
				11NAHT40MINUS	802.11n A-band 40 MHz channels. Select frequency channels lower than the primary control channel as the extension channel
				11NGHT40PLUS	802.11n G-band 40 MHz channels. Select frequency channels higher than the primary control channel as the extension channel
				11NGHT40MINUS	802.11n G-band 40 MHz channels. Select frequency channels lower than the primary control channel as the extension channel
				11ACVHT20	802.11ac A-band 20 MHz channels
				11ACVHT40PLUS	802.11ac A-band 40 MHz channels. Select frequency channels higher that control channel as the extension channel.
				11ACVHT40MINUS	802.11ac A-band 40 MHz channels. Select frequency channels lower that control channel as the extent ion channel.
				11ACVHT80	802.11ac A-band 80 MHz channels
				11ACVHT160	802.11ac A-band continuous 160 MHz channels.
				11ACVHT80_80	802.11ac A-band discontinuous 80+80 MHz channels.

cfreq2 get_ cfreq2	iwpriv athN cfreq2 center_ freq	Y	Y	This is only applicable when operating mode is 11ACVHT80_80. This sets center frequency for 2nd 80MHz band. The argument for cfreq2 is as provided as integer. This integer can be channel center frequency index (IEEE channel) or center frequency in MHz.
				`#iwpriv ath0 cfreq2 106 or`
				`#iwpriv ath0 cfreq2 5530`	

1.3.6 Debug parameters

Table 1-8 Debug parameters

Parameter	Format	DA	OL	Description

dbgLVL
getdbgLVL
	iwpriv ath*N* dbgLVL *{1|0}*	Y	Y	Controls the debug level of the VAP-based debug print statements. It is normally set to zero, eliminating all prints. The input value should be a hexadecimal value. See Table 1-9.
				`#iwpriv ath0 dbgLVL 0xffffffff`
				`# iwpriv ath0 getdbgLVL`
				`ath0 getdbgLVL:0xffffffff`

				0 — Disable debug prints
				1 — Enable debug prints (note that each bitmask has its own debug level)

HALDbg
GetHALDbg
	iwpriv wifi*N* HALDbg *{1|0}*			Sets the debug level in the HAL code; can be modified as required. The HAL must be built with the AH_DEBUG parameter defined for this command to be available; otherwise, it is conditionally compiled out. The value provided is a bitmask selecting specific categories of debug information from which to select.
				NOTE Some categories will produce copious amounts of output, and should be used sparingly for a few seconds. See Table 1-9 on page 25. The get parameter returns the current value in decimal format (convert to hexadecimal to match the list in the table). The default is 0 (no debugging), but it does not disable the unmaskable prints.
				For example, to set and get debug information for an 802.1x radius client:
				`#iwpriv wifi0 HALDbg 0x00008000`
				`#iwpriv wifi0 GetHALDbg`
				`wifi0 GetHALDbg:32768`

				0 — Disable debugging
				1 — Enable debugging

Table 1-9 802.11 Protocol layer debug bitmask

Symbolic name	Bit value	Description
IEEE80211_MSG_P2P_PROT	0x0100000000	P2P protocol driver debug
IEEE80211_MSG_RRM	0x0200000000	Radio resource measurement debug
IEEE80211_MSG_WNM	0x0400000000	Wireless network management debug
IEEE80211_MSG_PROXYARP	0x0800000000	Proxy ARP debug
IEEE80211_MSG_L2TIF	0x1000000000	Hotspot 2.0 L2 TIF debug

Table 1-9 802.11 Protocol layer debug bitmask (cont.)

Symbolic name	Bit value	Description
IEEE80211_MSG_WIFIPOS	0x2000000000	Wi-Fi positioning feature debug
IEEE80211_MSG_DFS	0x0400000000	DFS debug message
IEEE80211_MSG_MLME	0x80000000	MLME mode debug
IEEE80211_MSG_DEBUG	0x40000000	IFF_DEBUG equivalent
IEEE80211_MSG_DUMPPKTS	0x20000000	IFF_LINK2 equivalent
IEEE80211_MSG_CRYPTO	0x10000000	Crypto work
IEEE80211_MSG_INPUT	0x08000000	Input handling
IEEE80211_MSG_XRATE	0x04000000	Rate set handling
IEEE80211_MSG_ELEMID	0x02000000	Element ID parsing
IEEE80211_MSG_NODE	0x01000000	Node handling
IEEE80211_MSG_ASSOC	0x00800000	Association handling
IEEE80211_MSG_AUTH	0x00400000	Authentication handling
IEEE80211_MSG_SCAN	0x00200000	Scanning
IEEE80211_MSG_OUTPUT	0x00100000	Output handling
IEEE80211_MSG_STATE	0x00080000	State machine
IEEE80211_MSG_POWER	0x00040000	Power save handling
IEEE80211_MSG_DOT1X	0x00020000	802.1x authenticator
IEEE80211_MSG_DOT1XSM	0x00010000	802.1x state machine
IEEE80211_MSG_RADIUS	0x00008000	802.1x radius client
IEEE80211_MSG_RADDUMP	0x00004000	Dump 802.1x radius packets
IEEE80211_MSG_RADKEYS	0x00002000	Dump 802.1x keys
IEEE80211_MSG_WPA	0x00001000	WPA/RSN protocol
IEEE80211_MSG_ACL	0x00000800	ACL handling
IEEE80211_MSG_WME	0x00000400	WME protocol
IEEE80211_MSG_SUPG	0x00000200	SUPERG
IEEE80211_MSG_DOTH	0x00000100	802.11h
IEEE80211_MSG_INACT	0x00000080	Inactivity handling
IEEE80211_MSG_ROAM	0x00000040	STA-mode roaming
IEEE80211_MSG_ACTION	0x00000020	Action management frames
IEEE80211_MSG_WDS	0x00000010	WDS handling
IEEE80211_MSG_SCANENTRY	0x00000008	Scan entry
IEEE80211_MSG_SCAN_SM	0x00000004	Scan state machine
IEEE80211_MSG_ACS	0x00000002	Auto channel selection
IEEE80211_MSG_TDLS	0x00000001	TDLS
IEEE80211_MSG_ANY	0xFFFFFFFF	Anything

Table 1-10 HAL debug flags

Symbolic name	Enable Bit	Description
HAL_DBG_RESET	0x00000001	Information pertaining to reset processing and initialization
HAL_DBG_PHY_IO	0x00000002	PHY read/write states
HAL_DBG_REG_IO	0x00000004	Register I/O, including all register values. Use with caution.
HAL_DBG_RF_PARAM	0x00000008	RF parameter information and table settings
HAL_DBG_QUEUE	0x00000010	Queue management for WMM support
HAL_DBG_EEPROM_DUMP	0x00000020	Large EEPROM information dump; system must be compiled with a defined EEPROM_DUMP conditional variable
HAL_DBG_EEPROM	0x00000040	EEPROM read/write and status information
HAL_DBG_NF_CAL	0x00000080	Noise Floor calibration debug information
HAL_DBG_CALIBRATE	0x00000100	All other calibration debug information
HAL_DBG_CHANNEL	0x00000200	Channel selection and channel settings
HAL_DBG_INTERRUPT	0x00000400	Interrupt processing. **WARNING** This produces a LOT of output, use in short bursts.
HAL_DBG_DFS	0x00000800	DFS settings
HAL_DBG_DMA	0x00001000	DMA debug information
HAL_DBG_REGULATORY	0x00002000	Regulatory table settings and selection
HAL_DBG_TX	0x00004000	Transmit path information
HAL_DBG_TXDESC	0x00008000	Transmit descriptor processing
HAL_DBG_RX	0x00010000	Receive path information
HAL_DBG_RXDESC	0x00020000	Receive descriptor processing
HAL_DBG_ANI	0x00040000	Debug information for automatic noise immunity (ANI)
HAL_DBG_BEACON	0x00080000	Beacon processing and setup information
HAL_DBG_KEYCACHE	0x00100000	Encryption key management
HAL_DBG_POWER_MGMT	0x00200000	Power and Tx Power level management
HAL_DBG_MALLOC	0x00400000	Memory allocation
HAL_DBG_FORCE_BIAS	0x00800000	Force bias related processing
HAL_DBG_POWER_OVERRIDE	0x01000000	Tx power override processing
HAL_DBG_SPUR_MITIGATE	0x02000000	Mitigate
HAL_DBG_PRINT_REG	0x04000000	Print reg.
HAL_DBG_TIMER	0x08000000	Debug timer
HAL_DBG_UNMASKABLE	0xFFFFFFFF	Will be printed in all cases if AH_DEBUG is defined

1.3.7 Dynamic Channel Selection for Interference Mitigation (DCS-IM) Parameters

Table 1-11 DCS-IM parameters

<table>
<tr><th>Parameter</th><th>Format</th><th>DA</th><th>OL</th><th colspan="2">Description</th></tr>
<tr><td rowspan="5"><b>dcs_enable</b>
<b>get_dcs_enable</b></td><td rowspan="5">iwpriv wifi<i>N</i> dcs_ enable <i>value</i></td><td rowspan="5">Y</td><td rowspan="5">Y</td><td colspan="2">Enable or disable DCS.
<code>#iwpriv wifi0 dcs_enable 0</code>
<code>#iwpriv wifi0 get_dcs_enable</code>
<code>   wifi0 get_dcs_enable:0</code></td></tr>
<tr><td>0</td><td>Disable DCS</td></tr>
<tr><td>1</td><td>Enable DCS for CW interference mitigation (CW_IM).</td></tr>
<tr><td>2</td><td>Enable DCS for WLAN interference mitigation. Since the algorithm defined in this section primarily mitigates WLAN interferences, DCS for WLAN is referred to as WLAN interference mitigation (WLAN_IM).
<b>NOTE</b> This value is supported only in 5G</td></tr>
<tr><td>3</td><td>Enable both DCS for CW_IM and DCS for WLAN_IM
<b>NOTE</b> This value is supported only in 5G</td></tr>
<tr><td><b>set_dcs_intrth</b>
get_<b>dcs_intrth</b></td><td>iwpriv wifi<i>N</i> set_ dcs_intrth <i>value</i></td><td>Y</td><td>N</td><td colspan="2">Configures co-channel interference threshold (in percent) to trigger channel change. Default <i>value</i> of co-channel interference threshold is 30%.
<code>#iwpriv wifi0 set_dcs_intrth 30</code>
<code>#iwpriv wifi0 get_dcs_intrth</code>
<code>   wifi0 get_dcs_intrth:30</code></td></tr>
<tr><td><b>set_dcs_errth</b>
get_<b>dcs_errth</b></td><td>iwpriv wifi<i>N</i> set_ dcs_errth <i>value</i></td><td>Y</td><td>N</td><td colspan="2">Configures transmission failure rate threshold, used to indicates the presence of interference. Default <i>value</i> of transmission failure rate threshold is 30%.
<code>#iwpriv wifi0 set_dcs_errth 30</code>
<code>#iwpriv wifi0 get_dcs_errth</code>
<code>   wifi0 get_dcs_errth:30</code></td></tr>
<tr><td><b>s_dcs_phyerrth</b>
g_<b>dcs_phyerrth</b></td><td>iwpriv wifi<i>N</i> s_dcs_phyerrth <i>value</i></td><td>Y</td><td>Y</td><td colspan="2">Configures channel time wasted due to each PHY error (PHY error Penalty). Default <i>value</i> of PHY error penalty is set as 500 µsec.
<code>#iwpriv wifi0 s_dcs_phyerrth 500</code>
<code>#iwpriv wifi0 g_dcs_phyerrth</code>
<code>   wifi0 get_dcs_phyerrth:500</code></td></tr>
<tr><td>set_dcs_ coch_th
get_dcs_ coch_th</td><td>iwpriv wifi<i>N</i> set_ dcs_coch_th <i>value</i></td><td>N</td><td>Y</td><td colspan="2">Configures co-channel interference threshold (in percent) to trigger channel change. Default <i>value</i> of co-channel interference threshold is 30.
<code>#iwpriv wifi1 set_dcs_coch_th 30</code>
<code>#iwpriv wifi1 get_dcs_coch_th</code>
<code>   dcs_coch_th:30</code></td></tr>
</table>

Table 1-11 DCS-IM parameters (cont.)

Parameter	Format	DA	OL	Description

set_dcs_maxcu
get_dcs_maxcu
	iwpriv wifi*N* set_dcs_maxcu *value*	N	Y	Configures the maximum user channel utilization at which adjacent channel interference should be detected.Default value is 50.
				`#iwpriv wifi1 set_dcs_maxcu 50`
				`#iwpriv wifi1 get_dcs_maxcu`
				`   get_dcs_maxcu:50`

set_dcs_debug
get_dcs_debug
	iwpriv wifiN set_dcs_debug <value>	N	Y	Configuration to display debug info. Default value is 0.
				0 – disable debug info
				1 – Enable critical prints only
				`#iwpriv wifi1 set_dcs_debug 50`
				`#iwpriv wifi1 get_dcs_debug`
				`   get_dcs_debug:50`

NOTE DCS only supports CW detection on 2.4G radio. So the value it accept is either 0 or 1.

1.3.8 Green AP parameters

Table 1-12 Green AP Parameters

Parameter	Format	DA	OL	Description

ant_ps_on
get_ant_ps_on
	iwpriv ath*N* ant_ps_on *{1|0}*	Y	Y	Enables (1) or disables (0) green AP power save logic. The default value is 1.
				`#iwpriv ath0 ant_ps_on 1`
				`#iwpriv ath0 get_ant_ps_on`
				`   ath0 get_ant_ps_on:1`

ps_timeout
get_ps_timeout
	iwpriv ath*N* ps_timeout *transition_time*	Y	Y	Sets the transition time in seconds between power save off to power save on mode. The default value is 20.
				`#iwpriv ath0 ps_timeout 20`
				`#iwpriv ath0 get_ps_timeout`
				`   ath0 get_ps_timeout:20`

1.3.9 Hotspot 2.0

Table 1-13 Hotspot 2.0 parameters

Parameter	Format	DA	OL	Description

qbssload
get_qbssload
	iwpriv ath*N* qbssload *{1|0}*	Y	Y	Enables (1) or disables (0) BSS Load IE functionality. The get parameter returns the current value.
				`#iwpriv ath0 qbssload 1`
				`#iwpriv ath0 get_qbssload`
				`   ath0 get_qbssload:1`

proxyarp
get_proxyarp
	iwpriv ath*N* proxyarp *{1|0}*	Y	Y	Enables (1) or disables (0) ProxyARP functionality. The get parameter returns the current value.
				`#iwpriv ath0 proxyarp 1`
				`#iwpriv ath0 get_proxyarp`
				`   ath0 get_proxyarp:1`

Table 1-13 Hotspot 2.0 parameters (cont.)

Parameter	Format	DA	OL	Description

l2tif
get_l2tif
	iwpriv ath*N* l2tif *{1|0}*	Y	Y	Enables (1) or disables (0) Layer 2 Isolation Function (L2TIF). The get parameter returns the current value.
				`#iwpriv ath0 l2tif 1`
				`#iwpriv ath0 get_l2tif`
				`   ath0 get_l2tif:1`

dgaf_disable
g_dgaf_disable
	iwpriv ath*N* dgaf_ disable *{1|0}*	Y	Y	Enables (1) or disables (0) Downstream Group Address Forwarding Disable (DGAF Disable) functionality. The get parameter returns the current value.
				`#iwpriv ath0 dgaf_disable 1`
				`#iwpriv ath0 g_dgaf_disable`
				`   ath0 g_dgaf_disable:1`

1.3.10 HT20/HT40 coexistence parameters

Table 1-14 HT20/HT40 coexistence parameters

Parameter	Format	DA	OL	Description	

disablecoext
g_disablecoext
	iwpriv ath*N* disablecoext *1/0*	Y	N	Sets HT20/HT40 coexistence support. The default value is 0. The get parameter returns the current value.
				`#iwpriv ath0 disablecoext 0`
				`#iwpriv ath0 g_disablecoext`
				`   ath0 g_disablecoext:0`	

				0	Enable HT20/HT40 Coexistence support
				1	Disable HT20/HT40 Coexistence support

chscaninit
get_chscaninit
	iwpriv ath*N* chscaninit *interval_value*			Sets the overlapping BSS scan interval value. The get parameter returns the current value.
				`#iwpriv ath0 chscaninit`
				`#iwpriv ath0 get_chscaninit`
				`   ath0 get_chscaninit:`	

ht40intol
get_ht40intol
	iwpriv ath*N* ht40intol *1/0*			Sets support for HT20/HT40 coexistence management frame support. The default value is 0. The get parameter returns the current value.
				`#iwpriv ath0 ht40intol 0`
				`#iwpriv ath0 get_ht40intol`
				`   ath0 get_ht40intol:0`	

				0	Disable HT20/HT40 Coexistence Management frame support
				1	Enable HT20/HT40 Coexistence Management frame support

1.3.11 iQue parameters

Table 1-15 iQue parameters

Parameter	Format	DA	OL	Description
get_hbrstate	iwpriv ath*N* *get_hbrstate*	N	N	Displays Head of Line Block (HBR) related statistics: VoW, node address, state, trigger, block, dropped VI frames.
get_iqueconfig	iwpriv ath*N* *get_iqueconfig*	Y	N	Prints all iQUE configuration settings.
hbrparams	iwpriv ath*N* hbrparams *ac mode perlowbound*	N	N	Sets HBR mitigation. See Table 1-29 for access categories. For example, to enable HBR for video (vi) streams, use `iwpriv ath0 hbrparams 2 1 x`. The “x” value valid range is from 0-49, and indicates the lower bound PER; a PER better than this value causes HBR to unblock the node.
hbrPER_high **get_hbrPER_high**	iwpriv wifi*N* hbrPER_high *PER%*	N	N	Sets the upper bound PER (Packet Error Rate). If PER is greater than this value and MCS is low, HBR blocks the node (UDP video traffic to this node gets blocked). The PER is expressed as a percentage; for example, 25 means a 25% packet error rate. The get parameter returns the current value.
hbrPER_low **get_hbrPER_low**	iwpriv wifi*N* hbrPER_low *PER%*	N	N	Sets the lower bound PER. If PER is better than this value while probing, HBR unblocks the node (UDP video traffic to this node gets resumed). The PER is expressed as a percentage; for example, 25 means a 25% packet error rate. The get parameter returns the current value.

get_hbrtimer	`iwpriv athN`
	`get_hbrtimer`
		Y	N	Disabled internally.

hbrtimer
get_hbrtimer
	iwpriv ath*N* hbrtimer *timeout*	N	N	Sets the HBR timer timeout value in milliseconds. The default value is 2000 msec (2 seconds).
				`#iwpriv ath0 hbrtimer 2000`
				`#iwpriv ath0 get_hbrtimer`
				`   ath0 get_hbrtimer:2000`

Table 1-15 iQue parameters (cont.)

<table>
<tr><th>Parameter</th><th>Format</th><th>DA</th><th>OL</th><th colspan="3">Description</th></tr>
<tr><td rowspan="12"><b>mcastenhance</b>
<b>g_ mcastenhance</b></td><td rowspan="12">iwpriv ath<i>N</i> mcastenhance <i>mode</i></td><td rowspan="12">Y</td><td rowspan="12">Y</td><td colspan="3">Set multi-cast enhancement mode.
<code>#iwpriv ath0 mcastenhance 0</code>
<code>#iwpriv ath0 g_mcastenhance</code>
<code> ath0 g_mcastenhance:0</code></td></tr>
<tr><td colspan="3"><i>AP software versions 9.2/9.3/9.4</i></td></tr>
<tr><td>0</td><td colspan="2">Disable multi-cast enhancement</td></tr>
<tr><td>1</td><td colspan="2">Enable multi-cast enhancement; use tunneling mode. In OL chip tunneling is not supported.</td></tr>
<tr><td>2</td><td colspan="2">Enable multi-cast enhancement; use translating mode.</td></tr>
<tr><td colspan="3"><i>AP software versions 9.5/9.5.1/9.5.2/9.5.3</i></td></tr>
<tr><td>Value</td><td>Snooping</td><td>Multi-cast enhancement</td></tr>
<tr><td>0</td><td>Enabled</td><td>True multi-cast packet is send if any interested member is present.</td></tr>
<tr><td>1</td><td>Enabled</td><td>Tunneled unicast packet is send to interested members.</td></tr>
<tr><td>2</td><td>Enabled</td><td>Translated unicast packet is send to interested members.</td></tr>
<tr><td>4</td><td>Enabled</td><td>Disabled (Set bit 2 = 1)</td></tr>
<tr><td>5</td><td>Disabled</td><td>Enables Hy-Fi managed multi-cast functionality.</td></tr>
<tr><td><b>me_adddeny</b></td><td>iwpriv ath<i>N</i> me_ adddeny <i>groupaddresstbl</i></td><td>Y</td><td>N</td><td colspan="3">Adds the table of group addresses that are <i>not</i> to be learned. The <i>groupaddresstbl</i> value is to be entered as 4 integers (for example:- 239 255 255 1)
Two addresses exist in the snoop deny table by default: 224.0.0.1, 239.255.255.1</td></tr>
<tr><td><b>me_cleardeny</b></td><td>iwpriv ath<i>N</i> me_cleardeny <i>value</i></td><td>Y</td><td>N</td><td colspan="3">Clears the snoop deny table entries. <i>value</i> can be any integer; however, the parameter clears the snoop deny table regardless of <i>value</i>.</td></tr>
<tr><td><b>me_length</b>
<b>get_me_length</b></td><td>iwpriv ath<i>N</i> me_length <i>tablelength</i></td><td>Y</td><td>N</td><td colspan="3">Sets the snoop table length as number of entries. The default value is 64. The <i>param</i> range is 0 – 64. The get parameter returns the current value.</td></tr>
<tr><td><b>me_showdeny</b></td><td>iwpriv ath<i>N</i> me_showdeny <i>groupaddresstbl</i></td><td>Y</td><td>N</td><td colspan="3">Displays the table of group addresses that are <i>not</i> to be learned. Two addresses exist in the snoop deny table by default: 224.0.0.1, 239.255.255</td></tr>
<tr><td rowspan="6"><b>medebug</b>
<b>get_medebug</b></td><td rowspan="6">iwpriv ath<i>N</i> medebug <i>debuglevel</i></td><td rowspan="6">Y</td><td rowspan="6">N</td><td colspan="3">Sets the debug level for multicast enhancement. Param argument can accept any combination of the values below. The get parameter returns the current value.</td></tr>
<tr><td>0</td><td colspan="2">IEEE80211_ME_DBG_NONE</td></tr>
<tr><td>1</td><td colspan="2">IEEE80211_ME_DBG_INFO</td></tr>
<tr><td>2</td><td colspan="2">IEEE80211_ME_DBG_DEBUG</td></tr>
<tr><td>4</td><td colspan="2">IEEE80211_ME_DBG_DUMP</td></tr>
<tr><td>8</td><td colspan="2">IEEE80211_ME_DBG_ALL</td></tr>
</table>

Table 1-15 iQue parameters (cont.)

Parameter	Format	DA	OL	Description
medropmcast **get_ medropmcast**	iwpriv ath*N* medropmcast *{1|0}*	Y	N	Enables/disables medropmcast feature, which drops multi-cast packets if the snoop table is empty. The default value is 1. 0 Disables medropmcast 1 Enables medropmcast
medump	iwpriv ath*N* medump	Y	N	Dumps the snoop table for multi-cast enhancement.
medump_ dummy	`N/A`			Not supported; used by developers to debug the multi-cast to unicast feature.
metimeout **get_metimeout**	iwpriv ath*N* metimeout *timeoutper*	Y	N	Sets the timeout in ms for a STA to be removed from the snoop table if idle. The *param* value may be any unsigned integer value. The default is 120000 (2 minutes). The get parameter returns the current value.
metimer **get_metimer**	iwpriv ath*N* metimer *timer*	Y	N	Sets the timer in ms to check the status of the snoop table. The *timer* value may be any unsigned integer. The default is 30000 (30 seconds). The get parameter returns the current value.

1.3.12 Physical layer parameters

Table 1-16 Physical layer parameters

Parameter	Format	DA	OL	Description

LDPC
getLDPC
	iwpriv athN ldpc *{1|0}*	Y	N	Enables (1) or disables (0) the Low-density parity check feature, as described in 802.11n specification. The default value is 1.
				This option will have an effect only on chips supporting the LDPC feature. On other chips, this option will have no effect. Specific to 802.11n.
				`# iwpriv athN0 LDPC 1`
				`# iwpriv athN0 getLDPC`
				`    ath0 getLDPC:1`

setCountryID
getCountryID
	iwpriv wifi*N* setCountryID *countryidnum*	Y	Y	Sets the AP to the regulatory requirements of the country. See Table A-1 on page 176 for a full list of country IDs and strings. Default values are taken from the EEPROM. Country ID must be defined during initialization, as required for final system configuration. The get parameters return the current values.
				`#iwpriv wifi0 setCountryID 250`
				`#iwpriv wifi0 setCountry FR`
				`#iwpriv wifi0 getCountryID`
				`   wifi0 getCountryID:250`
				`#iwpriv wifi0 getCountry`
				`   wifi0 getCountry:FR`
				SetCountryID: Takes an integer value that represents the country, such as 250 for France
				setCountry: Takes an argument including the 2-character country string plus I (indoor) or O (outdoor)

Table 1-16 Physical layer parameters (cont.)

<table>
<tr><th>Parameter</th><th>Format</th><th>DA</th><th>OL</th><th colspan="2">Description</th></tr>
<tr><td rowspan="6"><b>txchainmask</b>
<b>rxchainmask</b>
<b>get_ txchainmask</b>
<b>get_ rxchainmask</b></td><td rowspan="6">iwpriv wifi<i>N</i> txchainmask <i>mask</i>
iwpriv wifi<i>N</i> rxchainmask <i>mask</i></td><td rowspan="6">Y</td><td rowspan="6">Y</td><td colspan="2">Sets the Tx and Rx chainmask values. For MIMO devices, indicates the number of Tx/Rx streams, and which chains are used. For some Qualcomm Atheros devices, up to 3 chains can be used, others are restricted to 3, 2 or 1.
The maximum number of chains available for the device. For dual chain devices, chain 2 is not available. Single chain devices only support chain 0. The chains are represented in the bit mask as:</td></tr>
<tr><td>Chain 0</td><td>0x01</td></tr>
<tr><td>Chain 1</td><td>0x02</td></tr>
<tr><td>Chain 2</td><td>0x04</td></tr>
<tr><td>Chain 4</td><td>0x08</td></tr>
<tr><td colspan="2">Chainmask selection can affect several performance factors. For a 3-chain device, an Rx chainmask of 0x05 (or 0x03) is used for 2x2 stream reception. For near range operations, a Tx chainmask of 0x05 (or 0x03) minimizes near range effects. For far range, a mask of 0x07 is used for Tx. The default chainmask values are stored in EEPROM. This iwpriv command overrides the chainmask settings. The get parameters returns the current values.
#iwpriv wifi0 txchainmask 0x05
#iwpriv wifiN rxchainmask 0x05
#iwpriv wifiN get_txchainmask
   wifi0 get_txchainmask:5
#iwpriv wifiN get_rxchainmask mask
   wifi0 get_rxchainmask:5</td></tr>
<tr><td><b>TXPowLim2G</b>
<b>TXPowLim5G</b>
<b>getTxPowLim2G</b>
<b>getTxPowLim5G</b></td><td>iwpriv wifi<i>N</i> TXPowLim2G <i>limit</i>
iwpriv wifi<i>N</i> TXPowLim5G <i>limit</i></td><td>Y</td><td>Y</td><td colspan="2">Sets the maximum transmit power limit for the 2 GHz band or 5 GHz band. The maximum transmit power is also governed by country-specific regulatory requirements set by the iwpriv setCountry or setCountryID parameters. The iwconfig txpower command is similar but sets maximum transmit power for all frequencies. The TxPowLim2G/TxPowLim5G settings can be overridden by TxPwrOvr. The TxPowLim2G/TxPowLim5G values may be also updated by other portions of the code, so the effect of the value may be temporary. The limit is expressed as an integer that equals +0.5 dBm for each value of 1. For example, 0 = 0 dBm; 10 = 5 dBm; 100 = 50 dBm. The default is 100 for both parameters. The get parameters return the current values.
#iwpriv wifi0 TXPowLim2G 20
#iwpriv wifi0 getTxPowLim2G
   wifi0 getTxPowLim2G:20</td></tr>
</table>

Table 1-16 Physical layer parameters (cont.)

Parameter	Format	DA	OL	Description

set_txpow_ mgmt
get_txpow_ mgmt
	iwpriv wifiN set_txpow_ mgmt *frame_ subtype transmit_ power*
	iwpriv wifiN get_txpow_ mgmt frame_subtype
		Y	Y	Used to configure transmit power for beacon, probe response, (re-)association request,(re-)association response, auth, disassociation and de-auth frames.The power value configured for a frame can be altered dynamically without any need for restart. The frame subtype is set as per standard IEEE conventions, for example to set the transmit power of beacon we use:
				`iwpriv wifi0 set_txpow_mgmt 0x80 8`
				`and to obtain the tx power set for beacon we use:`
				`iwpriv wifi0 get_txpow_mgmt 0x80`
				`wifi0 get_txpow_mgmt:8`
				The transmit power set for a particular frame type can be undone by setting the power to 255. The transmit power value is an 8 bit integer for both direct attach and offload radios.

antgain_2g
antgain_5g
	Iwpriv wifiX antgain_ 2g
	Iwpriv wifiX antgain_ 5g
		Y	Y	Set antenna gain for 2GHZ band and 5GHz band. The allowed values are between 0 and 30.
				Example:
				`iwpriv wifi0 antgain_2g 10`
				`Iwpriv wifi0 antgain_5g 10`

disp_tpc	Iwpriv wifi0 disp_tpc	Y	Y	Displays the tpc table. For each available transmission rate, it shows whether it supports TxBF/STBC/1-chain/2-chain/3-chain. In case of TxBF or STBC, it also shows the Tx power limit in txpower_txbf[] and txpower_ stbc[] respectively. Example: `iwpriv wifi0 disp_tpc 1`
get_minpower	Iwpriv ath0 get_ minpower	Y	Y	Get the minium tx power of radio.
get_maxpower	Iwpriv ath0 get_ maxpower	Y	Y	Get the max tx power of radio.

txstbc rxstbc
get_txstbc get_rxstbc
	iwpriv wifi*N* rxstbc *1/0*
	iwpriv wifi*N* txstbc *1/0*
		Y	N	Enables (1) or disables (0) the Space Time Coding Block (STBC) feature, as described in 802.11n specification, in the transmit (txstbc) or receive (rxtsbc) direction. The default value is 1. This option will have an effect only on chips supporting STBC. On other chips, this options will have no effect. Specific to 802.11n.
				`# iwpriv wifi0 txstbc 1`
				`# iwpriv wifi0 rxstbc 1`
				`# iwpriv wifi0 get_txstbc 1`
				`wifi0 get_txstbc:1`
				`# iwpriv wifi0 get_rxstbc 1`
				`wifi0 get_rxstbc:1`

promisc get_promisc	Iwpriv wifiN promisc 0|1 iworiv wifiN get_ promisc	N	Y	Enables or disables the promisc on device. Applicable only to QCA9980 `iwpriv wifi0 promisc 1`

1.3.13 Protection mechanism parameters

Table 1-17 Protection mechanism parameters

<table>
<tr><th>Parameter</th><th>Format</th><th>DA</th><th>OL</th><th colspan="2">Description</th></tr>
<tr><td rowspan="4"><b>protmode</b>
<b>get_protmode</b></td><td rowspan="4">iwpriv ath<i>N</i> protmode <i>{2|1|0}</i></td><td rowspan="4">Y</td><td rowspan="4">Y</td><td colspan="2">Enables or disables 802.11g protection mode. Causes RTS/CTS sequence (or CTS to self) to be sent when 802.11b devices are detected on the 802.11g network. Used to protect against Tx by devices that do not recognize OFDM modulated frames. The default is 0. The get parameter returns the current value.
<code>#iwpriv ath0 protmode 0</code>
<code>#iwpriv ath0 get_protmode</code>
<code>ath0 get_protmode:0</code></td></tr>
<tr><td>0</td><td>No protection</td></tr>
<tr><td>1</td><td>CTS to self</td></tr>
<tr><td>2</td><td>RTS/CTS</td></tr>
<tr><td rowspan="4"><b>extprotmode</b>
<b>get_ extprotmode</b></td><td rowspan="4">iwpriv ath<i>N</i> extprotmode <i>protectionmode</i></td><td rowspan="4">Y</td><td rowspan="4">Y</td><td colspan="2">Sets the protection mode used on the extension (secondary) channel when using 40 MHz channels. The default is 0. The get parameter returns the current value. Not applicable for OL.
<code>#iwpriv ath0 extprotmode 0</code>
<code>#iwpriv ath0 get_extprotmode</code>
<code>ath0 get_extprotmode:0</code></td></tr>
<tr><td>0</td><td>None, no protection</td></tr>
<tr><td>1</td><td>CTS to self</td></tr>
<tr><td>2</td><td>RTS/CTS</td></tr>
</table>

1.3.14 Radio-related parameters

Table 1-18 Radio-related parameters

<table>
<tr><th>Parameter</th><th>Format</th><th colspan="2">Description</th></tr>
<tr><td><b>6MBAck</b>
<b>Get6MBack</b></td><td>iwpriv wifi<i>N</i> 6MBAck <i>1|0</i></td><td colspan="2">This command enables (1) or disables (0) the use of the 6 MBps (OFDM) data rate for ACK frames. If disabled, ACK frames will be sent at the CCK rate. The default value is 0. The get parameter returns the current value. Not applicable for OL.
#iwpriv wifi0 6MBAck 1
#iwpriv wifi0 Get6MBAck
   wifi0 Get6MBAck:1</td></tr>
<tr><td rowspan="4"><b>AddSWBbo</b>
<b>SWBcnRespT</b>
<b>DMABcnRespT</b>
<b>GetAddSWBbo</b>
<b>GetSWBcnRespT</b>
GetDMABcnRespT</td><td rowspan="4">iwpriv wifi<i>N</i> SWBcnRespT
iwpriv wifi<i>N</i> DMABcnRespT
iwpriv wifi<i>N</i> AddSWBb0</td><td colspan="2">Adjust the calculation of the ready time for the QoS queues to adjust the QoS queue performance for optimal timing. These parameters are used for experimental adjustment of queue performance. In the AP application they are not relevant, so they should not be modified. Their default value is 0. Each get parameter returns the current value for its parameter. Not applicable for OL.
#iwpriv wifi0 SWBcnRespT 1
#iwpriv wifi0 DMABcnRespT 2
#iwpriv wifi0 AddSWBbo 10
#iwpriv wifi0 GetSWBcnRespT
   wifi0 GetSWBcnRespT:1
#iwpriv wifi0 GetDMABcnRespT
   wifi0 GetDMABcnRespT:2
#iwpriv wifi0 GetAddSWBbo
   wifi0 GetAddSWBbo:10</td></tr>
<tr><td>SWBcnRespT</td><td>Software beacon response time represents the time, in ms, required to process beacons in software</td></tr>
<tr><td>DMABcnRespT</td><td>DMA beacon response time, the time required to transfer a beacon message from memory to the MAC queue</td></tr>
<tr><td>AddSWBb0</td><td>Additional software beacon back-off is an estimated variable for final adjustment of the ready time offset</td></tr>
<tr><td rowspan="4"><b>AggrProt</b>
<b>AggrProtDur</b>
<b>AggrProtMax</b>
<b>getAggrProt</b>
<b>getAggrProtDur</b>
<b>getAggrProtMax</b></td><td rowspan="4">iwpriv wifi<i>N</i> AggrProt <i>1|0</i>
iwpriv wifi<i>N</i> AggrProtDur <i>duration</i>
iwpriv wifi<i>N</i> AggrProtMax <i>size</i></td><td colspan="2">Enable RTS/CTS protection on aggregate frames and control the size of the frames receiving RTS/CTS protection. Typically used as a test commands to set a specific condition in the driver. Each get parameter returns the current value for its parameter. Not applicable for OL.
#iwpriv wifi0 AggrProt 1
#iwpriv wifi0 AggrProtDuration 8192
#iwpriv wifi0 AggrProtMax 8192
#iwpriv wifi0 getAggrProt
   wifi0 getAggrProt:1
#iwpriv wifi0 getAggrProtDur
   wifi0 getAggrProtDur:8192
#iwpriv wifi0 getAggrProtMax
   wifi0 getAggrProtMax:8192</td></tr>
<tr><td>AggrProt</td><td>Enables (1) or disables (0 = Default) this function.</td></tr>
<tr><td>AggrProtDur</td><td>Indicates the amount of time to add to the duration of the CTS period to allow for additional packet bursts before a new RTS/CTS is required. Default is 8192 ms.</td></tr>
<tr><td>AggrProtMax</td><td>Indicates the largest aggregate size to receive RTS/CTS protection. Default is 8192 bytes.</td></tr>
</table>

Table 1-18 Radio-related parameters (cont.)

<table>
<tr><th>Parameter</th><th>Format</th><th colspan="2">Description</th></tr>
<tr><td>ANIEna
GetANIEna</td><td>iwpriv wifi*N* ANIEna *0*|*1*</td><td colspan="2">Enables the automatic noise immunity (ANI) processing in both the driver and the baseband unit. ANI mitigates unpredictable noise spurs in Rx channels that are due to the host system the device is installed in. This feature was added for CardBus and PCIE devices sold in the retail market not pre-installed in host systems. Most AP implementations do not enable ANI, preferring to limit noise spurs by design. The get parameter returns the current value. Not applicable for OL.
`#iwpriv wifi0 ANIEna 1`
`#iwpriv wifi0 GetANIEna`
`   wifi0 GetANIEna:1`</td></tr>
<tr><td rowspan="3">**AntSwap**
DivtyCtl
GetAntSwap
GetDivtyCtl</td><td rowspan="3">iwpriv wifi*N* AntSwap *1*|*0*
iwpriv wifi*N* DivtyCtl *AntSel*</td><td colspan="2">Control antenna switching behavior. For 802.11n devices, these control which chains are used for Tx. For Legacy devices, used to determine if diversity switching is enabled or disabled. The get parameters return the current values. Not applicable for OL.
`#iwpriv wifi0 AntSwap 1`
`#iwpriv wifi0 DivtyCtl 2`
`#iwpriv wifi0 GetAntSwap`
`   wifi0 GetAntSwap:0`
`#iwpriv wifi0 GetDivtyCtl`
`   wifi0 GetDivtyCtl:0`</td></tr>
<tr><td>AntSwap</td><td>Indicates when antenna A and B are swapped from the usual configuration, causing antenna A to be used by chain 1 or 2, and antenna B by chain 0. Default is 0 (that is, antennas are not swapped; antenna A to chain 0 and antenna B to chain 1, 2).</td></tr>
<tr><td>DivtyCtl</td><td>Enables/disables antenna switching altogether. If set to antenna A (1) or antenna B (2), the Tx antenna will not change based on receive signal strength. If set to variable (0), the Tx antenna is selected based on received signal strength.</td></tr>
<tr><td>**BcnNoReset**
getBcnNoReset</td><td>iwpriv wifi*N* BcnNoReset *1*|*0*</td><td colspan="2">Controls a debug flag that will either reset the chip or not when a stuck beacon is detected. If enabled (1), the system will NOT reset the chip upon detecting a stuck beacon, but will dump several registers to the console. Additional debug messages will be output if enabled, also. The default value is 0. The get parameter returns the current value. Not applicable for OL.
`#iwpriv wifi0 BcnNoReset 1`
`#iwpriv wifi0 getBcnNoReset`
`   wifi0 getBcnNoReset:1`</td></tr>
<tr><td>**CABlevel**
getCABlevel</td><td>iwpriv wifi*N* CABlevel *%Multicast*</td><td colspan="2">Sets the amount of space that can be used by Multi-cast traffic in the content after beacon (CAB) queue. CAB frames are also called beacon gated traffic frames and are sent attached to every beacon. In certain situations, so much multi-cast traffic may be transmitted that no time is left to send management or best effort (BE) traffic. TCP traffic gets starved out in these situations. This parameter controls how much of the CAB queue can be used by Multi-cast traffic, freeing the remainder for BE traffic. The default value of this parameter is 80 (80% Multi-cast). The get parameter returns the current value. Not applicable for OL.
`#iwpriv wifi0 CABlevel 50`
`#iwpriv wifi0 getCABlevel`
`   wifi0 getCABlevel:50`</td></tr>
</table>

Table 1-18 Radio-related parameters (cont.)

