AP 10.2 Command Line Interface (CLI)

User Guide

80-Y7207-3 Rev. F

October 23, 2014

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	November 2013	Initial release for AP 10.2 software. For 10.1 software, see 80-Y0984-1. The following is a summary of the changes documented between AP software versions 10.1 and 10.2. Added sections on channel loading, channel hopping, Intelligent Channel Selection (ICS), aggregate size scaling, Hy-Fi, 256QAM. Added get command to ANI. Added to DCS-IM commands. Added burst/get_burst commands to aggregation parameters. Added value 5 for mcastenhance. Added HMWDS/HMMC.
B	December 2013	Modified parameter spelling for set_dcs_coch_th, get_dcs_coch_th, set_dcs_maxcu, get_dcs_maxcu (Table 1-11); and getacsctrlflags (Table 1-51).
C	March 2014	Added content for Smart Antenna feature.
D	April 2014	In the following tables, removed “et” from the “get” command names: ■ Table 1-11: g_dcs_phyerrth ■ Table 1-13: g_dgaf_disable ■ Table 1-15: g_mcastenhance ■ Table 1-33: g_dscp_ovride ■ Table 1-51: g_acs_bkscanen, g_acs_rssivar, g_ acs_dbgtrace Updated Table 1-38. Added Section 1.3.31.1. Updated Table 1-29 with commands for Enhanced Independent Repeater feature
E	May 2014	In the following tables, removed “et” from the “get” command names: ■ Table 1-14: g_disablecoext ■ Table 1-22: g_periodicScan Added Section 1.3.33.
F	October 2014	Removed or reformatted red text in Table 1-25, Table 1-29, Table 1-31, and Table 1-47. The commands shown in revisions D and E in the subsection titled “Uncategorized protocol layer parameters” have been removed until further notice. Added Section 1.5.7, UCI Wireless configuration (qca-wifi).

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:

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 10.2 Programmer's Guide, 80-Y7207-1*
- *AP 10.2 Driver User's Guide, 80-Y7207-2*

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.

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.

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 10.2 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 and QCA955x.

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 and QCA989x.

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 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 11ac interface aggregation parameters

Parameter	Format	DA	OL	Description
addba **delba**	`iwpriv ath`*N* `addba AID` *AC BufSize* `iwpriv ath`*N* `delba AID AC initiator reason`			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 **setaddbaoper**). Both require the AID and AC specified. The AID value is shown by the **wlanconfig list** 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. `#iwpriv ath0 addba 1 0 32` `#iwpriv ath0 delba 1 0 1 36`
addbaresp	`iwpriv ath`*N* `addbaresp AID` *AC status*			Sends an addba response frame on the indicated AID and AC. AID is the value shown under the AID column using the command **wlanconfig list**. 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. `#iwpriv ath0 addbaresp 1 0 25`

Table 1-2 11ac interface aggregation parameters (cont.)

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...32}*	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. Specific to 802.11ac. `#iwpriv ath0 maxampdu 1` `#iwpriv ath0 get_maxampdu` `ath0 get_maxampdu:1`
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. Specific to 802.11ac. `#iwpriv ath0 vhtmaxampdu 1` `#iwpriv ath0 get_vhtmaxampdu` `ath0 get_vhtmaxampdu:1`
getaddbastatus	`iwpriv ath`*N* `getaddbastatus` *status*			Gets the ADDBA (Add Block Acknowledgement) status for AID (Association Identifier) and TID (Traffic Identifier). *What is the format of aid, tid when returned?* #iwpriv ath0 getaddbastatus aid ath0 getaddbastatus:
				aid — AID number of STA
				tid — TID number between 0-15
setaddbaoper **getaddbaoper**	`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` `#iwpriv ath0 getaddbaoper` `ath0 getaddbaoper:0`