Parameter	Format	Description

CCKTrgLow
CCKTrgHi
GetCCKTrgLow
GetCCKTrgHi
	iwpriv wifi*N* CCKTrgLow *Low Threshold*
	iwpriv wifi*N* CCKTrgHi *High Threshold*
		Not applicable for OL architecture. Controls the CCK PHY errors/second threshold settings for the ANI immunity levels. A PHY error rate below the low trigger causes the ANI algorithm to lower immunity thresholds, and a PHY error rate exceeding the high threshold causes immunity thresholds to increase. When a limit is exceed, the ANI algorithm modifies one of several baseband settings to either increase or decrease sensitivity. Thresholds are increased/decreased in this order:
		Increase
		■ Raise the noise immunity level to MAX from 0, if the spur immunity level is at MAX
		■ Raise the noise immunity level to next level from a non-zero value
		■ Raise spur immunity level
		■ (If using CCK rates) raise the CCK weak signal threshold and raise the FIR step level
		■ Disable the ANI PHY Err processing to reduce CPU load
		Decrease:
		■ Lower the noise immunity level
		■ Lower the FIR step level
		■ Lower the CCK weak signal threshold
		■ Lower the spur immunity level
		The default values for these settings are 200 errors/second for the high threshold, and 100 errors/second for the low threshold.
		The get parameters return the current values.
		`#iwpriv wifi0 CCKTrgLow 80`
		`#iwpriv wifi0 CCKTrgHi 220`
		`#iwpriv wifi0 GetCCKTrgLow`
		`   wifi0 GetCCKTrgLow:100`
		`#iwpriv wifi0 GetCCKTrgHi`
		`   wifi0 GetCCKTrgHi:200`

CCKWeakThr
GetCCKWeakThr
	iwpriv wifi*N* CCKWeakThr *1|0*	Not applicable for OL architecture. Selects either normal (0) or weak (1) CCK signal detection thresholds in the baseband; used to toggle between a more sensitive threshold and a less sensitive one, as part of the ANI algorithm. The actual settings are set at the factory and are stored in EEPROM. If ANI is enabled, this parameter may be changed independent of operator setting, so this command may be overridden during operation. The default value for this parameter is 0. The get parameter returns the current value.
		`#iwpriv wifi0 CCKWeakThr 1`
		`#iwpriv wifi0 GetCCKWeakThr`
		`   wifi0 GetCCKWeakThr:1`

Table 1-18 Radio-related parameters (cont.)

Parameter	Format	Description

chanbw
get_chanbw
	iwpriv ath*N* chanbw *channel bandwidth*	Sets manual channel bandwidth. The values indicate which channel bandwidth to use.
		NOTE: This command only applies to legacy rates; HT rates are controlled with the corresponding 802.11n commands.
		The default value is 0. The get parameter returns the current value.
		`#iwpriv ath0 chanbw 1`
		`#iwpriv ath0 get_chanbw`
		`   ath0 get_chanbw:1`
		Value – Description
		0 – Full channel bandwidth
		1 – Half channel bandwidth
		2 – Quarter channel bandwidth

CWMIgnExCCA
GetCWMIgnExCCA
	iwpriv wifi*N* CWMIgnExCCA *1|0*	Not applicable for OL architecture. Allows the system to ignore CCA on the extension channel for 802.11n devices operating in HT40 mode. Normally, to transmit, the device requires no energy detected on both the control and extension channels for a minimum of PIFS duration. This control allows for ignoring energy on the extension channel, is not in conformance with the latest draft of the 802.11n specifications, and should only be used in test mode. The default value is 0 (do not ignore extension channel CCA). The get parameter returns the current value.
		`#iwpriv wifi0 CWMIgnExCCA 1`
		`#iwpriv wifi0 GetCWMIgnExCCA`
		`   wifi0 GetCWMIgnExCCA:0`

extbusythres
g_extbusythres
	`iwpriv ath`*N* `extbusythres` *pctBusy*	Not applicable for OL architecture. Used as part of the channel width management state machine. This threshold is used to determine when to command the channel back down to HT20 mode when operating at HT40 mode. If the extension channel is busy more often then the specified threshold (in percent of total time), then CWM will shut down the extension channel and set the channel width to HT20. The default value is 30%. The get parameter returns the current value.
		`#iwpriv ath0 extbusythres 50`
		`#iwpriv ath0 g_extbusythres`
		`   ath0 g_extbusythres:50`

FIRStepLvl
GetFIRStepLvl
	iwpriv wifi*N* FIRStepLvl *level*	Not applicable for OL architecture. Adjusts the FIR filter parameter that determines when a signal is in band for weak signal detection. Raising this level reduces the likelihood of adjacent channel interference causing a large number of (low RSSI) PHY errors; lowering the level allows easier weak signal detection for extended range. It is also modified by the ANI algorithm, so it may change during operation, usually in steps of single units. The default value for this parameter is 0. The get parameter returns the initialization (starting) value and not the value currently in the operating registers.
		`#iwpriv wifi0 FIRStepLvl 1`
		`#iwpriv wifi0 GetFIRStepLvl`
		`   wifi0 GetFIRStepLvl:0`

Table 1-18 Radio-related parameters (cont.)

<table>
<tr><th>Parameter</th><th>Format</th><th colspan="2">Description</th></tr>
<tr><td rowspan="3"><b>ForceBias</b>
<b>ForBiasAuto</b>
<b>GetForceBias</b>
<b>GetForBiasAuto</b></td><td rowspan="3">iwpriv wifi<i>N</i> ForBiasAuto <i>1|0</i>
iwpriv wifi<i>N</i> ForceBias <i>Bias</i></td><td colspan="2">Not applicable for OL architecture. This command activates the force bias feature; used as a workaround to a directional sensitivity issue in the AR5133 PHY chip in 2.4 GHz bands. The get parameters return the current values.
<code>#iwpriv wifi0 ForBiasAuto 1</code>
<code>#iwpriv wifi0 ForceBias 2</code>
<code>#iwpriv wifi0 GetForBiasAuto</code>
<code>   wifi0 GetForBiasAuto:0</code>
<code>#iwpriv wifi0 GetForceBias</code>
<code>   wifi0 GetForceBias:1</code></td></tr>
<tr><td>ForBiasAuto</td><td>Automatically selects the bias level depending on the selected frequency.</td></tr>
<tr><td>ForceBias</td><td>Sets the bias to a value between 0 and 7. These commands are only available when the driver is compiled with the #define ATH_FORCE_BIAS parameter defined. Even when this switch is enabled, the default values for both parameters are 0 (disabled); they should only be enabled if the sensitivity issue is actually present.</td></tr>
<tr><td><b>getchaninfo</b></td><td colspan="3">Used by external applications to get channel information from the driver. An example application is the wlanconfig tool that uses this interface to get the channel information. The wireless tools do not know how to parse the information provided, since it is returned in an Atheros driver specific data structure. This command has no command line equivalent interface. The data structures used are defined as:
<code>struct ieee80211req_chaninfo {</code>
<code>u_intic_nchans;</code>
<code>struct ieee80211_channel ic_chans[IEEE80211_CHAN_MAX];</code>
<code>};</code>
<code>struct ieee80211_channel {</code>
<code>u_int16_t ic_freq;   /* setting in MHz */</code>
<code>u_int32_t ic_flags;   /* see below */</code>
<code>u_int8_t ic_flagext;   /* see below */</code>
<code>u_int8_t ic_ieee;   /* IEEE channel number */</code>
<code>int8_t   ic_maxregpower; /* max. regulatory Tx power in dBm */</code>
<code>int8_t   ic_maxpower;   /* max. Tx power in dBm */</code>
<code>int8_t   ic_minpower;   /* min. Tx power in dBm */</code>
<code>};</code></td></tr>
<tr><td><b>HTEna</b>
<b>GetHTEna</b></td><td>iwpriv wifi<i>N</i> HTEna <i>1|0</i></td><td colspan="2">Not applicable for OL architecture. Enables (1) or disables (0) 802.11n (HT) data rates. Normally, only used as a test command. The parameter is set to 1 (enabled) by default. The get parameter returns the current value.
<code>#iwpriv wifi0 HTEna 1</code>
<code>#iwpriv wifi0 GetHTEna</code>
<code>   wifi0 GetHTEna:1</code></td></tr>
<tr><td><b>mcast_rate</b>
<b>get_mcast_rate</b></td><td>iwpriv ath<i>N</i> mcast_ rate <i>rate</i></td><td colspan="2">Sets multi-cast to a fixed rate. The rate value is specified in units of kilobits per second (kbps). This allows the user to limit the impact of multi-cast on the overall performance of the system. Default is 11 Mbps in 2.4 GHz mode and 6 Mbps in 5 GHz mode. The get parameter returns the current value. For 5 GHz OFDM rates should be used.
<code>#iwpriv ath0 mcast_rate 12000</code>
<code>#iwpriv ath0 get_mcast_rate</code>
<code>   ath0 get_mcast_rate: 12000</code></td></tr>
</table>

Table 1-18 Radio-related parameters (cont.)

Parameter	Format	Description

NoiseImmLvl
GetNoiseImmLvl
	iwpriv wifi*N* NoiseImmLvl *level*	Not applicable for OL architecture. Selects a specific noise immunity level parameter during initialization. This command only has effect prior to creating a specific HAL instance and should be used only during system initialization. Each noise immunity level corresponds to a set of baseband parameters that adjust baseband receiver sensitivity. Values are set at the factory and selected as a set by this parameter. The level is also controlled by the ANI algorithm, so initial immunity level is modified during operation to select the optimal level for current conditions. The default is 4 and should not be changed without a specific reason. The get parameter returns the current value.
		`#iwpriv wifi0 NoiseImmLvl 3`
		`#iwpriv wifi0 GetNoiseImmLvl`
		`   wifi0 GetNoiseImmLvl:4`

OFDMTrgLow
OFDMTrgHi
GetOFDMTrgLow
GetOFDMTrgHi
	iwpriv wifi*N* OFDMTrgLow *Low Threshold*
	iwpriv wifi*N* OFDMTrgHi *High Threshold*
		Not applicable for OL architecture. Controls the OFDM PHY errors/second threshold settings for the ANI immunity levels. A PHY error rate below the low trigger causes the ANI algorithm to lower immunity thresholds, and a PHY error rate exceeding the high threshold increases immunity thresholds. When a limit is exceed, the ANI algorithm modifies one of several baseband settings to either increase or decrease sensitivity in this order:
		Increase:
		■ Raise the noise immunity level to MAX from 0, if the spur immunity level is at MAX
		■ Raise the noise immunity level to next level from a non-zero value
		■ Raise spur immunity level
		■ (If using CCK rates) raise the CCK weak signal threshold and raise the FIR step level
		■ Disable the ANI PHY Err processing to reduce CPU load
		Decrease:
		■ Lower the noise immunity level
		■ Lower the FIR step level
		■ Lower the CCK weak signal threshold
		■ Lower the spur immunity level OFDM weak signal detection on, with the existing spur immunity level 0
		The default values for these settings are 500 errors/second for the high threshold, and 200 errors/second for the low threshold. The get parameters return the current values.
		`#iwpriv wifi0 OFDMTrgLow 100`
		`#iwpriv wifi0 OFDMTrgHi 550`
		`#iwpriv wifi0 GetOFDMTrgLow`
		`   wifi0 GetOFDMTrgLow:200`
		`#iwpriv wifi0 GetOFDMTrgHi`
		`   wifi0 GetOFDMTrgHi:500`

Table 1-18 Radio-related parameters (cont.)

Parameter	Format	Description

OFDMWeakDet
GetOFDMWeakDet
	iwpriv wifi*N* OFDMWeakDet *1|0*	Not applicable for OL architecture. Selects normal (0) or weak (1) OFDM signal detection thresholds in the baseband register. The actual thresholds are factory set and are loaded in the EEPROM. This parameter corresponds to the initialization value for the ANI algorithm, and is only valid prior to system startup. The default value for this parameter is 1 (detect weak signals). The get parameter returns the initialization value only.
		`#iwpriv wifi0 OFDMWeakDet 0`
		`#iwpriv wifi0 GetOFDMWeakDet`
		`   wifi0 GetOFDMWeakDet:1`

RSSIThrLow
RSSIThrHi
GetRSSIThrLow
GetRSSIThrHi
	iwpriv wifi*N* RSSIThrLow *far threshold*
	iwpriv wifi*N* RSSIThrHi *near threshold*
		Not applicable for OL architecture. Determines the relative distance of the AP from the STA; used to determine how the ANI immunity levels are selected.
		■ If the average beacon RSSI of beacons from the AP > RSSIThrHi, the STA is determined to be at close-range
		■ If < RSSIThrHi but >RSSIThrLow, the STA is mid-range
		■ If <RSSIThrLow, the STA is long-range
		■ Defaults are 40 for the high (near) threshold and 7 for low (far).
		The get parameters return the current values.
		`#iwpriv wifi0 RSSIThrLow 6`
		`#iwpriv wifi0 RSSIThrHi 45`
		`#iwpriv wifi0 GetRSSIThrLow`
		`   wifi0 GetRSSIThrLow:7`
		`#iwpriv wifi0 GetRSSIThrHi`
		`   wifi0 GetRSSIThrHi:40`

set11NRates
get11NRates
	`iwpriv ath`*N* `Set11NRates` *rate_series*	When performing tests at fixed data rates, specifies the data rate. *rate_series* is specified as a group of 4 bytes in a 32-bit word. Each byte represents the MCS rate to use for each of 4 rate fallbacks. If hardware does not receive an ACK when transmitting at the first rate, it falls back to the second rate and retry, etc. through the fourth rate. As a convention, the high bit in the rate byte is always set, so for a rate of MCS-15 the rate value would be 0x8F. This command has a corresponding get parameter. It has no default value
		`#iwpriv ath0 set11NRates 0x8F8F8C8C`
		`#iwpriv ath0 get11NRates`
		`   ath0 get11NRates: 2408549516`

set11NRetries
get11NRetries
	`iwpriv ath`*N* set11NRetries *RetryCountPerStep*	For each rate in the rate series, the hardware can retry the same rate step multiple times. This value sets the number of retries for each step in the rate series. This is expressed as a group of 4 bytes in a 32-bit word, with each byte indicating the number of times to retry the rate step. Has a corresponding get parameter, and no default value.
		`#iwpriv ath0 set11NRetries 0x01010404`
		`#iwpriv ath0 get11NRetries`
		`   ath0 get11NRetries: 16843780`

setchanlist **getchanlist**	Used by an application to set the channel list manually. Channels that are not valid from a regulatory perspective will be ignored. This command is passed a byte array 255 bytes long that contains the list of channels required. A value of 0 indicates no channel, but all 255 bytes must be provided. getchanlist receives this array from the driver in a 255 byte array that contains the valid channel list. The response is a binary array that WLAN tools cannot parse; therefore this cannot be used on the command line.	

Table 1-18 Radio-related parameters (cont.)

Parameter	Format	Description

SpurImmLvl
GetSpurImmLvl
	iwpriv wifi*N* SpurImmLvl *level*	Not applicable for OL architecture. Sets the spur immunity level corresponding to the baseband parameter (**cyc_pwr_thr1**) that determines the minimum cyclic RSSI causing OFDM weak signal detection. Raising this level reduces the number of OFDM PHY errors/second (caused due to board spurs, or interferences with OFDM symbol periodicity). Lowering it allows detection of weaker OFDM signals (extending range). Note this value is the initialization, not the operating value. Default is 2. The get parameter returns the current value.
		`#iwpriv wifi0 SpurImmLvl 3`
		`#iwpriv wifi0 GetSpurImmLvl`
		`wifi0 GetSpurImmLvl:2`

g_chanstats_th chanstats_th	iwpriv wifiX g_ chanstats_th iwpriv wifiX chanstats_th <NEW_ THRESOLD>	Periodic channel stats are sent via net link broadcast events if obss channel utilization crosses chan_stats_th which is by default set as 40 percent. Chan stats threshold (chan_stats_th) can be read any time by issuing iwpriv command “iwpriv wifiX g_chanstats_th” and can be set by issuing command “iwpriv wifiX chanstats_th <NEW_THRESOLD>”.

pas_scanen
g_pas_scanen
	iwpriv wifiX pas_ scanen 1/0
	iwpriv wifiX g_pas_ scanen
		Enables strict passive scan in passive channels i.e.
		Station silently listens for beacon in passive channels without sending probe requests during channel scan
		`#iwpriv wifi1 pas_scanen 1 (Enables strict passive scan)`
		`#iwpriv wifi1 pas_scanen 0 (Disables strict passive scan)`

1.3.15 Radio resource management (802.11k)

The Radio Resource Management (RRM) functionality constitutes a partial implementation of the 802.11k specification. In this implementation, the AP attempts to gain information of the surrounding environment from the connected client by sending various messages to it and then receiving responses.

NOTE The 802.11k functions requires **wifitool** for configuration, after 802.11k functionality has been enabled with the **iwpriv rrm** command.

Table 1-19 Radio resource management (802.11k) parameters

Parameter	Format	Description	

quiet
get_quiet
	iwpriv ath*N* rrm	Enable (1) or disable (0) Radio Management Resource (RRM) and Quiet Period functions, which are part of the 802.11k specification. The default quiet period parameters are used when this feature is turned on.*get_quiet* returns the current status.
		`#iwpriv ath0 quiet 1`
		`#iwpriv ath0 get_quiet`
		`   ath0 get_quiet:1`	

rrm
get_rrm
	iwpriv ath*N* rrm	Enable (1) or disable (0) Radio Management Resource (RRM) functions, which are part of the 802.11k specification. *get_rrm* returns the current status.
		`#iwpriv ath0 rrm 1`
		`#iwpriv ath0 get_rrm`
		`   ath0 get_rrm:1`	

sendtsmrpt	wifitool ath*N* sendtsmrpt *num_rpt rand_ivl meas_dur tid dstmac bin0-range trig_cond avg_err_thresh cons_err_thresh delay_thresh trig_timeout*	Transmits a stream report	
		dstmac	Destination MAC address
		num_rpt	Number of repetition
		rand_ivl	Random interval
		meas_dur	Measurement duration
		tid	Traffic Identifier field contains the TID subfield.
		peermacaddr	Peer STA Address contains a MAC address indicating the RA in the MSDUs to be measured
		bin0-range	Bin 0 Range indicates the delay range of the first bin (Bin 0) of the Transmit Delay Histogram, expressed in units of TUs.
		trig_cond	Triggered Reporting. Refer to the IEEE 802.11k specification for details.
		avg_err_thresh	Average error threshold. Refer to the IEEE 802.11k specification for details.
		cons_err_thresh	Consecutive Error Threshold. Refer to the IEEE 802.11k specification for details.
		delay_thresh	Delay Threshold. Refer to the IEEE 802.11k specification for details.
		trig_timeout	Trigger Time-out. Refer to the IEEE 802.11k specification for details.
sendneigrpt	wifitool ath*N* sendneigrpt *mac_addr ssid dialog_token*	Transmits a neighbor report	
		mac_addr	Destination MAC address
		ssid	SSID for which report is required
sendlmreq	wifitool ath*N* sendlmreq *mac_addr*	Transmits a link measurement report	
		mac_addr	Destination MAC address

Table 1-19 Radio resource management (802.11k) parameters (cont.)

Parameter	Format	Description		
sendbcnrpt	wifitool ath*N* sendbcnrpt *dstmac* *regclass* *channum* *rand_ivl* *duration* *mode* *req_ssid* *rep_cond* *rpt_detail* *req_ie* *chanrpt_mode*	dstmac	Destination MAC address.	
		regclass	Regulatory class.	
		channum	Channel number set to zero if report required for all possible channel on that band.	
		rand_ivl	Random interval, see 802.11k specification for details	
		duration	Measurement duration, refer to 802.11k specification for definition.	
		mode	Measurement mode.	
			0	passive
			1	active
			2	beacon table
		req_ssid	Sets SSID matching requirement. If enabled (1), only reports matching to QCA BSS will be generated by the station. Default value is disabled (0).	
		rep_cond	The beacon reporting Information sub-element indicates the condition for issuing a beacon report. Default value is zero. Refer to the 802.11k specification for details.	
		rpt_detail	The reporting detail contains a 1-octet reporting detail data field that defines the level of detail per AP to be reported to the requesting STA. Default value is zero. Refer to 802.11k specification for details.	
		req_ie	For current implementation, this should be set to zero	
		chanrpt_mode	Reporting condition for beacon report. See 802.11k specification for details.	

Table 1-19 Radio resource management (802.11k) parameters (cont.)

Parameter	Format	Description		
sendstastats	wifitool ath*N* sendstastats *mac_addr* *duration* *gid*	mac_addr	Destination MAC address	
		duration	Measurement duration.	
		gid	Group Identity.	
			0	STA counters from dot11CountersTable
			1	STA counters from dot11CountersTable
			2	QoS STA counters for UP0 from dot11QosCountersTable
			3	QoS STA counters for UP1 from dot11QosCountersTable
			4	QoS STA counters for UP2 from dot11QosCountersTable
			5	QoS STA counters for UP3 from dot11QosCountersTable
			6	QoS STA counters for UP4 from dot11QosCountersTable
			7	QoS STA counters for UP5 from dot11QosCountersTable
			8	QoS STA counters for UP6 from dot11QosCountersTable
			9	QoS STA counters for UP7 from dot11QosCountersTable
			10	BSS Average Access
			11-25	Reserved.
sendchload	wifitool ath*N* sendchload *dstmac* *n_rpts* *regclass* *chnum* *rand_ivl* *mandatory_duration* *optional_condtion* *condition_val*	Transmits a channel load report		
		mac_addr	Destination MAC address	
		n_rpts	Number of repetitions client should perform. Refer to 802.11k specification for details.	
		regclass	Regulatory class.	
		chnum	Channel number.	
		rand_ivl	Random interval. Refer to 802.11k specification for details.	
		mandatory_ duration	Measurement duration. Refer to 802.11k specification for definition.	
		optional_condtion	Se optional condition to (1) if desired as part of request. Default is (0).	
		condition_val	Condition value if optional condition is true. Refer to 802.11k specification for details.	

Table 1-19 Radio resource management (802.11k) parameters (cont.)

Parameter	Format	Description	
sendnhist	wifitool ath*N* sendnhist *dstmac* *n_rpts* *regclass* *chnum* *rand_ivl* *mandatory_duration* *optional_condtion* *condition_val*	Transmits a noise histogram report	
		mac_addr	Destination MAC address
		n_rpts	Number of repetitions client should perform. Refer to 802.11k specification for details.
		regclass	Regulatory class.
		chnum	Channel number.
		rand_ivl	Random interval. Refer to 802.11k specification for details.
		mandatory_ duration	Measurement duration. Refer to 802.11k specification for definition.
		optional_condtion	Set optional condition to (1) if desired as part of request. Default is (0).
		condition_val	Condition value if optional condition is true. Refer to 802.11k specification for details.
sendlcireq	wifitool ath*N* sendlcireq *dstmac* *location* *latitude_res* *longitude_res* *altitude_res* *azimuth_res* *optional_condtion* *condition_val*	Transmits a noise histogram report	
		dstmac	Destination MAC address
		location	Location of requesting/reporting station refer 802.11k specifications for details
		latitude_res	Number of most significant bits (max 34) for fixed-point value of latitude. Refer to 802.11k specifications for details.
		longitude_res	Number of most significant bits (max 34) for fixed-point value of longitude. Refer to 802.11k specification for details.
		altitude_res	Number of most significant bits (max 30) for fixed-point value of altitude. Refer to 802.11k specification for details.
		azimuth_res	Number of most significant bits (max 9) for fixed-point value of Azimuth. Refer to 802.11k specification for details.
		optional_condtion	Set optional condition to (1) if desired as part of request. Default is (0).
		condition_val	Specifies report of azimuth of radio reception (0) or front surface (1) of reporting station. Refer to 802.11k specification for details.
rrmstats	wifitool ath*N* rrmstats *(mac_addr)*	Gets an RRM report in user space.	
		mac_addr	Optionally specifies MAC address of client. If not given, command will print all RRM statistics collected up to the command for all connected clients.
bcnrpt	wifitool ath*N* bcnrpt	Gets a beacon report in user space. Will provide most information received in a beacon report.	

1.3.16 Regulatory parameters

These commands interface with the regulatory information in the driver, and are used to control the settings affecting local requirements.

Table 1-20 Regulatory parameters

Parameter	Format	Description
doth_pwrtgt **get_doth_pwrtgt**	iwpriv ath*N* doth_pwrtgt *target*	Sets the desired maximum power on the current channel, as reported in the beacon and probe response messages. Used by STAs to set required output values. The value is capped by the regulatory maximum power value (=255). For large inputs the LSB 7 bits are used as the desired maximum power. The power value target is expressed in 0.5 dBm steps. The parameter has no default value. The get parameter returns the current value. #iwpriv ath0 doth_pwrtgt 25 #iwpriv ath0 get_doth_pwrtgt ath0 get_doth_pwrtgt:25

1.3.17 Security parameters

The security-related parameters relate to the security subsystem, and are specific interfaces required by the hostapd and wpa_supplicant programs. Table 1-21 lists a subset of the configurable security parameters. Other parameters are passed to the driver by iwconfig (for WEP) and by hostapd/wpa_supplicant (for WPA).

Table 1-21 Security-related parameters

Parameter	Format	DA	OL	Description
authmode get_authmode	iwpriv ath*N* authmode *mode {open|shared|auto}*	Y	Y	Sets the authentication mode for WEP operation. Authentication mode can be set to open, shared or auto. In 'auto' mode, both shared and open mode clients are allowed to authenticate. Default mode is open.The get parameter returns the current *mode* value. The terms open, shared, and auto may be given as 1, 2, or 4 instead, respectively. Result is correct; ignore error message in console.

Table 1-21 Security-related parameters (cont.)

Parameter	Format	DA	OL	Description	

authmode
get_authmode
	`iwpriv athN authmode mode`			Selects the authentication mode to configure the driver for. This command is also used by host_apd to configure the driver when host_apd is used as an authenticator. The user will normally not use these commands. The default value is 1. The get parameter returns the current value.
				`#iwpriv ath0 authmode 2`
				`#iwpriv ath0 get_authmode`
				`   ath0 get_authmode:2`
				The mode values are:	

				Value	Mode
				0	None specified
				1	Open authentication
				2	Shared key (WEP) authentication
				3	802.1x authentication
				4	Auto select/accept authentication (used by host_apd)
				5	WPA PSK with 802.1x PSK

countermeasures
get_ countermeas
	`iwpriv athN countermeasures 1|0`			Enables/disables WPA/WPA2 countermeasures, which perform additional processing on incoming authentication requests to detect spoof attempts, such as repeating authentication packets. A value of 1 enables countermeasures, and 0 disables them. This command has a corresponding get parameter.
				`#iwpriv ath0 countermeasures 1`
				`#iwpriv ath0 get_countermeas`
				`   ath0 get_countermeas:1`	

Table 1-21 Security-related parameters (cont.)

<table>
<tr><th>Parameter</th><th>Format</th><th>DA</th><th>OL</th><th colspan="4">Description</th></tr>
<tr><td rowspan="15"><b>driver_caps</b>
<b>get_driver_caps</b></td><td rowspan="15">iwpriv athN driver_caps caps</td><td></td><td></td><td colspan="4">Manually sets the driver capabilities flags; normally used for testing, because the driver fills in the proper capability flags. has a corresponding get parameter. has no default value.
#iwpriv ath0 driver_caps 0x034000003
#iwpriv ath0 get_driver_caps
   ath0 get_driver_caps:872415235
The flags are defined as:</td></tr>
<tr><td></td><td></td><td>0x00000001</td><td>WEP</td><td>0x00004000</td><td>Short Slot Time</td></tr>
<tr><td></td><td></td><td>0x00000002</td><td>TKIP</td><td>0x00008000</td><td>Short Preamble</td></tr>
<tr><td></td><td></td><td>0x00000004</td><td>AES</td><td>0x00010000</td><td>Monitor Mode</td></tr>
<tr><td></td><td></td><td>0x00000008</td><td>AES_CCM</td><td>0x00020000</td><td>TKIP MIC</td></tr>
<tr><td></td><td></td><td>0x00000010</td><td>HT Rates</td><td>0x01000000</td><td>WPA 2</td></tr>
<tr><td></td><td></td><td>0x00000020</td><td>CKIP</td><td>0x00800000</td><td>WPA 1</td></tr>
<tr><td></td><td></td><td>0x00000040</td><td>Fast Frame</td><td>0x02000000</td><td>Burst</td></tr>
<tr><td></td><td></td><td>0x00000080</td><td>Turbo</td><td>0x04000000</td><td>WME</td></tr>
<tr><td></td><td></td><td>0x00000100</td><td>IBSS</td><td>0x08000000</td><td>WDS</td></tr>
<tr><td></td><td></td><td>0x00000200</td><td>Power Management</td><td>0x10000000</td><td>WME TKIP MIC</td></tr>
<tr><td></td><td></td><td>0x00000400</td><td>Host AP</td><td>0x20000000</td><td>Background Scan</td></tr>
<tr><td></td><td></td><td>0x00000800</td><td>Ad Hoc Demo</td><td>0x40000000</td><td>UAPSD</td></tr>
<tr><td></td><td></td><td>0x00001000</td><td>Software Retry</td><td rowspan="2">0x80000000</td><td rowspan="2">Fast Channel Change</td></tr>
<tr><td></td><td></td><td>0x00002000</td><td>Tx Power Mgmt</td></tr>
<tr><td><b>dropunencrypted</b>
<b>get_dropunencry</b></td><td>iwpriv athN dropunencrypted 0|1</td><td></td><td></td><td colspan="4">Enables/disables dropping the unencrypted non-PAE frames received. Passing a value of 1 enables dropping of unencrypted non-PAE frames, a value of 0 disables. This command has a corresponding get parameter, and its default value is zero.
#iwpriv ath0 dropunencrypted 1
#iwpriv ath0 get_dropunencry
   ath0 get_dropunencry:1</td></tr>
<tr><td rowspan="6"><b>keymgtalgs</b>
<b>get_keymgtalgs</b></td><td rowspan="6">iwpriv athN keymgtalgs algs</td><td rowspan="6"></td><td rowspan="6"></td><td colspan="4">Used by host_apd to manage WPA keys (essentially the same as the WPA command). Has a corresponding get parameter.
#iwpriv ath0 keymgtalgs 3
#iwpriv ath0 get_keymgtalgs
   ath0 get_keymgtalgs:3
The algorithms supported are:</td></tr>
<tr><td><b>Value</b></td><td colspan="3"><b>Algorithm</b></td></tr>
<tr><td>0</td><td colspan="3">WPA_ASE_NONE</td></tr>
<tr><td>1</td><td colspan="3">WPA_ASE_8021X_UNSPEC</td></tr>
<tr><td>2</td><td colspan="3">WPA_ASE_8021X_PSK</td></tr>
<tr><td>3</td><td colspan="3">The command combines the supported algorithms, so a value of 3 indicates both unspecified and PSK support</td></tr>
</table>

Table 1-21 Security-related parameters (cont.)

<table>
<tr><th>Parameter</th><th>Format</th><th>DA</th><th>OL</th><th colspan="2">Description</th></tr>
<tr><td><b>mcastkeylen</b>
<b>get_mcastkeylen</b></td><td>iwpriv ath<i>N</i> mcastkeylen <i>length</i></td><td>Y</td><td>Y</td><td colspan="2">Only valid for WEP operations; sets the multicast/group key length of the WEP key. Key lengths of 5 (40 bits) or 13 (104 bits) are the only valid values, corresponding to 64 or 128 bit WEP encoding. Has no default value; has a corresponding get parameter.
<code>#iwpriv ath0 mcastkeylen 5</code>
<code>#iwpriv ath0 get_mcastkeylen</code>
<code>   ath0 get_mcastkeylen:5</code></td></tr>
<tr><td><b>privacy</b>
<b>get_privacy</b></td><td><code>iwpriv athN privacy 1|0</code></td><td></td><td></td><td colspan="2">Flag used to indicate WEP operations; not normally used by an application other than host_apd. WEP operations are normally configured through the appropriate iwconfig command. Has a corresponding get parameter, and its default value is 0.
<code>#iwpriv ath0 privacy 1</code>
<code>#iwpriv ath0 get_privacy</code>
<code>   ath0 get_privacy:1</code></td></tr>
<tr><td><b>rsncaps</b>
<b>get_rsncaps</b></td><td><code>iwpriv athN rsncaps flags</code></td><td></td><td></td><td colspan="2">Sets the RSN capabilities flags. The only valid capability flag is 0x01, RSN_CAP_PREAUTH, which configures the AP for pre-authorization functionality. Normally used only by host_apd when configuring the VAP. Has a corresponding get parameter.
<code>#iwpriv ath0 rsncaps 0x01</code>
<code>#iwpriv ath0 get_rsncaps</code>
<code>   ath0 get_rsncaps:1</code></td></tr>
<tr><td rowspan="11"><b>setfilter</b></td><td rowspan="11"><code>iwpriv athN setfilter filter</code></td><td></td><td></td><td colspan="2">Allows applications to specify the management frames it wants to receive from the VAP, causing the VAP to forward indicated frames to the networking stack. Normally used by host_apd to configure the VAP; has no corresponding get parameter.
<code>#iwpriv ath0 setfilter 0x08</code></td></tr>
<tr><td></td><td></td><td><b>Value</b></td><td>Algorithm</td></tr>
<tr><td></td><td></td><td>0x01</td><td>Beacon</td></tr>
<tr><td></td><td></td><td>0x02</td><td>Probe request</td></tr>
<tr><td></td><td></td><td>0x04</td><td>Probe response</td></tr>
<tr><td></td><td></td><td>0x08</td><td>Association request</td></tr>
<tr><td></td><td></td><td>0x10</td><td>Association response</td></tr>
<tr><td></td><td></td><td>0x20</td><td>Authentication</td></tr>
<tr><td></td><td></td><td>0x40</td><td>De-authentication</td></tr>
<tr><td></td><td></td><td>0x80</td><td>Disassociation</td></tr>
<tr><td></td><td></td><td>0xFF</td><td>ALL</td></tr>
<tr><td><b>setiebuf</b>
<b>getiebuf</b></td><td colspan="5">Used by an application to set/get application information elements into/from various frame types. The structure eee80211req_getset_appiebuf is passed as an argument to the IOCTL. These commands have no command line equivalent, but the command does show up as a valid iwpriv command. The definition of the required data structure is:
<code>struct ieee80211req_getset_appiebuf {</code>
<code>u_int32_t app_frmtype; /*mgmt frame type for which buffer is added */</code>
<code>u_int32_t app_buflen;   /*application supplied buffer length */</code>
<code>u_int8_t app_buf[];</code>
<code>};</code></td></tr>
</table>

Table 1-21 Security-related parameters (cont.)

Parameter	Format	DA	OL	Description

setkey
delkey
	`host_apd setkey`			The host_apd application must do periodic rekeying of the various connections. These commands allow for management of the key cache. The setkey command receives the argument ieee80211req_key structure. Neither command has any corresponding command line equivalents. This structure is:
				`struct ieee80211req_key {`
				`u_int8_t  ik_type;   /* key/cipher type */`
				`u_int8_t  ik_pad;`
				`u_int16_t ik_keyix;  /* key index */`
				`u_int8_t  ik_keylen; /* key length in bytes */`
				`u_int8_t  ik_flags;`
				`u_int8_t  ik_macaddr[IEEE80211_ADDR_LEN];`
				`u_int64_t ik_keyrsc; /* key Rx sequence counter */`
				`u_int64_t ik_keytsc; /* key Tx sequence counter */`
				`u_int8_tik_keydata[IEEE80211_KEYBUF_SIZE+IEEE80211_MICBUF_SIZE];`
				`};`

	`delkey`			Passes the structure ieee80211req_del_key:
				`struct ieee80211req_del_key {`
				`u_int8_t  idk_keyix;  /* key index */`
				`u_int8_t  idk_macaddr[IEEE80211_ADDR_LEN];`
				`};`

setmlme	Another of the host_apd support commands, this command performs direct access to the MLME layer in the driver, thus allowing an application to start or terminate a specific association. Note that the MLME_ASSOC sub command only makes sense for a STA (the AP will not start an association).
	This command has no command line equivalent. It passes the ieee80211req_mlme structure:
	`struct ieee80211req_mlme {`
	`   u_int8_tim_op;     /* operation to perform */`
	`defineIEEE80211_MLME_ASSOC1/* associate STA */`
	`defineIEEE80211_MLME_DISASSOC2/* disassociate STA */`
	`defineIEEE80211_MLME_DEAUTH3/* deauthenticate STA */`
	`defineIEEE80211_MLME_AUTHORIZE4/* authorize STA */`
	`defineIEEE80211_MLME_UNAUTHORIZE5/* unauthorize STA */`
	`   u_int16_tim_reason;/* 802.11 reason code */`
	`   u_int8_tim_macaddr[IEEE80211_ADDR_LEN];`
	`};`			

ucastcipher
get_uciphers
	`iwpriv athN ucastcipher`			Used mainly by the host_apd authenticator, and sets the unicast cipher type to the indicated value. See the **mcastcipher** command for the definition of the values. There is no default value. The get parameter returns the current value.
				`#iwpriv ath0 ucastcipher 2`
				`#iwpriv ath0 get_uciphers`
				`   ath0 get_uciphers:2`

ucastciphers **get_ucastciphers**	`iwpriv athN ucastciphers cipher_types`			Set support for cipher types. The values are preserved here to maintain binary compatibility with applications such as **wpa_supplicant** and **hostapd**. The default value is 7.

Table 1-21 Security-related parameters (cont.)

Parameter	Format	DA	OL	Description	

ucastkeylen
get_ucastkeylen
	`iwpriv athN ucastkeylen length`			Only valid for WEP operations. This command is used to set the key length of the WEP key for unicast frames. Key lengths of 5 (40 bits) or 13 (104 bits) are the only valid values, corresponding to 64 or 128 bit WEP encoding, respectively. Has no default value. The get parameter returns the current value.
				`#iwpriv ath0 ucastkeylen 5`
				`#iwpriv ath0 get_ucastkeylen`
				`   ath0 get_ucastkeylen:5`	

wpa
get_wpa
	`iwpriv athN wpa WPA Mode`			Sets the desired WPA modes. Typically overridden by the setting in the hostapd configuration file, which uses the same interface to set the WPA mode. Thus, this command is not normally used during configuration. The default value is 0. The get parameter returns the current value.
				`#iwpriv ath0 wpa 3`
				`#iwpriv ath0 get_wpa`
				`   ath0 get_wpa:0`
				The value of WPA Mode indicates the level of support:	

				0	No WPA support
				1	WPA support
				2	WPA2 support
				3	Both WPA and WPA2 support

wps
get_wps
	`iwpriv athN wps WPS Mode`			Sets the desired WPS mode. The default is 0. The get parameter returns the current value.
				`#iwpriv ath0 wps 0`
				`#iwpriv ath0 get_wps`
				`   ath0 get_wps:0`	

				0	Disable WPS mode.
				>=1	Enable WPS mode.

1.3.18 STA parameters

Table 1-22 STA parameters

Parameter	Format	Description	
autoassoc **get_autoassoc**	*iwpriv* `ath`*N* *autoassoc 1|0*	Sets the auto-association mode. Default is 0.	

eospdrop
get_eospdrop
	iwpriv `ath`*N*
	eospdrop *1|0*
		Sets support for forcing uapsd EOSP drop (AP only). The get parameter returns the current value.
		`#iwpriv ath0 eospdrop 0`
		`#iwpriv ath0 get_eospdrop`
		`   ath0 get_eospdrop:0`	

		0	Disable forcing uapsd EOSP drop
		1	Enable forcing uapsd EOSP drop

Table 1-22 STA parameters (cont.)

Parameter	Format	Description	

periodicScan
g_periodicScan
	iwpriv athN *periodicScan enable and set*	Sets STA periodic scan support. 0 is disable and other values are enable. If the value is less than 30000, it will be set to 30000. The get parameter returns the current value.
		`#iwpriv ath0 periodicScan 0`
		`#iwpriv ath0 g_periodicScan`
		`   ath0 g_periodicScan:0`	

		0	Disable periodic scan
		>0	Enable periodic scan and set periodic scan period
powersave **get_powersave**	iwpriv athN powersave *powersave mode*	Sets support for the STA power save mode. The default is 0. The get parameter returns the current value.	
		0	STA power save none
		1	STA power save low
		2	STA power save normal
		3	STA power save maximum

1.3.19 Turbo parameters

Table 1-23 Turbo parameters

Parameter	Format	Description	

burst
get_burst
	iwpriv athN *burst 1|0*	Enables (1) or disables (0) Atheros super AG bursting support in the driver. Passing a value of 1 to the driver enables Super G bursting. Passing a value of 0 to the driver disables Super A/G bursting; not normally used when using 802.11n devices. The default value is 0. The get parameter returns the current value.
		`#iwpriv ath0 burst 0`
		`#iwpriv ath0 get_burst`
		`   ath0 get_burst:0`	

compression
get_compression
	iwpriv athN *compression 1|0*	Enables/disables Data compression support Atheros supper G The get parameter returns the current value. Not valid for partial offload.
		`#iwpriv ath0 compression 0`
		`#iwpriv ath0 get_compression`
		`   ath0 get_compression:0`	

		0	Disable
		1	Enable

ff
get_ff
	iwpriv athN *ff 1|0*	Enables/disables fast frames support of Atheros supper G. The get parameter returns the current value. Not valid for partial offload.
		`#iwpriv ath0 ff 0`
		`#iwpriv ath0 get_ff`
		`   ath0 get_ff:0`	

		0	Disable
		1	Enable

Table 1-23 Turbo parameters (cont.)

Parameter	Format	Description	

periodicScan
get_periodicScan
	iwpriv athN *periodicScan enable_and_set*	Sets STA periodic scan support. 0 is disable and other values are enable. If the value is less than 30000, it will be set to 30000. The get parameter returns the current value.
		`#iwpriv ath0 periodicScan 0`
		`#iwpriv ath0 get_periodicScan`
		`ath0 get_periodicScan:0`	

		0	Disable periodic scan
		>0	Enable periodic scan and set periodic scan period

1.3.20 Tx beamforming parameters

The 802.11ac standard transmit beam forming (TxBF) features are available. Tx beam forming parameters must be set before association with the station.

Table 1-24 Tx beamforming parameters

Parameter	Format	Description	
Vhtsubfer	iwpriv ath***N*** vhtsubfer *{0|1}*	Single-user beam former	
		0	Disable single-user beam former
		1	Enable single-user beam former
Vhtsubfee	iwpriv ath***N*** vhtsubfee *{0|1}*	Single-user beam formee	
		0	Disable single-user beam formee
		1	Enable single-user beam formee
Vhtmubfer	iwpriv ath***N*** vhtmubfer *{0|1}*	Multiple-user beam former	
		0	Disable multiple-user beam former
		1	Enable multiple-user beam former
Vhtmubfee	iwpriv ath***N*** vhtmubfee *{0|1}*	Multiple-user beam formee	
		0	Disable multiple-user beam formee
		1	Enable multiple-user beam formee

1.3.20.1 TxBF configuration

Following are the recommended sequences for setting parameters on the AP/STA:

Beamformer (AP):

```
sudo iwpriv ath0 vhtsubfer 1
sudo iwpriv ath0 vhtsubfee 0
sudo iwpriv ath0 vhtmubfer 0
sudo iwpriv ath0 vhtmubfee 0
sudo iwpriv ath0 implicitbf 0
```

Beamformee (STA):

```
sudo iwpriv ath0 vhtsubfer 0
sudo iwpriv ath0 vhtsubfee 1
sudo iwpriv ath0 vhtmubfer 0
sudo iwpriv ath0 vhtmubfee 0
sudo iwpriv ath0 implicitbf 0
```

1.3.20.2 TxBF statistics

You can accesses TxBF statistics by using the **iwpriv** command **txrx_fw_stats** with parameters, 1to 19. Following are examples for accessing TxBF statistics:

Section 1.3.21.7

Section 1.3.21.10

1.3.21 Firmware Statst

1.3.21.1 Target physical device stats

NOTE The physical device target device stats shows the number of times various expected and unexpected transmit and receive events have happened.

Output (STA)

```
[ 1277.221252] ### Tx ###
[ 1277.221256] comp_queued       :      128
[ 1277.221258] comp_delivered    :      180
[ 1277.221261] msdu_enqued       :      12218
[ 1277.221264] wmm_drop          :      0
[ 1277.221267] local_enqued      :      12218
[ 1277.221269] local_freed       :      12218
[ 1277.221273] hw_queued         :      12457
[ 1277.221276] hw_reaped         :      12457
[ 1277.221278] mac underrun      :      0
[ 1277.221282] phy underrun      :      0
[ 1277.221284] hw_paused         :      8
[ 1277.221287] seq_posted        :      12221
[ 1277.221291] mu_seq_posted     :      0
[ 1277.221293] seq_failed        :      0
[ 1277.221296] seq_restarted     :      236
[ 1277.221300] tx_abort          :      0
[ 1277.221302] mpdus_requed      :      239
[ 1277.221304] mpdus_sw_flush    :      0
[ 1277.221307] mpdus_hw_filter   :      8
[ 1277.221309] mpdus_truncated   :      0
[ 1277.221312] mpdus_ack_failed  :      318
[ 1277.221314] mpdus_expired     :      0
[ 1277.221317] excess retries    :      318
[ 1277.221319] last rc           :      0
[ 1277.221321] sched self trig   :      0
[ 1277.221325] ampdu retry failed:      0
[ 1277.221328] illegal rate errs :      0
[ 1277.221330] pdev cont xretry  :      0
[ 1277.221334] pdev tx timeout   :      0
[ 1277.221336] pdev resets       :      1
[ 1277.221339] ppdu txop ovf     :      0
[ 1277.221342] mcast Drop        :      0
[ 1277.221344]
[ 1277.221344] ### Rx ###
[ 1277.221346] ppdu_route_change :      0
[ 1277.221349] status_rcvd       :      255
[ 1277.221351] r0_frags          :      0
[ 1277.221354] r1_frags          :      0
[ 1277.221356] r2_frags          :      0
[ 1277.221360] htt_msdus         :      0
[ 1277.221362] htt_mpdus         :      0
[ 1277.221365] loc_msdus         :      243
[ 1277.221369] loc_mpdus         :      243
[ 1277.221371] oversize_amsdu    :      0
[ 1277.221373] phy_errs          :      0
[ 1277.221376] phy_errs dropped  :      0
[ 1277.221378] mpdu_errs         :      12
[ 1277.221381] pdev_rx_timeout   :      0
[ 1277.221385] ovfl_mpdu_errs    :      0
[ 1277.221392]
[ 1277.221392]  ### TX extended stats ###
[ 1277.221395] tx_de_cache_miss            128
[ 1277.221398] tx_enqueue_peer_invalid     0
[ 1277.221400] tx_burst_disable_cong_ctrl 0
[ 1277.221403] tx_de_lookup_failed         128
[ 1277.221407] tx_delay_proc               0
[ 1277.221410] tt_prefetch_suspend         0
[ 1277.221412] short_pkt                   0
[ 1277.221416] total_enqueue               12218
[ 1277.221418] peer_id_invalid             0
[ 1277.221421] host_inspect                0
[ 1277.221424] pdev_zero_discard           0
[ 1277.221427] tickle_proc_sched           0
[ 1277.221430] residue_cleanup             0
```

Output interpretation

Tx:

comp_queued: # of remote MSDUs (data frames) completed and put into completion queued.

comp_delivered: # of remote MSDUs in completion queue been sent to host

msdu_enqueue: # of MSDUs queued to WAL. This includes remote and local MSDUs

wmm_drop: # of MSDUs dropped due to WMM limitation. This counter also mean that MSDUs are getting dropped due to limited pool. The large ratio of wmm_drop/msdu_enqueu would potentially indicate throughput problem.

local_enqued: # of local MSDUs (non-data frames) queued to WAL

local_freed: # of local MSDUs completed

hw_queued: # of PPDUs queued to hardware

hw_reaped: # of PPDUs completed from hardware

underrun: # of times Tx under run occurred

tx_abort: N/A

mpdus_requed: # of MPDUs retried

excess_retries: # of times excess tries happened

last_rc: the last hardware rate code used for transmission.

The rate code is encoded as follows:

```
b’7..b’6: Preamble (0-OFDM, 1-CCK, 2 HT and 3 VHT)
b’5..b’4: NSS (0- 1x1, 1-2x2, 2-3x3, 3-4x4)
b’3..b’0: Rate/MCS
                 OFDM :     0: OFDM 48 Mbps
                            1: OFDM 24 Mbps
                            2: OFDM 12 Mbps
                            3: OFDM 6 Mbps
                            4: OFDM 54 Mbps
                            5: OFDM 36 Mbps
                            6: OFDM 18 Mbps
                            7: OFDM 9 Mbps
                 CCK (preamble == 1)
                            0: CCK 11 Mbps Long
                            1: CCK 5.5 Mbps Long
                            2: CCK 2 Mbps Long
                            3: CCK 1 Mbps Long
                            4: CCK 11 Mbps Short
                            5: CCK 5.5 Mbps Short
                            6: CCK 2 Mbps Short
                HT/VHT (preamble == 2/3)
                         0..7: MCS0..MCS7 (HT, HT MCS > 7 are represented
using this field and NSS)
                            0..9: MCS0..MCS9 (VHT)
```

sched_self_trig:

Number of times, firmware retry PPDU transmissions, which were not given to hardware due to PPDU airtime exceeding desired length, e.g., BT limits the duration and it may happen that PPDU was not fitting in the duration set by BT.

ampdu retry failed:

Number of times, all AMPDU retries failed. After all AMPDU retries exhausted BAR is sent.

illegal rate errs:

Number of times hardware encountered illegal VHT rate PHY errors.

pdev cont xretry:

Number of times firmware encountered persistent excess retries

pdev reset:

Number of times hardware reset, for events like firmware workaround for PHY hangs etc.

Rx

ppdu_route_change:

Number of times for a received PPDU, part of MPDUs are data frames and part of the MPDUs are non-data frames

status_rcvd:

# of Rx status is used. One Rx status usually represents one MSDU

r0_frags:

# of buffer fragmentation happened in Ring 0. The buffer fragmentation means that a MSDU occupies more than one Rx buffer

r1_frags: # of buffer fragmentation happened in Ring 1

r2_frags: # of buffer fragmentation happened in Ring 2

r3_frags: # of buffer fragmentation happened in Ring 3

htt_msdus: # of data MSDUs received

htt_mpdus: # of data MPDUs received

loc_msdus: # of non-data MSDUs received

loc_mpdus: # of non-data MPDUs received

oversize_amsdu: # of the times that receiving an A-MSDU which has SDUs more than the size of Rx status ring

TX extended stats

tx**_de_cache_miss:** # of times de_cache missed

tx**_enqueue_peer_invalid:** Data packet is dropped because peer is invalid

tx_burst_disable_cong_ctrl: # of times burst is disabled

tx_de_lookup_failed: Data packets are dropped in tx_de_input_ext()

tx_delay_proc: Pre fetch delay schedule

tt_prefetch_suspend: # of times pre fetch got suspended

short_pkt: # of Short pkt (less than 1280 size)

total_enqueue: # of pkt enqueuer (mgmt+data packets)

peer_id_invalid: # of data pkt sent by host with invalid peer id

host_inspect: # of data pkt sent back to host for inspection

pdev_zero_discard: # of data pkt got discarded due to no vdev

tickle_proc_sched: Pre-fetch schedule based on tickle

residue_cleanup: Pre-fetch schedule based on waitq residue cleanup

1.3.21.2 Rx reorder stats

Output (STA)

```
[75111.141380] Rx reorder statistics:
[75111.141390]   0 non-QoS frames received
[75111.141397]   2258 frames received in-order
[75111.141402]   0 frames flushed due to timeout
[75111.141408]   0 frames flushed due to moving out of window
[75111.141414]   0 frames flushed due to receiving DELBA
[75111.141420]   37 frames discarded due to FCS error
[75111.141426]   515 frames discarded due to invalid peer
[75111.141432]   0 frames discarded due to duplication (non aggregation)
[75111.141438]   0 frames discarded due to duplication in reorder queue
[75111.141444]   0 frames discarded due to processed before
[75111.141455]   0 times reorder timeout happened
[75111.141460]   0 times bar ssn reset happened
[75111.141464]   0 times incorrect bar received
```

Output interpretation

Non-QoS frames received: # of MPDUs that came from a peer without aggregation configured

Frames received in-order: # of MPDUs received and are in-order, i.e. deliver to upper stack

Frames flushed due to timeout: # of MPDUs been flushed due to timeout.

NB: those frames are discarded

Frames flushed due to moving out of window: # of MPDUs been flushed due to receiving a new MPDU that moves the reorder window forward.

NB: These frames are delivered to upper stack

Frames flushed due to receiving DELBA: # of MPDUs been flushed due to DELBA.

NB: These frames are discarded

Frames discarded due to FCS error: # of MPDUs discarded due to FCS error

Frames discarded due to invalid peer: # of MPDUs discarded because we cannot find the corresponding peer.

Frames discarded due to duplication (non-aggregation): # of MPDUs came from a peer without aggregation configured which are duplication of previous received MPDU.

Frames discarded due to duplication in reorder queue: # of MPDUs which are duplication of frames in Rx reorder queue

Frames discarded due to processed before: # of MPDUs which are received before.

NB: If the incoming sequence number of a MPDU has more than 2047 offset of expected sequence number in sequence number space, it is considered as processed before.

Times reorder timeout happened: # of times reorder timer has expired.

Bar ssn reset happened: Number of times reorder sequence windows was reset due to reception of BAR

Incorrect bar received: Number of times received BAR was not valid.

1.3.21.3 Rx rate info stats

Output (STA)

```
[16462.210781] RX Rate Info:
[16462.210785] MCS counts (0..9): 0, 0, 8358, 1498, 360, 968, 313, 5611, 12149,
31245
[16462.210791] SGI counts (0..9): 0, 0, 0, 0, 0, 0, 0, 1812, 12083, 1844
[16462.210796] NSS  counts: 1x1 10681, 2x2 33237, 3x3 16584, 4x4 0
[16462.210800] NSTS count: 1643
[16462.210802] BW counts: 20MHz 0, 40MHz 40499, 80MHz 20003
[16462.210806] Preamble counts: 2053, 0, 0, 60502, 0, 300
[16462.210810] STBC rate counts (0..9): 0, 0, 0, 0, 0, 1, 162, 311, 316, 0, 853
[16462.210815] LDPC TXBF Counts: 49472, 0
[16462.210818] RSSI (data, mgmt): 45, 16
[16462.210821] RSSI Chain 0 (0x80 0x12 0x2a 0x29)
[16462.210824] RSSI Chain 1 (0x80 0x0f 0x29 0x28)
[16462.210827] RSSI Chain 2 (0x80 0x0f 0x27 0x27)
```

Output interpretation

- **MCS counts:** These are counters for each MCSs 0..9 in case of VHT, and MCS0..7 in the case of the HT association. For 802.11n MCS8..23 combine this field with NSS field, e.g, MCS8 is NSS 2 MCS0.

NOTE The MCS count does not capture legacy OFDM/CCK rates.

- **SGI counts:** Counters for each SGI enabled MCS
- **NSS counts:** Captures number of spatial streams. Indicate whether 1x1, 2x2 or 3x3 rate is being used. Combined with MCS gives actual (802.11n) MCS in case of HT
- **NSTS count:** Indicates whether the frames are being sent with STBC enabled and the transmission is at a 1x1 rate. The NSTS count can be seen to be equal to the sum of STBC counts
- **BW counts:** Indicate number of received frames on 20, 40, and 80 MHz. Useful to debug which all BWs are being used currently by the transmitter STA
- **Preamble counts:** Index 0 counts legacy (CCK/OFDM) ppdus, 1 HT, 2 HT with BF (on QCA9880 always 0), three VHT and four VHT with BF (on QCA9880 always zero), five all other, e.g., PHY error
- **STBC rate counts:** Similar to MCS counts give what all MCSs have STBC enabled
- **LDPC TXBF counts:** the first counter increments for each received LDPC ppdu. Second one increment for each received TxBF frames, which is not supported by QCA9880

- **RSSI (data, management):** Absolute RSSI value as seen in received MAC descriptor for data and management frame respectively
- **RSSI chain 0:** (sec80, sec40, sec20, pri20) gives RSSI seen in MAC descriptor for given chain across primary/secondary channels. This could be quite useful to make sure, all chains are balanced.

NOTE When rate is fixed, only one of the MCS count would increment with autorate. Most of the MCSs would be used depending on the environment.

1.3.21.4 Tx rate info stats

```
root@OpenWrt:/# iwpriv ath0 txrx_fw_stats 6
TX Rate Info:
MCS counts (0..9):    0, 0, 0, 0, 0, 0, 0, 0, 0, 0
MCS counts SU (0..9): 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
MCS counts MU (0..9): 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
SGI counts (0..9): 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
NSS  counts: 1x1 0, 2x2 0, 3x3 0 4x4 0
BW counts: 20MHz 0, 40MHz 0, 80MHz 0
Preamble (O C H V) counts: 8661123, 0, 0, 41
STBC rate counts (0..9): 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
LDPC Counts: 0
RTS Counts: 0
Ack RSSI: -128
```

1.3.21.5 Copy engine and host stats

```
root@OpenWrt:/# iwpriv ath0 txrx_fw_stats 8
++++++++++ CE STATISTICS +++++++++++
CE0 Host sw_index (dst_ring):     0
CE0 Host write_index (dst_ring):  0
CE1 Host sw_index (dst_ring):     9
CE1 Host write_index (dst_ring):  8
CE2 Host sw_index (dst_ring):     54
CE2 Host write_index (dst_ring):  53
CE3 Host sw_index (dst_ring):     0
CE3 Host write_index (dst_ring):  0
CE4 Host sw_index (dst_ring):     0
CE4 Host write_index (dst_ring):  0
CE5 Host sw_index (dst_ring):     190
CE5 Host write_index (dst_ring):  189
CE6 Host sw_index (dst_ring):     0
CE6 Host write_index (dst_ring):  0
CE7 Host sw_index (dst_ring):     0
CE7 Host write_index (dst_ring):  1
CE8 Host sw_index (dst_ring):     0
CE8 Host write_index (dst_ring):  127
CE9 Host sw_index (dst_ring):     0
CE9 Host write_index (dst_ring):  0
CE10 Host sw_index (dst_ring):     0
CE10 Host write_index (dst_ring):  0
CE11 Host sw_index (dst_ring):     0
CE11 Host write_index (dst_ring):  0
++++++++++ HOST TX STATISTICS +++++++++++
Ol Tx Desc In Use:  0
Ol Tx Desc Failed:  0
CE Ring (4) Full :  0
DMA Map Error    :  0
Tx pkts completed:  0
Tx bytes completed:  0
Tx pkts from stack:  0
++++++++++ HOST RX STATISTICS +++++++++++
Rx pkts completed:  0
Rx bytes completed:  0
++++++++++ HOST GENERIC STATISTICS +++++++++++
Fast Path on CPU[0]:  8405
Fast Path on CPU[1]:  59466
Fast Path on CPU[2]:  0
Fast Path on CPU[3]:  0
Fast Path on CPU[4]:  0
Fast Path on CPU[5]:  0
Fast Path on CPU[6]:  0
Fast Path on CPU[7]:  0
Non Fast Path on CPU[0]:  4023668
Non Fast Path on CPU[1]:  6475825
Non Fast Path on CPU[2]:  0
Non Fast Path on CPU[3]:  0
Non Fast Path on CPU[4]:  0
Non Fast Path on CPU[5]:  0
Non Fast Path on CPU[6]:  0
Non Fast Path on CPU[7]:  0
Fast Tasklet on CPU[0]:  6108975
Fast Tasklet on CPU[1]:  7891404
Fast Tasklet on CPU[2]:  0
Fast Tasklet on CPU[3]:  0
Fast Tasklet on CPU[4]:  0
Fast Tasklet on CPU[5]:  0
Fast Tasklet on CPU[6]:  0
Fast Tasklet on CPU[7]:  0
Reg. Tasklet on CPU[0]:  3082
Reg. Tasklet on CPU[1]:  0
Reg. Tasklet on CPU[2]:  0
Reg. Tasklet on CPU[3]:  0
Reg. Tasklet on CPU[4]:  0
Reg. Tasklet on CPU[5]:  0
Reg. Tasklet on CPU[6]:  0
Reg. Tasklet on CPU[7]:  0
++++++++++ HOST FLOW CONTROL STATISTICS +++++++++++
Receive from stack count: 0
non queued pkt count: 33961
queued pkt count: 0
queue overflow count: 0
```

1.3.21.6 Host multi-task enhance stats

```
root@OpenWrt:/# iwpriv ath0 txrx_fw_stats 12
++++++++++ HOST MCAST Ehance STATISTICS +++++++++++
Mcast recieved: 0
ME converted: 0
ME dropped (Map): 0
ME dropped (alloc): 0
ME dropped(internal): 0
ME bufs in use: 0
```

1.3.21.7 Tx beamforming data info stats

```
root@OpenWrt:/# iwpriv ath0 txrx_fw_stats 13
TXBF Data Info:
VHT Tx TxBF counts(0..9): 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
VHT Rx TxBF counts(0..9): 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
HT Tx TxBF counts(0..7): 0, 0, 0, 0, 0, 0, 0, 0,
OFDM Tx TxBF counts(0..7): 0, 0, 0, 0, 0, 0, 0, 0,
```

1.3.21.8 Tx beamforming sounding info stats

```
root@OpenWrt:/# iwpriv ath0 txrx_fw_stats 14
TXBF Sounding Info:
Sounding User 1   : 20Mhz 0, 40Mhz 0, 80Mhz 0
Sounding User 2   : 20Mhz 0, 40Mhz 0, 80Mhz 0
Sounding User 3   : 20Mhz 0, 40Mhz 0, 80Mhz 0
CBF 20 (Nc 1 2 3 4):0, 0, 0, 0
CBF 40 (Nc 1 2 3 4):0, 0, 0, 0
CBF 80 (Nc 1 2 3 4):0, 0, 0, 0
```

1.3.21.9 Packet error stats

```
root@OpenWrt:/# iwpriv ath0 txrx_fw_stats 15
HWSCH Error  (0..3):0, 175, 0, 3

SchCmdResult (0..7):0, 0, 0, 38, 2188833, 0, 0, 3
SIFS Status   (0..7):0, 6063747, 404313, 38, 0, 0, 0, 137

URRN_stats Error  (0..3):23, 23, 0,
Flush Error  (0..9):0, 0, 0, 0, 0, 3, 0, 0, 0, 0
Flush Error  (10..17):0, 0, 0, 0, 0, 0, 0, 0

Phy Error    (0..9):0, 0, 0, 0, 0, 0, 0, 0, 0, 0
Phy  Error   (9.17):0, 0, 0, 0, 0, 0, 0, 0
```

1.3.21.10 Tx self gen stats

```
root@OpenWrt:/# iwpriv ath0 txrx_fw_stats 16
TX_SELFGEN Info:
su_ndpa           :0
su_ndp            :0
su_bar            :0
SU-BAR is typically used to reset the BA window state on the peer.
su_cts2self       :0
su_ndpa_err       :0
su_ndp_err        :0
mu_ndpa           :0
mu_ndp            :0
mu_brpoll_1       :0
mu_brpoll_2       :0
mu_bar_1          :0
mu_bar_2          :0
mu_cts2self       :0
```

```
mu_ndpa_err         :0
mu_ndp_err        :0
mu_brp1_err       :0
mu_brp2_err       :0
```

Output interpretation

su_ndpa: Number of times SU-NDPA frames transmitted

su_ndp: Number of times SU-NDP frames transmitted

su_bar: Number of times the BAR frames transmitted in SU seq to flush BA state

SU-BAR is typically used to reset the BA window state on the peer.

su_cts2self: Number of times the CTS2SELf frames to extend SU-data burst

su_ndpa_err: Number of times the SU-NDPA frames not transmitted due to hardware pause

su_ndp_err: Number of times the SU-NDP frames that didn't receive correct CBF

mu_ndpa: Number of times the MU-NDPA frames transmitted

mu_ndp: Number of times the MU-NDP frames transmitted

mu_brpoll_1: Number of times the BRPOLL frames transmitted to second user

mu_brpoll_2: Number of times the BRPOLL frames transmitted to third user

mu_bar_1: Number of times the BAR frames to second user in MU-PPDU

mu_bar_2: Number of times the BAR frames to third user in MU-PPDU

mu_cts2self : Number of times the CTS2SELF frames to extend MU-data burst

mu_ndpa_err: Number of times the MU-NDPA frames not transmitted due to hardware pause

mu_ndp_err: Number of times the MU-NDP frames that did not receive correct CBF

mu_brp1_err: Number of times the BRPOLL frames to second user that didn't receive correct CBF

mu_brp2_err: Number of times the BRPOLL frames to third user that didn't receive correct CBF

1.3.21.11 Tx multi-user info stats

```
root@OpenWrt:/# iwpriv ath0 txrx_fw_stats 17
TX_MU Info:
mu_sch_nusers_2          :0
mu_sch_nusers_3          :0
mu_mpdus_queued_usr0     :0
mu_mpdus_queued_usr1     :0
mu_mpdus_queued_usr2     :0
mu_mpdus_queued_usr3     :0
mu_mpdus_tried_usr0      :0
mu_mpdus_tried_usr1      :0
mu_mpdus_tried_usr2      :0
```

```
mu_mpdus_tried_usr3     :0
mu_mpdus_failed_usr0    :0
mu_mpdus_failed_usr1    :0
mu_mpdus_failed_usr2    :0
mu_mpdus_failed_usr3    :0
mu_mpdus_requeued_usr0  :0
mu_mpdus_requeued_usr1  :0
mu_mpdus_requeued_usr2  :0
mu_mpdus_requeued_usr3  :0
mu_err_no_ba_usr0       :0
mu_err_no_ba_usr1       :0
mu_err_no_ba_usr2       :0
mu_err_no_ba_usr3       :0
mu_mpdu_underrun_usr0   :0
mu_mpdu_underrun_usr1   :0
mu_mpdu_underrun_usr2   :0
mu_mpdu_underrun_usr3   :0
mu_ampdu_underrun_usr0  :0
mu_ampdu_underrun_usr1  :0
mu_ampdu_underrun_usr2  :0
mu_ampdu_underrun_usr3  :0
```

1.3.21.12 SIFS response stats

```
root@OpenWrt:/# iwpriv ath0 txrx_fw_stats 18
SIFS RESP RX stats:

s-poll trigger           :0   num ps-poll trigger frames
 u-apsd trigger            :0   num uapsd trigger frames
 qboost trigger data[exp]  :0   num data trigger frames; idx 0: explicit
and idx 1: implicit
 qboost trigger bar[exp]   :0   num bar trigger frames;  idx 0: explicit
and idx p1: implicit
 qboost trigger data[imp]  :0
 qboost trigger bar[imp]   :0

SIFS RESP TX stats:

 SIFS response data           :0   num ppdus transmitted at SIFS interval
 SIFS response timing err     :0    num ppdus failed to meet SIFS resp
timing
```

Output interpretation:

s-poll trigger:0 Number of times the s-poll trigger frames

u-apsd trigger:0 Number of times the uapsd trigger frames

qboost trigger data [exp]:0 Number of times the data trigger frames; idx 0: explicit and idx 1: implicit

qboost trigger bar [exp]:0 Number of times the bar trigger frames; idx 0: explicit and idx p1: implicit

SIFS RESP Tx stats:

SIFresponse data:0 Number of times the ppdus transmitted at SIFS interval

SIFS response timing err: 0 Number of times the ppdus failed to meet SIFS resp timing

1.3.21.13 iwpriv ath0 txrx_fw_stats 19

```
root@OpenWrt:/# iwpriv ath0 txrx_fw_stats 19
RESET stats:

warm reset              :1
cold reset              :0
 tx flush                :0
 tx glb reset            :0
 tx txq reset            :0
 rx timeout rese :0
hw status mismatch      :0
 hw status multi mismatch :0
```

Output interpretation

warm reset: Number of warm resets from reboot

cold reset: Number of cold resets from reboot

tx flush: Number of resets to recover from Tx hang

tx glb reset: Number of resets because of Tx queue timeout

tx txq reset: Number of resets because of Tx hw stuck

rx timeout rese: Number of mismatches between status and schedule id in hardware scheduler

hw status mismatch: Number of mismatches between status and schedule id in hardware scheduler

hw status multi mismatch: Number of resets because status and schedule id out of sync in hardware scehdulert: Number of resets because of Rx hardware timeout hw status mismatch

1.3.21.14 Clearing firmware statistics

To clear the stats for a particular bit set in the specified mask:

Execute the following command:

```
iwpriv wlan0 txrx_fw_st_rst < mask>
```

For example a mask of 0x3fff would clear stats 1, 2, 3, 4, and 5 and 6.

NOTE The command argument values, 4 and 10 are obsolete.

1.3.22 Unassociated power consumption improvement parameters

Table 1-25 Unassociated power consumption improvement parameters

Parameter	Format	Description	

ignore11d
get_ignore11d
	iwpriv `ath`*N ignore11d 1|0*	Processes or ignores 11d beacon
		Default value is 1
		`#iwpriv ath0 ignore11d 0`
		`#iwpriv ath0 get_ignore11d`
		`ath0 get_ignore11d:0`	

		0	Process 11d beacon
		1	Ignore 11d beacon

1.3.23 Smart antenna

To change default settings for smart antenna and to read Smart Antenna settings, iwprivs are implemented. These iwprivs are tied with wifiN interface instead of athN because Smart Antenna treats all the VAPs created over a physical (wifiN) device in same manner.

Table 1-26 Smart antenna parameters

Parameter	Format	DA	OL	Description	
set_sa_param **get_sa_param**	iwpriv wifidev [*wifi0*|*wifi1]* set_sa_param *dword1 dword2 dword3 dword4* iwpriv wifidev [*wifi0*|*wifi1]* get_sa_param *dword1 dword2 dword3*	Y	Y	Sets and gets Smart Antenna parameters. Each dword attribute is defined as given in this table. At any time, iwpriv wifi0 set_sa_param 0 0 0 0 can be used to list all the paramID that are used as a third argument in set_sa_param. **NOTE** dword4 is not required on the get command.	
				dword1: 0xAABBCCDD	
				0xAA	param type: 0 = radio param, 1 = node param For radio param, MAC is 00:00:00:00:00:00. For node param, proper MAC address must be specified.
				0xBB	Reserved (should be 00)
				0xCCDD	bytes 5 and 6 of MAC
				dword2: 0xEEFFGGHH	
				0xEEFFGGHH	bytes 4, 3, 2, 1 of MAC. For example, if the MAC address is 00:03:7f:48:d8:73 then: CC = 00, DD = 03, EE = 7f, FF = 48, GG = d8, HH = 73.
				dword3: paramID. See definitions and descriptions in Table 1-27.	
				dword4: paramValue (required only for set_sa_param)	

Table 1-27 dword3 parameters

ParamName	Param ID	Node or Radio param	Description
SMART_ANT_PARAM_ HELP	0	Radio	Displays current available commands list
SMART_ANT_PARAM_ TRAIN_MODE	1	Radio	Self-packet generation or existing traffic mode. Currently only existing traffic mode is supported. 0 = existing; 1 = mixed.
SMART_ANT_PARAM_ TRAIN_ PER_THRESHOLD	2	Radio	Smart antenna lower, upper and per diff thresholds. Here byte 0 is lower_bound, byte 1 is upper_bound, byte 3 is per_ diff_threshold and byte 4 is config. By default lower bound is 20, upper bound is 80, per_diff_ threshold is 3 and config is 1. Config is a bit map of 4 possible values: #define SA_CONFIG_INTENSETRAIN 0x1 /* setting this bit in config indicates training with double number of packets */ #define SA_CONFIG_EXTRATRAIN 0x2 /* setting this bit in config indicates to do extra traing in case of conflits in first metric */ #define SA_CONFIG_SLECTSPROTEXTRA 0x4 /* setting this bit in config indicates to protect extra training frames with self CTS */ #define SA_CONFIG_SLECTSPROTALL 0x8 /* setting this bit in config indicates to protect all training frames with self CTS */
SMART_ANT_PARAM_ PKT_LEN	3	Radio	Packet length of proprietary generated training packet. By default is 1536.
SMART_ANT_PARAM_ NUM_PKTS	4	Radio	Number of packets used for training. If not set, default value of 640 will be used.
SMART_ANT_PARAM_ TRAIN_START	5	Node	Start smart antenna training.
SMART_ANT_PARAM_ TRAIN_ENABLE	7	Radio + Node	Bitmap for init, periodic & performance triggers. #define SA_INIT_TRAIN_EN 0x1 #define SA_PERIOD_TRAIN_EN 0x2 #define SA_PERF_TRAIN_EN 0x4 #define SA_RX_TRAIN_EN 0x10
SMART_ANT_PARAM_ RETRAIN_INTERVAL	9	Radio	Periodic retrain interval in milliseconds. By default it is 2 minutes.
SMART_ANT_PARAM_ GOODPUT_AVG_ INTERVAL	12	Radio	Good put averaging interval. By default it is 2 seconds.
SMART_ANT_PARAM_ DEFAULT_ANTENNA	13	Radio	Default antenna for Rx, Tx multicast and Tx broadcast. By default it is antenna 0.
SMART_ANT_PARAM_ DEFAULT_TX_ANTENNA	14	Radio	Default Tx antenna for Tx. By default it is antenna 0. Once a new node connects, by default this antenna is used as unicast Tx antenna.

Table 1-27 dword3 parameters (cont.)

ParamName	Param ID	Node or Radio param	Description
SMART_ANT_PARAM_ TX_ANTENNA	15	Node	Once this command is set, no training will be done for this node and this antenna will be used for all unicast Tx.
SMART_ANT_PARAM_ DBG_LEVEL	16	Radio	It’s a 4 bit value used for controlling the prints. By default it is log level 1. Bit 1 controls log level 1, bit 2 controls log level 2, bit 3 controls log level 3 and bit 4 controls log level 4.
SMART_ANT_PARAM_ PRETRAIN_PKTS	17	Radio	Number of pre train packets. Once a node is connected these many packets are sent before starting the training. By default it is 600.
SMART_ANT_PARAM_ OTHER_BW_PKTS_TH	18	Radio	Threshold for other bw packets to detect bandwidth change. By default it is 5.
SMART_ANT_PARAM_ GOODPUT_IGNORE_ INTERVAL	19	Radio	By default good put ignoring interval is 1 second.
SMART_ANT_PARAM_ MIN_PKT_TH_BW20	20	Radio	Minimum number of packets in 20 MHz BW to indicate active BW. By default it is 20.
SMART_ANT_PARAM_ MIN_PKT_TH_BW40	21	Radio	Minimum number of packets in 40 MHz BW to indicate active BW. By default it is 10.
SMART_ANT_PARAM_ MIN_PKT_TH_BW80	22	Radio	Minimum number of packets in 80 MHz BW to indicate active BW. By default it is 5.
SMART_ANT_PARAM_ DEBUG_INFO	23	Node	Displays Last training time, Periodic triggers and performance triggers for specific node.
SMART_ANT_PARAM_ MAX_TRAIN_PPDU	24	Radio	Max number of train ppdus in train command. By default it is 50
SMART_ANT_PARAM_ PERF_HYSTERESIS	25	Radio	Hysteresis for performance based trigger By default it is 3.
SMART_ANT_PARAM_ BCN_ANTENNA	27	Radio	To Configure Beacon Antenna
SMART_ANT_PARAM_ TRAIN_RATE_TESTMODE	28	Node	Train rate code in test mode
SMART_ANT_PARAM_ TRAIN_ANTENNA_ TESTMODE	29	Node	Train antenna in test mode
SMART_ANT_PARAM_ TRAIN_PACKETS_ TETSMODE	30	Node	Number of train packets in test mode
SMART_ANT_PARAM_ TRAIN_START_ TETSMODE	31	Node	Start Training in test mode

1.3.24 WDS parameters

Table 1-28 WDS parameters

Parameter	Format	DA	OL	Description	
nobeacon **get_nobeacon**	iwpriv ath*N* nobeacon	N	N	Enables/disables VAP to transmit beacon and probe response. The get parameter returns the current value. An AE test by Frank Yang determined that these commands are invalid as of 5/2/2011. The macro "ATH_SUPPORT_AP_WDS_COMBO" controls whether the commands are supported.	
				0	Disable
				1	Enable
extap **get_extap**	iwpriv ath*N* extap *{0-3}*	Y	Y	Sets Extender AP support. The get parameter returns the current value. #iwpriv ath0 extap 0 #iwpriv ath0 get_extap ath0 get_extap:0	
				0	Disable Extender AP support
				1	Enable Extender AP support
				2	Enable Extender AP support and purge the DEBUG info.
				3	Print out debug info for Extender AP, does not enable or disable the Extender AP support.
athnewind get_athnewind	iwpriv ath*N athnewind* *{1|0}*	Y	N	Enables (1) or disables (0) enhanced independent repeater mode. If this option is enabled, the STA VAP will scan for the Root AP in all the available channels and connect to it. The AP VAP will start and continue to transmit beacons independently of the STA VAP connection status. The default value is 0. The get parameter returns the current value. #iwpriv ath0 athnewind 1 #iwpriv ath0 get_athnewind ath0 get_athnewind:1	
wds **get_wds**	iwpriv ath*N* wds *{1|0}*	Y	Y	Enables (1) or disables (0) 4-address frame format for this VAP. Used for WDS configurations (see “Wi-Fi Distribution System (WDS)” in the *AP Driver User’s Guide* for details). The default value is 0. The get parameter returns the current value. #iwpriv ath0 wds 1 #iwpriv ath0 get_wds ath0 get_wds:1	

1.3.25 WMM parameters

WMM parameters manage the WMM link settings. To set parameters, each command must specify the access category (AC) and mode (STA or AP).

Table 1-29 Access categories and modes

Value	Symbol	Description
Access Categories		
0	AC_BE	Best effort
1	AC_BK	Background
2	AC_VI	Video
3	AC_VO	Voice
Mode Parameter		
0	AP	AP mode: Update the AP WMM table
1	STA	STA mode: Update the STA WMM tables

The parameters accessible for WMM operations are specified in the WMM (including WMM Power Save) Specifications. These parameters control the way in which the time slots or TXOPs are metered out for each traffic stream. Table lists the parameters accessible in the Qualcomm Atheros driver.

Table 1-30 WMM parameters

Parameter	Format	DA	OL	Description

acparams	iwpriv ath*N*
	acparams
	ac {0-3}
	rts {1|0}
	aggrscaling {0-3}
	min_rate[Mbps]
		Y	Y	Configures the access category. See Table 1-29
				Access category:
				0: BE; 1: BK; 2: VI; 3: VO
				Enable RTS/CTS: Applies to all rate series.
				Aggregate scaling: Controls the maximum air time that the aggregates can use.
				0: Disable, ≥ 4 ms; 1: ≥ 2 ms; 2: ≥ 1 ms; 3: ≥ 0.5 ms
				Minimum Rate: Sets the per-access category lower threshold rate, which used by the voice (VO) and video (VI) rate algorithm. If the operating rate drops below this threshold, then HBR applies.