Table 1-3 11na and 11ac interfaces specific statistics

Parameter	Format	DA	OL	Description
addba **delba**	`iwpriv athN addba AID` *AC BufSize* `iwpriv athN delba AID AC initiator reason`			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 **setaddbaoper**). Both require the AID and AC specified. The AID value is shown by the **wlanconfig list** 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. `#iwpriv ath0 addba 1 0 32` `#iwpriv ath0 delba 1 0 1 36`
addbaresp	`iwpriv athN addbaresp AID` *AC status*			Sends an addba response frame on the indicated AID and AC. AID is the value shown under the AID column using the command **wlanconfig list**. 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. `#iwpriv ath0 addbaresp 1 0 25`
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 peregrine cards. It is 0 for the rest. Specific to 802.11n. `#iwpriv wifi0 burst 1` `#iwpriv wifi0 get_burst` `get_burst:1`
txrx_fw_stats	iwpriv athN txrx_fw_stats *{1...6}*	N	Y	Tx and Rx related statistics from target `#iwpriv ath0 txrx_fw_stats 1`
getaddbastatus	`iwpriv athN getaddbastatus` *status*			Gets the ADDBA (Add Block Acknowledgement) status for AID (Association Identifier) and TID (Traffic Identifier). *What is the format of aid, tid when returned?* `#iwpriv ath0 getaddbastatus aid` `ath0 getaddbastatus:`
				aid: AID number of STA
				tid: TID number between 0-15
setaddbaoper **getaddbaoper**	`iwpriv athN 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` `#iwpriv ath0 getaddbaoper` `ath0 getaddbaoper:0`

1.3.2 ANI parameters

Table 1-4 ANI parameters

Parameter	Format	DA	OL	Description
addba **delba**	`iwpriv ath`*N* `addba AID` *AC BufSize* `iwpriv ath`*N* `delba AID AC initiator reason`			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 **setaddbaoper**). Both require the AID and AC specified. The AID value is shown by the **wlanconfig list** 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. `#iwpriv ath0 addba 1 0 32` `#iwpriv ath0 delba 1 0 1 36`
addbaresp	`iwpriv ath`*N* `addbaresp AID` *AC status*			Sends an addba response frame on the indicated AID and AC. AID is the value shown under the AID column using the command **wlanconfig list**. 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. `#iwpriv ath0 addbaresp 1 0 25`
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`
getaddbastatus	`iwpriv ath`*N* `getaddbastatus` *status*			Gets the ADDBA (Add Block Acknowledgement) status for AID (Association Identifier) and TID (Traffic Identifier). *What is the format of aid, tid when returned?* #iwpriv ath0 getaddbastatus aid ath0 getaddbastatus:
				aid: AID number of STA
				tid: TID number between 0-15
setaddbaoper **getaddbaoper**	`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` `#iwpriv ath0 getaddbaoper` `ath0 getaddbaoper:0`

1.3.3 Association/ACL parameters

Table 1-5 Association/ACL parameters

Parameter	Format	DA	OL	Description
addmac **delmac** **getmac** **maccmd** **get_maccmd**	iwpriv ath*N* addmac *macaddr* iwpriv ath*N* delmac *macaddr* iwpriv ath*N* maccmd *cmd*	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`
ap_bridge **get_ap_bridge**	iwpriv ath*N* ap_bridge *mode*	Y	Y	Enables or disables 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*			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`

1.3.4 Beacon configuration parameters

Table 1-6 Beacon configuration parameters

Parameter	Format	DA	OL	Description
ampdudensity **get_ampdudensity**	iwpriv ath*N* ampdudensity *MPDU_density*			Sets the value of MPDU density. The default is 6. The get parameter returns the current value. `#iwpriv ath0 ampdudensity 6` `#iwpriv ath0 get_ampdudensity` `ath0 get_ampdudensity:6`
				0 — No time restriction
				1 — 0.25 µs
				2 — 0.5 µs
				3 — 1 µs
				4 — 2 µs
				5 — 4 µs
				6 — 8 µs
				7 — 16 µs
amsdu **get_amsdu**	iwpriv ath*N* amsdu *0/1*			Sets the aggregated MSDUs (AMSDUs) transmission setting. `#iwpriv ath0 amsdu 0` `#iwpriv ath0 get_amsdu` `ath0 get_amsdu:0`
				0 — Disable AMSDU transmission
				1 — Enable AMSDU transmission
amsdulimit **get_amsdulimit**	iwpriv ath*N* amsdulimit *AMPDU_limits*			Sets the value of 11n A-MPDU limits. `#iwpriv ath0 amsdulimit` `#iwpriv ath0 get_amsdulimit` `ath0 get_amsdulimit:`
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 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. `#iwpriv ath0 bintval 400` `#iwpriv ath0 get_bintval` `ath0 get_bintval:200`

Table 1-6 Beacon configuration parameters (cont.)

Parameter	Format	DA	OL	Description
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`
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. The get parameter returns the current value. `#iwpriv ath0 dtim_period 5` `#iwpriv ath0 get_dtim_period` `wifi0 get_dtim_period:1`

Table 1-6 Beacon configuration parameters (cont.)

Parameter	Format	DA	OL	Description
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
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`

Table 1-6 Beacon configuration parameters (cont.)

Parameter	Format	DA	OL	Description
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
vap_ind **get_vap_ind**	iwpriv ath*N* vap_doth *1/0*			Enables/disables VAP WDS independance set
				0 — Disable wds independance set
				1 — Enable wds independance set

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 (sSecondary) 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>Description</th></tr>
<tr><td rowspan="5">chwidth
get_chwidth</td><td rowspan="5">iwpriv athN
chwidth channelwidth</td><td rowspan="5">Y</td><td rowspan="5">Y</td><td>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 cwmode, because it can be automatically overridden.
<pre>#iwpriv ath0 chwidth 0
#iwpriv ath0 get_chwidth
   ath0 get_chwidth:0</pre></td></tr>
<tr><td>0 Use the device settings</td></tr>
<tr><td>1 20 MHz</td></tr>
<tr><td>2 20/40 MHz</td></tr>
<tr><td>3 VHT80</td></tr>
<tr><td>cwmenable
get_cwmenable</td><td>iwpriv athN cwmenable {1|0}</td><td>Y</td><td>Y</td><td>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.
<pre>#iwpriv ath0 cwmenable 1
#iwpriv ath0 get_cwmenable
   ath0 get_cwmenable:1</pre></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

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. See Table 1-9. `#iwpriv ath0 dbgLVL 256` `# iwpriv ath0 getdbgLVL` `ath0 getdbgLVL:256`
				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
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

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

Symbolic name	Bit value	Description
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.

Table 1-10 HAL debug flags (cont.)

Symbolic name	Enable Bit	Description
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

Parameter	Format	DA	OL	Description
dcs_enable **get_dcs_enable**	iwpriv wifi*N* dcs_enable *value*	Y	Y	Enable or disable DCS. `#iwpriv wifi0 dcs_enable 0` `#iwpriv wifi0 get_dcs_enable` `wifi0 get_dcs_enable:0`
				0 — Disable DCS
				1 — Enable DCS for CW interference mitigation (CW_IM).
				2 — 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).
				3 — Enable both DCS for CW_IM and DCS for WLAN_IM
set_dcs_intrth **get_dcs_intrth**	iwpriv wifi*N* set_dcs_intrth *value*	Y	N	Configures co-channel interference threshold (in percent) to trigger channel change. Default *value* of co-channel interference threshold is 30%. `#iwpriv wifi0 set_dcs_intrth 30` `#iwpriv wifi0 get_dcs_intrth` `wifi0 get_dcs_intrth:30`
set_dcs_errth **get_dcs_errth**	iwpriv wifi*N* set_dcs_errth *value*	Y	Y	Configures transmission failure rate threshold, used to indicates the presence of interference. Default *value* of transmission failure rate threshold is 30%. `#iwpriv wifi0 set_dcs_errth 30` `#iwpriv wifi0 get_dcs_errth` `wifi0 get_dcs_errth:30`
s_dcs_phyerrth **g_dcs_phyerrth**	iwpriv wifi*N* s_dcs_phyerrth *value*	Y	Y	Configures channel time wasted due to each PHY error (PHY error Penalty). Default *value* of PHY error penalty is set as 500 ?sec. `#iwpriv wifi0 s_dcs_phyerrth 500` `#iwpriv wifi0 g_dcs_phyerrth` `wifi0 g_dcs_phyerrth:500`
set_dcs_coch_th **get_dcs_coch_th**	iwpriv wifi*N* set_dcs_coch_th *value*	N	Y	Configures co-channel interference threshold (in percent) to trigger channel change. Default *value* of co-channel interference threshold is 30. `#iwpriv wifi1 set_dcs_coch_th 30` `#iwpriv wifi1 get_dcs_coch_th` `dcs_coch_th:30`

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`

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_htol40intol` `ath0 get_htol40intol: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 1-30 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.
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.)