Table 1-30 WMM parameters

setwmmparams
getwmmparams
	iwpriv ath*N* setwmmparam *wmeparam {1-6}* *ac {0-3}* *bss {1|0}* *wmevalue*	Y	Y	Sets WMM sub-parameters. The range and units of measure for *wmevalue* are listed with the WME parameter below. The get parameter returns the current settings.
				`#iwpriv ath0 setwmmparams 1 0 0 4`
				`#iwpriv ath0 getwmmparams 1 0 0`
				`   ath0 getwmmparams:4`
				Each WME parameter can be executed independently, without using “setwmmparams” or “getwmmparams”, as shown in the following examples. The access category, BSS/local, and value arguments remain the same. Each set parameter has a corresponding get parameter that returns the current value. For example, the `cwmin` parameter may be given as follows:
				`#iwpriv ath0 cwmin 3 1 2`
				`#iwpriv ath0 get_cwmin 3 1`
				`   ath0 get_cwmin: 2`
				The WME parameters may thus be given as follows:
				`#iwpriv athN acm`
				`#iwpriv athN aifs`
				`#iwpriv athN cwmax`
				`h#iwpriv athN cwmin`
				`#iwpriv athN noackpolicy`
				`#iwpriv athN txoplimit`
				Note for noackpolicy:
				`#iwpriv athN noackpolicy 2 0 1 (Second aparameter needs to be zero (bss = 0) for this to work)`	

				WME Parameters (*wmeparam*, *wmevalue*)	
				1	CWMIN (*wmevalue* = 0-15, in units of slot time)
				2	CWMAX (*wmevalue* = 0-15, in units of slot time)
				3	AIFS (*wmevalue* = 0-15, in units of slot time)
				4	TXOPLIMIT (*wmevalue* = 0-8192, in units of 32 μs)
				5	ACM (*wmevalue* = 0 for disable, 1 for enable)
				6	NOACKPOLICY (*wmevalue* = 0 for disable, 1 for enable)
				Access Category Parameters (*ac*)	
				0	Best effort (BE)
				1	Background (BK)
				2	Video (VI)
				3	Voice (VO)
				BSS/Local Parameters (*bss*)	
				1	BSS (channel parameters broadcast to STAs)
				0	Local (channel parameters applied to self)

Table 1-30 WMM parameters

uapsd
get_uapsd
	iwpriv ath*N* uapsd *{1|0}*	Y	Y	Enables (1) or disables (0) the corresponding bit in the capabilities field of the beacon and probe response messages; has no other effect. The default value is 1. This get parameter returns the current value.
				`#iwpriv ath0 uapsd 1`
				`#iwpriv ath0 get_uapsd`
				`ath0 get_uapsd:1`

wmm
get_wmm
	iwpriv ath*N* wmm *{1|0}*	Y	Y	Enables (1) or disables (0) WMM capabilities in the driver. The WMM capabilities perform special processing for multimedia stream data including voice and video data. This command has a corresponding get parameter, and its default is 1 (WMM enabled).
				`#iwpriv ath0 wmm 1`
				`#iwpriv ath0 get_wmm`
				`ath0 get_wmm:1`

1.3.26 256QAM rate support parameters

Table 1-31 256QAM parameters

Parameter	Format	DA	OL	Description

vht_11ng
get_vht_11ng
	iwpriv ath*N* vht_11ng *{1/0}}*	N	Y	Enables (1) or disables (0) 256QAM rate support. The default value is 0. This command enables 256QAM rate support in 2.4GHz band HT modes only (such as 11NGHT20, 11NGHT40PLUS, 11NGHT40MINUS)
				The get parameter returns the current value.
				`#iwpriv ath0 vht_11ng 1`
				`#iwpriv ath0 get_vht_11ng`
				`ath0 get_vht_11ng:1`

1.3.27 Hy-Fi options – WMM DSCP prioritization

Table 1-32 Hy-Fi parameters

Parameter	Format	DA	OL	Description
aldstats	iwpriv wifix aldstats *{1/0}*	Y	N	To enable/disable few Hy-Fi link metrics stats. This option should be enabled to collect packet drops to no buffs, excessive retries and transmitted packet count stats per access category per destination node. This command is applicable only for direct attach VAPs `#iwpriv wifi0 aldstats 1`

s_dscp_ovride
g_dscp_ovride
	iwpriv wifix set_dscp_ ovride *{1/0}*	Y	Y	To enable/disable dscp override feature. Packets with specific dscp value set can be mapped to a specific TID through this feature.
				`#iwpriv wifi0 set_dscp_ovride 1`
				`#iwpriv wifi0 get_dscp_ovride`
				`get_dscp_ovride:1`

Table 1-32 Hy-Fi parameters

Parameter	Format	DA	OL	Description
reset_dscp_map	iwpriv wifix reset_dscp_map <tid>	Y	N	To reinitialize all the dscp's with a default tid value. This command is not available for offload vap. `#iwpriv wifi0 reset_dscp_map 1`
set_dscp_tid_map get_dscp_tid_map	iwpriv wifix s_dscp_tid_map <dscp> (hex_val)> <td (0-8)>	Y	Y	To configure a specific tid for specific dscp value. Iwpriv option set_dscp_ovride should be set to 1. `#iwpriv wifi0 s_dscp_tid_map` 0xe0 1 `#iwpriv wifi0 g_dscp_tid_map` 0xe0 `g_dscp_tid_map:1`

sIgmpDscpOvrid
gIgmpDscpOvrid
	iwpriv wifix sIgmpDscpOvrid 1	Y	Y	To enable IGMP TID override.
				`#iwpriv wifix sIgmpDscpOvrid 1`
				`#iwpriv wifix gIgmpDscpOvrid`
				`gIgmpDscpOvrid:1`

sIgmpDscpTidMap
gIgmpDscpTidMap
	iwpriv wifix sIgmpDscpTidMap <tid>	Y	Y	To configure a specific TID for IGMP packets. All IGMP transmitted will go through the TID configured. Iwpriv option sIgmpDscpOvrid should be set to 1 for this command to work.
				`#iwpriv wifix sIgmpDscpTidMap <tid>`
				`#iwpriv wifix gIgmpDscpTidMap`
				`sIgmpDscpTidMap: <tid>`

sHmmcDscpOvrid
gHmmcDscpOvrid
	iwpriv wifix sHmmcDscpOvrid *{1/0}*	Y	Y	To enable/disable hmmc dscp override. To push all multi-cast to unicast converted packets through a specific TID
				`#iwpriv wifix sHmmcDscpOvrid`
				`#iwpriv wifix gHmmcDscpOvrid`
				`gHmmcDscpOvrid`

sHmmcDscpTidMap
gHmmcDscpTidMap
	iwpriv wifix sHmmcDscpTidMap <tid>	Y	Y	To configure a specific tid for unicast packets derived from multi-cast packets. Iwpriv option sHmmcDscpOvrid should be set to 1 for this command to work
				`#iwpriv wifix sHmmcDscpTidMap <tid>`
				`#iwpriv wifix gHmmcDscpTidMap`
				`gHmmcDscpTidMap:<tid>`

setBlkReportFld
getBlkReportFld
	iwpriv wifix setBlkReportFld *{1/0}*	Y	Y	To enable/disable report flooding. Enabling this feature would block flooding reports to other STAs associated with the AP.
				`#iwpriv wifix setBlkReportFld 1`
				`#iwpriv wifix getBlkReportFld`
				`getBlkReportFld: 1`

setDropSTAQuery
getDropSTAQuery
	iwpriv wifix setDropSTAQuery *{1/0}*	Y	Y	To enable/disable DropSTAQuery feature. Enabling feature would drop IGMP Querys from STA
				`#iwpriv wifix setDropSTAQuery 1`
				`#iwpriv wifix getDropSTAQuery`
				`getDropSTAQuery:1`

nopbn
get_nopbn
	iwpriv athX nopbn *{1/0}*	Y	Y	To disable VAPs being notified when jump start button gets pushed.
				`#iwpriv ath0 nopbn 1`
				`#iwpriv ath0 get_nopbn 1`
				`get_nopbn:1`

Table 1-32 Hy-Fi parameters

Parameter	Format	DA	OL	Description
reset_dscp_map	iwpriv wifix reset_dscp_map <tid>	Y	N	To reinitialize all the dscp's with a default tid value. This command is not available for offload vap. `#iwpriv wifi0 reset_dscp_map 1`
set_dscp_tid_map get_dscp_tid_map	iwpriv wifix s_dscp_tid_map <dscp> (hex_val)> <td (0-8)>	Y	Y	To configure a specific tid for specific dscp value. Iwpriv option set_dscp_ovride should be set to 1. `#iwpriv wifi0 s_dscp_tid_map` 0xe0 1 `#iwpriv wifi0 g_dscp_tid_map` 0xe0 `g_dscp_tid_map:1`

sIgmpDscpOvrid
gIgmpDscpOvrid
	iwpriv wifix sIgmpDscpOvrid 1	Y	Y	To enable IGMP TID override.
				`#iwpriv wifix sIgmpDscpOvrid 1`
				`#iwpriv wifix gIgmpDscpOvrid`
				`gIgmpDscpOvrid:1`

sIgmpDscpTidMap
gIgmpDscpTidMap
	iwpriv wifix sIgmpDscpTidMap <tid>	Y	Y	To configure a specific TID for IGMP packets. All IGMP transmitted will go through the TID configured. Iwpriv option sIgmpDscpOvrid should be set to 1 for this command to work.
				`#iwpriv wifix sIgmpDscpTidMap <tid>`
				`#iwpriv wifix gIgmpDscpTidMap`
				`sIgmpDscpTidMap: <tid>`

sHmmcDscpOvrid
gHmmcDscpOvrid
	iwpriv wifix sHmmcDscpOvrid *{1/0}*	Y	Y	To enable/disable hmmc dscp override. To push all multi-cast to unicast converted packets through a specific TID
				`#iwpriv wifix sHmmcDscpOvrid`
				`#iwpriv wifix gHmmcDscpOvrid`
				`gHmmcDscpOvrid`

sHmmcDscpTidMap
gHmmcDscpTidMap
	iwpriv wifix sHmmcDscpTidMap <tid>	Y	Y	To configure a specific tid for unicast packets derived from multi-cast packets. Iwpriv option sHmmcDscpOvrid should be set to 1 for this command to work
				`#iwpriv wifix sHmmcDscpTidMap <tid>`
				`#iwpriv wifix gHmmcDscpTidMap`
				`gHmmcDscpTidMap:<tid>`

setBlkReportFld
getBlkReportFld
	iwpriv wifix setBlkReportFld *{1/0}*	Y	Y	To enable/disable report flooding. Enabling this feature would block flooding reports to other STAs associated with the AP.
				`#iwpriv wifix setBlkReportFld 1`
				`#iwpriv wifix getBlkReportFld`
				`getBlkReportFld: 1`

setDropSTAQuery
getDropSTAQuery
	iwpriv wifix setDropSTAQuery *{1/0}*	Y	Y	To enable/disable DropSTAQuery feature. Enabling feature would drop IGMP Querys from STA
				`#iwpriv wifix setDropSTAQuery 1`
				`#iwpriv wifix getDropSTAQuery`
				`getDropSTAQuery:1`

nopbn
get_nopbn
	iwpriv athX nopbn *{1/0}*	Y	Y	To disable VAPs being notified when jump start button gets pushed.
				`#iwpriv ath0 nopbn 1`
				`#iwpriv ath0 get_nopbn 1`
				`get_nopbn:1`

Table 1-32 Hy-Fi parameters

Parameter	Format	DA	OL	Description

set_dscp_ovride
get_dscp_ovride
	iwpriv athX set_dscp_ ovride *{1/0}*	Y	Y	To enable/disable dscp override feature. Packets with specific dscp value set can be mapped to a specific TID through this feature on per vap basis.
				`#iwpriv ath0 set_dscp_ovride 1`
				`#iwpriv ath0 get_dscp_ovride get_ dscp_ovride:1`

s_dscp_tid_map g_ dscp_tid_map	iwpriv athX s_dscp_tid_ map <dscp> (hex_val)>
	<tid (0-8)>
		Y	Y	To configure a specific tid for specific dscp value. Iwpriv option vap based set_dscp_ovride should be set to 1.
				`#iwpriv ath0 s_dscp_tid_map 0xe0 1`
				`#iwpriv ath0 g_dscp_tid_map 0xe0 g_ dscp_tid_map:1`

vappriority
get_vappriority
	iwpriv athX vappriority [0-3]
	iwpriv athX get_ vappriority
		Y	Y	To set outgoing traffic from a vap to specific priority. To use this feature, vap based dscp_override feature must be set.
				`#iwpriv ath0 vappriority 3`
				`#iwpriv ath0 get_vappriority`
				`get_vappriority:3`

1.3.28 Channel loading/Channel hopping parameters

Table 1-33 Channel loading/Channel hopping parameters

Parameter	Format	DA	OL	Description

acsmindwell
get_acsmindwell
	iwpriv ath*N* acsmindwell *value_in_ms*	Y	Y	Minimum time in milliseconds to spend on each channel even if channel is idle.
				`#iwpriv ath0 acsmindwell 100`
				`#iwpriv ath0 get_acsmindwell`
				`ath0 get_acsmindwell:100`

acsmaxdwell
get_acsmaxdwell
	iwpriv ath*N* acsmaxdwell *value_in_ ms*	N	N	Maximum time in milliseconds than can be spent on a channel. Default value is 300 msec.
				The value to be set should be greater than or equal to acsmindwell. So check the value of acsmindwell and choose a value accordingly, else the command returns error.
				`#iwpriv ath0 acsmaxdwell 100`
				`#iwpriv ath0 get_acsmaxwell`
				`ath0 get_acsmindwell:100`

acsreprt	Iwpriv ath*N* acsreport *value*	Y	Y	Enable (1) or disable (0) channel loading. `#iwpriv ath0 acsreport 1|0`

ch_hop_en
get_ch_hop_en
	iwpriv ath*N* ch_hop_en *{1|0}*	Y	N	Enables (1) or disables (0) channel hopping feature
				`#iwpriv ath0 ch_hop_en 1`
				`#iwpriv ath0 get_ch_hop_en`
				`ath0 get_ch_hop_en:1`

Table 1-33 Channel loading/Channel hopping parameters (cont.)

Parameter	Format	DA	OL	Description
ch_long_dur get_ ch_long_dur	iwpriv ath*N* ch_long_dur *value_in_seconds*	Y	N	Set/get long duration timer value in seconds #iwpriv ath0 ch_long_dur 60 #iwpriv ath0 get_ch_long_dur ath0 get_ch_long_dur:60
ch_nhop_dur get_ ch_nhop_dur	Iwpriv ath*N* ch_nhop_dur*{value in seconds}*	Y	N	Set/get no hop duration for channel hopping #iwpriv ath0 ch_nhop_dur 60 #iwpriv ath0 get_ch_nhop_dur ath0 get_ch_nhop_dur:60
ch_cntwn_dur get_ ch_cntwn_dur	Iwpriv ath*N* ch_cntwn_dur *{value in seconds}*	Y	N	Set/get counter window duration for channel hopping #iwpriv ath0 ch_cntwn_dur 60 #iwpriv ath0 g_ch_cntwn_dur g_ch_cntwn_dur:60
ch_noise_th get_ch_noise_th	Iwpriv athN ch_noise_th {value}	Y	N	Set/get noise threshold in dB iwpriv ath0 ch_noise_th -90 #iwpriv ath0 get_ch_noise_th get_ch_noise_th:-90
ch_cnt_th get_ch_cnt_th	Iwpriv athN ch_cnt_th {value}	Y	N	Set/get counter threshold iwpriv ath0 ch_cnt_th 60 iwpriv ath0 get_ ch_cnt_th get_ ch_cnt_th:60
use_custom_chan	iwpriv athx use_custom_chan 1 or 0	Y	N	Set/get custom channel list usage iwpriv athx use_custom_chan 1 or 0 iwpriv athx get_custom_chan 1 or 0

1.3.29 802.11k parameters

Table 1-34 802.11k Parameters

Parameter	Format	DA	OL	Description
rrm get_rrm	iwpriv ath*N* rrm {1|0}	Y	Y	Enables or disables 802.11k. Default is disabled. # iwpriv ath0 rrm 1 # iwpriv ath0 get_rrm get_rrm:1

1.3.30 Block channel list parameters

Table 1-35 Block Channel List Parameters

Parameter	Form	DA	OL	Descriptio

acs_bmode
g_acs_bmode
	iwpriv wifi*N* acs_bmode *{3|2|1|0}*	Y	Y	Sets the channel blocking mode. Setting bit 0 blocks the channel from manual selection, and setting bit 1 blocks this channel from being used as a secondary channel. By default, the channel is excluded from being selected as a primary channel when auto channel selection runs.
				`# iwpriv wifi0 acs_bmode 3`
				`# iwpriv wifi0 g_acs_bmode`

1.3.31 Aggregate size scaling parameters

Table 1-36 Aggregate Size Parameters

Parameter	Format	DA	OL	Description

acparams	iwpriv athN acparams
	{AC-0,1,2,3,4} {0}
	{Scaling factor: 0-3} {0}
		N	Y	Configures aggregate size scaling factor for the AC.
				`#iwpriv ath0 acparams 0 0 1 0`
				`#iwpriv ath0 acparams 2 0 1 0`

1.3.32 Wifitool Utility

Qualcomm Atheros provides proprietary Wifitool utility for Linux-based distribution. The primary purpose of this utility is to get stats and configure various features like 802.11k and channel loading or any other feature that requires a large number of parameters as input and output.

1.3.32.1 802.11k

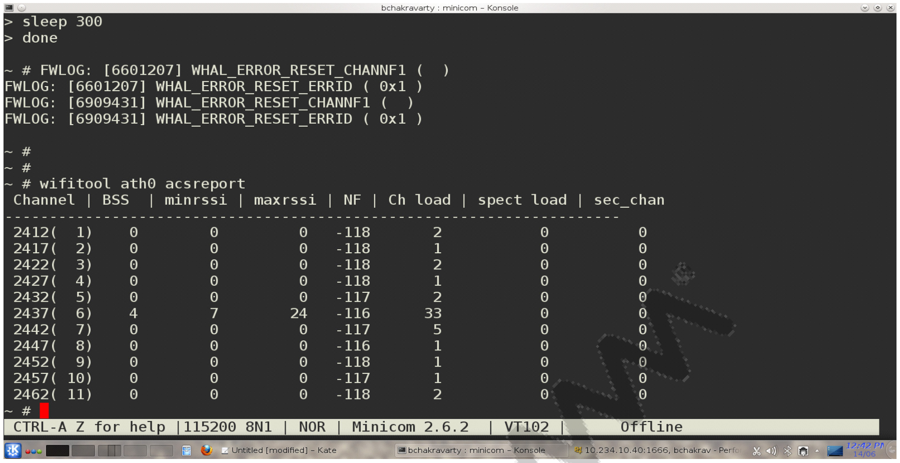
Table 1-37 Wifitool 802.11k parameters

Parameter	Format	Description
sendbcnrpt	wifitool *interface_name sendbcnrpt dest_mac bssid chan_num reg_class*	Beacon report ■ dest mac address: MAC address of associated station to which beacon request is sent. ■ bssid is the BSSID of desired AP (RSSI to be determined). ■ chan_num: chan number for which stats are to be determined ■ reg_class: reg class of the operating channel.
sendchload	wifitool *interface_name sendchload cmd reg_class destmac channel*	■ cmd: reserved for future use, in current implementation it should be passed as any positive value greater then zero. ■ reg_class: reg class of operating channel. ■ destmac: MAC address of associated station. ■ channel: channel on which we want station to calculate channel load.
sendstastats	wifitool *interface_name sendstastats dst_mac duration gid*	■ dst mac: MAC address of associated client ■ duration: interval for which we want to take this statistics. Value is in ms. ■ gid: group id, this value is taken from 802.11k specification.
sendnhist	wifitool *interface_name sendnhist dstmac duration regclass channel*	■ dst mac: MAC address of associated client. ■ duration: interval for which we want to take this statistics. Value is in msec. ■ regclass: reg class of operating channel. ■ channel: channel on which station will calculate channel load will be calculated.

1.3.32.2 Channel loading

Table 1-38 Wifitool channel loading parameters

Parameter	Format	Description
acsreport	wifitool *athN acsreport*	Get channel loading in user layers with the wifitool utility
setchanlist	wifitool athN setchanlist ch1 ch2….chN	To set list of channels for participating in the channel loading algorithm
getchanlist	wifitool athN getchanlist	To get the list of valid channels for channel loading
custom_chan_list	wifitool athX custom_chan_list [-a 1-101 1-165] [-n 1-101 1-165]	-a number of channels followed by IEEE number(s) of the channel(s) when STA is connected -n number of channels followed by IEEE number(s) of the channel(s)


bchakravarty : minicom – Konsole
Untitled [modified] – Kate
bchakravarty : minicom – Konsole
10.234.10.40:1666, bchakrav - Perfo
12:42 PM
14/06

```
> sleep 300
> done

~ # FWLOG: [6601207] WHAL_ERROR_RESET_CHANNF1 (   )
FWLOG: [6601207] WHAL_ERROR_RESET_ERRID ( 0x1 )
FWLOG: [6909431] WHAL_ERROR_RESET_CHANNF1 (   )
FWLOG: [6909431] WHAL_ERROR_RESET_ERRID ( 0x1 )

~ #
~ #
~ # wifitool ath0 acsreport
 Channel | BSS  | minrssi | maxrssi | NF | Ch load | spect load | sec_chan
-------------------------------------------------------------------
 2412(  1)    0        0         0   -118       2          0          0
 2417(  2)    0        0         0   -118       1          0          0
 2422(  3)    0        0         0   -118       2          0          0
 2427(  4)    0        0         0   -118       1          0          0
 2432(  5)    0        0         0   -117       2          0          0
 2437(  6)    4        7        24   -116      33          0          0
 2442(  7)    0        0         0   -117       5          0          0
 2447(  8)    0        0         0   -116       1          0          0
 2452(  9)    0        0         0   -118       1          0          0
 2457( 10)    0        0         0   -117       1          0          0
 2462( 11)    0        0         0   -118       2          0          0
~ #
 CTRL-A Z for help |115200 8N1 | NOR | Minicom 2.6.2   | VT102 |      Offline
```

Figure 1-1 Channel loading

1.3.32.3 Block channel list

To block any set of channel from participating in the ACS algorithm, this command can be used.

Table 1-39 Block channel list

Parameter	Format	Description
block_acs_ channel	wifitool athN block_acs_channel channel1,channel2,channel3……channel N	Set list of channels to be blocked from the ACS channel selection

Max value for N is 255.

After successful execution of this API, user should view the following log on console:

“Following channels are blocked from ACS”

[channel 1] [channel2]…….[channel]

Every successful execution of this command will amend the previously stored list in the driver. If the user wants to flush the previously stored list, then they should execute this command with single channel with value as zero.

e.g

```
Wifitool ath0 block_acs_channel 1,2,3
Wifitool ath0 block_acs_channel 4,5,6
```

Will block ACS to block channel 1 2 3 4 5 6 to participate in acs channel selection, where as

```
Wifitool ath0 block_acs_channel 1,2,3
Wifitool ath0 block_acs_channel 0
Wifitool ath0 block_acs_channel 4,5,6
```

Will block only channel 4, 5, and 6

1.3.32.4 FIPS validation

wifitool tool can be used to validate FIPS implementation in QCA900B. Tool expects input in the following format and provides pass or fail.

<CommandID><space><Mode><space><KeyLength><space><InputDataLength><Key><InputData><ExpectedOutput><space><IV><newline>

Following is an example text file:

```
Fips.txt
0 1 16 16 2b7e151628aed2a6abf7158809cf4f3c
6bc1bee22e409f96e93d7e117393172a 3ad77bb40d7a3660a89ecaf32466ef97
f0f1f2f3f4f5f6f7f8f9fafbfcfdfeff
```

And run the following command

```
#wifitool ath0 fips input_file
```

1.3.32.5 Chainmask per client

wifitool tool can be used to set chainmask per client in QCA900B. Tool expects input in the following format and provides pass or fail.

```
wifitool athX chmask_persta <mac_addr><nss>
```

Run the following command

```
#wifitool ath0 chmask_persta 00:34:12:34:56:78 4
```

1.3.32.6 Set antenna switch

wifitool tool can be used to set antenna switch in QCA900B. Tool expects input in the following format and provides pass or fail.

```
wifitool athx set_antenna_switch <ctrl_cmd_1> <ctrl_cmd_2>
```

Run the following command

```
# wifitool athx set_antenna_switch 1 1
```

1.3.32.7 Set user control table

wifitool tool can be used to set user control table in QCA900B. Tool expects input in the following format and provides pass or fail.

```
wifitool athX set_usr_ctrl_tbl val1 val2 .....n
```

Run the following command

```
# wifitool athx set_antenna_switch 2,3,4,5…n
```

1.3.32.8 Block Acknowledge

Table 1-40 Block Acknowledge

<table>
<tr><th>Parameter</th><th>Format</th><th colspan="2">Description</th></tr>
<tr><td><b>sendaddba</b>
senddelba</td><td>wifitool ath<i>N</i> sendaddba AID <i>AC BufSize</i>
wifitool ath<i>N</i> senddelba AID AC initiator reason</td><td colspan="2">Test commands used to manually add or delete block acknowledge aggregation streams. Automatic addba/delba processing must be turned off prior to using these commands (see <b>setaddbaoper</b>). Both require the AID and AC specified. The AID value is shown by the <b>wlanconfig list</b> command. When adding an aggregation link with addba, BufSize must be set to the maximum number of subframes sent in an aggregate. When deleting aggregation links, the initiator field indicates whether this link was initiated by the AP (1) or the remote STA (0). The 8-bit code indicates the reason the link shut down. No corresponding get parameters or default values.
<code>#wifitool ath0 sendaddba 1 0 32</code>
<code>#wifitool ath0 senddelba 1 0 1 36</code></td></tr>
<tr><td>setaddbaresp</td><td>wifitool ath<i>N</i> <i>set</i>addbaresp AID <i>AC status</i></td><td colspan="2">Sends an addba response frame on the indicated AID and AC. AID is the value shown under the AID column using the command <b>wlanconfig list</b>. The status value is an 8-bit value indicating the status field of the response. Normally used only during testing of the aggregation interface. The command does not have a corresponding get parameter, nor does it have a default value.
<code>#wifitool ath0 setaddbaresp 1 0 25</code></td></tr>
<tr><td rowspan="3">getaddbastats</td><td rowspan="3">wifitool ath<i>N</i> getaddbastats <i>status</i></td><td colspan="2">Gets the ADDBA (Add Block Acknowledgment) status for AID (Association Identifier) and TID (Traffic Identifier).
<i>What is the format of aid, tid when returned?</i>
<code>#wifitool ath0 getaddbastats aid ath0 getaddbastats:</code></td></tr>
<tr><td>aid</td><td>AID number of STA</td></tr>
<tr><td>tid</td><td>TID number between 0-15</td></tr>
</table>

1.3.33 Target recovery parameters

Parameter	Format	DA	OL	Description

seth.get_ fwrecovery	iwpriv wifi1 *set_fw_ recovery {1|0}*	Y	Y	Enables (1) or disables (0) the target recovery mechanism for the QCA9880 radio
				`#iwpriv wifi1 set_fw_recovery 1 (enable)`
				`#iwpriv wifi1 set_fw_recovery 0 (disable)`

get_fw_ recovery	iwpriv wifi1 get_fw_ recovery	Y	Y	This parameter is used to check if the target recovery mechanism is enabled or disabled.

1.3.34 Uncategorized radio layer parameters

NOTE Most commands are not applicable for Partial Offload, until explicitly mentioned.

Table 1-41 Uncategorized radio layer parameters

Parameter	Format	Description
acktimeout **get_acktimeout**	*iwpriv* `wifi`*N acktimeout*	Need description.
amemPrint	*iwpriv* `wifi`*N* amemPrint	Note: Applicable for Partial Offload
ATHdebug getATHdebug	*iwpriv* `wifi`*N ATHdebug*	Need description.
DisASPMWk **get_**DisASPMWk	*iwpriv* `wifi`*N DisASPMWk*	Need description.
DisPACal **get_**DisPACal	*iwpriv* `wifi`*N DisPACal*	Need description.
DisTurboG **get_**DisTurboG	*iwpriv* `wifi`*N DisTurboG*	Need description.
EnaASPM **get_EnaASPM**	*iwpriv* `wifi`*N EnaASPM*	Need description.
FIRStepLvl **get_FIRStepLvl**	*iwpriv* `wifi`*N FIRStepLvl*	Need description.
immunity **get_immunity**	*iwpriv* `wifi`*N immunity*	Need description.
LDPC **getLDPC**	*iwpriv* `wifi`*N LDPC*	Need description.
limit_legacy **get_limit_legacy**	*iwpriv* `wifi`*N* limit_legacy	Need description.
PCIEClkReq PCIEClkReq	*iwpriv* `wifi`*N* PCIEClkReq	Need description.
PCIEDETACH PCIEDETACH	*iwpriv* `wifi`*N* PCIEClkReq	Need description.
PCIEL1SKPEn PCIEL1SKPEn	*iwpriv* `wifi`*N* PCIEL1SKPEn	TBD
PCIEPwRset PCIEPwRset	*iwpriv* `wifi`*N* PCIEPwRset	TBD
PCIEPwrSvEn PCIEPwrSvEn	*iwpriv* `wifi`*N* PCIEPwrSvEn	TBD
PCIERestore PCIERestore	*iwpriv* `wifi`*N* PCIERestore	TBD
PCIEWAEN PCIEWAEN	*iwpriv* `wifi`*N* PCIEWAEN	TBD
rb **get_rb**	*iwpriv* `wifi`*N* rb	TBD
rbdetect **get_rbdetect**	*iwpriv* `wifi`*N* rbdetect	TBD
rbskipthresh get_rbskipthresh	*iwpriv* `wifi`*N* rbskipthresh	TBD

Table 1-41 Uncategorized radio layer parameters (cont.)

Parameter	Format	Description
rbto **get_rbto**	*iwpriv* `wifi`*N* rbto	TBD
RegRead_base GetRegRead	*iwpriv* `wifi`*N* RegRead_base	TBD
rximt_first get_rximt_first	*iwpriv* `wifi`*N* rximt_first	TBD
rximt_last get_rximt_last	*iwpriv* `wifi`*N* rximt_last	TBD
rxstbc get_rxstbc	*iwpriv* `wifi`*N* rxstbc	TBD
set_ledcustom get_ledcustom	*iwpriv* `wifi`*N* *set_ledcustom*	TBD
set_swapled get_swapled	*iwpriv* `wifi`*N* set_swapled	TBD
setHALparam getHALparam	*iwpriv* `wifi`*N* *setHALparam*	TBD
setPhyRestartWar getPhyRestartWar	*iwpriv* `wifi`*N* *setPhyRestartWar*	TBD
tpscale get_tpscale	*iwpriv* `wifi`*N* tpscale	**NOTE** Applicable for Partial Offload Tx Power scaling. Valid values are [0-4]
tximt_first get_tximt_first	*iwpriv* `wifi`*N* *tximt_first*	TBD
tximt_last get_tximt_last	*iwpriv* `wifi`*N* *tximt_last*	TBD
txstbc get_txstbc	*iwpriv* `wifi`*N* get_txstbc	TBD

1.3.35 Uncategorized protocol layer parameters

Table 1-42 Uncategorized protocol layer parameters

Parameter	Format	Description	

htweptkip
get_htweptkip
	`iwpriv ath`*N*
	`htweptkip`
	1|0
		Enable/disable 11n support in WEP or TKIP mode. The get parameter returns the current value.
		`#iwpriv ath0 htweptkip 1`
		`#iwpriv ath0 get_htweptkip`
		`ath0 get_htweptkip:1`	

		0	Disable
		1	Enable

Table 1-42 Uncategorized protocol layer parameters (cont.)

Parameter	Format	Description
rc_vivo get_rcvivo	`iwpriv athN rc_vivo`	Removed in latest releases.
setparam getparam	iwpriv athN setparam	For sub-ioctl handlers, and usually not used directly. For example, `iwpriv ath0 ampdu 0` should be equivalent to `iwpriv ath0 setparam 73 0`. (Internal note: `iwpriv athN ampdu` and `iwpriv athN get_amdpu` have been deemed unnecessary to document. Use `iwpriv wifiN AMPDU` and `iwpriv wifiN getAMPDU` instead.)

1.3.36 2.4 GHz VHT 256-QAM Broadcom interoperability support

Table 1-43 lists the parameters for 2.4 GHz VHT 256-QAM Broadcom interoperability support.

Table 1-43 2.4 GHz VHT 256-QAM Broadcom interoperability support

Parameter	Format	DA	OL	Description

11ngvhtintop
g_11ngvhtintop
	iwpriv ath*N* *11ngvhtintop* {1/0}}	N	Y	Enables (1) or disables (0) 2.4 GHz 256-QAM interoperability support with Broadcom based devices. The default value is 0. This command enables VHT 256-QAM rate support with Broadcom based devices.
				The get parameter returns the current value.
				`#iwpriv ath0 11ngvhtintop 1`
				`#iwpriv ath0 g_11ngvhtintop`

1.3.37 QWRAP

Table 1-44 lists the QWRAP debug parameters.

Table 1-44 QWRAP debug

Parameter	Format	DA	OL	Description
get_proxysta	iwpriv ath*N* get_proxysta	Y	Y	This is a debug command to check whether VAP is proxy sta vap or not in QWRAP. No set command available

mcast_echo
g_mcast_echo
	iwpriv wifi*N* mcast_echo {1/0	N	Y	This command is use to set multicast/broadcast echo support for physical device in Qwrap isolation mode.
				The set and get parameter returns the current value.
				`#iwpriv ath0 mcast_echo 1`
				`#iwpriv ath0 g_mcast_echo`

1.3.38 Airtime Fairness (ATF) Parameters

Table 1-45 lists the ATF parameters.

Table 1-45 ATF parameters

Parameter	Format	DA	OL	Description

commitatf	*iwpriv athN*
	commitatf {1/0}
		Y	Y	This is commit command, it must be issued once user finishes any setting for ssid/sta percentage configuration by wlanconfig tools.
				The example is the following.
				`#iwpriv ath0 commitatf 1`
				`/*setting effective*/`
				`#iwpriv ath0 commitatf 0`
				`/*setting ineffective*/`

get_commitatf	iwpriv ath*N* get_commitatf	Y	Y	This command displays the value set for commitatf Usage: `Iwpriv ath0 get_commitatf: Displays if commitatf is set or cleared.`

atfstrictsched	iwpriv wifi*N* atfstrictsched
	{1/0}
		Y	Y	This command is for enabling or disabling ATF strict scheduling.
				Example command:
				`/* enable strict scheduling */`
				`#iwpriv wifi0 atfstrictsched 1`
				`/* disable strict scheduling – enabled Fair queue scheduling */`
				`#iwpriv wifi0 atfstrictsched 0`
				NOTE Fair-queue ATF Scheduling is enabled by default in both direct attach and partial offload architecture

gatfstrictsched	iwpriv wifi*N* gatfstricsched	Y	Y	The command is used to check whether ATF strict scheduling is enabled or disabled. Example command: `# iwpriv wifi0 gatfstrictsched wifi0 gatfstrictsched:1`

atfgrouppolicy	iwpriv wifiN
	atfgrouppolicy
		Y	Y	The command is used for selecting the inter group scheduling policy (Across groups)
				Example command:
				`/* enable strict scheduling across groups */`
				`#iwpriv wifi0 atfgrouppolicy 1`
				`/* disable strict scheduling across groups –`
				`enabled Fair queue scheduling across*/`
				`#iwpriv wifi0 atfgrouppolicy 0`
				NOTE Fair-queue ATF scheduling across groups is enabled by default in both direct attach and partial offload architecture

Table 1-45 ATF parameters

Parameter	Format	DA	OL	Description
gatfgrouppolicy	iwpriv wifiN gatfgrouppolicy	Y	Y	The command is used to checking the inter group scheduling (Across groups) Example command: `# iwpriv wifi0 gatfgrouppolicy wifi0gatfgrouppolicy:1` **NOTE** denotes strict scheduling across groups and 0 denotes fair scheduling across groups. This is the same for both direct attach and partial offload architecture.
atf_sched_dur	iwpriv wifi*N* atf_sched_ dur ac {0-3} dur {ac is the access category and dur is the token}	N	Y	This command is used to set the number of tokens to be allocated for a particular access category Example Command: `iwpriv wifi1 atf_sched_dur 2 5` The first parameter is the Access Category which should be between 0 and 3 0 – Best Effort 1 – Background – Video – Voice The second parameter should be the number of tokens which should be a positive value.

atfobsssched	iwpriv wifiN atfobsssched 1/0	Y	N	This command is used to enable ATF OBSS module, which considers interference from other APs, before distributing the tokens to the associated STAs
				Example Command:
				`#iwpriv wifi0 atfobsssched 1`
				`/* enable OBSS scheduling */`
				`#iwpriv wifi0 atfobsssched 0`
				/* disable OBSS scheduling */

g_atfobsssched	iwpriv wifiN g_ atfobsssched	Y	N	This command is used to get the current state of ATF OBSS module, if it is enabled or not. `# iwpriv wifi0 gatfobsssched wifi0 gatfobsssched:0`

atf_shr_buf	iwpriv ath*N* atf_shr_ buf
	{1/0}
		Y	N	This command is used to enable/disable sharing of Tx Buffers between the clients in the ratio of airtime.
				Example command:
				`#iwpriv ath0 atf_shr_buf 1`
				`/* enable Tx Buffer sharing */`

g_atf_shr_buf	iwpriv ath*N* g_atf_shr_ buf	Y	N	This command is used to query whether Tx Buffer sharing between clients is enabled or not. Example Command: `#iwpriv ath0 g_atf_shr_buf ath0 g_atf_ shr_buf:1`
atfmaxclient	iwpriv ath*N* atfmaxclient	Y	N	This command is used to enable maxclient support on direct attach architecture. This feature is disabled by default `#iwpriv ath0 atfmaxclient`

Table 1-45 ATF parameters

Parameter	Format	DA	OL	**Description**
g_atfmaxclient	iwpriv ath*N g_* atfmaxclient	Y	N	This command is used to query whether maxclient support is enabled `#iwpriv ath0 g_atfmaxclient`
atfssidgroup	iwpriv athN atfssidgroup 1/0	Y	Y	This command is used to enable or disable ssid grouping feature. `#iwpriv ath0 atfssidgroup 1/0`
g_atfssidgroup	iwpriv athN g_ atfssidgroup	Y	Y	This command is used to query whether ssid grouping is enabled or not. `#iwpriv ath0 g_atfssidgroup`

atfssidsched	iwpriv ath*N*
	atfssidsched
	{1/0}
		Y	N	This command is for setting per SSID ATF scheduling policy. Default set to 0.
				Example command:
				`/* enable strict scheduling */`
				`#iwpriv ath0 atfssidsched 1`
				`/* disable strict scheduling – enabled Fair queue scheduling */`
				`#iwpriv ath0 atfssidsched 0`

g_atfssidsched	iwpriv ath*N* g_ atfssidsched	Y	N	This command is used to query current per SSID scheduling policy. #iwpriv ath0 g_atfssidsched

NOTE In ATF SSID grouping the policies within and across the groups should be given after the ssid is added to a group and the groups are configured with certain percentage.

A combination of Strict-queue scheduling within group and Fair-queue scheduling across groups is invalid. i.e.,

```
iwpriv wifiN atfstrictsched 1
iwpriv wifiN atfgrouppolicy 0
```

The above configuration is invalid.

The following is the example of the sequence of commands for two ssids and two groups

```
wlanconfig athx addatfgroup private ssid1
wlanconfig athx addatfgroup public ssid2
wlanconfig athx configatfgroup private 80
wlanconfig athx configatfgroup public 20
iwpriv wifiN atfgrouppolicy 1
iwpriv wifiN atfstrictsched 0
```

1.3.39 Wake on wireless – AP assist

Table 1-46 lists the wake-on-wireless AP assist parameters.

Table 1-46 Wake-on-wireless AP assist parameters

Parameter	Format	DA	OL	Description
sendwowpkt	iwpriv ath*N* sendwowpkt <mac addr >	N	Y	This command sends WoW magic packet to specified associated node. No get command available

1.3.40 Dynamic Frequency Selection (DFS) parameters

Table 1-47 DFS parameters

Parameter	Format	DA	OL	Description

`staDFSEnable`	iwpriv wifiN
	`staDFSEnable`
	{1/0}
		Y	Y	For a radio (in STA mode) whose TX Power > 23dBm should support DFS. STA mode DFS can be enabled/Disabled by using the command.
				NOTE At present, STA mode CAC is performed only for ETSI domain.
				Usage:
				`#iwpriv wifi0 staDFSEnable 1`
				`#iwpriv wifi0 staDFSEnable 0`

`getstaDFSEnable`	iwpriv wifiN
	`getstaDFSEnable`
		Y	Y	This command is used to query whether sta mode DFS is Enabled
				Usage:
				`#iwpriv wifi1 getstaDFSEnable wifi1get_`
				`staDFSEnable:0`

Table 1-48 Repeater DFS channel switch options

Parameter	Format	DA	OL	Description

CSwOpts	iwpriv wifiN CSwOpts {1/2/4/8/10/20/F/0}	Y	Y	This command is used to set the channel switch options in Repeater AP. Each bit in the parameter value represents one option. Below are the options:
				1. IEEE80211_CSH_OPT_NONDFS_RANDOM 0x00000001. Enable Random selection of channel from Non-DFS channel list when local radar detected. If a Non-DFS channel is not found in the list it chooses a DFS channel.
				2. IEEE80211_CSH_OPT_IGNORE_CSA_DFS 0x00000002. When a CSA comes from Root AP and the channel in the CSA is DFS, choose a new random Non-DFS channel and use this new channel in the CSA announcement in Repeater AP's BSS.
				3. IEEE80211_CSH_OPT_CAC_APUP_BYSTA 0x00000004. Do a CAC before joining Root AP if the Root AP is in DFS channel.
				4. IEEE80211_CSH_OPT_CSA_APUP_BYSTA 0x00000008. Announce a CSA in Repeater BSS when the Repeater connects to the Root AP and the Root AP's channel is not same as Repeater AP's channel.
				5. IEEE80211_CSH_OPT_RCSA_TO_UPLINK 0x00000010. Enable sending RCSA to uplink/parent/Root AP when a repeater AP detects RADAR.
				6. IEEE80211_CSH_OPT_PROCESS_RCSA 0x00000020. Enable processing of RCSA from any downlink/client/RE/Repeater AP.
				Both IEEE80211_CSH_OPT_RCSA_TO_UPLINK and IEEE80211_CSH_OPT_PROCESS_RCSA bits should be set for the Repeater and Root to work together and to change the channel after RADAR detection. The bit IEEE80211_CSH_ OPT_PROCESS_RCSA can be disabled to simulate the situation that Root/uplink does not respond to the RCSA.
				Usage:
				Enable option 1
				`#iwpriv wifi0 CSwOpts 0x1`
				Enable option 2
				`#iwpriv wifi0 CSwOpts 0x2`
				Enable option 5 and 6
				`#iwpriv wifi0 CSwOpts 0x30`

get_CSwOpts	iwpriv wifiN get_ CSwOpts	Y	Y	This command is used to query the channel switch options enabled in Repeater AP. `#iwpriv wifi0 get_CSwOpts`

1.3.41 NSS Offload

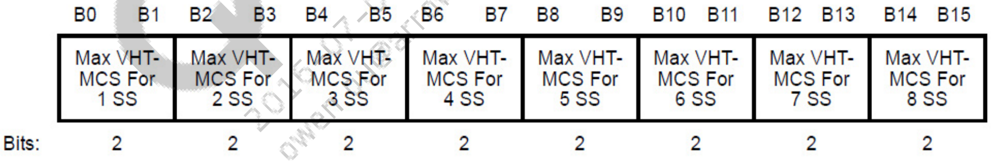
Table 1-49 NSS Offload - Tx Flow Control buffer pool size

Parameter	Format	DA	OL	Description

fc_bufX_ max
X : {0/1/2/3}
	iwpriv wifiN
	fc_bufX_ max
	N – 0/1/2 (Radio_ num)
	X – 0/1/2/3 (Peer Range)
		N	Y	This command is used to set the Max Buffer Queue Size for WiFi Tx Queue (used in peer based flow control). In NSS Offload mode, this prevents NSS hoarding buffers greater than the number configured for a given peer range.
				This parameter should be set for 4 different peer ranges -
				0: 1 – 64 Clients
				1: 65 – 128 Clients
				2: 129 – 256 Clients
				3: > 256 Clients
				Usage (for 5 GHz radio)
				`#iwpriv wifi0 fc_buf0_max 8192`
				`#iwpriv wifi0 fc_buf1_max 16384`
				`#iwpriv wifi0 fc_buf2_max 24576`
				`#iwpriv wifi0 fc_buf3_max 32768`
				E.g., When number of clients connected to 5 GHz radio are in 1-64 range, the Tx queue is limited to 8192, and NSS would request buffers accordingly As more number of clients connect, the max Tx queue size is adjusted accordingly and NSS would request more buffers as it moves to upper peer range. 2 GHz radio generally requires half the buffers as required by 5 GHz radio.
				NOTE If these parameters are increased from the default values, following NSS related parameters also need to be changed accordingly.
				■ n2h_high_water_core0 – The high_water parameter is the limit on total number of network buffers that NSS offload processor0 can use at any point of time. The default value is set to 59392, which comprises of 32768 for 5 GHz radio Tx queues, 16384 for 2 GHz Tx Queues and rest 10240 required for general WiFi and NSS operation.
				Usage:
				`# sysctl -w dev.nss.n2hcfg.n2h_high_water_core0=59392`
				■ n2h_wifi_pool_buf – This parameters specifies the number of buffers allocated for WiFi Tx Queues, of the total pool size specified as n2h_high_water_core0. Note that 8192 buffers for 5 GHz and 4096 buffers for 2 GHz are already allocated during WiFi initialization, so 12288 needs to subtracted from total WiFi requirement. E.g. if 5 GHz radio requires 32768 (assuming max peers i.e. > 256) and 2 GHz radio requires 16384(assuming max peers i.e. > 256) then wifi pool size should be set as (32768+16384-12288) = 36864
				Usage :
				`# sysctl -w dev.nss.n2hcfg. n2h_wifi_pool_buf=36864`

1.3.42 Disable Selected MCS For Given SSID iwpriv options

Parameter`	Format	DA	O	Description
disable11nmcs	iwpriv athX disable11nmcs 0xFFFFFFF0	Y	Y	Every bit in this command represents a MCS rate. LSB bit 0 represents and MSB bit 31 represents MCS 31. If a particular bit is set, then that MCS rate will be disabled for that vap.
g_disable11nmcs	iwpriv athX g_ disable11nmcs	Y	Y	iwpirv athX g_disable11nmcs will display the configured value on the vap.
conf_11acmcs	iwpriv athX conf_ 11acmcs 0xFFFFFFF5	N	Y	The LSB 16 bits passed to conf_11acmcs represents the 16 bits in Rx MCS Map and Tx MCS Map field of VHT Capabilities IE. The MSB 16 bits should be 0xFFFF always. Please refer the below picture1 on how to interpret the LSB bits of Rx MCS Map and Tx MCS Map fields of VHT Capabilities IE. Only those bits are that valid w.r.t the number of tx and rx spatial streams will be considered.
g_conf_11acmcs	iwpriv athX g_conf_ 11acmcs	N	Y	iwpirv athX g_conf_11acmcs will display the configured value on the vap



iwpirv athX conf_11acmcs & g_conf_11acmcs configuration examples.

```
Iwpriv athX conf_11acmcs 0xFFFFFFC0
```

This command enables VHTMCS 0 – 7 in Spatial Streams 1,2,3 and says Spatial Streams (SS) 4 to 8 is not supported.

```
Iwpriv athX conf_11acmcs 0xFFFFFFF5
```

This command enables VHTMCS 0 – 8 in Spatial Streams 1,2 and says Spatial Streams (SS) is not 3 to 8 supported.

```
Iwpriv athX conf_11acmcs 0xFFFFFFFA
```

This command enables VHTMCS 0 – 9 in Spatial Streams 1,2 and says Spatial Streams (SS) is not 3 to 8 supported.

```
iwpriv athX   g_conf_11acmcs
```

This command will display the configured value on the vap

iwpirv athX *disable11nmcs & g_disable11nmcs* configuration examples.

```
iwpriv athX disable11nmcs 0xFFFFFFF0
```

This command disables HTMCS 4 to 31 and enables only MCS 0 to 3.

```
iwpriv athX disable11nmcs 0xFFFF0FFF
```

This command enables MCS 12 to 15 and disables rest of the HTMCS rates.

```
iwpirv athX   g_disable11nmcs
```

This command displays the configured value on the VAP.

1.3.43 Disable selected legacy rates for given SSID (Using iwpriv)

There are 12 legacy rates and every bit in the value passed in iwpriv command represents a legacy rate. LSB bit 0 (B0) represents 1 Mbps and MSB bit (B11) represents 54 Mbps. If a particular bit is set, then that legacy rate will be disabled for that vap.

For 802.11 B

B3	B2	B1	B0
11	5.5	2	1

For 802.11G / 802.11NG

B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
54	48	36	24	18	12	9	6	11	5.5	2	1

For 802.11A/802.11NA/802.11AC

B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
54	48	36	24	18	12	9	6	Reserved	Reserved	Reserved	Reserved

List of legacy rates per desired modes

Desired mode	Legacy rates (in Mbps)	Basic rates (in Mbps)
802.11b	1,2,5.5,11	1,2,5.5,11
802.11G, 802.11NG	1,2,5.5,11,6,9,12,18,24,36,48,54	1,2,5.5,11,6,12,24
802.11A,802.11NA,802.11AC	6,9,12,18,24,36,48,54	6,12,24

IWPRIV command

iwpriv athX dis_legacy 0x (value) : Disables selected legacy rates for the VAP

iwpriv athX g_dis_legacy : Displays the configured value on the VAP

Examples

- iwpriv athX dis_legacy 0x9

If the desired PHY mode is 802.11B /802.11G/802.11NG, then this command disables 1 Mbps and 11 Mbps from the supported legacy rate set and sets the management frame rate and RTS / CTS rate to 2 Mbps. If AP is in 802.11b mode then data frames will get transmitted at 5.5 Mbps.

If the desired PHY mode is 802.11A / 802.11NA/ 802.11AC, then the above command do not disable any rates since the first four bits that represent: 1 Mbps, 2 Mbps, 5.5 Mbps and 11 Mbps are not available in case of 802.11A, 802.11NA and 802.11AC. Hence it is required to pass a value greater than 0xF.

- iwpriv athX dis_legacy 0x1f

NOTE The above command is not applicable in 802.11b mode since a user is not allowed to disable all the basic supported rates.

If the desired PHY mode is 802.11G or 802.11NG, then this command disables all CCK rates (802.11b basic rates) along with 6 Mbps from the supported legacy rate set and sets the management frame rate to 12 Mbps instead of 9 Mbps, since 12 Mbps is the next basic rate after 6 Mbps.

If the desired PHY mode is 802.11A / 802.11NA/ 802.11AC, then the above command disables 6 Mbps rate only and sets the management frame rate to 12 Mbps.

- iwpriv athx dis_legacy 0x0

This command enables all the legacy rates and sets the management frame rate to default rate i.e: 1 Mbps (802.11b/802.11g/802.11ng) and 6 Mbps (802.11a/802.11na/802.11ac).

NOTE Do not disable all the basic supported rates.

Example:

```
iwpriv athX dis_legacy 0x000f (Not allowed in 11b only)
iwpriv athX dis_legacy 0x015f (Not allowed in 11b, 11G, 11NG, 11A, 11NA,
11AC)
iwpriv athX dis_legacy 0x0150 (Not allowed in 11A, 11NA, 11AC)
```

1.3.44 Revised signaling for 160/80+80 MHz

Parameter	Format		Description

revsig160
get_revsig160
	iwpriv athN revsig160 *1|0*	Enable/disable revised signaling for 160 & 80+80 MHz
		Default value is 1
		`#iwpriv ath0 revsig160 0`
		`#iwpriv ath0 get_revsig160`	

		0	Disable revised signaling for 160 & 80+80 MHz
		1	Enable revised signaling for 160 & 80+80 MHz

1.3.45 AP Diagnostics for Carrier

Parameter	Format	DA	OL	Description
set_diag_enable	iwpriv wifiN set_diag_enable {1/0}	Y	Y	AP diagnostics can be enabled/disabled by using this command. Usage: #iwpriv wifi0 set_diag_enable 1
get_diag_enable	iwpriv wifiN get_diag_enable	Y	Y	This command is used to query whether AP diagnostics is enabled. Usage: #iwpriv wifi0 get_diag_enable
set_err_thres	iwpriv athN set_err_thres {datarate in mbps}	Y	Y	This command is used to set the error condition data rate threshold. Default value is 26 Usage: #iwpriv ath0 set_err_thres 13
get_err_thres	iwpriv athN get_err_thres	Y	Y	This command is used to get the configured error condition data rate threshold. Usage: #iwpriv ath0 get_err_thres
set_warn_thres	iwpriv athN set_warn_thres {datarate in mbps}	Y	Y	This command is used to set the warning condition data rate threshold. Default value is 65 Usage: #iwpriv ath0 set_warn_thres 52
get_warn_thres	iwpriv athN get_warn_thres	Y	Y	This command is used to get the configured warning condition data rate threshold. Usage: #iwpriv ath0 get_warn_thres

1.3.46 Reject clients with low SNR

Parameter	Format	D	O	Description
set_min_snr_en	iwpriv wifiN set_min_snr_en {1/0}	Y	Y	Reject clients with low SNR facility can be enabled/disabled by using this command. Usage: #iwpriv wifi0 set_min_snr_en 1
get_min_snr_en	iwpriv wifiN get_min_snr_en	Y	Y	This command is used to query whether reject clients with low SNR facility is enabled. Usage: #iwpriv wifi0 get_min_snr_en
set_min_snr	iwpriv wifiN set_min_snr {SNR value in dB}	Y	Y	This command is used to set the client reject snr threshold. Usage: #iwpriv wifi0 set_min_snr 15
get_min_snr	iwpriv wifiN get_min_snr	Y	Y	This command is used to get the configured client reject snr threshold Usage: #iwpriv wifi0 get_min_snr

1.3.47 VLAN tagging

Following commands provide control to enable/disable VLAN Tagging to be sent out over Wi-Fi This control is applicable only for TX packets to be sent out over VLAN interfaces created over Wi-Fi (i.e. for ath0.1, ath0.2 etc.)

Parameter	Format	D	O	Description
vlan_tag	iwpriv athN vlan_tag {1/0}	Y	Y	To enable VLAN Tagging: #Iwpriv athN vlan_tag 1 To disable VLAN Tagging: #Iwpriv athN vlan_tag 0
get_vlan_tag	iwpriv athN get_vlan_tag	Y	Y	This command is used get the current state (enabled/disabled) of VLAN tagging for the interface. Usage: #iwpriv athN get_vlan_tag
vlan_tag	iwpriv athN vlan_tag {1/0}	Y	Y	This command is used to enable/disable the VLAN Tag to be sent out over Wi-Fi Usage: Following enables VLAN Tag to be sent out over WIFI for the VLAN interface created over WIFI (i.e. for ath0.1, ath0.2 etc.) #Iwpriv athN vlan_tag 1 Following disables VLAN Tag to be sent out over WIFI even for the VLAN interface created over Wi-Fi. #Iwpriv athN vlan_tag 0

Parameter	Format	DA	OL	Description
get_vlan_tag	iwpriv athN get_vlan_tag	Y	Y	This command is used get the current state (enabled/disabled) of VLAN tagging for the interface. Usage: `#iwpriv athN get_vlan_tag`

1.3.48 Mesh Support

Following commands provide control to configure mesh

Parameter	Format	D	O	Description
addlocalpeer	iwpriv athN addlocalpeer *0x<peer flags><byte5><byte 4> 0x<byte3><byte2>< byte1><byte0>*	N	Y	To add peer: `#iwpriv athN` *0x12040034 0x56788797 mac addr 00:34:56:78:87:97 and peer flags 0x1204* Each bit in peer flags indicate below capability. Dual Stream support BIT(0) i.e 0x1 Three Stream support BIT(1) i.e 0x02 Four Stream support BIT(2) i.e 0x04 Reserved bits BIT(3) to BIT(7) 0xF8 Peer support HT20 BIT(8) i.e 0x0100 Peer support HT20 & HT40 BIT(9) i.e 0x0200 Peer support 11ACVHT20 BIT(10) i.e 0x0400 Peer support 11ACVHT40 BIT(11) i.e 0x0800 Peer support 11ACVHT80 BIT(12) i.e 0x1000 Peer support 11ACVHT80_80 BIT(13) i.e 0x2000 Peer support 11ACVHT160 BIT(14) i.e 0x4000 Reserved bit BIT(15) ie. 0x8000
Allowdata	Iwpriv athN allowdata 0x<byte5><byte4> 0x<byte3><byte2><byte 1><byte0>	N	Y	Authorize key for the local peer. Usage: `# iwpriv athN allow data 0x8cfd 0xf0021f9c`
rmode_pktsim	iwpriv athN rmode_ pktsim	N	Y	Allow packets in rawmode Usage `# iwpriv athN rmode_pktsim 1`
Meshdbg	iwpriv athN meshdbg	N	Y	BIT usage: To add the meta header in Ethernet format BIT(0) i.e 0x1 Rx debug info BIT(1) i.e 0x2 To set the meta header BIT(16-31) Usage `iwpriv athN meshdbg 1`

Parameter	Format	D	O	Description
Conf_meshtx	Iwpriv athN conf_meshtx	N	Y	Prevent the mesh vap from sending disassociation frame Usage: `# iwpriv conf_meshtx 1`
Mesh_rxfilter	Iwpriv athN mesh_rxfilter	N	Y	To apply Rx filters for mesh Below is the bitwise info: Drop DS Frames BIT(0) i.e 0x1 Drop ToDS Frames BIT(1) i.e0x2 Drop NoDS Frames BIT(2) i.e 0x4 Drop RA Frames BIT(3) i.e0x8 Drop TA Frames BIT(4) i.e 0x10

en_ol_txstats	Iwpriv wifiN en_ol_ txstats	N	Y	To get Tx completion handler stats
				Ol stats have to be enabled before enabling these stats
				Usage:
				`iwpriv wifiN enable_ol_stats 1`
				`iwpriv wifiN en_ol_txstats 1`

1.3.49 DBDC Repeater Support

Following commands provide control to configure DBDC Repeater.

Parameter	Format	D	O	Description

primaryradio
getprimaryradio
	iwpriv wifi*N* *primaryradio*
	{1|0}
		Y	Y	Set (1) or unset (0) primary radio on DBDC repeater mode.
				If this option is set, Ethernet client traffic of DBDC Repeater reach DBDC RootAP through current radio STA VAP.
				If this option is unset, Ethernet client traffic of DBDC Repeater reach DBDC RootAP through other radio STA VAP.
				The default value is 1 for wifi0 and 0 for other radios.
				The get parameter returns the current value.
				`#iwpriv wifi0 primaryradio 1`
				`#iwpriv wifi0 getprimaryradio`
				`wifi0 getprimaryradio:1`

delay_stavapup g_ delay_stavapup	iwpriv wifi*N* *delay_stavapup*
	{1|0}
		Y	Y	Enable (1) or disable (0) delay_stavapup on dbdc repeater mode.
				If this option is enabled, it delays secondary radio's sta vap connection with RootAP during first scan. This delay avoids looping of multicast packets when Hyfi bridge takes time to attach and dbdc_enable flag is disabled.
				The default value is 0. The get parameter returns the current value.
				`#iwpriv wifi0 delay_stavapup 1`
				`#iwpriv wifi0 g_delay_stavapup`
				`wifi0 g_delay_stavapup:1`

1.3.50 TR-69 Related Parameters

Table 1-50 shows list of WEXT commands

Table 1-50 WEXT commands

Parameter	Format	DA	OL	Description
g_phyofdmerr	iwpriv wifiN g_ phyofdmerr	Y	Y	Command to get OFDM Phy error count from the Radio Usage: `#iwpriv wifi0 g_phyofdmerr`
g_fcserr	iwpriv wifiN g_fcserr	Y	Y	Command to get FCS error count from the Radio Usage: `#iwpriv wifi1 g_fcserr`
g_chanutil	Iwpriv wifiN g_chanutil	Y	Y	Command to get Channel utilization from the Radio Usage: `#iwpriv wifi1 g_chanutil`

Table 1-51 shows the list of ‘wifitool’ commands

Table 1-51 wifitool commands

Parameter	Format	DA	OL	Description
tr069_get_ success_retrans	wifitool athN tr069_ get_success_ retrans	Y	Y	Command to get successful retransmission count from the VAP Usage: `#wifitool ath0 tr069_get_success_retrans`
tr069_get_fail_ retrans	wifitool athN tr069_ get_fail_retrans	Y	Y	Command to get failed retransmission count from the VAP Usage: `#wifitool ath0 tr069_get_fail_retrans`
tr069_get_ success_mul_ retrans	wifitool athN tr069_ get_success_mul_ retrans	Y	Y	Command to get successful retries after multiple attempts from the VAP Usage: `#wifitool ath0 tr069_get_success_mul_retrans`
tr069_get_ack_ failures	wifitool athN tr069_ get_ack_failures	Y	Y	Command to get ACK failures count from the VAP Usage: `#wifitool ath0 tr069_get_ack_failures`
tr069_get_retrans	wifitool athN tr069_ get_retrans	Y	Y	Command to get total re-transmission count from the VAP Usage: `#wifitool ath0 tr069_get_retrans`
tr069_get_aggr_ pkts	wifitool athN tr069_ get_aggr_pkts	Y	Y	Command to get aggregate packets from the VAP. Usage: `#wifitool ath0 tr069_get_aggr_pkts`

Table 1-51 wifitool commands

Parameter	Format	DA	OL	Description
tr069_get_sta_bytes_sent	wifitool athN tr069_get_sta_bytes_sent <STA MAC>	Y	Y	Command to get bytes sent to the specific station from the VAP. Usage: `#wifitool ath0 tr069_get_sta_bytes_sent <STA MAC>`
tr069_get_sta_bytes_rcvd	wifitool athN tr069_get_sta_bytes_rcvd <STA MAC>	Y	Y	Command to get bytes received from the specific station in the VAP. Usage: `wifitool ath0 tr069_get_sta_bytes_rcvd <STA MAC>`

1.3.51 Rx Monitor Filters

Table 1-52 List of monitor filters

Parameter	Format	DA	OL	Description
Set_monrxfilter	Iwpriv set_monrxfilter	N	Y	Used to set various filters in monitor/ap_monitor mode BIT(0) Enable/Disable all filters (1/0) i.e 0x1 BIT(1) Enable mac address based filtering i.e0x2 Mac address needs to be added to the monitor mode Before giving the above iwpriv command acfg_tool acfg_mon_addmac athN <mac address BIT(2) Drop unicast data frames i.e 0x04 BIT(3) Drop multicast data frames i.e 0x08 BIT(4) Drop control/management frames i.e0x10

1.4 wlanconfig utility

The Qualcomm Atheros **wlanconfig** utility manages VAP instances. It is an integral part of the configuration scripts and provides the primary method to:

- Create a VAP
- List VAP parameters
- Delete an interface

NOTE Although commands may have adverse effects, not all effects may have been documented. Consider the nature of multiple VAP configurations that use multiple radios, and use caution when changing parameters.

1.4.1 Create a VAP

Creating a VAP requires parameters indicating the specific nature of the VAP. A VAP can be either a client node (managed node) or an infrastructure node (master node).

```
#wlanconfig ATH[N] create wlandev wifiN wlanmode
[ap|sta|mon|adhoc|wrap][wlanaddr <mac_addr>] [mataddr <mac_addr>]
[bssid|-bssid] [nosbeacon] [bssid|-bssid] [nosbeacon]
```

Where:

Argument	Description	
ATH*[N]*	VAP name. If the number at the end of the name is omitted, the system will automatically use the next available interface number. The VAP name ATH is not required, any text string will do. Note that when the index is occupied by another VAP, **create VAP** will fail.	
create	Create action	
wlandev wifi*N*	Indicates to which interface the VAP will attach. The interface number is required for this argument. For dual concurrent operations, *N* indicates which radio to attach the VAP to.	
wlanmode *mode*	Indicates the mode to open the VAP into. The valid modes are:	
	ap	AP (infrastructure) mode
	sta	STA (client) mode
	wrap	AP mode to be used under special repeater mode called QWRAP
bssid *-bssid*	Optional parameter indicating that the MAC address should be cloned from the first VAP for this interface. Not normally specified. **NOTE** bssid is not supported by wlanconfig, but is supported by the Qualcomm Atheros driver.	
nosbeacon	Indicates that no beacons will be transmitted from this VAP. Used as part of STA mode.	
mataddr		Original mac of wired/wireless station connected on QWRAP AP
wlanaddr		Virtual/changed mac address to be used for proxy sta with respect to wired/wireless clients in QWRAP mode

1.4.2 List VAP parameters

The argument to the **list** command defines the type of listing to produce. Each type is described in this section:

- AP list elements
- STA list Elements
- Channel list elements
- Capabilities list elements
- WME list elements
- Keys list elements

The list command provides an extended listing of parameters from the VAP, depending on the type of list for each associated STA. The list command generates a print of the VAP association list with the associated parameters:

```
# wlanconfig athN list [ap|sta|chan|caps|wme|keys]
```

1.4.2.1 AP list elements

Table 1-53 describes the AP list elements. It only applies to VAPs that are STA VAPs. This scan result provides a list of nearby APs. The following is an example:

```
# wlanconfig athN list ap
SSID             BSSID              CHAN  RATE  S:N   INT   CAPS
Atheros Guests   00:0b:85:5b:a6:e1  52    54M   13:0  100   E
ney-11a          00:03:7f:00:de:ea  60    54M   22:0  100   Es  WME
perseus-cis      00:1d:45:29:39:50  36    54M   30:0  100   E   WME
BILL-AP          00:03:7f:00:ce:ee  36    54M   27:0  100   Es  WME
apps-atheros1    00:03:7f:00:ce:d3  36    54M   26:0  100   EPs WME  ATH
```

Table 1-53 AP list elements

Element	Description					
BSSID	BSSID value of the AP. Takes the form of a MAC address					
CAPS	Current capabilities of the AP These are alphanumeric characters corresponding to specific 802.11 capability bits in the beacon and probe response Responses are defined as:					
	E	ESS	**P**	Privacy	**s**	Short Slot Time
	I	IBSS	**S**	Short Preamble	**D**	DSSS/OFDM
	c	Pollable	**B**	PBCC		
	C	Poll Request	**A**	Channel Agility		
CHAN	Channel the AP is servicing					
INT	Beacon interval, in ms					
RATE	Maximum rate of the AP					
S:N	Signal to Noise ratio. The first number is the last received RSSI from the device, and the last number is the noise value.					
SSID	Name string of the AP as broadcast in the beacon					

Table 1-53 AP list elements

Element	Description					
(No Header)	All information elements (IE) for the attached STA are printed. They have the values:					
	WPA	WPA IE	**ATH**	Qualcomm Atheros Vendor IE	**RSN**	aRSN IE
	WME	WMM IE	**VEN**	Vendor-Specific IE	**???**	Unknown IE

1.4.2.2 STA list Elements

Table 1-54 describes the list elements for each STA associated with the indicated VAP. This listing is produced:

```
root@OpenWrt:/# wlanconfig ath0 list sta
ADDR AID CHAN TXRATE RXRATE RSSI IDLE  TXSEQ  RXSEQ  CAPS ACAPS ERP STATE
MAXRATE(DOT11) HTCAPS ASSOCTIME IEs MODE PSMODE
88:1d:fc:55:84:61 1 11 0M 1M 23 0 0 65535 Es 0 5 0 Q 00:00:20 IEEE80211_MODE_11G  0
```

NOTE The data for the ACAPS element data is no longer reported. In the example output above, the data 0, 33, Q, and WME correspond to ERP, STATE, HTCAPS, and (no header) elements listed in Table 1-54.

Table 1-54 STA list elements

Element	Description					
ADDR	MAC address of the STA					
AID	Association ID; determines the specific AP/STA association pair used in 802.11n test commands					
CAPS	**E**	ESS	**P**	Privacy	**s**	Short Slot Time
	I	IBSS	**S**	Short Preamble	**D**	DSSS/OFDM
	c	Pollable	**B**	PBCC		
	C	Poll Request	**A**	Channel Agility		
CHAN	Channel the device is associated on					
ERP	Extended Rate PHY capabilities in dBm. A value of 0 indicates a legacy STA. Printed in hex.					
HTCAPS	HT capabilities flags; these are character indicators that represent a capability of the 802.11n STA					
	A	Advanced coding	**Q**	Static MIMO power save	**S**	Short GI enabled (HT40)
	W	HT40 channel width	**R**	Dynamic MIMO power save	**D**	Delayed block ACK
	P	MIMO power save enabled	**G**	Greenfield preamble	**M**	Max AMSDU size
IDLE	Current setting of the STA inactivity timer. This is the time in ms when the STA will go into power save of no activity occurs on the link.					
RATE	Current data rate of the association					
RSSI	Signal strength of the last received packet. For MIMO devices, this is an average value over all active receive chains.					

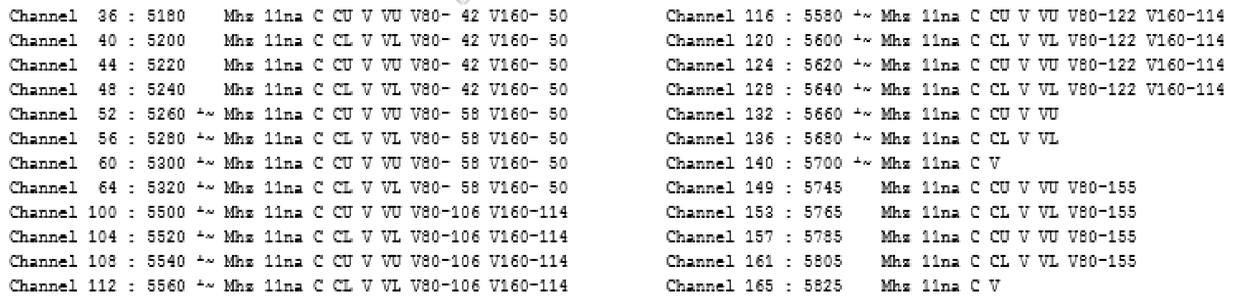
Table 1-54 STA list elements (cont.)

Element	Description					
RXSEQ	Receive sequence number of the last received packet					
STATE	Current state of the STA. This is an hexadecimal value that consists of these bits:					
	0x0001	Authorized for Data Transfer	0x0010	Power Save Mode Enabled	0x0100	uAPSD SP in Progress
	0x0002	QoS enabled	0x0020	Auth Reference held	0x0200	An ATH Node
	0x0004	ERP Enabled	0x0040	uAPSD Enabled	0x0400	WDS Workaround Req.
	0x0008	HT Rates Enabled	0x0080	uAPSD Triggerable	0x0800	WDS Link
TXSEQ	Transmit sequence number of the last received packet					
(No Header)	All information elements (IE) for the attached STA are printed. They have the values:					
	WPA	WPA IE	**ATH**	Qualcomm Atheros Vendor IE	**RSN**	RSN IE
	WME	WMM IE	**VEN**	Vendor-Specific IE	**???**	Unknown IE

1.4.2.3 Channel list elements

Table 1-55 describes the channel list elements, listing available channels and frequencies followed by strings indicating specific VAP channel capabilities. This example lists channels with channel number and frequency in MHz:

```
# wlanconfig ath0 list chan
```



```
Channel  36 : 5180    Mhz 11na C CU V VU V80- 42 V160- 50      Channel 116 : 5580 *~ Mhz 11na C CU V VU V80-122 V160-114
Channel  40 : 5200    Mhz 11na C CL V VL V80- 42 V160- 50      Channel 120 : 5600 *~ Mhz 11na C CL V VL V80-122 V160-114
Channel  44 : 5220    Mhz 11na C CU V VU V80- 42 V160- 50      Channel 124 : 5620 *~ Mhz 11na C CU V VU V80-122 V160-114
Channel  48 : 5240    Mhz 11na C CL V VL V80- 42 V160- 50      Channel 128 : 5640 *~ Mhz 11na C CL V VL V80-122 V160-114
Channel  52 : 5260 *~ Mhz 11na C CU V VU V80- 58 V160- 50      Channel 132 : 5660 *~ Mhz 11na C CU V VU
Channel  56 : 5280 *~ Mhz 11na C CL V VL V80- 58 V160- 50      Channel 136 : 5680 *~ Mhz 11na C CL V VL
Channel  60 : 5300 *~ Mhz 11na C CU V VU V80- 58 V160- 50      Channel 140 : 5700 *~ Mhz 11na C V
Channel  64 : 5320 *~ Mhz 11na C CL V VL V80- 58 V160- 50      Channel 149 : 5745    Mhz 11na C CU V VU V80-155
Channel 100 : 5500 *~ Mhz 11na C CU V VU V80-106 V160-114      Channel 153 : 5765    Mhz 11na C CL V VL V80-155
Channel 104 : 5520 *~ Mhz 11na C CL V VL V80-106 V160-114      Channel 157 : 5785    Mhz 11na C CU V VU V80-155
Channel 108 : 5540 *~ Mhz 11na C CU V VU V80-106 V160-114      Channel 161 : 5805    Mhz 11na C CL V VL V80-155
Channel 112 : 5560 *~ Mhz 11na C CL V VL V80-106 V160-114      Channel 165 : 5825    Mhz 11na C V
```

Table 1-55 Channel list elements

Column 1	**FHSS**	FHSS channel
Column 2	**11na**	5 GHz band 802.11n capable
	11a	5 GHz band legacy
	11ng	2.4 GHz band 802.11n capable
	11g	2.4 GHz band legacy
	11b	2.4 GHz band DSSS only

Table 1-55 Channel list elements

Column 3	**C**	802.11n control channel capable
	CU	802.11n upper extension channel enabled
	CL	802.11n lower extension channel enabled
Column 4	**V**	80211ac (VHT - 20 MHz band) control channel capable
Column 5	**VU**	80211ac (VHT - 40 MHz band) upper extension channel enabled
	VL	80211ac (VHT - 40 MHz band) lower extension channel enabled
Column 6	**V80-<CH>**	80211ac (VHT - 80 MHz band) channel. With center frequency CH
Column 7	V160-<CH>	80211ac (VH -160 MHz band) channel with center frequency CH.

1.4.2.4 Capabilities list elements

Table 1-56 describes the capabilities list strings; the list provides a list of the VAP capabilities output as a comma-delimited string.

```
# wlanconfig ath0 list caps
ath0=3782e41f<WEP,TKIP,AES,AES_
CCM,HOSTAP,TXPMGT,SHSLOT,SHPREAMBLE,TKIPMIC,WPA1,WPA2,
BURST,WME>
```

Table 1-56 Capabilities list elements

AES	AES OCB available	**MONITOR**	Monitor mode	**TXPMGT**	Tx power mgmt.
AES_CCM	AES CCM	**PMGT**	Power mgmt. available	**WEP**	WEP available
AHDEMO	Ad hoc demo mode	**SHPREAMBLE**	Short GI preamble available	**WME**	WME capable
BURST	Frame bursting capable	**SHSLOT**	Short Slot available	**WPA1**	WPA1 available
CKIP	CKIP available	**SWRETRY**	Tx software retry	**WPA2**	WPA2 available
HOSTAP	Host AP mode	**TKIP**	TKIP available		
IBSS	IBSS mode available	**TKIPMIC**	TKIP MIC available		

1.4.2.5 WME list elements

This list provides the current settings of the VAP WME settings:

```
# wlanconfig ath0 list wme
    AC_BE cwmin 4 cwmax 6 aifs 3 txopLimit  0
       cwmin 4 cwmax 10 aifs 3 txopLimit  0
    AC_BK cwmin 4 cwmax 10 aifs 7 txopLimit  0
       cwmin 4 cwmax 10 aifs 7 txopLimit  0
    AC_VI cwmin 3 cwmax 4 aifs 1 txopLimit 3008
       cwmin 3 cwmax 4 aifs 2 txopLimit 3008
    AC_VO cwmin 2 cwmax 3 aifs 1 txopLimit 1504
       cwmin 2 cwmax 3 aifs 2 txopLimit 1504
```

1.4.2.6 Keys list elements

This list provides the current keys that are set and the one that is being used:

```
#wlanconfig ath0 list keys
ath0 3 key sizes : 40, 104, 128bits
4 keys available :
[1]: 1234-5678-90 (40 bits)
[2]: off
[3]: off
[4]: off
Current Transmit Key: [1]
Security mode:restricted
```

1.4.3 Delete an interface

The VAP must be down before deleting an interface to avoid bad interactions with other VAPs. This command applies only to the VAP interface specified and uses the form:

```
# wlanconfig athN destroy
```

1.4.4 NAWDS configuration parameters

The NAWDS parameter has several subparameters, each of which may have its own set of options and settings. For example, the *add-repeater* subparameter has *mac_addr* and *caps* as options. Each NAWDS subparameter is listed in Table 1-57 as a separate entry.

Table 1-57 Configure NAWDS parameters

Parameter	Format	DA	OL	Description	
add-repeater	wlanconfig ath*N* nawds *add-repeater mac_ addr caps*	Y	Y	Add a NAWDS AP with the specified MAC address and capability. The definition of CAPS is the same as the CAPS mentioned in defcaps.	
				mac_addr	MAC address
				caps	Capabilities

Table 1-57 Configure NAWDS parameters (cont.)

<table>
<tr><th>Parameter</th><th>Format</th><th>DA</th><th>OL</th><th colspan="8">Description</th></tr>
<tr><td rowspan="7"><b>defcaps</b></td><td rowspan="7">wlanconfig ath<i>N</i> nawds <i>defcaps caps</i></td><td rowspan="7">Y</td><td rowspan="7">Y</td><td colspan="8"></td></tr>
<tr><td colspan="2">0x0</td><td colspan="6">When a NAWDS AP is operating in learning mode, it must discover which capability the NAWDS AP peer has. In this situation, defcaps would be used. The CAPS is defined as follows:
#define NAWDS_REPEATER_CAP_DS 0x01
#define NAWDS_REPEATER_CAP_TS 0x02
#define NAWDS_REPEATER_CAP_4S 0x04
#define NAWDS_REPEATER_CAP_HT20 0x0100
#define NAWDS_REPEATER_CAP_HT2040 0x0200
#define NAWDS_REPEATER_CAP_11ACVHT20 0x0400
#define NAWDS_REPEATER_CAP_11ACVHT40 0x0800
#define NAWDS_REPEATER_CAP_11ACVHT80 0x1000
#define NAWDS_REPEATER_CAP_11ACVHT80_80 0x2000
#define NAWDS_REPEATER_CAP_11ACVHT160 0x4000
If CAPS equals 0, the HT rate would be disabled. To enable NAWDS_REPEATER_CAP_DS, at least one of NAWDS_REPEATER_CAP_HT20 and NAWDS_REPEATER_CAP_HT2040 must be specified.
The range of CAPS values are defined as follows:</td></tr>
<tr><td>Nxn</td><td>HT20</td><td>HT40</td><td>VHT20</td><td>VHT40</td><td>VHT80</td><td>VHT80_80</td><td>VHT160</td></tr>
<tr><td>1x1</td><td>0x0100</td><td>0x0200</td><td>0x0400</td><td>0x0800</td><td>0x1000</td><td>0x2000</td><td>0x4000</td></tr>
<tr><td>2x2</td><td>0x0101</td><td>0x0201</td><td>0x0401</td><td>0x0801</td><td>0x01001</td><td>0x2004</td><td>0x4004</td></tr>
<tr><td>3x3</td><td>0x102</td><td>0x0202</td><td>0x0402</td><td>0x0802</td><td>0x01002</td><td>N/A</td><td>N/A</td></tr>
<tr><td>4x4</td><td>0x0102</td><td>0x0204</td><td>0x0404</td><td>0x0804</td><td>0x1004</td><td>N/A</td><td>N/A</td></tr>
<tr><td rowspan="2"><b>del-repeater</b></td><td rowspan="2">wlanconfig ath<i>N</i> nawds <i>del-repeater mac_addr</i></td><td rowspan="2">Y</td><td rowspan="2">Y</td><td colspan="8">Delete a NAWDS AP with the specified MAC address.</td></tr>
<tr><td colspan="2">mac_addr</td><td colspan="6">MAC address</td></tr>
<tr><td><b>list</b></td><td>wlanconfig ath<i>N</i> nawds <i>list</i></td><td>Y</td><td>Y</td><td colspan="8">Display current NAWDS configurations.</td></tr>
</table>

Table 1-57 Configure NAWDS parameters (cont.)

Parameter	Format	DA	OL	Description	
mode	wlanconfig ath*N* nawds *mode value*	Y	Y	Configures the mode in which NAWDS AP is operating. Whenever the mode is changed, the NAWDS MAC table would be cleared. *value* may specify one of the following:	
				0	NAWDS Disabled
				1	STATIC Repeater mode
				2	STATIC Bridge mode
				3	LEARNING Repeater mode
				4	LEARNING Bridge mode
override	wlanconfig ath*N* nawds *override value*	Y	Y	Enables (1) or disables (0) override command. *value* may specify one of the following:	
				0	No more MAC address my be added to the NAWDS table when the table is full.
				1	When running out of entry space in NAWDS MAC table (either by configuring too many NAWDS APs or by learning too many AP using the learning feature), enabling the override would delete MAC addresses occupied by dead NAWDS APs.

1.4.4.1 Configuration examples

Static bridge and peer node supports HT20 rates

```
Set SSID, Mode and PRIMARY_CH using UCI commands.
Bring the AP up

Iwpriv ath0 wds 1
Wlanconfig ath0 nawds mode 2
Wlanconfig ath0 nawds add-repeater 00:03:7f:xx:xx:xx:xx 0x0100
```

Learning bridge and by default peer NAWDS AP supports HT40/DS rates

```
Set SSID, Mode and PRIMARY_CH using UCI commands.
Bring the AP up

Iwpriv ath0 wds 1
Wlanconfig ath0 nawds mode 4
Wlanconfig ath0 nawds defcaps 0x0201
```

Static bridge and peer node supports VHT rates

```
Set SSID, Mode and PRIMARY_CH using UCI commands.
Bring the AP up

Iwpriv ath0 wds 1
```

```
Wlanconfig ath0 nawds mode 2
Wlanconfig ath0 nawds add-repeater 00:03:7f:xx:xx:xx:xx 0x1002
```

VHT Example Rates: 3x3

```
wlanconfig ath0 nawds add-repeater <mac> 0x1002 - 3x3 HT80
wlanconfig ath0 nawds add-repeater <mac> 0x802 - 3x3 HT40
wlanconfig ath0 nawds add-repeater <mac> 0x402 - 3x3 HT20
```

VHT Example Rates: 2x2

```
wlanconfig ath0 nawds add-repeater <mac> 0x1001 - 2x2 HT80
wlanconfig ath0 nawds add-repeater <mac> 0x801 - 2x2 HT40
wlanconfig ath0 nawds add-repeater <mac> 0x401- 2x2 HT20
```

VHT Example Rates: 1x1

```
wlanconfig ath0 nawds add-repeater <mac> 0x1000 - 1x1 HT80
wlanconfig ath0 nawds add-repeater <mac> 0x800 - 1x1 HT40
wlanconfig ath0 nawds add-repeater <mac> 0x400 - 1x1 HT20
```

1.4.5 HMWDS/HMMC commands

Table 1-58 Configure HMWDS/HMMC parameters

Parameter	Format	DA	OL	Description
hmmc add	wlanconfig athX hmmc add *<ipv4 mcast address> <netmask>*	Y	Y	To add a range of multicast address defined by <ipv4mcastaddr>/<netmask> for which all mcast packets should converted to unicast for all the stations associated to the ap.
hmmc del	wlanconfig athX hmmc del <ipv4 mcast address> <netmask>	Y	Y	To delete the mcast ip range of address.
hmmc dump	wlanconfig athX hmmc dump	Y	Y	To display the ranges configured so far.
hmwds add_addr	wlanconfig ath0 hmwds add_addr <wds_mac_addr> <peer_mac_addr>	Y	Y	To add a managed WDS address through an associated peer.
hmwds reset_addr	wlanconfig ath0 hmwds reset_addr <mac_addr>	Y	Y	Resets all the managed WDS entries in the global WDS table if both <wds_mac_addr> and <peer_mac_addr> are not specified.
hmwds read_addr	wlanconfig ath0 hmwds read_addr <peer_mac_addr>	Y	Y	Lists all the managed WDS addresses behind the given peer.
hmwds read_table	wlanconfig ath0 hmwds read_table	Y	Y	Lists all the managed WDS addresses configured.