Parameter	Format	DA	OL	Description
mcastenhance **g_** **mcastenhance**	iwpriv ath*N* mcastenhance *mode*	Y	Y	Set multi-cast enhancement mode. `#iwpriv ath0 mcastenhance 0` `#iwpriv ath0 g_mcastenhance` `ath0 g_mcastenhance:0`
				AP software versions 9.2/9.3/9.4
				0 Disable multi-cast enhancement
				1 Enable multi-cast enhancement; use tunneling mode.
				2 Enable multi-cast enhancement; use translating mode.
				AP software versions 9.5/9.5.1/9.5.2/9.5.3
				Value / **Snooping** / **Multi-cast enhancement**
				0 / Enabled / True multi-cast packet is send if any interested member is present.
				1 / Enabled / Tunnelled unicast packet is send to interested members.
				2 / Enabled / Translated unicast packet is send to interested members.
				4 / Enabled / Disabled (Set bit 2 = 1)
				5 / Disabled / Enables Hy-Fi managed multi-cast functionality.
me_adddeny	iwpriv ath*N* me_adddeny *groupaddresstbl*	Y	N	Adds the table of group addresses that are ***not*** to be learned. The *groupaddresstbl* 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
me_cleardeny	iwpriv ath*N* me_cleardeny *value*	Y	N	Clears the snoop deny table entries. *value* can be any integer; however, the parameter clears the snoop deny table regardless of *value*.
me_length **get_me_length**	iwpriv ath*N* me_length *tablelength*	Y	N	Sets the snoop table length as number of entries. The default value is 32. The *param* range is 0 – 32. The get parameter returns the current value.
me_showdeny	iwpriv ath*N* me_showdeny *groupaddresstbl*	Y	N	Displays the table of group addresses that are ***not*** to be learned. Two addresses exist in the snoop deny table by default: 224.0.0.1, 239.255.255
medebug **get_medebug**	iwpriv ath*N* medebug *debuglevel*	Y	N	Sets the debug level for multicast enhancement. Param argument can accept any combination of the values below. The get parameter returns the current value.
				0 IEEE80211_ME_DBG_NONE
				1 IEEE80211_ME_DBG_INFO
				2 IEEE80211_ME_DBG_DEBUG
				4 IEEE80211_ME_DBG_DUMP
				8 IEEE80211_ME_DBG_ALL

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.
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.
retrydur get_retrydur	iwpriv wifi*N* retrydur *threshold_period*	Y	N	Sets the retry threshold in μs. Feature disabled if set at 0. `#iwpriv wifi0 retrydur 0` `#iwpriv wifi0 get_retrydur` `wifi0 get_retrydur: 0`

1.3.12 Physical layer parameters

Table 1-16 Physical layer parameters

Parameter	Format	DA	OL	Description
noedgech **get_noedgech**	iwpriv ath*N* noedgech			Forces the AP to avoid band edge channels when selecting a channel. `#iwpriv ath0 noedgech` `#iwpriv ath0 get_noedgech` `ath0 get_noedgech:`
LDPC **getLDPC**	iwpriv wifi*N* 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 wifi0 LDPC 1` `# iwpriv wifi0 getLDPC` `wifi0 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 98 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>Description</th></tr>
<tr><td>txchainmask
rxchainmask
get_
txchainmask
get_
rxchainmask</td><td>iwpriv wifiN
txchainmask mask
iwpriv wifiN
rxchainmask mask</td><td>Y</td><td>Y</td><td>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 2 or 1.
NOTE: 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:
<table><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></table>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.
<pre>#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</pre></td></tr>
</table>

Table 1-16 Physical layer parameters (cont.)

Parameter	Format	DA	OL	Description
TXPowLim2G **TXPowLim5G** **getTxPowLim2G** **getTxPowLim5G**	iwpriv wifi*N* TXPowLim2G *limit* iwpriv wifi*N* TXPowLim5G *limit*	Y	Y	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 100` `#iwpriv wifi0 getTxPowLim2G` `wifi0 getTxPowLim2G:100`
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`

1.3.13 Protection mechanism parameters

Table 1-17 Protection mechanism parameters

Parameter	Format	DA	OL	Description
protmode **get_protmode**	iwpriv ath*N* protmode *{1|0}*	Y	Y	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. `#iwpriv ath0 protmode 0` `#iwpriv ath0 get_protmode` `ath0 get_protmode:0`
				0 — No protection
				1 — CTS to self
				2 — RTS/CTS
extprotmode **get_** **extprotmode**	iwpriv ath*N* extprotmode *protectionmode*	Y	Y	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. `#iwpriv ath0 extprotmode 0` `#iwpriv ath0 get_extprotmode` `ath0 get_extprotmode:0`
				0 — None, no protection
				1 — CTS to self
				2 — RTS/CTS

1.3.14 Radio-related parameters

Table 1-18 Radio-related parameters

Parameter	Format	Description	
6MBAck **Get6MBack**	`iwpriv wifi`*N* `6MBAck` *1|0*	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. `#iwpriv wifi0 6MBAck 1` `#iwpriv wifi0 Get6MBAck` `wifi0 Get6MBAck:1`	
AddSWBbo **SWBcnRespT** **DMABcnRespT** **GetAddSWBbo** **GetSWBcnRespT** **GetDMABcnRespT**	`iwpriv wifi`*N* `SWBcnRespT` `iwpriv wifi`*N* `DMABcnRespT` `iwpriv wifi`*N* `AddSWBb0`	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. `#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`	
		SWBcnRespT	Software beacon response time represents the time, in ms, required to process beacons in software
		DMABcnRespT	DMA beacon response time, the time required to transfer a beacon message from memory to the MAC queue
		AddSWBb0	Additional software beacon back-off is an estimated variable for final adjustment of the ready time offset

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

Parameter	Format	Description	
AggrProt **AggrProtDur** **AggrProtMax** **getAggrProt** **getAggrProtDur** **getAggrProtMax**	`iwpriv wifi`*N* `AggrProt` *1|0* `iwpriv wifi`*N* `AggrProtDur` *duration* `iwpriv wifi`*N* `AggrProtMax` *size*	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. `#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`	
		AggrProt	Enables (1) or disables (0 = Default) this function.
		AggrProtDur	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.
		AggrProtMax	Indicates the largest aggregate size to receive RTS/CTS protection. Default is 8192 bytes.
ANIEna **GetANIEna**	iwpriv wifi*N* ANIEna *0|1*	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. `#iwpriv wifi0 ANIEna 1` `#iwpriv wifi0 GetANIEna` `wifi0 GetANIEna:1`	

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

Parameter	Format	Description	
AntSwap **DivtyCtl** **GetAntSwap** **GetDivtyCtl**	iwpriv wifi*N* AntSwap *1|0* iwpriv wifi*N* DivtyCtl *AntSel*	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. `#iwpriv wifi0 AntSwap 1` `#iwpriv wifi0 DivtyCtl 2` `#iwpriv wifi0 GetAntSwap` `wifi0 GetAntSwap:0` `#iwpriv wifi0 GetDivtyCtl` `wifi0 GetDivtyCtl:0`	
		AntSwap	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).
		DivtyCtl	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.
BcnNoReset **getBcnNoReset**	iwpriv wifi*N* BcnNoReset *1|0*	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. `#iwpriv wifi0 BcnNoReset 1` `#iwpriv wifi0 getBcnNoReset` `wifi0 getBcnNoReset:1`	
CABlevel **getCABlevel**	iwpriv wifi*N* CABlevel *%Multicast*	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. `#iwpriv wifi0 CABlevel 50` `#iwpriv wifi0 getCABlevel` `wifi0 getCABlevel:50`	

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*	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*	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.)