1.4.6 ATF configuration commands

Table 1-59 Configure/show ATF parameters

Parameter	Format	D	O	Description
addssid	wlanconfig athX addssid *<ssid name>* <airtime percentage>	Y	Y	Assign percentage of airtime to the SSID. The airtime percentage value range is 0~100. Example: `#wlanconfig ath0 addssid BEE0 12`
delssid	wlanconfig athX delssid <ssid name>	Y	Y	Delete the SSID assigned. Example: `# wlanconfig ath0 delssid BEE0`
addsta	wlanconfig athX addsta <sta mac addr> <airtime percentage>	Y	Y	Assign percentage of airtime to the STA. The airtime percentage value range is 0~100. Example: `#wlanconfig ath0 addsta 220011abef6660`
delsta	wlanconfig athX delsta <sta mac addr>	Y	Y	Delete the STA assigned. Example: `# wlanconfig ath0 delsta 220011abef66`
showatftable	wlanconfig athX showatftable	Y	Y	Displays the ATF table. The SSIDs and STAs part of the ATF table will be listed Example: `#wlanconfig ath0 showatftable`
showairtime	wlanconfig athX showairtime	Y	Y	Lists all STAs and percentage of ATF. Example: `#wlanconfig ath0 showairtime` Displays STA's added in the ATF table. The airtime value shown is in terms of 1000.
flushatftable	wlanconfig athX flushatftable	Y	Y	This flushes all the configurations and data present in the atf table. Example `#wlanconfig ath0 flushatftable` When this command is issued there is a reset of bss and the stations which are connected will get disconnected and connected again.
addatfgroup	wlanconfig athX addatfgroup <group name> <ssid name>	Y	N	Creates a new atfgroup, if the group doesn’t exist already and adds the SSID to the group created. If the group already exists, the SSID will be added to the group. Example: `#wlanconfig ath0 addatfgroup group1 myvap`
configatfgroup	wlanconfig athX configatfgroup <group name> <airtime percentage>	Y	N	Assigns/Configures airtime percentage to the group. Example: `#wlanconfig ath0 configatfgroup group1 80`

Table 1-59 Configure/show ATF parameters

delatfgroup	wlanconfig athX delatfgroup <group name>	Y	N	Deletes a group. Example: `#wlanconfig ath0 delatfgroup group1`
showatfgroup	wlanconfig athX showatfgroup	Y	N	Displays all groups configured, SSIDs for each group & the airtime percentage assigned to each group Example: `#wlanconfig ath0 showatfgroup`

addtputsta	wlanconfig athX addtputsta <sta mac addr> <tput in kbps> <max airtime percentage (optional)>	Y	N	To configure 25 Mbps throughput for a client (and with max airtime of 20%)
				Example:
				`#wlanconfig ath0 addtputsta 220011abef66 25000`
				`#wlanconfig ath0 addtputsta 220011abef66 25000 20`

addtputsta	wlanconfig athX addtputsta ffffffffffff <tput in kbps (ignored)> <reserved airtime percentage>	Y	N	To configure reserved airtime as 5% Example: `#wlanconfig ath0 addtputsta ffffffffffff 25000 5`
deltputsta	wlanconfig athX addtputsta ffffffffffff <tput in kbps (ignored)> <reserved airtime percentage>	Y	N	To remove throughput configuration for a client Example: `#wlanconfig ath0 deltputsta 220011abef66`
deltputsta	wlanconfig athX deltputsta ffffffffffff	Y	N	To remove throughput configuration for all clients Example: `#wlanconfig ath0 deltputsta ffffffffffff`
showtputtbl	wlanconfig athX showtputtbl	Y	N	To show throughput configuration of all clients and the reserved airtime Example: `#wlanconfig ath0 showtputtbl`

1.5 Other commands

The following tables describe additional commands and parameters beyond iwconfig, iwpriv, and wlanconfig.

1.5.1 Athssd parameters

Table 1-60 Athssd Parameters

Configuration	Format	DA	OL	Description	
nobeacon **get_nobeacon**	`iwpriv ath`*N* `nobeacon`			Enables/disables VAP to transmit beacon and probe response. The get parameter returns the current value.	
				0	Disable
				1	Enable
Standalone Scan	athssd –i wifiN –j ath*N* –s *val*			Start athssd, configuring it to carry out a standalone scan on channel *val*. *val* can be 0, in which case the current channel will be used.	
		Y	Y	s=0	
		Y	N	s>0	
External GUI	athssd –i wifi0 –j ath*N* –s	N	N	Start athssd, configuring it to work with external GUI. Typically the GUI is an internal tool.	

1.5.2 DFS

Configuring the AP for DFS involves setting up certain parameters. They can be set by using UCI commands or appropriate iwpriv commands. Please refer to iwpriv command reference for further details.

1. Use UCI command Wi-Fi detect to get the default parameters of the radio.
2. Set up the following parameters:
   a. Set up country code
   b. Select the proper RADIO
   c. Select the proper mode. Possible modes are
      i. 11A
      ii. 11NAHT20
      iii. 11NAHT40PLUS
      iv. 11NAHT40MINUS
      v. 11ACVHT20
      vi. 11ACVHT40PLUS
      vii. 11ACVHT40MINUS
      viii. 11ACVHT80

ix. 11ACVHT160

x. 11ACVHT80_80

d. Select the appropriate channel

e. For FCC testing following extra set up is necessary:

i. Set rate control to manual mode.

ii. Set manual rate to 9 Mbps (Use uci set wireless.@wifi-iface[0].set11NRates=0x80808080)

iii. Iwpriv command can also be used for (i) and (ii)

iv. Commit the configuration using uci commit

1.5.3 NAT parameters

For Host Network Address Translation (HNAT), the rules are programmed through Linux command “iptables”.

The simple NAT rule for egress and ingress TCP traffic is as follows:

```
iptables -t nat -A POSTROUTING -o eth1.2 -p tcp -j MASQUERADE
iptables -t nat -A PREROUTING  -i eth1.2 -p tcp -j DNAT --to 192.168.1.100
```

For further information about syntax and usage, refer to http://www.linuxhomenetworking.com/wiki/index.php/Quick_HOWTO_:_Ch14_:_Linux_Firewalls_Using_iptables

1.5.4 Radartool

Table 1-61 Radartool parameters

Parameter	Format	DA	OL	Description	
usenol	radartool -i wifi [*0*|*1]* usenol *[0/1]*	Y	Y	usenol 0: Sets the test system in test mode so that it stays in the same channel during the test. By default the AP will switch channel when it detects radar. usenol 1: Causes the AP to switch channels when radar is detected.	
dfsdebug	radartool -i wifi [*0*|*1]* dfsdebug *debug_level*	Y	Y	Sets the debug level.	
				0x00000100	minimal DFS debug
				0x00000200	normal DFS debug
				0x00000400	maximal DFS debug
				0x00000800	display matched filter ID
				0x00001000	display TLV related information
				0x00002000	display readar NOL
				0x00004000	display PHY error summary
				0x00008000	display PHY error FFT reports

Table 1-61 Radartool parameters (cont.)

Parameter	Format	DA	OL	Description
shownol	radartool -i wifi [*0|1]* shownol *debug_level*	Y	Y	Displays the NOL list. Set dfsdebuglevel to 0x2000 before using command
enable	radartool –I wifiX enable	Y	Y	Enables dfs on a particular channel
disable	radartool –I wifiX disable	Y	Y	Disables dfs on a particular channel
ignorecac	radartool –I wifiX ignorecac 0|1	Y	Y	Used to set ignore cac value which, when set to 1, waits for cactimeout value.
shownolhistory	radartool -i wifi [0|1] shownolhistory	Y	Y	Displays the NOL History. The NOL History is meaningful when the Wi-Fi device supports STA (station) mode DFS. STA mode DFS can be enabled or disabled by using the command: iwpriv wifi[0/1] `staDFSEnable` 1/0. NOL history bit is set for a channel if radar is seen in the channel at least once. The NOL history persists until the wireless driver is removed from the Operating System.

1.5.5 Spectraltool parameters

Table 1-62 Spectraltool parameters

Parameter	Format	DA	OL	Description
fft_period	spectraltool -i wifi*N* fft_period *val*	Y	N	Set skip interval for FFT reports. *(Not applicable for 11ac chipsets.)*
scan_period	spectraltool -i wifi*N* scan_period *val*	Y	Y	Set Spectral Scan period. Period increment resolution is 256*Tclk, where Tclk = 1/44 MHz (Gmode), 1/40 MHz (Amode)
scan_count	spectraltool -i wifi*N* scan_count *val*	Y	Y	Set number of reports to return
short_report	spectraltool -i wifi*N* short_report *{1|0}*	Y	N	Set to 1 to report only one set of FFT results per spectral_scan_period. *(Not applicable for 11ac chipsets.)*
priority	spectraltool -i wifi*N* priority *{1|0}*	Y	Y	Set priority.
fft_size	spectraltool -i wifi*N* fft_size *val*	N	Y	Set the number of FFT data points to compute, defined as a log index: num_fft_pts = 2^fft_size Value can range from 2 (num_fft_pts=4) to 9 (num_fft_pts=512). *(Only for 11ac chipsets)*
gc_ena	spectraltool -i wifi*N* gc_ena *{1|0}*	N	Y	Set to enable targeted gain change before starting the spectral scan FFT. *(Only for 11ac chipsets)*
noise_floor_ref	spectraltool -i wifi*N* noise_floor_ref *val*	N	Y	Set noise floor reference number (signed) for the calculation of bin power (dBm). *(Only for 11ac chipsets)*
init_delay	spectraltool -i wifi*N* init_delay *val*	N	Y	Disallow spectral scan triggers after Tx/Rx packets by setting this delay value to roughly SIFS time period or greater. Delay timer counts in units of 0.25 µs. *(Only for 11ac chipsets)*

Table 1-62 Spectraltool parameters (cont.)

Parameter	Format	DA	OL	Description
nb_tone_thr	spectraltool -i wifi*N* nb_tone_thr *val*	N	Y	Set number of strong bins (inclusive) per sub-channel, below which a signal is declared a narrow band tone. *(Only for 11ac chipsets)*
str_bin_thr	spectraltool -i wifi*N* str_bin_thr *val*	N	Y	Set bin/max_bin ratio threshold over which a bin is declared strong, for spectral scan bandwidth analysis. *(Only for 11ac chipsets)*
wb_rpt_mode	spectraltool -i wifiN wb_rpt_mode *{1/0}*	N	Y	Set this to 1 to report spectral scans as EXT_BLOCKER (phy_error=36), if none of the sub-channels are deemed narrow band. *(Only for 11ac chipsets)*
rssi_thr	spectraltool -i wifi*N* rssi_thr *val*	N	Y	ADC RSSI must be greater than or equal to this threshold (signed Db) to ensure spectral scan reporting with normal PHY error codes (see rssi_rpt_mode in this table). *(Only for 11ac chipsets)*
pwr_format	spectraltool -i wifi*N* pwr_format *{0/1}*	N	Y	Format of frequency bin magnitude for spectral scan triggered FFTs. *(Only for 11ac chipsets)* 0 – linear magnitude 1 – log magnitude (20*log10(lin_mag), 1/2 dB step size)
rpt_mode	spectraltool -i wifi*N* rpt_mode *val*	N	Y	Format of per-FFT reports to software for spectral scan triggered FFTs.*(Only for 11ac chipsets)* 0 – No FFT report (only pulse end summary) 1 – 2-dword summary of metrics for each completed FFT 2 – 2-dword summary + 1x-oversampled bins (in-band) per FFT. In the case of QCA9984/QCA9888, eight additional bins are reported, four bins to the left and right of the band-edge. 3 – 2-dword summary + 2x-oversampled bins (all) per FFT
bin_scale	spectraltool -i wifi*N* bin_scale *val*	N	Y	Number of LSBs to shift out to scale the FFT bins for spectral scan triggered FFTs.*(Only for 11ac chipsets)*
dBm_adj	spectraltool -i wifi*N* dBm_adj *{1/0}*	N	Y	Set to 1 (with pwr_format=1), to report bin magnitudes converted to dBm power using the noisefloor calibration results. *(Only for 11ac chipsets)*
chn_mask	spectraltool -i wifi*N* chn_mask *val*	N	Y	Set per chain enable mask to select input ADC for search FFT. *(Only for 11ac chipsets)*

1.5.6 Intelligent channel manager parameters

ICM is a channel selection application external to the driver. It is intended to provide a number of advantages over the current in-driver ACS, the main ones being flexibility and use of spectral data to identify non-802.11 interferences during channel selection.

Future potential benefits include use of historical data, utilization of other radios to speed up scan, an so on. It can be used either standalone, or as a server carrying out scans and ranking for an external entity. We describe only the former below, since the latter functionality is currently for QCA internal use.

Since ICM has functionality similar to ACS, the configuration settings are similar to those for ACS. The only difference is that if a channel is set to a static value while ICM is enabled, ICM will still come up and rank the channels for future use with DCS, but it will not set the best channel at bring-up (compared to ACS, which will not be activated in the first place). Since ICM also interacts with DCS, the DCS settings apply as-is.

We only provide the following additional configurations specific to ICM:

Standalone configuration

```
cfg -a ICM_ENABLE=1 or uci set wireless.wifi0.icm_enable = 0
cfg –a ICM_MODE=”standalone”
```

ICM_MODE can also be set to “server” (`cfg –a ICM_MODE=”server”`). However, this is currently for QCA internal use only, as noted previously.

1.5.6.1 Enabling selection debug information

In case it is desired to view additional debug information pertaining to the selection process (e.g. number of APs on every channel, Noise floor, Noise floor threshold, presence of various interferer's, etc.), there are two options available: Console prints and CSV dump. The CSV dump is much more detailed than the Console prints.

1.5.6.1.1 Console prints

In case it is desired to view selection debug information on the console, then the ICM debug level should be lowered from 3 to 2 by setting ICM_DEBUG_LEVEL to 2 (The valid values are described in section 1.5.6.2 below – see option ‘-q’ for setting debug level). A table will be printed on the console at the end of each selection algorithm run. Refer to legend printed before table to understand contents. It is highly recommended to disable kernel console prints at this ICM Debug Level, else such prints can pop up in-between and make it hard to understand the tables.

```
echo 0 > /proc/sys/kernel/printk
```

Console print configuration

```
cfg -a ICM_DEBUG_LEVEL=2
```

CSV dump

In case it is desired to view a very detailed selection debug information dump in CSV format, then this can be enabled by setting ICM_ENABLE_SELDEBUG_DUMP to 1.

CSV dump configuration

```
cfg -a ICM_ENABLE_SELDEBUG_DUMP=1
```

The CSV file created is /tmp/icmseldebug.csv. It can be TFTP’ied to the host and viewed in a suitable application such as MS Excel. If the file is already present on the AP when ICM is launched, its contents are first emptied. Information is appended to the file for every run of the selection algorithm. A column titled ‘Record Set No.’ is updated for every run. Row entries having the same record set number correspond to the same algorithm run.

NOTE Unlike the ICM Console Print method, this does not require disabling Kernel Prints.

1.5.6.2 ICM command line parameters (debugging only)

ICM is intended to be invoked from a bring-up script such as apup. The command line parameters need not be used directly except for debugging purposes.

NOTE The following ICM command line parameters are only for reference.

Table 1-63 ICM command line parameters

Parameter	Format	Description	
-e	icm -e	Run as daemon. By default, non-daemon execution is used.	
-f	icm –f	Enable use of Dynamic noise floor	
-h	icm –h	Display help	
-s	icm –s *val*	Server mode: Socket type to listen on for messages from external entity. Not applicable for standalone mode. Listed here only for completeness. The default value is 1.	
		0	TCP
		1	UDP
-t	icm –t	Enable some internal unit tests. **NOTE** This is only for developers. It is not intended for	
-v	icm –v *val*	Enable (1) or Disable (0) server mode. It is disabled by default.	
-i	icm –i	Dump selection debug information to /tmp/icmseldebug.csv	
-q	icm –q *val*	Set debug level. The default is 3.	
		1	Minor
		2	Default
		3	Major
		4	Critical

Table 1-63 ICM command line parameters

Parameter	Format	Description	
-u	icm –u *val*	Set debug module bitmap, formed by O-ring bit positions corresponding to each module. The default is 0xFF.	
		0x01	Main
		0x02	Scan
		0x04	Selector
		0x08	Utilities
		0x10	Test
		0x20	Socket
		0x40	Spectral
		0x80	Command

1.5.6.3 ACS/DCS/OBSS enhancements: iwpriv CLI commands

Table 1-64 ACS/DCS/OBSS iwpriv commands

Command	Format	Description
acs_bkscanen	iwpriv wifi1 *acs_bkscanen* <value>	Bit ‘1” – Enabled ACS/OBSS background scan depending on the value "acs_ctrlflags". Bit “0” – Disables acs/obss background scan timer.
"g_acs_bkscanen"	iwpriv wifi1 get_acs_ bkscanen	ACS/OBSS background scan value
acs_bkscanintvl	iwpriv wifi1 acs_scanintvl <value>	Set the background scan value default is one minute
get_acsscanintvl	iwpriv wifi1 g_acsscanintvl	Display the background scan timer value
acs_rssivar	iwpriv wifi1 acs_rssivar <value>	Set the RSSI variance. Used for ignoring the difference between two channel. If the two channel differ with value less then rssivar then both channel are considered as having same RSSI Default Value: 10
g_acs_rssivar	get_acs_rssivar	Display RSSI variance value
acs_chloadvar	iwpriv wifi1 acs_chloadvar <value>	Set the channel load variance If two channel differ with channel load value less then ch load variance .They are treated as having same channel load for next level evaluation Default Value: 20
g_acschloadvar	Iwpriv wifiN g_acschloadvar	Value of channel load variance. Default is 10
.acs_lmtobss	iwpriv wifi1 acs_lmtobss 1	Enable limited BSS check.
get_acslmtobss	iwpriv wifi1 get_acslmtobss <value>	Status of limited BSS check enable/disable

Table 1-64 ACS/DCS/OBSS iwpriv commands

Command	Format	Description
acs_ctrlflags	iwpriv wifi1 acs_ctrlflags 0xx	Back ground scan ACS control flags 0x1 – Full ACS check 0x2 -Only OBSS check this is used for manual configuration of channel.
getacsctrlflags	iwpriv wifi1 getacsctrlflags <value>	Get value of ACS control flag set
acs_dbgtrace	iwpriv wifi1 acs_dbgtrace 0xxx	Set ACS run time debug option The values signify EACS_DBG_DEFAULT 0x1 EACS_DBG_FUNC 0x8000 EACS_DBG_CHLOAD 0x4 EACS_DBG_RSSI 0x80 EACS_DBG_OBSS 0x100 EACS_DBG_REGPOWER 0x200 EACS_DBG_NF 0x400 EACS_DBG_SCAN 0x800 EACS_DBG_ADJCH 0x1000
g_ acs_dbgtrace	iwpriv wifi1 g_acs_dbgtrace	Display the debug option specified
obss_rssi_th	iwpriv wifiX obss_rssi_th <value>	Configure OBSS RSSI threshold. If OBSSI RSSI is greater than configured value then only move to 20 MHz. Value range: 0 – 127.
gobss_rssi_th	iwpriv wifiX gobss_rssi_th	Retrieve OBSS RSSI threshold
obss_rx_rssi_th	iwpriv wifiX obss_rx_rssi_th <value>	Configure RSSI threshold for received frame with 40 MHz intolerance bit. If RSSI of received is greater than configured value then only move to 20 MHz. Value range: 0–127.
acs_txpwr_opt	iwpriv wifiX acs_txpwr_opt <value>	Configures ACS Tx Power parameter option. Values range: 1-2 1. Tx Power for good throughput 2. Tx Power for maximum range
g_acs_txpwr_opt	iwpriv wifiX g_acs_txpwr_opt	Retrieves the Tx power type values
antenna_plzn	iwpriv wifiX antenna_plzn <value>	Configures antenna polarization. Value – 32 Bit value Bits [24-31] – specifies to enable antenna polarization (0xFF – enable, 0 - disable). Bits [0-23] – Antenna value. Default value – 0xFF00000A (VHVH antenna polarization) V – Vertical, H – Horizontal.

Table 1-64 ACS/DCS/OBSS iwpriv commands

Command	Format	Description
acs_2g_allchan	iwpriv wifi*N acs_2g_allchan* {1|0}	Enable (1) or disable (0) overlap channel selection in 2.4GHz band with ACS If this option is enabled, AP can select one of the overlapping channels with ACS. The default value is 0. `#iwpriv wifi0 acs_2g_allchan 1`

g_acs_2g_allchan	iwpriv wifi*N g_acs_2g_allchan*	The get parameter returns the current value of overlap channel selection.
		`#iwpriv wifi0 g_acs_2g_allchan`
		`wifi0 g_acs_2g_allchan:1`

1.5.7 Dynamic Encap/Decap configuration

The format in which the offload host driver exchanges frames with firmware/hardware can be configured dynamically. This is currently available only for QCA9980.

Table 1-65 Dynamic Encap/Decap configuration

Command	DA	OL	Description	
iwpriv athN encap_type <*value*>	N	Y	Set the transmit encapsulation type	
			0	Raw 802.11 mode
			1	Native Wi-Fi (Only config support. Rest of host data path to be added separately when required)
			2	Ethernet II mode
iwpriv athN get_encap_type	N	Y	Get the transmit encapsulation type	
			0	Raw 802.11 mode
			1	Native Wi-Fi (Only config support. Rest of host data path to be added separately when required)
			2	Ethernet II mode
iwpriv athN decap_type <*value*>	N	Y	Set the receive decapsulation type	
			0	Raw 802.11 mode
			1	Native Wi-Fi (Only config support. Rest of host data path to be added separately when required)
			2	Ethernet II mode
iwpriv athN get_decap_type	N	Y	Get the receive decapsulation type	
			0	Raw 802.11 mode
			1	Native Wi-Fi (Only config support. Rest of host data path to be added separately when required)
			2	Ethernet II mode

1.5.8 Raw mode simulation

A simulation is available for QCA internal testing of Raw 802.11 encap/decap. This simulation converts Ethernet Type II to and from raw 802.11 MPDUs at offload driver entry/exit points, so that the AP can exchange Ethernet Type II frames with external hosts connected to it via Ethernet cables while exchanging raw 802.11 frames along the internal data paths.

Table 1-66 Raw mode simulation

Command	DA	OL	Description	
iwpriv athN rawsim_ txagr <*value*>	N	Y	Enable/Disable use of multiple fragments during Tx by Raw 802.11 mode simulation. This will result in creation of A-MSDUs, with one MSDU per fragment.	
			0	Enable
			1	Disable
iwpriv athN get_rawsim_ txagr	N	Y	Return whether use of multiple fragments during Tx by raw mode simulation is enabled or not.	
			0	Enabled
			1	Disabled
iwpriv athN rawsim_stats	N	Y		Print raw mode simulation module internal statistics. Note: These are not exhaustive and do not cover events outside the simulation module, such as higher layer failure to process successfully decapped MPDUs, etc. These are meant for QCA internal debug purposes only.
iwpriv athN clr_rawsim_ stats 0	N	Y	Clear raw mode simulation module internal statistics.	
iwpriv athN rawsim_ debug <*value*>	N	Y	Enable/Disable dumping of additional debug data by raw mode simulation. Currently this consists of hex dumps of frames before/after encap/decap	
			0	Enable
			1	Disable
iwpriv athN get_rawsim_ debug	N	Y	Return whether dumping of additional debug data by raw mode simulation is enabled or not.	
			0	Enabled
			1	Disabled

1.5.9 Thermal mitigation

Thermal Mitigation supports two ways for changing the thermal configuration.

1.5.9.1 Thermal tool

Thermal tool is a user space tool implemented to configure the various thermal mitigation parameters.

Following are the available options which can be used for configuration:

-set - Specifies set operation

-get - Specifies get operation. Reads config from driver and displays on screen.

-i - Interface name wifi0 or wifi1

-e - 1: enable, 0: disable

-et - Event time in duty cycle units [E.g. 10 means each 10 duty cycle FW will send 1 event]

-dc - Duty cycle in milliseconds

-dl - It is a bitmap of four log levels. By default, only log level 1 (only error messages) is enabled.

-pN - Thermal policy for level/zone N [only policy Queue Pause:1 is supported as of now]

-loN - Low threshold for level N

-hiN - High threshold for level N

-offN - Tx Off percentage for level N

-qpN - Disable all Tx queues having priority less than configured value for level N

NOTE Option “-set” must be provided while setting the configuration and to read, the config “-get” option should be used. Help string will be displayed whenever a mistake has been made while typing the command.

Uses

Set operation:

```
#thermaltool –i wifiN –set –e 1 –dc XXX …
```

Get operation:

```
#thermaltool –i wifiN -get
```

1.5.9.2 SYSFS entries

WLAN driver provides the following SYSFS entries:

1. /sys/class/net/wifiN/thermal/mode [permission: RW, possible values: “enabled” and “disabled”]
2. /sys/class/net/wifiN/thermal/temp [permission: R, possible values: Int]
3. /sys/class/net/wifiN/thermal/thlvl [permission: RW, possible values: 0, 1, 2, 3 (in future more levels may be added)]
4. /sys/class/net/wifiN/thermal/dc [permission: RW, possible values (milliseconds): +ve int]
5. /sys/class/net/wifiN/thermal/off [permission: RW, possible values (off percent): [0, 100]]

mode

Can be set as “enabled” to enable thermal mitigation and “disabled” to disable thermal mitigation. “enabled” and “disabled” are mapped to 1 and 0 in WLAN driver respectively.

temp

Read only entry meant for reading sensor temperature reported by FW to host.

thlvl/off

These two entries are provided to facilitate setting the off percent for a specific thermal zone. These can be set in any order. Setting both of these correspond to 1 configuration. If a read operation is issued on these entries, will return 0 or the last value set depending upon whether the command is pending or completed.

dc

It affects the duty cycle of specific radio (Duty cycle of all thermal zones/levels).

Default values

Execute the command,

```
#thermaltool -i wifiN -get
```

for reading the default values. The command returns the current configuration values and if the configuration has not been changed, the outcome of the command will be default configuration -> addition

1.5.10 RDK-B HAL API PHASE 2

Parameter	Format	D	O	Description
set_waterm_th	Iwpriv athX set_waterm_th <threshold value >	Y	Y	Assign the high watermark threshold value Example: `#iwpriv ath0 set_waterm_th 40`
get_waterm_th	iwpriv athX get_waterm_th	Y	Y	Get the high watermark threshold value Example: `#iwpriv ath0 get_waterm_th`
get_th_reach	iwpriv athX get_th_reach	Y	Y	Get the number of times the threshold has been breached Example: `#iwpriv ath0 get_th_reach`

get_assoc_reach	Iwpriv athX get_assoc_ reach	Y	Y	Get the maximum number of associated devices post the threshold value Example: `#iwpriv ath0 get_assoc_reach`
get_assoc_dev_ watermark_time	Wifitool athX get_assoc_ dev_watermark_time	Y	Y	Displays the time at which the maximum number of devices has been associated post the threshold Example: `#wifitool ath0 get_assoc_dev_ watermark_time`
set_traf_rate	Iwpriv athX set_traf_rate	Y	Y	Set the traffic rate/time at which the received signal level is to be measured `#iwpriv ath0 set_traf_rate 300`
set_traf_int	Iwpriv athX set_traf_int	Y	Y	Set the traffic interval until which the received signal level has to be measured. Example `#iwpriv ath0 set_Traf_int 1800`
set_traf_stat	Iwpriv athX set_ traf_stat	Y	Y	Enable/disable the Traffic measurement statistics for the specific rate and interval. Example: `#iwpriv ath0 set_traf_stat 1/0`
display_traffic_ statistics	Wifitool athX display_traffic_ statistics	Y	Y	Display the statistics. i.e the Received signal level maximum,minum,and median at each rate until the interval time is elapsed. Example: `# wifitool ath0 display_traffic_`

1.6 Location Wireless Interface (LOWI)

LOWI-test tool provides a command line (minicom shell) facility for the end user to communicate with LOWI-server by providing commands over command line environment of AP to trigger various requests to Wi-Fi host for the following features:

- Ranging measurements (Single-sided and double-sided Round Trip Time (RTT))
- Location Civic Report (LCR) configuration
- Location Civic Information (LCI) configuration
- Where are you request to remote STA
- Fine Timing Measurement (FTM) range request

LOWI-test must be run from the minicom shell. User must first open a minicom shell.

1.6.1 Ranging measurements

LOWI-test tool can generate ranging request to LOWI using input parameters from user and receives response measurements from LOWI. It also converts measurements received to user readable format.

Once in minicom shell to issue ranging request to STA(s) use following command format:

```
lowi-test -r <input_xml_file> -n <number_of_measurements> -d <delay_in_
ms> -o <raw_output_file> -s <summary_file>
```

As a minimum the following command can be issued:

```
lowi-test -r <input_xml_file>
```

In this case, the other parameters would be default values.

1.6.1.1 Command line options

Following table list the command line options:

Option	Description
-r <input_xml_file>	Ranging Measurement Request and input XML. Input XML file can include the path in the device's file system.
-n <num>	Number of requests. Default is 1.(Optional)
-d <delay>	Delay between measurements (msec). Default is 3000. (Optional)
-o <raw_output_file>	Output file including path on device where the raw output of the test will be stored. Default is '/usr/share/location/lowi/lowi_ap_res.csv' (Optional)
-s <summary_file>	Summary file including path on device where the summary of the test will be stored. Default is '/usr/share/location/lowi /lowi_ap_summary.csv' (Optional)

1.6.1.2 User input

User input is provided via an XML file that LOWI-test parces. This input XML file should be created on some accessible location on device and should have the following format.

```
<body>
    <ranging>
        <ap>
           <!-- Specifies band for which measurement request can be made.
                 0 = 2.4 GHz, 1 = 5 GHz                 -->
            <band>x</band>

            <!-- RTT type for measurement request.
                 2 = Single sided RTT
    3 = Double sided RTT                   -->
            <rttType>x</rttType>

            <!-- Number of packets for each RTT measurement.-->
            <numFrames>x</numFrames>

<!-- Burst Duration.   Default 15.
Applicable only for double-sided RTT.
Refer to Specification 802.11 mc (D5.2)for values-->
            <burstduration>xx</burstduration>

            <!-- Frequency in Mhz of the center of total BW.   -->
            <center_freq1>x</center_freq1>

            <!-- Frequency in Mhz of the center of the second 80MHZ Lobe
if BW is
            80MHz + 80MHz.                              -->
            <center_freq2>x</center_freq2>

            <!-- Bandwidth to be used  for ranging.
                 0 = 20MHZ, 1 = 40MHZ, 2 = 80MHZ, 3 =160MHZ        -->
            <bw>x</bw>

            <!-- Channel number.                        -->
            <ch>xxx</ch>
<!-- MacId of the target.                     -->
            <mac>xx:xx:xx:xx:xx:xx</mac>
        </ap>
    </ranging>
</body>
```

More <ap> tags can be added for ranging with multiple STAs or APs

1.6.1.3 Output files

The results can be obtained either by channeling command line output to a file or by accessing raw result and summary files.

Default raw result and summary files are as follows:

/usr/share/location/lowi/lowi_ap_res.csv

/usr/share/location/lowi /lowi_ap_summary.csv

1.6.1.4 Example and test result

1.6.1.4.1 Single sided RTT

Command line command

```
lowi-test -r /tmp/ap_list.xml -n 1
```

Contents of ap_list.xml file

```
<body>
    <ranging>
        <ap>
            <band>1</band>
            <rttType>2</rttType>
            <numFrames>5</numFrames>
            <burstduration>15</burstduration>
            <center_freq1>5500</center_freq1>
            <center_freq2>0</center_freq2>
            <ch>100</ch>
            <mac>8c:fd:f0:01:e6:bd</mac>
        </ap>
    </ranging>
</body>
```

Command line output

```
*****STARTING RTT Measurement (1)*****
TIME Read = 1459251052.611184058 = 1459251052611msec
[234840.35][QCALOG-LOWIUtils] inPostcardToResponse - FROM: LOWI-SERVER,
TO:   1077154768-LOWIClient, RESP:  LOWI_RANGING_SCAN

Ranging scan results obtained!
=== Ranging SCAN RESULTS ===
[1]  8c:fd:f0:01:e6:bd 100 24090000 -92 2016/3/29:11:30:52:705
[1] 8c:fd:f0:01:e6:bd 100 23790000 -96 2016/3/29:11:30:52:705
[1] 8c:fd:f0:01:e6:bd 100 21290000 -92 2016/3/29:11:30:52:705
[1] 8c:fd:f0:01:e6:bd 100 21590000 -92 2016/3/29:11:30:52:705
[1] 8c:fd:f0:01:e6:bd 100 21290000 -96 2016/3/29:11:30:52:706
=== RANGING SCAN RESULTS END (1 APs Found) ===
TIME Read = 1459251052.709778247 = 1459251052709msec
RTT Measurement request (1 of 1)- SUCCESS, Rsp time 98[234840.36]
[QCALOG-LOWIClientReceiver] ~LOWIClientReceiver - After join complete
Issued scan Type: RANGING
Summary stats: Scan Type: RANGING
Avg Response Time: Ranging: 98 ms
AP Chan Detection rate RSSI(dBm) RTT(psec)
Min  Max Avg(dBm) Avg(W)   Min   Max   Avg
8c:fd:f0:01:e6:bd 100    1/1   (100%)  -48   -46   -46   -46 21290000
24090000 22410000
```

1.6.1.4.2 Double sided RTT

Command line command

```
lowi-test -r /tmp/ap_list.xml -n 1
```

Contents of ap_list.xml file

```
<body>
    <ranging>
        <ap>
            <band>1</band>
            <rttType>3</rttType>
            <numFrames>5</numFrames>
            <burstduration>15</burstduration>
            <center_freq1>5500</center_freq1>
            <center_freq2>0</center_freq2>
            <ch>100</ch>
            <mac>8c:fd:f0:01:e6:bd</mac>
        </ap>
    </ranging>
</body>
```

Command line output

```
STARTING RTT Measurement (1)*****
TIME Read = 1460160568.821491027 = 1460160568821msec
[139416.52][QCALOG-LOWIUtils] inPostcardToResponse - FROM: LOWI-SERVER,
TO:   1074631632-LOWIClient, RESP:  LOWI_RANGING_SCAN
=== Ranging SCAN RESULTS ===
[1]itr 8c:fd:f0:01:d1:15 149 555100 -80 2016/4/9:0:9:28:844
[1]itr 8c:fd:f0:01:d1:15 149 476900 -82 2016/4/9:0:9:28:845
=== RANGING SCAN RESULTS END (1 APs Found) ===
TIME Read = 1460160568.847545385 = 1460160568847msec
RTT Measurement request (1 of 1) - SUCCESS, Rsp time 26[139416.53][QCALOG-
LOWIClientReceiver] ~LOWIClientReceiver - After join complete
Issued scan Type: RANGING
Summary stats: Scan Type: RANGING
Avg Response Time: Ranging: 26 ms
   AP Chan   Detection rate RSSI(dBm) RTT(psec)
   Min  Max Avg(dBm) Avg(W)   Min   Max   Avg
8c:fd:f0:01:d1:15 149   1/1   (100%)  -41   -40   -40   -40 476900 555100
516000
```

1.6.2 LCR configuration

LOWI-test tool can generate a request using civic report inputs from user to set LCR information

To issue a LCR configuration request use following command format:

```
lowi-test –lcr <input_xml_file>
```

1.6.2.1 Command line options

Following table list the command line options:

Option	Description
-lcr <input_xml_file>	LCR configuration request and input XML

Input XML file which contains LCR information should be created on some accessible location on device and should have the following format.

```
<lcr_info>
    <!-- Country Code                     -->
    <country_code>xx</country_code>

    <!-- Civic Info in hex bytes format (Optional)
    Refer to Specification 802.11 mc (D5.2) Section 9.4.2.22.13
    Formatted according to IETF RFC 4776(#6005) starting at the
    country code field                       -->
    <civic_address>xxxxxxxx</civic_address>
</lcr_info>
```

1.6.2.2 Output files

The results can be obtained by channeling command line output to a file.

1.6.2.3 Example and test result

Command line command

```
lowi-test -lcr /tmp/lcr.xml
```

Contents of lcr.xml file

```
<lcr_info>
<country_code>US</country_code>        <civic_
address>0102030405060708090a0b0c0d0e0f101112131415161718191a1b1c1d1e1f20
2122232425262728292a2b2c2d2e2f303132333435363738393a3b3c3d3e3f4041424344
45464748494a4b4c4d4e4f505152535455565758595a5b5c5d5e5f606162636465666768
696a6b6c6d6e6f707172737475767778797a7b7c7d7e7f808182838485868788898a8b8c
8d8e8f909192939495969798999a9b9c9d9e9f00010203040506070809aaabacadae</ci
vic_address>
</lcr_info>
```

Command line output

```
*****SET LCR INFORMATION (1)*****
SET LCR INFORMATION (1)- SUCCESS
```

1.6.3 LCI configuration

LOWI-test tool can generate a request using civic information inputs from user to set LCI

To issue a LCI configuration request use following command format:

```
lowi-test -lci <input_xml_file>
```

1.6.3.1 Command line options

Following table list the command line options:

Option	Description
-lci <input_xml_file>	LCI configuration request and input XML

Input XML file which contains LCI information should be created on some accessible location on device and should have the following format.

When LCI input is in hex bytes format:

```
<lci_info>
    <!-- Latitude in degrees       -->
    <latitude>xxxx</latitude>

    <!-- Longitude in degrees      -->
    <longitude>xxxxx</longitude>

    <!-- Altitude in units of 1/256 m-->
    <altitude>xxxx</altitude>

    <!-- Latitude uncertainty(Optional)         -->
    <latitude_unc>xxxx</latitude_unc>

    <!-- Longitude uncertainty (Optional)  -->
    <longitude_unc>xxxxx</longitude_unc>

    <!-- Altitude uncertainty(Optional)     -->
    <altitude_unc>xxxxx</altitude_unc>

    <!-- Motion Pattern. (Optional)
                 0 = Motion Not expected
    1 = Motion Expected
    2 = Motion unknown                   -->
    <motion_pattern>x</motion_pattern>

    <!-- Floor in units 1/16th of floor # if known. (Optional)
               value is 80000000 if unknown                  -->
    <floor>x</floor>

    <!-- Height above floor in units of 1/64 m   (Optional)  -->
    <height_above_floor>xxxx</height_above_floor>

    <!-- Height uncertainty (Optional)
          value 0 means unknown, values 1-18 are valid   -->
    <height_unc>xxxx</height_unc>
</lci_info>
```

1.6.3.2 Output files

The results can be obtained by channeling command line output to a file.

1.6.3.3 Example and test result

Command line command

```
lowi-test -lci /tmp/lci.xml
```

Contents of lci.xml file

```
<lci_info>
    <latitude>37.3740</latitude>
    <longitude>-121.9960</longitude>
    <altitude>7</altitude>
    <latitude_unc>61.8212</latitude_unc>
    <longitude_unc>61.9912</longitude_unc>
    <altitude_unc>59.9624</altitude_unc>
    <motion_pattern>0</motion_pattern>
    <floor>3</floor>
    <height_above_floor>2</height_above_floor>
    <height_unc>77</height_unc>
</lci_info>
```

Command line output

```
*****SET LCI INFORMATION (1)*****
SET LCI INFORMATION (1)- SUCCESS
```

1.6.4 Where are you (LCI request)

LOWI-test tool can send an LCI (Location Configuration Information) request to an associated remote STA by issuing the command with the following format:

```
lowi-test -w <remote_sta_mac_address>
```

1.6.4.1 Command line options

Following table list the command line options:

Option	Description
-w <remote_sta_mac_address>	MAC address of remote sta in aa:bb:cc:dd:ee:ff format

1.6.4.2 Output files

The results can be obtained by channeling command line output to a file.

1.6.4.3 Example and test results

Command line command

```
lowi-test -w 8c:fd:f0:01:e6:bd
```

Command line output:

```
*****WHERE ARE YOU REQUEST (1)*****
TIME Read = 1459017322.473032633 = 1459017322473msec
WHERE ARE YOU REQUEST (1)- SUCCESS, Rsp time 1271558167
```

1.6.5 Fine Timing Measurement Range Request (FTMRR)

LOWI-test tool can generate a FTM range request to associated remote STA

To issue a FTM range request use following command format:

```
lowi-test -ftmrr <remote_sta_mac_address> <rand_interval> <input_xml_
file>
```

1.6.5.1 Command line options

Following table list the command line options:

Option	Description
-ftmrr	FTM range request
<remote_sta_mac_address>	Remote MAC address
<rand_interval>	Random interval
<input_xml_file>	Input XML file with node information

Input XML file contains target node information with which remote STA indicated by **<remote_sta_mac_address>** should do the ranging. This file should be created on some accessible location on device and should have the following format. For multiple FTMRR nodes add <element> tags to the XML.

```
<ftmrr>
    <element>
        <!-- BSSID of remote target  -->
        <bssid>xx:xx:xx:xx:xx:xx</bssid>

        <!-- BSSID info in hex bytes format.
        Refer to Specification 802.11 mc (D5.2) Section 9.4.2.37 -->
        <info_bssid>xxxxxxxx</info_bssid>

        <!-- Channel -->
        <ch>xxx</ch>

        <!-- Channel center 1  (optional) -->
        <center_ch1>xxx</center_ch1>

        <!-- Channel center 2  (optional) -->
        <center_ch2>xxxx</center_ch2>

        <!-- Operating channel width  (optional)
            0 = 20 MHz
            1 = 40 MHz
            2 = 80 MHz
            3 = 160 MHz
            4 =  80+80 MHz          -->
        <width_ch>xx</width_ch>

        <!-- Operating Class indicates the channel set of the AP
         indicated by this BSSID. Valid Operating classes are listed in
         Specification 802.11 mc (D5.2) Annex E -->
        <op_class>xx</op_class>
```

```
            <!-- PHY type of the AP indicated by this BSSID.
             Integer value coded according to the value of the dot11PHYType
             as listed in Specification 802.11 mc (D5.2) Annex C.-->
            <phy_type>xx</phy_type>
        </element>
    </ftmrr>
```

1.6.5.2 Output files

The results can be obtained by channeling command line output to a file.