<table>
<tr><th>Parameter</th><th>Format</th><th>Description</th></tr>
<tr><td>chanbw
get_chanbw</td><td>iwpriv athN
chanbw
channel bandwidth</td><td>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.
<pre>#iwpriv ath0 chanbw 1
#iwpriv ath0 get_chanbw
   ath0 get_chanbw:1</pre>
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0</td><td>Full channel bandwidth</td></tr>
<tr><td>1</td><td>Half channel bandwidth</td></tr>
<tr><td>2</td><td>Quarter channel bandwidth</td></tr>
</table></td></tr>
<tr><td>CWMIgnExCCA
GetCWMIgnExCCA</td><td>iwpriv wifiN
CWMIgnExCCA 1|0</td><td>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.
<pre>#iwpriv wifi0 CWMIgnExCCA 1
#iwpriv wifi0 GetCWMIgnExCCA
   wifi0 GetCWMIgnExCCA:0</pre></td></tr>
<tr><td>extbusythres
g_extbusythres</td><td><code>iwpriv athN
extbusythres</code> pctBusy</td><td>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.
<pre>#iwpriv ath0 extbusythres 50
#iwpriv ath0 g_extbusythres
   ath0 g_extbusythres:50</pre></td></tr>
<tr><td>fastcc
get_fastcc</td><td>iwpriv <code>ath</code>N fastcc 1|0</td><td>Enables fast channel change. A value of 1 indicates that channel changes within band will be done without resetting the chip. A value of 0 indicates that any channel change will require a chip reset. The default value is 0. The get parameter returns the current value.
<pre>#iwpriv ath0 fastcc 1
#iwpriv ath0 get_fastcc
   ath0 get_fastcc:1</pre></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>FIRStepLvl
GetFIRStepLvl</td><td>iwpriv wifiN FIRStepLvl level</td><td colspan="2">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.
<code>#iwpriv wifi0 FIRStepLvl 1
#iwpriv wifi0 GetFIRStepLvl
wifi0 GetFIRStepLvl:0</code></td></tr>
<tr><td rowspan="3">ForceBias
ForBiasAuto
GetForceBias
GetForBiasAuto</td><td rowspan="3">iwpriv wifiN ForBiasAuto 1|0
iwpriv wifiN ForceBias Bias</td><td colspan="2">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
#iwpriv wifi0 ForceBias 2
#iwpriv wifi0 GetForBiasAuto
wifi0 GetForBiasAuto:0
#iwpriv wifi0 GetForceBias
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>getchaninfo</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 {
u_intic_nchans;
struct ieee80211_channel ic_chans[IEEE80211_CHAN_MAX];
};
struct ieee80211_channel {
u_int16_t ic_freq; /* setting in MHz */
u_int32_t ic_flags; /* see below */
u_int8_t ic_flagext; /* see below */
u_int8_t ic_ieee; /* IEEE channel number */
int8_t ic_maxregpower; /* max. regulatory Tx power in dBm */
int8_t ic_maxpower; /* max. Tx power in dBm */
int8_t ic_minpower; /* min. Tx power in dBm */
};</code></td></tr>
</table>

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

Parameter	Format	Description
HTEna **GetHTEna**	iwpriv wifi*N* HTEna *1|0*	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. `#iwpriv wifi0 HTEna 1` `#iwpriv wifi0 GetHTEna` `wifi0 GetHTEna:1`
mcast_rate **get_mcast_rate**	iwpriv ath*N* mcast_rate *rate*	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. `#iwpriv ath0 mcast_rate 10000` `#iwpriv ath0 get_mcast_rate` `ath0 get_mcast_rate: 10000`
NoiseImmLvl **GetNoiseImmLvl**	iwpriv wifi*N* NoiseImmLvl *level*	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`

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

Parameter	Format	Description
OFDMTrgLow **OFDMTrgHi** **GetOFDMTrgLow** **GetOFDMTrgHi**	iwpriv wifi*N* OFDMTrgLow *Low Threshold* iwpriv wifi*N* OFDMTrgHi *High Threshold*	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`
OFDMWeakDet **GetOFDMWeakDet**	iwpriv wifi*N* OFDMWeakDet *1|0*	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`