1.6.5.3 Example and test result

Command line command

```
lowi-test -ftmrr 8c:fd:f0:01:e6:bd 20 /tmp/ftmrr.xml
```

Contents of ftmrr.xml file

```
<ftmrr>
    <element>
            <bssid>8C:FD:F0:07:1B:DD</bssid>
            <info_bssid>c91c0000</info_bssid>
            <ch>149</ch>
            <center_ch1>155</center_ch1>
            <center_ch2>0</center_ch2>
            <width_ch>2</width_ch>
            <op_class>128</op_class>
            <phy_type>9</phy_type>
    </element>
    <element>
            <bssid>8C:FD:F0:01:E6:BD</bssid>
            <info_bssid>c91c0000</info_bssid>
            <ch>149</ch>
            <op_class>128</op_class>
            <phy_type>9</phy_type>
    </element>
</ftmrr>
```

Command line output

```
*****FTM RANGE REQUEST (1)*****
TIME Read = 1459020136.590402291 = 1459020136590msec
FTM RANGE REQUEST (1)- SUCCESS, Rsp time 1268744050
```

1.6.6 References

For more details refer IEEE 802.11 mc Specifications.

2 UCI Wireless Configuration (QCA-Wi-Fi)

For details of UCI, visit http://wiki.openwrt.org/doc/uci.

QSDK supports qca-wifi driver natively. The UCI database section to configure it is called ‘wireless’. It can be accessed using the following command:

```
uci show wireless
```

This command will show the whole wireless section; it will be organized in the following subsection

- **radioN:** a radio subsection represents an actual radio hardware. One subsection will be initialized per-radio during the first boot. This sub-sections contains configuration parameters such as mode (11n, 11ac...), channel (1, 6 11, 36...).
- **wifi-iface[N]:** a Wi-fi-iface section represents a Wiifi VAP. It supports configuration parameters such as SSID, shortgi... The underlying radio interface is specified using the configuration item ‘device’. It should refer to a radioN section as specified above.

The driver will create the wifiN interface at init time, and the initial boot sequence will use /sys to detect these network devices and populate the UCI database accordingly.

When enabling a wifi interface, the driver will read the UCI database, and create VAPs interface using wlanconfig - one per wifi-iface subsection.

The interface names will use the following convention:

- For radio 0, the vap network devices will be called: ath0, ath01, ath02, ath03...
- For radio 1, the vap network devices will be called: ath1, ath11, ath12, ath13...

If more than 10 VAPs are create, numbering will continue as expected: ath010/ath011/ath012... for radio 0, and ath110/ath111/ath112... for radio 1.

2.1 Per VAP configuration parameters

These parameters should be set in the wireless.wifi-iface[...] section corresponding to the VAP you want to configure.

They will be applied to this particular VAP.

repacd_auto_create_vaps

Table 2-1 Per VAP configuration parameters

Parameter	Format	Description
scanband	uci set wireless.@wifi-iface[0].scanband=1	ALL (0), 2G_ONLY (1), 5G_ONLY (2)
periodicScan	uci set wireless.@wifi-iface[0].periodicScan=18000	This command sets support of sta periodic scan. 0 is disable and other value is enable. If the value is less than 30000, it will be set to 30000.
nawds_mode	uci set wireless.@wifi-iface[0].nawds_mode=3	DISABLED = 0, STATIC_REPEATER = 1, STATIC_BRIDGE = 2, LEARNING_REPEATER = 3, LEARNING_BRIDGE = 4
nawds_override	uci set wireless.@wifi-iface[0].nawds_override=00:03:7F:10:00:86	When disabled, no more MAC entry can be added to the NAWDS list when the list is full. If enabled, new MAC entry will override the dead NAWDS AP entry.
nawds_add_repeater	uci add_list wireless.@wifi-iface[0].nawds_add_repeater='00:03:7F:10:00:85 0x1'	Adds nawds repeater mac address with its capabilities into the list. More than one can be added.
nawds_defcaps	uci set wireless.@wifi-iface[0].nawds_defcaps=0x2	Set the default capability for nawds mode. HT20(0x1), HT2040(0x2), DS(0x4)
nawds_del_repeater	uci set wireless.@wifi-iface[0].nawds_del_repeater=00:03:7F:10:00:85	Remove the nawds repeater MAC from the nawds list.
revsig160	uci set wireless.@wifi-iface[0].revsig160=0 uci set wireless.@wifi-iface[0].revsig160=1	Enable/disable revised signaling for 160/80+80 MHz Default value is 1
repacd_security_unmanaged	uci set wireless.@wifi-iface[0].repacd_security_unmanaged=0 uci set wireless.@wifi-iface[0].repacd_security_unmanaged=1	If set, it disables passing of credentials, including ssid and key, to the unmanaged VAP. Default is 0.
repacd_auto_create_vaps	uci set wireless.wifi0.repacd_auto_create_vaps=0 uci set wireless.wifi0.repacd_auto_create_vaps=1	If not set, new VAPs are not created on this radio. Default is 1.

2.2 Example UCI configuration

802.11ac open mode

```
uci set wireless.wifi0=wifi-device
 uci set wireless.wifi0.type=qcawifi
 uci set wireless.wifi0.macaddr=00:60:02:00:c9:c9
 uci set wireless.wifi0.hwmode=11ac
 uci set wireless.wifi0.disabled=0
 uci set wireless.wifi0.htmode=HT80
 uci set wireless.wifi0.channel=100
 uci set wireless.wifi0.txchainmask=15
 uci set wireless.wifi0.rxchainmask=15
 uci set wireless.wifi0.mode=ap
 uci set wireless.@wifi-iface[0]=wifi-iface
 uci set wireless.@wifi-iface[0].device=wifi0
 uci set wireless.@wifi-iface[0].network=lan
 uci set wireless.@wifi-iface[0].mode=ap
 uci set wireless.@wifi-iface[0].ssid=5g_open
 uci set wireless.@wifi-iface[0].encryption=none
 uci commit wireless
```

802.11ac open mode 11ACVHT80+80 operating mode

```
uci set wireless.wifi0=wifi-device
uci set wireless.wifi0.type=qcawifi
uci set wireless.wifi0.macaddr=00:60:02:00:c9:c9
uci set wireless.wifi0.hwmode=11ac
uci set wireless.wifi0.disabled=0
uci set wireless.wifi0.htmode=HT80_80
uci set wireless.wifi0.channel=36
uci set wireless.wifi0.txchainmask=15
uci set wireless.wifi0.rxchainmask=15
uci set wireless.wifi0.mode=ap
uci set wireless.@wifi-iface[0]=wifi-iface
uci set wireless.@wifi-iface[0].device=wifi0
uci set wireless.@wifi-iface[0].network=lan
uci set wireless.@wifi-iface[0].cfreq2=106
uci set wireless.@wifi-iface[0].mode=ap
uci set wireless.@wifi-iface[0].ssid=5g_open
uci set wireless.@wifi-iface[0].encryption=none
uci commit wireless
```

802.11ac WPA2-PSK

```
uci set wireless.wifi0=wifi-device
 uci set wireless.wifi0.type=qcawifi
 uci set wireless.wifi0.macaddr=00:60:02:00:c9:c9
 uci set wireless.wifi0.hwmode=11ac
 uci set wireless.wifi0.disabled=0
 uci set wireless.wifi0.htmode=HT80
 uci set wireless.wifi0.channel=100
 uci set wireless.wifi0.txchainmask=15
 uci set wireless.wifi0.rxchainmask=15
 uci set wireless.wifi0.mode=ap
 uci set wireless.@wifi-iface[0]=wifi-iface
 uci set wireless.@wifi-iface[0].device=wifi0
 uci set wireless.@wifi-iface[0].network=lan
 uci set wireless.@wifi-iface[0].mode=ap
 uci set wireless.@wifi-iface[0].ssid=5g_wpa2
 uci set wireless.@wifi-iface[0].encryption=psk2+ccmp
 uci set wireless.@wifi-iface[0].key=12345678
 uci commit wireless
```

For other WPA2 security modes replace encryption type

```
uci set wireless.@wifi-iface[0].encryption=psk2+ccmp
 uci set wireless.@wifi-iface[0].encryption=psk2+ccmp-256
 uci set wireless.@wifi-iface[0].encryption=psk2+gcmp
 uci set wireless.@wifi-iface[0].encryption=psk2+gcmp-256
```

For WPA-PSK security mode replace encryption type

```
uci set wireless.@wifi-iface[0].encryption=tkip
```

For PMF (protected management frames) enabled AP

```
uci set wireless.wifi0=wifi-device
 uci set wireless.wifi0.type=qcawifi
 uci set wireless.wifi0.macaddr=00:60:02:00:c9:c9
 uci set wireless.wifi0.hwmode=11ac
 uci set wireless.wifi0.disabled=0
 uci set wireless.wifi0.htmode=HT80
 uci set wireless.wifi0.channel=100
 uci set wireless.wifi0.txchainmask=15
 uci set wireless.wifi0.rxchainmask=15
 uci set wireless.wifi0.mode=ap
 uci set wireless.@wifi-iface[0]=wifi-iface
 uci set wireless.@wifi-iface[0].device=wifi0
 uci set wireless.@wifi-iface[0].network=lan
 uci set wireless.@wifi-iface[0].mode=ap
 uci set wireless.@wifi-iface[0].ssid=5g_pmf
 uci set wireless.@wifi-iface[0].encryption=psk2+ccmp
 uci set wireless.@wifi-iface[0].key=12345678
 uci set wireless.@wifi-iface[0].ieee80211w=2
 uci set wireless.@wifi-iface[0].group_mgmt_cipher=AES-128-CMAC
 uci commit wireless
```

For enabling BIP CMAC/GMAC

```
uci set wireless.@wifi-iface[0].group_mgmt_cipher=AES-128-CMAC
 uci set wireless.@wifi-iface[0].group_mgmt_cipher=BIP-GMAC-128
 uci set wireless.@wifi-iface[0].group_mgmt_cipher=BIP-GMAC-256
 uci set wireless.@wifi-iface[0].group_mgmt_cipher=BIP-CMAC-256
```

WEP security configuration

```
uci set wireless.wifi0=wifi-device
 uci set wireless.wifi0.type=qcawifi
 uci set wireless.wifi0.macaddr=00:60:02:00:c9:c9
 uci set wireless.wifi0.hwmode=11ac
 uci set wireless.wifi0.disabled=0
 uci set wireless.wifi0.htmode=HT80
 uci set wireless.wifi0.channel=100
 uci set wireless.wifi0.txchainmask=15
 uci set wireless.wifi0.rxchainmask=15
 uci set wireless.wifi0.mode=ap
 uci set wireless.@wifi-iface[0]=wifi-iface
 uci set wireless.@wifi-iface[0].device=wifi0
 uci set wireless.@wifi-iface[0].network=lan
 uci set wireless.@wifi-iface[0].mode=ap
 uci set wireless.@wifi-iface[0].ssid=5g_wep
 uci set wireless.@wifi-iface[0].encryption=wep
 uci set wireless.@wifi-iface[0].key1=1111111111
 uci set wireless.@wifi-iface[0].key2=2222222222
 uci set wireless.@wifi-iface[0].key3=3333333333
 uci set wireless.@wifi-iface[0].key4=4444444444
 uci set wireless.@wifi-iface[0].key=3
 uci commit wireless
```

For WEP+Shared configuration, change encryption type

```
uci set wireless.@wifi-iface[0].encryption=wep+shared
```

WAPI security configuration

```
uci set wireless.wifi0=wifi-device
 uci set wireless.wifi0.type=qcawifi
 uci set wireless.wifi0.macaddr=00:60:02:00:c9:c9
 uci set wireless.wifi0.hwmode=11ac
 uci set wireless.wifi0.disabled=0
 uci set wireless.wifi0.htmode=HT80
 uci set wireless.wifi0.channel=100
 uci set wireless.wifi0.txchainmask=15
 uci set wireless.wifi0.rxchainmask=15
 uci set wireless.wifi0.mode=ap
 uci set wireless.@wifi-iface[0]=wifi-iface
 uci set wireless.@wifi-iface[0].device=wifi0
 uci set wireless.@wifi-iface[0].network=lan
 uci set wireless.@wifi-iface[0].mode=ap
 uci set wireless.@wifi-iface[0].ssid=5g_wapi
 uci set wireless.@wifi-iface[0].encryption=wapi-psk
 uci set wireless.@wifi-iface[0].key=12345678
 uci commit wireless
```

2.3 QWRAP configuration (basic)

2.3.1 QWRAP per radio configuration

```
uci set wireless.wifi0.channel=36
 uci set wireless.wifi0.qwrap_enable=1
 uci set wireless.wifi0.wlanaddr='00:00:00:00:00:00'
 uci set wireless.wifi0.disabled=0
```

2.3.2 QWRAP 'wrap' interface

```
uci set wireless.@wifi-iface[0].mode=wrap
 uci set wireless.@wifi-iface[0].ssid=QWRAP_ROOT2
 uci set wireless.@wifi-iface[0].encryption=psk2+ccmp
 uci set wireless.@wifi-iface[0].key=1234567890abcdexyz
uci set wireless.@wifi-iface[0].wpa_group_rekey=2000
uci set wireless.@wifi-iface[0].device=wifi0
uci set wireless.@wifi-iface[0].network=lan
```

2.3.3 QWRAP 'sta' interface

```
uci add wireless wifi-iface
uci set wireless.@wifi-iface[1].mode=sta
uci set wireless.@wifi-iface[1].device=wifi0
uci set wireless.@wifi-iface[1].network=lan
uci set wireless.@wifi-iface[1].encryption=psk2+ccmp
uci set wireless.@wifi-iface[1].key=1234567890abcdexyz
uci set wireless.@wifi-iface[1].wpa_group_rekey=2000
uci set wireless.@wifi-iface[1].ssid=QWRAP_ROOT1
uci commit wireless
uci export wireless
wifi
```

2.3.4 Automatic addition of proxystas

UCI configurations

```
uci set wireless.qcawifi=qcawifi
```

NOTE This command is mandatory if any of the following uci commands are used.

To change the bridge name: (Default is br-lan)

```
wireless.qcawifi.qwrap_br_name=br0
```

To change the Ethernet interface name: (Default is eth1)

```
wireless.qcawifi.qwrap_eth_name=eth0
```

To change the wired station limit: (Default is 20)

```
wireless.qcawifi.qwrap_sta_limit=10
```

To change the poll timer: (Default is 1 second)

```
wireless.qcawifi.qwrap_poll_timer=5
```

To enable automatic addition feature: (it is disabled on default)

```
wireless.qcawifi.qwrap_eth_sta_add_en=1
```

To enable automatic deletion feature: (it is disabled on default)

```
wireless.qcawifi.qwrap_eth_sta_del_en=1
```

2.4 QWRAP configuration (DBDC)

2.4.1 QWRAP DBDC configuration 1

One radio has STA VAP (connect to Root AP) and the other radio has AP VAP.

```
rm -rf /etc/config/wireless
 wifi detect > /etc/config/wireless
uci set wireless.wifi1=wifi-device
 uci set wireless.wifi0.type=qcawifi
 uci set wireless.wifi0.channel=36
 uci set wireless.wifi0.macaddr=8c:fd:f0:24:fa:d7
 uci set wireless.wifi0.hwmode=11ac
 uci set wireless.wifi0.qwrap_dbdc_enable=1
 uci set wireless.@wifi-iface[0]=wifi-iface
 uci set wireless.@wifi-iface[0].device=wifi0
 uci set wireless.@wifi-iface[0].network=lan
 uci set wireless.@wifi-iface[0].mode=ap
 uci set wireless.@wifi-iface[0].ssid=kris
 uci set wireless.@wifi-iface[0].encryption=psk2+ccmp
 uci set wireless.@wifi-iface[0].key=12345678
 uci set wireless.@wifi-iface[0].qwrap_ap=1
 uci set wireless.wifi1=wifi-device
 uci set wireless.wifi1.type=qcawifi
 uci set wireless.wifi1.channel=6
 uci set wireless.wifi1.macaddr=8c:fd:f0:24:fa:d8
 uci set wireless.wifi1.wlanaddr=00:00:00:00:00:00
 uci set wireless.wifi1.hwmode=11ng
 uci set wireless.wifi1.qwrap_enable=1
 uci set wireless.@wifi-iface[1]=wifi-iface
 uci set wireless.@wifi-iface[1].device=wifi1
 uci set wireless.@wifi-iface[1].network=lan
 uci set wireless.@wifi-iface[1].mode=sta
 uci set wireless.@wifi-iface[1].ssid=kris_bee
 uci set wireless.@wifi-iface[1].encryption=psk2+ccmp
 uci set wireless.@wifi-iface[1].key=12345678
```

2.4.2 QWRAP DBDC configuration 2

One radio has AP VAP and STA VAP (connect to Root AP) and the other radio has AP VAP

```
rm -rf /etc/config/wireless
 wifi detect > /etc/config/wireless

uci set wireless.wifi0=wifi-device
 uci set wireless.wifi0.type=qcawifi
 uci set wireless.wifi0.channel=36
 uci set wireless.wifi0.macaddr=8c:fd:f0:24:fa:d8
 uci set wireless.wifi0.hwmode=11ac
 uci set wireless.wifi0.qwrap_enable=1

uci set wireless.@wifi-iface[0]=wifi-iface
 uci set wireless.@wifi-iface[0].device=wifi0
 uci set wireless.@wifi-iface[0].network=lan
 uci set wireless.@wifi-iface[0].mode=wrap
 uci set wireless.@wifi-iface[0].ssid=dbdc-ap1
 uci set wireless.@wifi-iface[0].encryption=psk2+ccmp
 uci set wireless.@wifi-iface[0].key=12345678

uci add wireless wifi-iface
 uci set wireless.@wifi-iface[1]=wifi-iface
 uci set wireless.@wifi-iface[1].device=wifi0
 uci set wireless.@wifi-iface[1].network=lan
 uci set wireless.@wifi-iface[1].mode=sta
 uci set wireless.@wifi-iface[1].ssid=kris_bee
 uci set wireless.@wifi-iface[1].encryption=psk2+ccmp
 uci set wireless.@wifi-iface[1].key=12345678

uci set wireless.wifi1=wifi-device
 uci set wireless.wifi1.type=qcawifi
 uci set wireless.wifi1.channel=11
 uci set wireless.wifi1.macaddr=8c:fd:f0:24:fa:d7
 uci set wireless.wifi1.hwmode=11ng
 uci set wireless.wifi1.qwrap_dbdc_enable=1

uci set wireless.@wifi-iface[2]=wifi-iface
 uci set wireless.@wifi-iface[2].device=wifi1
 uci set wireless.@wifi-iface[2].network=lan
 uci set wireless.@wifi-iface[2].mode=ap
 uci set wireless.@wifi-iface[2].ssid=dbdc-ap2
 uci set wireless.@wifi-iface[2].encryption=psk2+ccmp
 uci set wireless.@wifi-iface[2].key=12345678
 uci set wireless.@wifi-iface[2].qwrap_ap=1
```

2.4.3 QWRAP DBDC configuration 3

Both radios has AP VAP and STA VAP (connect to Root AP).

```
rm -rf /etc/config/wireless
wifi detect > /etc/config/wireless

uci set wireless.wifi0=wifi-device
uci set wireless.wifi0.type=qcawifi
uci set wireless.wifi0.channel=1
uci set wireless.wifi0.macaddr=00:03:7f:12:34:56
uci set wireless.wifi0.hwmode=11ng
uci set wireless.wifi0.htmode=HT40
uci set wireless.wifi0.disabled=0
uci set wireless.wifi0.qwrap_enable=1
uci set wireless.@wifi-iface[0]=wifi-iface
uci set wireless.@wifi-iface[0].device=wifi0
uci set wireless.@wifi-iface[0].network=lan
uci set wireless.@wifi-iface[0].mode=wrap
uci set wireless.@wifi-iface[0].ssid=hb_qwrapap
uci set wireless.@wifi-iface[0].encryption=psk2+ccmp
uci set wireless.@wifi-iface[0].key=12345678
uci add wireless wifi-iface
uci set wireless.@wifi-iface[1]=wifi-iface
uci set wireless.@wifi-iface[1].device=wifi0
uci set wireless.@wifi-iface[1].network=lan
uci set wireless.@wifi-iface[1].mode=sta
uci set wireless.@wifi-iface[1].ssid=hb_rootap
uci set wireless.@wifi-iface[1].encryption=psk2+ccmp
uci set wireless.@wifi-iface[1].key=12345678

uci set wireless.wifi1=wifi-device
uci set wireless.wifi1.type=qcawifi
uci set wireless.wifi1.channel=36
uci set wireless.wifi1.macaddr=00:03:7f:77:22:32
uci set wireless.wifi1.hwmode=11ac
uci set wireless.wifi1.htmode=HT80
uci set wireless.wifi1.disabled=0
uci set wireless.wifi1.qwrap_enable=1
uci set wireless.@wifi-iface[2]=wifi-iface
uci set wireless.@wifi-iface[2].device=wifi1
uci set wireless.@wifi-iface[2].network=lan
uci set wireless.@wifi-iface[2].mode=wrap
uci set wireless.@wifi-iface[2].ssid=besra_qwrapap
uci set wireless.@wifi-iface[2].encryption=psk2+ccmp
uci set wireless.@wifi-iface[2].key=12345678
uci add wireless wifi-iface
uci set wireless.@wifi-iface[3]=wifi-iface
uci set wireless.@wifi-iface[3].device=wifi1
uci set wireless.@wifi-iface[3].network=lan
uci set wireless.@wifi-iface[3].mode=sta
uci set wireless.@wifi-iface[3].ssid=besra_rootap
uci set wireless.@wifi-iface[3].encryption=psk2+ccmp
uci set wireless.@wifi-iface[3].key=12345678
uci commit wireless
```

2.5 DBDC Repeater mode

Both radios has AP VAP and STA VAP (connect to Root AP). Both radios are configured on EXTAP mode.

```
rm -rf /etc/config/wireless
wifi detect > /etc/config/wireless

uci set wireless.wifi0=wifi-device
uci set wireless.wifi0.type=qcawifi
uci set wireless.wifi0.channel=1
uci set wireless.wifi0.macaddr=00:03:7f:12:34:56
uci set wireless.wifi0.hwmode=11ng
uci set wireless.wifi0.htmode=HT40
uci set wireless.wifi0.disabled=0
uci set wireless.@wifi-iface[0]=wifi-iface
uci set wireless.@wifi-iface[0].device=wifi0
uci set wireless.@wifi-iface[0].network=lan
uci set wireless.@wifi-iface[0].mode=ap
uci set wireless.@wifi-iface[0].ssid=hb_extap
uci set wireless.@wifi-iface[0].encryption=psk2+ccmp
uci set wireless.@wifi-iface[0].key=12345678
uci add wireless wifi-iface
uci set wireless.@wifi-iface[1]=wifi-iface
uci set wireless.@wifi-iface[1].device=wifi0
uci set wireless.@wifi-iface[1].network=lan
uci set wireless.@wifi-iface[1].mode=sta
uci set wireless.@wifi-iface[1].extap=1
uci set wireless.@wifi-iface[1].ssid=hb_rootap
uci set wireless.@wifi-iface[1].encryption=psk2+ccmp
uci set wireless.@wifi-iface[1].key=12345678

uci set wireless.wifi1=wifi-device
uci set wireless.wifi1.type=qcawifi
uci set wireless.wifi1.channel=36
uci set wireless.wifi1.macaddr=00:03:7f:77:22:32
uci set wireless.wifi1.hwmode=11ac
uci set wireless.wifi1.htmode=HT80
uci set wireless.wifi1.disabled=0

uci set wireless.@wifi-iface[2]=wifi-iface
uci set wireless.@wifi-iface[2].device=wifi1
uci set wireless.@wifi-iface[2].network=lan
uci set wireless.@wifi-iface[2].mode=ap
uci set wireless.@wifi-iface[2].ssid=besra_extap
uci set wireless.@wifi-iface[2].encryption=psk2+ccmp
uci set wireless.@wifi-iface[2].key=12345678
uci add wireless wifi-iface
uci set wireless.@wifi-iface[3]=wifi-iface
uci set wireless.@wifi-iface[3].device=wifi1
uci set wireless.@wifi-iface[3].network=lan
uci set wireless.@wifi-iface[3].mode=sta
uci set wireless.@wifi-iface[3].extap=1
uci set wireless.@wifi-iface[3].ssid=besra_rootap
uci set wireless.@wifi-iface[3].encryption=psk2+ccmp
uci set wireless.@wifi-iface[3].key=12345678
uci commit wireless
```

2.6 Enabling NSS Wi-Fi Offload

The default release configuration does not enable NSS wifi offload. User need to provide the below user level configuration to enable NSS wifi offload mode.

```
uci set wireless.qcawifi=qcawifi
uci set wireless.qcawifi.nss_wifi_olcfg=7
uci commit
```

These are per device level parameter.

UCI Config value specification

Each bit of the value specified represent for which radio nss offload need to be enabled

Bit 0 = 1 Radio 0 enabled for nss wifi offload

Bit 1 = 1 Radio 1 enabled for nss wifi offload

Bit 2 = 1 Radio 2 enabled for nss wifi offload

E.g.

```
nss_wifi_olcfg=7   - All bits are on so all 3 radio’s are enabled for nss
wifi offload
nss_wifi_olcfg=3   - Bit 0  and Bit 1 is set ,
       which implies the radio 0 and radio 1 enabled for nss wifi offlaod
nss_wifi_olcfg=1 - Bit 0. Radio 0 is enabled for nss wifi offlaod mode.
nss_wifi_olcfg=0 - None of radios is enabled for nss wifi offlaod mode.
```

2.7 Enabling WPS enhancement for range extenders and enabling repeater WPS configuration via a single push button

This feature can be used to propagate the WPS button push event to different VAPs in a configurable order and for a configurable duration. By default, this feature is disabled i.e. the WPS button push event is delivered to all VAPs simultaneously and they remain in active WPS mode for two minutes. Also, the SSID and security credentials of a STA VAP, which is received from the rootap, can be propagated to AP VAP(s).

Setting this to 1 enables this feature.

```
uci set wireless.qcawifi=qcawifi
uci set wireless.qcawifi.wps_pbc_extender_enhance=1
```

Setting this to 1 enables propagation of a STA VAP’s SSID and security credentials to all AP VAPs, when the STA VAP connects to the rootap.

```
uci set wireless.qcawifi.wps_pbc_overwrite_ap_settings_all=1
```

Setting this to 1 enables propagation of a STA VAP’s SSID and security credentials to the AP VAP of the same radio, when the STA VAP connects to the rootap.

```
uci set wireless.wifiX.wps_pbc_overwrite_ap_settings=1
```

This is the SSID suffix to be added to all AP VAPs, while propagating the SSID from a STA VAP.

```
uci set wireless.qcawifi.wps_pbc_overwrite_ssid_suffix=”-REPT”
```

This is the SSID suffix to be added for a particular radio.

```
uci set wireless.wifiX.wps_pbc_overwrite_ssid_band_suffix=”-BAND”
```

If set to 1, the WPS button push event will be passed to this VAP (as per the ‘start_time’ and ‘duration’) settings

```
uci set wireless.@wifi-iface[X].wps_pbc_enable=1
```

This is the delay in seconds, relative to the actual button press, the event is passed to this VAP.

```
uci set wireless.@wifi-iface[X].wps_pbc_start_time=0-240
```

This is the time in seconds, the VAP remains in active PBC mode.

```
uci set wireless.@wifi-iface[X].wps_pbc_duration=0-120
```

If set to 0, the WPS button push event will not be passed to a STA VAP, if the STA VAP is already connected to the rootap.

```
uci set wireless.wifiX.wps_pbc_try_sta_always=1
```

If set to 0, the WPS button push event will be passed to an AP VAP, even if the STA VAP of the same radio is not connected to the rootap.

```
uci set wireless.wifiX.wps_pbc_skip_ap_if_sta_disconnected=1
```

2.8 Enabling Wi-Fi memory pre-allocation

The WLAN driver feature pre-allocates memory for a specified number of VAPs, associating clients and scan entries. The default values are 16 VAPs, 124 clients and 256 scan entries.

This pre-allocation is disabled by default and to enable it, prealloc_disabled=0 must be passed as module param.

The module param, max_vaps can be used to specify the maximum number of VAPs that needs to be pre-allocated.

Additional buffer of 16 peer entries were allocated since it was noticed that one VAP might use more than one peer entry when there are some pending frames in the firmware. When these additional buffer of entries are not used by VAP structure, it can be used by the associating clients. Hence, one might notice additional clients (more than configured) getting associated.

UCI config values:

Eg:

```
uci set wireless.qcawifi=qcawifi
uci set wireless.qcawifi.prealloc_disabled=0
uci set wireless.qcawifi.max_vaps=12
```

2.9 UCI command to enable ATF

Execute the following UCI commands to enable ATF. These commands would add a module param ‘atf_ mode’ to umac module to enable/disable the ATF feature.

NOTE These commands are available only in openwrt build environment.

Implement similar commands to enable ATF on solutions using other build environments (for example, buildroot).

```
uci set wireless.qcawifi=qcawifi
uci set wireless.qcawifi.atf_mode=1
```

2.10 Single AP Band Steering Daemon (lbd) parameters

Single AP band steering is controlled by the Load Balancing Daemon (**lbd**). This is an optional daemon that is not enabled by default. To start the load balancing daemon, you need to enable at least one VAP per band. Then enable the load balancing feature either via UCI or from the web interface.

To enable and start it via UCI, use the following commands:

```
uci set lbd.@config[0].Enable=1
uci commit lbd
/etc/init.d/lbd start
```

Table 2-2 shows the parameters that can be updated in the **lbd** UCI configuration file.

Note the following about these parameters:

- Those parameters in a section with an *_Adv* suffix have been selected for functional and performance reasons. It is not recommended that they are modified without careful consideration. Those for which OEM changes are specifically not recommended have a WARNING in their description box.
- Some of these parameters are only relevant when operating in multiple AP mode and are denoted with **Multi-AP mode only** in the description. See Section 2.11 for details on starting the daemon that provides the multi-AP steering functionality.

NOTE RSSI values are in the units reported by the *wlanconfig athX list* command.

These parameters are not read directly by lbd. Rather, the /etc/init.d/lbd script generates a /tmp/lbd.conf file which is what ultimately is read by lbd. After updating the parameters via uci and committing them, be sure to restart the daemon using `/etc/init.d/lbd restart`

Table 2-2 Band and AP Steering Configurable Parameters

Configuration Type	Section	Option	Description	Default
config	config	Enable	Whether the load balancing logic is enabled or not	0
config	config	MatchingSSID	The SSID to match when limiting band steering to only a single SSID. Normally band steering will manage all SSIDs within the LAN network. This allows restricting it to a single SSID.	-
config	config	PHYBasedPrioritization	Boolean flag indicating whether preference should be given to putting or keeping 802.11ac clients on 5 GHz or not. **NOTE** This feature is currently not supported on platforms with three radios.	0
config	config_Adv	AgeLimit	The maximum age (in seconds) for measured values before they are considered too out of date from which to make a steering decision.	5
IdleSteer	IdleSteer	NormalInactTimeout	Number of seconds for the inactivity value under no overload conditions on both bands. **NOTE** This value is set to ensure the ability to detect certain clients that periodically send higher layer keep alive messages (such as an ARP request to the gateway).	10
IdleSteer	IdleSteer	OverloadInactTimeout	Number of seconds for the inactivity value when the serving band is overloaded	10
IdleSteer	IdleSteer	InactCheckInterval	How frequently (in seconds) to check for inactive associated STAs on both bands	1

Table 2-2 Band and AP Steering Configurable Parameters

Configuration Type	Section	Option	Description	Default
IdleSteer	IdleSteer	RSSISteeringPoint_DG	The point at which the measured or estimated RSSI on 2.4 GHz dictates a node associated on 5 GHz should be steered to 2.4 GHz. This default value effectively disables downgrade steering due to its limited benefit with modern clients.	5
IdleSteer	IdleSteer	RSSISteeringPoint_UG	The point at which the measured or estimated RSSI on 5 GHz dictates a node associated on 2.4 GHz should be steered to 5 GHz.	20
ActiveSteer	ActiveSteer	TxRateXingThreshold_UG	The rate (in Kbps) at which a rate crossing event should be generated for a potential active client upgrade to 5 GHz.	50000
ActiveSteer	ActiveSteer	RateRSSIXingThreshold_UG	The value (in dB) the uplink RSSI on 2.4 GHz must be above to be considered for active steering to 5 GHz. This threshold is in conjunction with the `TxRateXingThreshold_UG`.	30
ActiveSteer	ActiveSteer	TxRateXingThreshold_DG	The rate (in Kbps) at which a rate crossing event should be generated for a potential active client downgrade to 2.4 GHz. This threshold is an additional trigger (an OR condition) for downgrade (with the other trigger being the `RawRSSIXingThreshold_DG`. This value effectively disables downgrade active steering due to its limited usefulness with modem clients.	6000
ActiveSteer	ActiveSteer	RateRSSIXingThreshold_DG	The value (in dB) the uplink RSSI on 5 GHz may be below to be considered for active steering to 2.4 GHz. This threshold is an additional trigger (an OR condition) for downgrade (with the other trigger being the `TxRateXingThreshold_DG`). This value effectively disables downgrade active steering due to its limited usefulness with modem clients.	0
IAS	IAS	Enable_W2	Whether to enable Interfernce Avoidance Steering on 2.4 GHz. If this is set to 0, no attempt will be made to detect interference for a STA associated on 2.4 GHz.	1

Table 2-2 Band and AP Steering Configurable Parameters

Configuration Type	Section	Option	Description	Default
IAS	IAS	Enable_W5	Whether to enable Interference Avoidance Steering on 5 GHz. If this is set to 0, no attempt will be made to detect interference for a STA associated on 5 GHz.	1
IAS	IAS	MaxPollutionTime	The number of seconds after which a BSS that was previously marked as polluted is considered no longer polluted. **NOTE** BSS may also have its pollution cleared by output from the accumulator.	1200 (20 minutes)
IAS	IAS	UseBestEffort	Whether best effort steering should be used when the reason for the steering is IAS. When best effort steering is used, any failure will not update the client steering classification state machines.	0
Offload	Offload	MUAvgPeriod	Number of seconds to average before generating a new utilization report on both bands	60
Offload	Offload	MUOverloadThreshold_W2	Medium utilization threshold (in percentage) for an overload condition on 2.4 GHz	70
Offload	Offload	MUSOverloadThreshold_ W5	Medium utilization threshold (in percentage) for an overload condition on 5 GHz	70
Offload	Offload	MUSafetyThreshold_W2	The percentage of medium utilization that the measured plus projected utilization is allowed to reach before all further upgrade steering is disallowed until a new utilization measurement is done.	50
Offload	Offload	MUSafetyThreshold_W5	The percentage of medium utilization that the measured plus projected utilization is allowed to reach before all further upgrade steering is disallowed until a new utilization measurement is done.	60
Offload	Offload	OffloadingMinRSSI	Uplink RSSI (in dB) above which pre-association steering and post-association offloading is allowed.	20
StaDB	StaDB	IncludeOutOfNetwork	Whether out of network devices should be included in the database or not.	1

Table 2-2 Band and AP Steering Configurable Parameters

Configuration Type	Section	Option	Description	Default
StaDB	StaDB_Adv	AgingSizeThershold	The number of entries allowed in the station database before periodic aging is triggered.	100
StaDB	StaDB_Adv	AgingFrequency	Once aging is triggered, how frequently (in seconds) to perform aging of the station database.	60
StaDB	StaDB_Adv	OutOfNetworkMaxAge	The number of seconds that must elapse since the last update for an out-of-network entry before it is considered too old and is removed from the database.	300
StaDB	StaDB_Adv	InNetworkMaxAge	The number of seconds that must elapse since the last update for an in-network entry before it is considered too old and is removed from the database. Only unassociated entries will be considered for removal.	2592000 (30 days)
StaDB	StaDB_Adv	NumRemoteBSSes	Multi-AP mode only. The maximum number of statistics to store for BSSes other than those provided by the serving AP.	4
StaMonitor	StaMonitor_Adv	RSSIMeasureSamples_W2	Number of RSSI measurements to average using QoS Null Data Packets before generating a RSSI report on 2.4 GHz	5
StaMonitor	StaMonitor_Adv	RSSIMeasureSamples_W5	Number of RSSI measurements to average using QoS Null Data Packets before generating a RSSI report on 5 GHz	5
BandMonitor	BandMonitor _Adv	MUCheckInterval_W2	How frequently (in seconds) to check the medium utilization on 2.4 GHz	10
BandMonitor	BandMonitor _Adv	MUCheckInterval_W5	How frequently (in seconds) to check the medium utilization on 5 GHz	10
BandMonitor	BandMonitor _Adv	ProbeCountThreshold	The number of consecutive probe request RSSI values that must be available to consider using the average RSSI when making pre-association steering decisions. **WARNING** Changing this value is not recommended as larger values are likely to reduce the chance of pre-association steering.	1

Table 2-2 Band and AP Steering Configurable Parameters

Configuration Type	Section	Option	Description	Default
BandMonitor	BandMonitor_ Adv	MUReportPeriod	Multi-AP mode only. How often (in seconds) the medium utilization information should be collected from all nodes in the network.	30
BandMonitor	BandMonitor_ Adv	LoadBalancingAllowedMax Period	Multi-AP mode only. The amount of time that must be remaining in the `MUReportPeriod` for a load balancing slot to be assigned. This allows a second device to attempt load balancing if the first device assigned had nothing to do.	15
BandMonitor	BandMonitor_ Adv	NumRemoteChannels	Multi-AP mode only. The maximum number of channels that may be in use in the network on nodes other than the current one. This should be generally set based on the maximum number of radios in the devices being deployed in the network.	3
Estimator_ Adv	Estimator_ Adv	RSSIDiff_EstW5FromW2	Difference when estimating 5 GHz RSSI value from the one measured on 2.4 GHz.	-15
Estimator_ Adv	Estimator_ Adv	RSSIDiff_EstW2FromW5	Difference when estimating 2.4 GHz RSSI value from the one measured on 5 GHz.	5
Estimator_ Adv	Estimator_ Adv	ProbeCountThreshold	The number of consecutive probe request RSSI values that must be available to consider using the average RSSI on the unassociated band when making steering decisions. **WARNING** Reducing this value may lead to unnecessary steering in cases where there is higher variability in probe request RSSI.	3
Estimator_ Adv	Estimator_ Adv	StatsSampleInterval	The amount of time (in seconds) between consecutive samples of the byte count statistics for a STA when estimating its data rate.	1

Table 2-2 Band and AP Steering Configurable Parameters

Configuration Type	Section	Option	Description	Default
Estimator_ Adv	Estimator_ Adv	11kProhibitTime	The minimum amount of time (in seconds) to enforce between consecutive 802.11k Beacon Requests. **WARNING** Reducing this value could increase the likelihood of clients rejecting requests that occur too frequently.	30
Estimator_ Adv	Estimator_ Adv	PhyRateScalingForAirtime	The factor by which to scale the estimate PHY rate to arrive at an approximate effective MAC rate.	50%
Estimator_ Adv	Estimator_ Adv	EnableContinuousThroughpu t	Run with throughput sampling always enabled (for demo or debugging purposes only). With this option enabled, the current throughput for each associated STA will be logged every second.	0
Estimator_ Adv	Estimator_ Adv	BcnrptActiveDuration	Duration (in milliseconds) for an active mode 802.11k Beacon Request. This is used on non-DFS channels. **WARNING** Reducing this value could lead to a lower success rate for measurements.	50
Estimator_ Adv	Estimator_ Adv	BcnrptPassiveDuration	Duration (in milliseconds) for a passive mode 802.11k Beacon Request. This is used on DFS channels. **WARNING** Reducing this value could lead to a lower success rate for measurements.	200
Estimator_Adv	Estimator_ Adv	FastPollutionDetectBufSize	The size of the accumulator buffer when looking for interference starting from the no interference state. **WARNING** This value was carefully selected based on experimentation. Modification is not recommended.	5

Table 2-2 Band and AP Steering Configurable Parameters

Configuration Type	Section	Option	Description	Default
Estimator_Ad	Estimator_ Adv	NormalPollutionDetectBufSi ze	The size of the accumulator buffer after interference has been detected on the BSS. **WARNING** This value was carefully selected based on experimentation. Modification is not recommended.	5
Estimator_Adv	Estimator_ Adv	PollutionDetectThreshold	Minimum percentage of detected samples in the accumulator buffer for the BSS to be declared polluted. **WARNING** This value was carefully selected based on experimentation. Modification is not recommended.	60
Estimator_Adv	Estimator_ Adv	PollutionClearThreshold	Maximum percentage of detected samples in the accumulator buffer (after it is full) for the BSS to be declared no longer polluted. **WARNING** This value was carefully selected based on experimentation. Modification is not recommended.	40
Estimator_Adv	Estimator_ Adv	InterferenceAgeLimit	The base time used to determine when detector samples should be aged out of the accumulator buffer. Samples that are older than InterferenceAgeLimit multiplied by the buffer size will be removed.	15
Estimator_Adv	Estimator_Adv	IASLowRSSIThreshold	Minimum RSSI (in dB) required for a sample to be fed to the interference detector. Below this value, interference will be assumed not to be present as any low downlink rate (MCS) value is likely due to the path loss. **WARNING** This value was carefully selected based on experimentation. Modification is not recommended.	12

Table 2-2 Band and AP Steering Configurable Parameters

Configuration Type	Section	Option	Description	Default
SteerExec	SteerExec	SteeringProhibitTime	Number of seconds to wait prior to steering the client again after a steering when either the legacy steering mechanism is used or the 802.11v BSS Transition Management mechanism is used but the client still attempts to authenticate on a BSS other than the target one. **WARNING** Reducing this value could lead to a client blacklisting the AP due to too frequent steering.	300
SteerExec	SteerExec	BTMSteeringProhibitShortTime	The time period to wait prior to steering an 11v-capable client again after a successful steering within BTMAssociationTime. **WARNING** Reducing this value is not recommended.	30
SteerExec	SteerExec_ Adv	TSteering	Number of seconds allowed for the client to reconnect before AP aborts steering when performing legacy steering. **WARNING** Reducing this value is not recommended.	15
SteerExec	SteerExec_ Adv	InitialAuthRejCoalesceTime	Number of seconds to coalesce multiple authentication rejects down to a single one when counting consecutive auth rejects. This parameter is used in conjunction with `AuthRejMax` below to abort steering when a client is not moving to the desired BSS. **WARNING** This value was carefully selected based on client testing. Modification is not recommended.	2

Table 2-2 Band and AP Steering Configurable Parameters

Configuration Type	Section	Option	Description	Default
SteerExec	SteerExec_ Adv	AuthRejMax	The number of consecutive authentication rejects that cause steering to be aborted and the device to be marked as steering unfriendly **WARNING** This value was carefully selected based on client testing. Modification is not recommended.	3
SteerExec	SteerExec_ Adv	SteeringUnfriendlyTime	The amount of time a device is considered steering unfriendly before another attempt. This is used as the base for an exponential back-off scheme when a STA repeatedly fails legacy steering. **WARNING** This value was carefully selected based on client testing. Modification is not recommended.	600
SteerExec	SteerExec_ Adv	MaxSteeringUnfriendlyTime	Maximum time (in seconds) for the legacy steering unfriendly timer. This is used in conjunction with `SteeringUnfriendlyTime`.	604800 (1 week)
SteerExec	SteerExec_ Adv	TargetLowRSSIThreshold_ W2	RSSI threshold (in dB) indicating 2.4 GHz band is not strong enough for association. When steering to 2.4 GHz, if the uplink RSSI (as measured by probe requests) falls below this value, steering will be aborted.	5
SteerExec	SteerExec_ Adv	TargetLowRSSIThreshold_ W5	RSSI threshold (in dB) indicating 5 GHz band is not strong enough for association. When steering to 5 GHz, if the uplink RSSI (as measured by probe requests) falls below this value, steering will be aborted.	15
SteerExec	SteerExec_ Adv	BlacklistTime	The amount of time (in seconds) before automatically removing the blacklist (independent of RSSI conditions, but still subject to overload checks).	900 (15 minutes)
SteerExec	SteerExec_ Adv	BTMResponseTime	Maximum response delay for 802.11v BSS Transition Management Request.	10

Table 2-2 Band and AP Steering Configurable Parameters

Configuration Type	Section	Option	Description	Default
SteerExec	SteerExec_ Adv	BTMAssociationTime	The maximum time allowed for an 11v-capable client to reconnect before AP aborts steering the client, releases the blacklist (if in use) for the client, and marks the BTM steering attempt as having failed.	6
SteerExec	SteerExec_ Adv	BTMUnfriendlyTime	The time period to wait prior to steering an 11v-capable client again upon BTM steering failures (subject to exponential back-off). **WARNING** Reducing this value is not recommended as it may contribute to a higher steering failure rate.	600
SteerExec	SteerExec_ Adv	MaxBTMUnfriendly	Maximum time (in seconds) for the BTM steering unfriendly timer for idle steering.	86400 (1 day)
SteerExec	SteerExec_ Adv	MaxBTMActiveUnfriendly	Maximum time (in seconds) for the BTM steering unfriendly timer for active steering.	604800 (1 week)
SteerExec	SteerExec_ Adv	MinRSSIBestEffort	The RSSI (in dB) below which BTM-based steering will operate in best effort mode (where no blacklists are installed).	12
SteerExec	SteerExec_ Adv	LowRSSIXingThreshold	RSSI threshold (in dB) to generate an indication when a client crosses it (in dB)	10
SteerAlg_ Adv	SteerAlg_ Adv	MinTxRateIncreaseThreshold	Minimum amount the 5 GHz PHY rate (in Kbps) must be above the 2.4 GHz PHY rate when determining if the channel is good enough. This is only used in overload scenarios.	53
SteerAlg_Avd	SteerAlg_Adv	MaxSteeringTargetCount	The maximum number of candidates to include in an 802.11v BSS Transition Management Request or legacy steering operation. The default value reflects the fact that devices currently do not make good use of the preference value included in BTM request.	1

Table 2-2 Band and AP Steering Configurable Parameters

Configuration Type	Section	Option	Description	Default
APSteer	APSteer	LowRSSIAPSteeringThreshold_CAP	Multi-AP mode only RSSI value (in dB) below which the uplink RSSI of a STA associated to the Central AP (CAP) must fall for it to be considered as a candidate for AP steering.	20
APSteer	APSteer	LowRSSIAPSteerThreshold_RE	Multi-AP mode only RSSI value (in dB) below which the uplink RSSI of a STA associated to a Range Extender (CAP) must fall for it to be considered as a candidate for AP steering.	45
APSteer	APSteer	APSteerToRootMinRSSIIncThreshold	Multi-AP mode only The amount (in dB) the RSSI of the CAP must be better than that of the serving RE, as measured by an 802.11k Beacon Measurement, for the STA to be steered to the CAP.	5
APSteer	APSteer	APSteerToLeafMinRSSIincThreshold	Multi-AP mode only The amount (in dB) the RSSI of an RE must be better than that of the CAP, as measured by an 802.11k Beacon Measurement, for the STA to be steered to the RE.	10
APSteer	APSteer	APSteerToPeerMinRSSIIncThreshold	Multi-AP mode only The amount (in dB) the RSSI of an RE must be better than that of the serving RE, as measured by an 802.11k Beacon Measurement, for the STA to be steered to the RE.	10
APSteer	APSteer	DownlinkRSSIThreshold_W5	Multi-AP mode only The value (in dBm) the downlink RSSI, as measured using 802.11k Beacon Measurement, must be above for a 5 GHz channel to be preferred over 2.4 GHz when AP steering is used. If the downlink RSSI is not above this value, the 2.4 GHz channel will be selected for AP steering so long as its Tx power is at least as good as the 5 GHz channel that was measured using 802.11k.	-65

2.11 Multi-AP Coordinated Steering and Adaptive Path Selection parameters

The multi-AP coordinated steering and Adaptive Path Selection features are implemented within the daemon named **hyd**. To make use of these features, some additional Wi-Fi settings must be enabled and then the daemon must be started. Typically this is done through the RE Placement and Auto-Configuration Daemon (**repacd**). Steps for using this daemon are in Section 2.12

If configuring this feature manually, use the following steps on the CAP:

1. Configure the SSID and pass-phrase on the Wi-Fi interfaces as desired, enabling an AP interface on each band.
2. Enable the RRM and WDS features on each AP interface using the following commands (which assume a 2 radio device with one wireless AP interface configured on each radio):

```
uci set wireless.wifi-iface[0].wds=1
uci set wireless.wifi-iface[0].rrm=1
uci set wireless.wifi-iface[1].wds=1
uci set wireless.wifi-iface[1].rrm=1
uci commit wireless
```

3. Bring up the wireless interfaces using the `wifi` command.
4. Disable **mcsd** (as it cannot run at the same time as **hyd**):

```
uci set mcsd.config.Enable=0
uci commit mcsd
/etc/init.d/mcsd stop
```

5. Enable and start **hyd**:

```
uci set hyd.@config[0].Enable=1
uci commit hyd
/etc/init.d/hyd start
```

Then on the range extender(s), use the following steps:

1. Configure the device to not respond to DHCP requests:

```
uci set dhcp.lan.ignore=1
uci commit dhcp
/etc/init.d/dnsmasq restart
```

2. Configure the device as a pure bridge in one of two modes:

   a. Dynamic address

```
uci set network.lan.ifname='eth0 eth1'
uci set network.lan.proto=dhcp
uci delete network.wan
uci commit network
/etc/init.d/network restart
```

   b. Static IP address

```
uci set network.lan.ifname='eth0 eth1'
uci set network.lan.proto=static
uci set network.lan.ipaddr=<desired ip>
uci set network.lan.gateway=<IP of gateway>
uci set network.lan.dns=<IP of gateway>
uci delete network.wan
uci commit network
/etc/init.d/network restart
/etc/init.d/dnsmasq restart
```

3. Create Wi-Fi STA and AP interface on each radio with the appropriate SSID and pass-phrase.
4. Enable the RRM and WDS features on each AP interface using the following commands (which assume a 2 radio device with one wireless AP interface configured on each radio):

```
uci set wireless.wifi-iface[0].wds=1
uci set wireless.wifi-iface[0].rrm=1
uci set wireless.wifi-iface[1].wds=1
uci set wireless.wifi-iface[1].rrm=1
uci commit wireless
```

5. Bring up the wireless interfaces using the `wifi` command.
6. Disable **mcsd** (as it cannot run at the same time as **hyd**):

```
uci set mcsd.config.Enable=0
uci commit mcsd
/etc/init.d/mcsd stop
```

7. Enable and start **hyd**, telling it to operate as a range extender:

```
uci set hyd.@config[0].Enable=1
uci set hyd.@config[0].Mode=HYCLIENT
uci commit hyd
/etc/init.d/hyd start
```

Beyond this basic configuration, the parameters for configuring Adaptive Path Selection (APS) and coordinated steering are described in the table below. Note that although this daemon has its own configuration file, it also uses the **lbd** configuration file. For those parameters specific to AP steering, see Section 2.10

Table 2-3 Multi-AP Coordinated Steering and Adaptive Path Selection Parameters

Configuration Type	Section	Option	Description	Default
config	config	DisableSteering	Whether the steering feature should be disabled. This is primarily intended for use when testing APS where no steering of clients is desired.	0
hy	hy	ConstrainTCPMedium	Whether the less dominant direction of a TCP connection should be forced onto the same interface as the dominant direction of the connection. Generally allowing for each direction to use a different interface will result in better performance, so this feature is defaulted to off.	0

Table 2-3 Multi-AP Coordinated Steering and Adaptive Path Selection Parameters

Configuration Type	Section	Option	Description	Default
PathChWlan	PathChWlan	ScalingFactorHighRate_W5	Rate (in Mbps) above which the scaling factor for high rate links on 5 GHz should be applied. See `ScalingFactorHigh` below.	750
PathChWlan	PathChWlan	ScalingFactorHighRate_W2	Rate (in Mbps) above which the scaling factor for high rate links on 2.4 GHz should be applied. See `ScalingFactorHigh` below.	200
PathChWlan	PathChWlan	ScalingFactorLow	Conversion factor (as a percentage) when deriving a UDP capacity value from a PHY rate that falls below the low rate threshold (as determined by `LinkCapacityThreshold` below) The PHY rate is multiplied by this value to estimate the full UDP capacity.	60%
PathChWlan	PathChWlan	ScalingFactorMedium	Conversion factor (as a percentage) when deriving a UDP capacity value from a PHY rate that falls between the low rate (as determined by `LinkCapacityThreshold` below) and high rate thresholds. The PHY rate is multiplied by this value to estimate the full UDP capacity.	85%
PathChWlan	PathChWlan	ScalingFactorHigh	Conversion factor (as a percentage) when deriving a UDP capacity value from a PHY rate that falls above the high rate threshold. The PHY rate is multiplied by this value to estimate the full UDP capacity.	60%
PathChWlan	PathChWlan	ScalingFactorTCP	Conversion factor (as a percentage) when deriving a TCP capacity value from a UDP capacity value.	90%
PathChWlan	PathChWlan	UseWHCAlgorithm	Boolean flag to control whether the above scheme is used to compute the capacity or the old scheme (that relies on questionable firmware stats) is used. The WHC algorithm should be enabled for all Hy-Fi testing.	1
PathSelect	PathSelect	LinkCapacityThreshold	The threshold value (in Mbps) used for the low rate in when determining the scaling factor to use for the UDP and TCP capacity estimates.	20

Table 2-3 Multi-AP Coordinated Steering and Adaptive Path Selection Parameters

Configuration Type	Section	Option	Description	Default
SteerMsg	SteerMsg	AvgUtilReqTimeout	The number of seconds to wait for the average utilization report to be sent back to the CAP after sending the average utilization request before timing out.	1
SteerMsg	SteerMsg	LoadBalancingCompleteTimeout	The number of seconds to allow for an RE assigned a load balancing slot to send back the complete message before the CAP assumes it was lost over the air and moves on to the next device.	90
SteerMsg	SteerMsg	RspTimeout	The number of seconds to allow for a response message to come back from a node that was sent a Prepare for Steering Request, Abort Request, or STA Info Request.	2

2.12 Range Extender Placement and Auto-configuration Daemon

The RE Placement and Auto-configuration Daemon (**repacd**) simplifies the placement and configuration of range extenders in a home. It is recommended that this be used when using the Multi-AP Coordinated Steering and Adaptive Path Selection features.

The steps to configure this feature are different on the Central AP (CAP) and any range extenders (REs). Use the following steps on the CAP:

1. Configure the SSID and pass-phrase on the Wi-Fi interfaces as desired. It is not necessary to enable the Wi-Fi interfaces as repacd will do this itself.
2. Disable **mcsd** (as it cannot run at the same time as **repacd** since **repacd** may enable **hyd**):

```
uci set mcsd.config.Enable=0
uci commit mcsd
/etc/init.d/mcsd stop
```

3. Enable and start **repacd**:

```
uci set repacd.repacd.Enable=1
uci commit repacd
/etc/init.d/repacd start
```

Then on the range extender(s), use the following steps:

1. Configure the device to not respond to DHCP requests:

```
uci set dhcp.lan.ignore=1
uci commit dhcp
/etc/init.d/dnsmasq restart
```

2. Configure the device as a pure bridge in one of two modes:

a. Dynamic address

```
uci set network.lan.ifname='eth0 eth1'
uci set network.lan.proto=dhcp
uci delete network.wan
uci commit network
/etc/init.d/network restart
```

b. Static IP address

```
uci set network.lan.ifname='eth0 eth1'
uci set network.lan.proto=static
uci set network.lan.ipaddr=<desired ip>
uci set network.lan.gateway=<IP of gateway>
uci set network.lan.dns=<IP of gateway>
uci delete network.wan
uci commit network
/etc/init.d/network restart
/etc/init.d/dnsmasq restart
```

3. Unless using a platform that contains NSS, disable ECM (as this generally leads to improved performance):

```
/etc/init.d/qca-nss-ecm stop
/etc/init.d/qca-nss-ecm disable
```

4. Disable **mcsd** (as it cannot run at the same time as **repacd** since **repacd** may enable **hyd**):

```
uci set mcsd.config.Enable=0
uci commit mcsd
/etc/init.d/mcsd stop
```

5. Enable and start **repacd**:

```
uci set repacd.repacd.Enable=1
uci commit repacd
/etc/init.d/repacd start
```

Once these steps are done, press the WPS button on both the CAP and the RE and wait for a few minutes for the configuration steps to complete. Note that the same steps to configure the RE apply when using this feature against existing APs that do not support the Wi-Fi SON feature set. In this case the RE will fall back to its inter-operable mode of range extension.

Beyond this basic configuration, the parameters in the table below can further control the behavior of **repacd**.

Table 2-4 Replacement and Auto-Configuration Daemon Parameters

Configuration Type	Section	Option	Description	Default
config	repacd	Enable	Whether the RE placement and auto-configuration logic is enabled or not	0
config	repacd	ManagedNetwork	The name of the network where the Wi-Fi interfaces being managed will reside.	lan

Table 2-4 Replacement and Auto-Configuration Daemon Parameters

Configuration Type	Section	Option	Description	Default
config	repacd	DeviceType	The primary role of the device. Must be one of `RE` or `Client`. In `Client` mode, the device will only operate as a range extender if its connection to the CAP falls into the desired range.	RE
config	repacd	Role	The current role `CAP` or `NonCAP` for this device. This should generally not be changed directly, as the value is set by the init script and read by the daemon.	NonCAP
config	repacd	ConfigREMode	The mechanism to use for range extension. Supported values are: `auto`, `son`, `wds`, `qwrap`, and `extap`. In `auto` mode, the RE will configure itself based on the detected configuration of the root AP. The exact behavior is further controlled by the `DefaultREMode` parameter below. Note that QWrap and ExtAP mode do not currently support the full credential cloning logic.	wds
config	repacd	DefaultREMode	The fallback mode to use when the central AP is not detected to be running in full Wi-Fi SON or WDS mode. This can be one of `qwrap` or `extap`.	qwrap
config	repacd	BlockDFSChannels	Whether to disable DFS channels when creating AP interfaces. This may be removed in the future now that the existing VAP configuration can be reused when starting repacd.	0
config	repacd	EnableSteering	When operating in WDS mode, whether single AP band steering should be enabled.	1
config	repacd	EnableSON	When operating in full Wi-Fi SON mode, whether to enable Multi-AP Coordinated Steering. This is generally only intended for use in debugging or testing where steering is not desired. For best performance in production, leaving this feature enabled is recommended.	1
config	repacd	LinkCheckDelay	The amount of time (in seconds) to wait between successive link checks. Note that the actual amount of time between two link checks may be 1 second larger than this (due to implementation considerations).	2
WiFiLink	WiFiLink	MinAssocCheckPostWPS	The number of times the association must be deemed up after a WPS button press before it is considered stable enough before an RSSI measurement can begin.	5

Table 2-4 Replacement and Auto-Configuration Daemon Parameters

Configuration Type	Section	Option	Description	Default
config	repacd	TrafficSeparationEnabled	When operating in full Wi-Fi SON mode, whether to enable multi SSID and traffic separation support.	0
config	repacd	NetworkGuest	When multi SSID and traffic separation is enabled this parameter is used to hold the guest network's bridge name. If multi SSID and traffic separation is disabled then this parameter has no effect.	guest
WiFiLink	WiFiLink	WPSTimeout	The amount of time (in seconds) to wait for an association to take place after the WPS button is pressed. If this amount of time elapses without the STA interface associating, the device will be assumed to be too far from the CAP.	180
WiFiLink	WiFiLink	AssociationTimeout	The amount of time (in seconds) to wait for the STA interface to associate before considering the device as too far from the CAP. Note that a WPS push button cancels this timer and runs the WPS timeout instead.	300
WiFiLink	WiFiLink	RSSINumMeasurements	The number of measurements to take to arrive at an average RSSI to compare against the near/far thresholds.	5
WiFiLink	WiFiLink	RSSIThresholdFar	The signal level (in dBm) below which the RE is considered too far from the CAP and should be moved closer.	-75
WiFiLink	WiFiLink	RSSIThresholdNear	The signal level (in dBm) above which the RE is considered too close to the CAP and should be moved farther.	-60
WiFiLink	WiFiLink	RSSIThresholdMin	The signal level (in dBm) above which a device whose primary role is as a client is eligible to become a range extender (so long as it does not exceed RSSIThresholdNear).	-75
LEDState	Varies	Name_1 Name_2	The name of the LED configuration section (in `/etc/config/system`) to use to resolve this to a SysFS name.	
LEDState	Varies	Trigger_1 Trigger_2	The mode in which the LED should operate. `none` - Solid on or off `timer` - Blinking	
LEDState	Varies	Brightness_1	The value to set for the LED brightness. At least on AP148, the brightness does not seem to matter, so a value of `1` should be used for on and a value of 0 for off.	

Table 2-4 Replacement and Auto-Configuration Daemon Parameters

Configuration Type	Section	Option	Description	Default
LEDState	Varies	DelayOn_1 DelayOn_2	The amount of time (in milliseconds) the LED should stay on. This is only relevant if the corresponding trigger is set to `timer`.	
LEDState	Varies	DelayOff_1	The amount of time (in milliseconds) the LED should stay off. This is only relevant if the corresponding trigger is set to `timer`.	
WiFiLink	WiFiLink	RSSIThresholdPrefer2GBackhaul	The signal level (in dBm) on 5 GHz STA VAP, below which the 5 GHz STA VAP is brought down.	-100
WiFiLink	WiFiLink	2GBackhaulSwitchDownTime	The duration of time during which the 2.4 GHz backhaul interface must be down before the 5 GHz backhaul is brought back up.	10
WiFiLink	WiFiLink	MaxMeasuringStateAttempts	Number of consecutive times 5 GHz interface can go down while doing an RSSI measurement before it is considered unstable and is forcefully disabled.	3

Table 2-5 Wireless configuration file parameters

Configuration Type	Option	Description	Default
wifi-iface	repacd_security_unmanaged	If set, passing of credentials to the unmanaged VAP is disabled including SSID, and key.	0
wifi-device	repacd_auto_create_ vaps	Prevents creation of new VAPs on desired radio.	1

NOTE The LEDState sections, the section name can take one of the following values:

- `NotAssociated` - STA interface is still trying to associate
- `WPSInProgress` - WPS button was pressed and the timeout has not yet occurred
- `Measuring` - STA is associated and an average downlink RSSI value is being computed
- `WPSTimeout` - Failed to establish an association within `WPSTimeout` seconds
- `AssociationTimeout` - Failed to establish an association within
- `AssociationTimeout` seconds
- `RE_MoveCloser` - RSSI is too weak or the STA was unable to associate
- `RE_MoveFarther` - RSSI is too strong (duplicating coverage)

- `RE_LocationSuitable` - RSSI is sufficient for the backhaul without too much coverage overlap
- `CL_LinkSufficient` - RSSI is sufficient for the device to act as a client device but is not sufficient for it to become a range extender.
- `CL_LinkInadequate` - RSSI is too weak or the device cannot even associate.
- `CL_ActingAsRE` - RSSI is in the sweet spot to allow the device to act as an RE while continuing to meet the client requirements.

This allows the LED scheme to be tweaked through the configuration file. The value names have a suffix to allow for up to 2 LEDs to be controlled in a given state. All parameters with the same suffix apply to the same LED.

2.13 Co-ordinated ATF between RootAP & Repeater

Co-ordinated ATF feature allows the user to modify ATF configurations of Repeater AP from the Root AP. The user would enter ATF configurations for a repeater including the Repeater MAC address at the RootAP which would then be passed on to the Repeater AP & configurations are applied. Coordinated ATF feature uses wsplcd daemon to achieve this. The following sections details the changes done in the wsplcd configuration file to support this feature

2.13.1 Enable Co-ordinated ATF feature in wsplcd config file

The option 'atfConfigEnable' need to be set to '1' to enable Coordinated ATf feature. Note that this option need to be enabled on both RootAP & Repeater side for this feature to work.

```
uci set wsplcd.config.atfConfigEnable=1
```

NOTE Pre-requisites to run wsplcd would be to enable , 'wps_pbc' option on all AP interfaces in the wireless config file & enabling 'HyfiSecurity option in wsplcd config file.

wsplcd daemon need to be restarted everytime there is a configuration change. After uci commands are executed, restart the wsplcd daemon using the following command:

```
/etc/init.d/wsplcd restart
```

2.13.2 Config section 'atf-config'

This is a per repeater Section which includes the Repeater MAC address (MAC address of STA interface on repeater end) & a Repeater name which is used as an identifier to map subsequent sections to the Repeater. Note that there can be only 1 such section for a repeater. UCI commands corresponding to this section is as below:

```
uci set wsplcd.Repeater1=atf-config
uci set wsplcd.Repeater1.REmac=<mac address of the repeater>
uci set wsplcd.Repeater1.REname=<identifier>
```

NOTE If 'REmac' mentioned is 'ff:ff:ff:ff:ff:ff', the same configuration will be sent to all connected Repeaters

2.13.3 Config section 'atf-re-config'

This section need to be created per Repeater & per ATF rule. Each section can have one ATF configuration say, either 1 SSID based or 1 Peer based. Multiple rules should not be clubbed together under a single section. If there are multiple ATF rules to be configured for a repeater, multiple such sections should be created with Rename pointing to the same Repeater Identifier. In other words, the number of 'atf-re-config' section depends on the number of ATF rules that is intended to be configured. 'REname' is used to map 'atf-re-config' section with 'atf-config' section & thereby to the Repeater for which this configuration is intended for.

The following uci commands would create 3 'atf-re-config' sections for the repeater, RE0; the first section for ssid based ATF, 2nd for Peer based ATF & 3rd section for Group based ATF.

SSID Based ATF configuration

```
uci add wsplcd atf-re-config
uci set wsplcd.@atf-re-config[X].REname=RE0           # The repeater
identifier as mentioned in atf-config section
uci set wsplcd.@atf-re-config[X].ssid=RE_1            # SSID for which
the rule is created
uci set wsplcd.@atf-re-config[X].ssid_cmd=addssid     #command can be
either 'addssid' or 'delssid'
uci set wsplcd.@atf-re-config[X].ssid_wifidev=ath01   #Interface name on
which the rule is to be applied
uci set wsplcd.@atf-re-config[X].ssid_val=60          #The airtime
percentage that need to be applied
```

Peer Based ATF configuration

```
uci add wsplcd atf-re-config                          #creates a new
'atf-re-config' section
uci set wsplcd.@atf-re-config[X].REname=RE0           # The repeater
identifier as mentioned in atf-config section
uci set wsplcd.@atf-re-config[X].sta=00:33:33:33:33:33 #MAC address of
the station for which the rule is created
uci set wsplcd.@atf-re-config[X].sta_cmd=addsta       #command should be
either 'addsta' or 'delsta'
uci set wsplcd.@atf-re-config[X].sta_wifidev=ath01    #Interface name on
which the rule is to be applied
uci set wsplcd.@atf-re-config[X].sta_val=10           #The airtime
percentage that need to be applied
```

Group Based ATF configuration

```
uci add wsplcd atf-re-config                          #creates a new
'atf-re-config' section
uci set wsplcd.@atf-re-config[X].REname=RE0           # The repeater
identifier as mentioned in atf-config section
uci set wsplcd.@atf-re-config[X].group=group1         #group name to be
created
uci set wsplcd.@atf-re-config[X].group_cmd=addgroup   #command should be
either 'addgroup' or 'delgroup'
uci add_list wsplcd.@atf-re-config[X].group_ssid=ssid1 #List of SSIDs
that need to be added to the group
uci add_list wsplcd.@atf-re-config[X].group_ssid=ssid2#List of SSIDs that
need to be added to the group
uci add_list wsplcd.@atf-re-config[X].group_ssid=ssid3#List of SSIDs that
need to be added to the group
uci set wsplcd.@atf-re-config[X].group_wifidev=ath01  #Interface name on
which the rule is to be applied
uci set wsplcd.@atf-re-config[X].group_val=20         #The airtime
percentage that need to be applied
uci set wsplcd.@atf-re-config[X].group_enable=1       # command to enable
group. Group commands can be applied only if                group
feature is enabled.
```

2.13.4 Config section 'atf-re-radioparams-config'

This section includes ATF radio parameter configurations. This would include ATF scheduling policy, obss enable/disable & Inter group scheduling policy setting. This is per Repeater, per Radio section.

In other words, a separate section has to be created for each radio interfaces on the same Repeater.

```
uci add wsplcd atf-re-radioparams-config                    # Creates a
new 'atf-re-radioparams-config' section
uci set wsplcd.@atf-re-radioparams-config[1].REname=RE0     # The
repeater identifier as mentioned in atf-config
section
uci set wsplcd.@atf-re-radioparams-config[1].radio_name=wifi0# Radio
name on which the rule is to be applied
uci set wsplcd.@atf-re-radioparams-config[1].sched_policy=STRICT
#Scheduling policy can be either 'STRICT' or 'FAIR'
uci set wsplcd.@atf-re-radioparams-config[1].sched_group_policy=FAIR
#Inter group scheduling policy can be either
'STRICT' or 'FAIR'
uci set wsplcd.@atf-re-radioparams-config[1].sched_obss_enable=1   # Set
this field to '1' or '0' to enable or disable OBSS scheduling
respectively
```

NOTE There can be a maximum of 8 Repeaters that can be configured at a time. Each Repeaters can have upto 8 ATF SSID based configurations, 8 ATF Peer Based configurations, 3 ATF radio param configurations , 2 ATF Group configurations with 4 SSIDS per group at a time.

Sample configuration file

The following sample configuration file represents ATF configuration for 3 Repeaters

Repeater 1: 1 SSID based, 1 Peer based & 1 Group configuration

Repeater 2: 3 SSID based configurations

Repeater 3: 2 Peer based configurations

```
config atf-config 'Repeater1'
   option REmac '00:01:02:03:04:05'
   option REname 'RE0'

config atf-config 'Repeater2'
   option REmac '00:01:02:03:04:06'
   option REname 'RE1'

config atf-config 'Repeater3'
   option REmac '00:01:02:03:04:07'
   option REname 'RE2'

config atf-re-config
   option REname 'RE0'
   option ssid 'rootap_AP'
   option ssid_wifidev 'ath01'
   option ssid_val '60'
   option ssid_cmd 'addssid'

config atf-re-config
   option REname 'RE0'
   option sta '00:33:33:33:33:33'
   option sta_wifidev 'ath01'
   option sta_val '10'
   option sta_cmd 'addsta'

config atf-re-config
   option REname 'RE0'
   option group 'group1'
   option group_wifidev 'ath01'
   option group_val '20'
   list group_ssid 'ssid1'
   list group_ssid 'ssid2'
   list group_ssid 'ssid3'
   option group_cmd 'addgroup'
   option group_enable '1'

config atf-re-config
   option REname 'RE1'
   option ssid 'ssid_RE1'
   option ssid_cmd 'addssid'
   option ssid_wifidev 'ath0'
   option ssid_val '10'
```

config atf-re-config

```
option REname 'RE1'
option ssid 'ssid_RE1_2'
option ssid_cmd 'addssid'
option ssid_wifidev 'ath0'
option ssid_val '20'
```

config atf-re-config

```
option REname 'RE1'
option ssid 'ssid_RE1_3'
option ssid_cmd 'addssid'
option ssid_wifidev 'ath0'
option ssid_val '20'
```

config atf-re-config

```
option REname 'RE2'
option sta '00:22:22:22:22:22'
option sta_wifidev 'ath01'
option sta_val '10'
option sta_cmd 'addsta'
```

config atf-re-config

```
option REname 'RE2'
option sta '00:44:44:44:44:44'
option sta_wifidev 'ath01'
option sta_val '10'
option sta_cmd 'addsta'
```

2.14 Avoiding module reload during wifi up/down

Instead of reloading all the wifi modules during wifi up/down, just a firmware reload could help in reducing memory free/alloc cycles. During the last VAP removal, the firmware is unloaded and during the first VAP creation, the firmware is re-initialized.

With the new implementation, ‘wifi’ will just unload and reload the firmware. For normal configuration changes, ‘wifi’ is sufficient.

If any module parameters are changed or if the user specifically wants to reload the modules, a new command ‘wifi load’ has been added to reload the modules.

List of module params for which wifi load needs to be passed so that the modules are reloaded:

```
testmode
vow_config
ol_bk_min_free
ol_be_min_free
ol_vi_min_free
ol_vo_min_free
ar900b_emu
frac
intval
atf_mode
atf_msdu_desc
```

```
atf_peers
atf_max_vdevs
fw_dump_options
enableuartprint
ar900b_20_targ_clk
qca9888_20_targ_clk
max_descs
max_peers
qwrap_enable
otp_mod_param
max_active_peers
enable_smart_antenna
nss_wifi_olcfg
max_clients
max_vaps
enable_smart_antenna_da
prealloc_disabled
lteu_support
enable_mesh_support
enable_mesh_peer_cap_update
```

For other uci changes, normal wifi command is sufficient.

A Country Code Definitions

Table A-1 identifies the country definition, country string, and country code used to set the country ID for 802.11d and regulatory requirements.

Table A-1 Country code definitions

Country definition	Country string	Country ID
CTRY_DEBUG	DB	0
CTRY_DEFAULT	NA	0
CTRY_ALBANIA	AL	8
CTRY_ALGERIA	DZ	12
CTRY_ARGENTINA	AR	32
CTRY_ARMENIA	AM	51
CTRY_AUSTRALIA	AU	36
CTRY_AUSTRALIA2	AU	5000
CTRY_AUSTRIA	AT	40
CTRY_AZERBAIJAN	AZ	31
CTRY_BAHRAIN	BH	48
CTRY_BELARUS	BY	112
CTRY_BELGIUM	BE	56
CTRY_BELGIUM2	BE	5002
CTRY_BELIZE	BZ	84
CTRY_BOLIVIA	BO	68
CTRY_BOSNIA_HERZ	BA	70
CTRY_BRAZIL	BR	76
CTRY_BRUNEI_DARUSSALAM	BN	96
CTRY_BULGARIA	BG	100
CTRY_CANADA	CA	124
CTRY_CANADA2	CA	5001
CTRY_CHILE	CL	152
CTRY_CHINA	CN	156
CTRY_COLOMBIA	CO	170
CTRY_COSTA_RICA	CR	188
CTRY_CROATIA	HR	191

Table A-1 Country code definitions (cont.)

Country definition	Country string	Country ID
CTRY_CYPRUS	CY	196
CTRY_CZECH	CZ	203
CTRY_DENMARK	DK	208
CTRY_DOMINICAN_REPUBLIC	DO	214
CTRY_ECUADOR	EC	218
CTRY_EGYPT	EG	818
CTRY_EL_SALVADOR	SV	222
CTRY_ESTONIA	EE	233
CTRY_FAEROE_ISLANDS	FO	234
CTRY_FINLAND	FI	246
CTRY_FRANCE	FR	250
CTRY_GEORGIA	GE	268
CTRY_GERMANY	DE	276
CTRY_GREECE	GR	300
CTRY_GUATEMALA	GT	320
CTRY_HONDURAS	HN	340
CTRY_HONG_KONG	HK	344
CTRY_HUNGARY	HU	348
CTRY_ICELAND	IS	352
CTRY_INDIA	IN	356
CTRY_INDONESIA	ID	360
CTRY_IRAN	IR	364
CTRY_IRAQ	IQ	368
CTRY_IRELAND	IE	372
CTRY_ISRAEL	IL	376
CTRY_ITALY	IT	380
CTRY_JAMAICA	JM	388
CTRY_JAPAN	JP	392
CTRY_JAPAN1	JP	393
CTRY_JAPAN2	JP	394
CTRY_JAPAN3	JP	395
CTRY_JAPAN4	JP	396
CTRY_JAPAN5	JP	397
CTRY_JAPAN6	JP	4006
CTRY_JAPAN7	JP	4007
CTRY_JAPAN8	JP	4008

Table A-1 Country code definitions (cont.)

Country definition	Country string	Country ID
CTRY_JAPAN9	JP	4009
CTRY_JAPAN10	JP	4010
CTRY_JAPAN11	JP	4011
CTRY_JAPAN12	JP	4012
CTRY_JAPAN13	JP	4013
CTRY_JAPAN14	JP	4014
CTRY_JAPAN15	JP	4015
CTRY_JAPAN16	JP	4016
CTRY_JAPAN17	JP	4017
CTRY_JAPAN18	JP	4018
CTRY_JAPAN19	JP	4019
CTRY_JAPAN20	JP	4020
CTRY_JAPAN21	JP	4021
CTRY_JAPAN22	JP	4022
CTRY_JAPAN23	JP	4023
CTRY_JAPAN24	JP	4024
CTRY_JAPAN25	JP	4025
CTRY_JAPAN26	JP	4026
CTRY_JAPAN27	JP	4027
CTRY_JAPAN28	JP	4028
CTRY_JAPAN29	JP	4029
CTRY_JAPAN30	JP	4030
CTRY_JAPAN31	JP	4031
CTRY_JAPAN32	JP	4032
CTRY_JAPAN33	JP	4033
CTRY_JAPAN34	JP	4034
CTRY_JAPAN35	JP	4035
CTRY_JAPAN36	JP	4036
CTRY_JAPAN37	JP	4037
CTRY_JAPAN38	JP	4038
CTRY_JAPAN39	JP	4039
CTRY_JAPAN40	JP	4040
CTRY_JAPAN41	JP	4041
CTRY_JAPAN42	JP	4042
CTRY_JAPAN43	JP	4043
CTRY_JAPAN44	JP	4044

Table A-1 Country code definitions (cont.)

Country definition	Country string	Country ID
CTRY_JAPAN45	JP	4045
CTRY_JAPAN46	JP	4046
CTRY_JAPAN47	JP	4047
CTRY_JAPAN48	JP	4048
CTRY_JAPAN49	JP	4049
CTRY_JAPAN50	JP	4050
CTRY_JAPAN51	JP	4051
CTRY_JAPAN52	JP	4052
CTRY_JAPAN53	JP	4053
CTRY_JAPAN54	JP	4054
CTRY_JAPAN55	JP	4055
CTRY_JAPAN56	JP	4056
CTRY_JAPAN57	JP	4057
CTRY_JAPAN58	JP	4058
CTRY_JAPAN59	JP	4059
CTRY_JORDAN	JO	400
CTRY_KAZAKHSTAN	KZ	398
CTRY_KENYA	KE	404
CTRY_KOREA_NORTH	KP	408
CTRY_KOREA_ROC	KR	410
CTRY_KOREA_ROC3	KR	412
CTRY_KUWAIT	KW	414
CTRY_LATVIA	LV	428
CTRY_LEBANON	LB	422
CTRY_LIBYA	LY	434
CTRY_LIECHTENSTEIN	LI	438
CTRY_LITHUANIA	LT	440
CTRY_LUXEMBOURG	LU	442
CTRY_MACAU	MO	446
CTRY_MACEDONIA	MK	807
CTRY_MALAYSIA	MY	458
CTRY_MALTA	MT	470
CTRY_MEXICO	MX	484
CTRY_MONACO	MC	492
CTRY_MOROCCO	MA	504
CTRY_NETHERLANDS	NL	528

Table A-1 Country code definitions (cont.)

Country definition	Country string	Country ID
CTRY_NETHERLANDS_ANTILLES	AN	530
CTRY_NEW_ZEALAND	NZ	554
CTRY_NICARAGUA	NI	558
CTRY_NORWAY	NO	578
CTRY_OMAN	OM	512
CTRY_PAKISTAN	PK	586
CTRY_PANAMA	PA	591
CTRY_PARAGUAY	PY	600
CTRY_PERU	PE	604
CTRY_PHILIPPINES	PH	608
CTRY_POLAND	PL	616
CTRY_PORTUGAL	PT	620
CTRY_PUERTO_RICO	PR	630
CTRY_QATAR	QA	634
CTRY_ROMANIA	RO	642
CTRY_RUSSIA	RU	643
CTRY_SAUDI_ARABIA	SA	682
CTRY_SERBIA_MONTENEGRO	CS	891
CTRY_SINGAPORE	SG	702
CTRY_SLOVAKIA	SK	703
CTRY_SLOVENIA	SI	705
CTRY_SOUTH_AFRICA	ZA	710
CTRY_SPAIN	ES	724
CTRY_SRI_LANKA	LK	144
CTRY_SWEDEN	SE	752
CTRY_SWITZERLAND	CH	756
CTRY_SYRIA	SY	760
CTRY_TAIWAN	TW	158
CTRY_THAILAND	TH	764
CTRY_TRINIDAD_Y_TOBAGO	TT	780
CTRY_TUNISIA	TN	788
CTRY_TURKEY	TR	792
CTRY_UAE	AE	784
CTRY_UKRAINE	UA	804
CTRY_UNITED_KINGDOM	GB	826
CTRY_UNITED_STATES	US	840

Table A-1 Country code definitions (cont.)

Country definition	Country string	Country ID
CTRY_UNITED_STATES2	US	841
CTRY_UNITED_STATES_FCC49	PS	842
CTRY_URUGUAY	UY	858
CTRY_UZBEKISTAN	UZ	860
CTRY_VENEZUELA	VE	862
CTRY_VIET_NAM	VN	704
CTRY_YEMEN	YE	887
CTRY_ZIMBABWE	ZW	716