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

Parameter	Format	Description
rate11a **rate11b** **rate11g** **get_rate11a** **get_rate11b** **get_rate11b**	iwpriv `ath`*N* rate11a *roaming_rate* iwpriv `ath`*N* rate11b *roaming_rate* iwpriv `ath`*N* rate11g *roaming_rate*	Sets the roaming rate for each band usage. These rates are used to determine if a new AP is required. If the data rate on the link drops below these values, the scan module will determine if a better AP on the same ESS can be used. Values are specified in 500 Kbps increments, so a value of 48 indicates a rate of 24 Mbps. This command has a corresponding get parameter, and its default value is 48 for 802.11a and 18 for 802.11bg. `#iwpriv ath0 rate11a 32` `#iwpriv ath0 rate11b 2` `#iwpriv ath0 rate11g 10` `#iwpriv ath0 get_rate11a` `ath0 get_rate11a:32` `#iwpriv ath0 get_rate11b` `ath0 get_rate11b:2` `#iwpriv ath0 get_rate11g` `ath0 get_rate11g:10`
rssi11a **rssi11b** **rssi11g** **get_rssi11a** **get_rssi11b** **get_rssi11g**	iwpriv `ath`*N* rssi11a *roaming_threshold* iwpriv `ath`*N* rssi11b *roaming_threshold* iwpriv `ath`*N* rssi11g *roaming_threshold*	Sets the RSSI threshold for roaming in 802.11a, 802.11b, and 802.11g modes. These thresholds are used to make roaming decisions based on signal strength from the current set of APs available. The values are provided in dB. Have corresponding get parameters. The default value for 802.11b and 802.11g is 24 dBm. `#iwpriv ath0 rssi11a 30` `#iwpriv ath0 rssi11b 30` `#iwpriv ath0 rssi11g 30` `#iwpriv ath0 get_rssi11a` `ath0 get_rssi11a:30` `#iwpriv ath0 get_rssi11b` `ath0 get_rssi11b:30` `#iwpriv ath0 get_rssi11g` `ath0 get_rssi11g:30`
RSSIThrLow **RSSIThrHi** **GetRSSIThrLow** **GetRSSIThrHi**	iwpriv wifi*N* RSSIThrLow *far threshold* iwpriv wifi*N* RSSIThrHi *near threshold*	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`

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

Parameter	Format	Description
rtparams	iwpriv wifiN rtparams *ratetable_indexPER value probe_interval*	Configures the rate table. No get parameter available. *Rate Table Index:* 0: BE/BK traffic 1: VI/VO traffic *PER Value:* If the PER value for a rate is above this threshold, then this rate will not be included in the rate series. *Probe Interval:* Determines probe frequency. `#iwpriv wifi0 rtparams`
scanvalid **get_scanvalid**	iwpriv `ath`*N* scanvalid *period*	Sets the period that scan data is considered value for roaming purposes. If scan data is older than this period, a scan is forced to determine if roaming is required. The period is specified in seconds. This command has a corresponding get parameter, and its default is 60 seconds. `#iwpriv ath0 scanvalid 30` `#iwpriv ath0 get_scanvalid` `ath0 get_scanvalid:30`
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`

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

Parameter	Format	Description
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.	
SpurImmLvl **GetSpurImmLvl**	iwpriv wifi*N* SpurImmLvl *level*	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`

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

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`

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

sendtsmrpt	wifitool ath*N* sendtsmrpt *dstmac num_rpt rand_ivl meas_dur tid peermacaddr 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*	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.)

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.)

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.)

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

doth_pwrtgt **get_doth_pwrtgt**	`iwpriv athN 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. 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`
doth_reassoc	`iwpriv athN doth_ reassoc` *value*	Instructs the driver to generate a re association request. The single value provided is not used; it is more of a single-shot action than a setting. This command has no default, and no corresponding get parameter. `#iwpriv ath0 doth_reassoc 1`

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.
authmode **get_authmode**	iwpriv ath*N* 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 0. The get parameter returns the current value. *Note that the get parameter will be implemented in a future release.* `#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_** **countermeasures**	iwpriv ath*N* 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_countermeasures` `ath0 get_countermeasures:1`

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

Parameter	Format	DA	OL	Description
driver_caps **get_driver_caps**	iwpriv ath*N* driver_caps caps			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:

		0x00000001	WEP	0x00004000	Short Slot Time
		0x00000002	TKIP	0x00008000	Short Preamble
		0x00000004	AES	0x00010000	Monitor Mode
		0x00000008	AES_CCM	0x00020000	TKIP MIC
		0x00000010	HT Rates	0x01000000	WPA 2
		0x00000020	CKIP	0x00800000	WPA 1
		0x00000040	Fast Frame	0x02000000	Burst
		0x00000080	Turbo	0x04000000	WME
		0x00000100	IBSS	0x08000000	WDS
		0x00000200	Power Management	0x10000000	WME TKIP MIC
		0x00000400	Host AP	0x20000000	Background Scan
		0x00000800	Ad Hoc Demo	0x40000000	UAPSD
		0x00001000	Software Retry	0x80000000	Fast Channel Change
		0x00002000	Tx Power Mgmt		

dropunencrypted **get_dropunencry**	iwpriv ath*N* dropunencrypted *0*|*1*			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`

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

Parameter	Format	DA	OL	Description
keymgtalgs **get_keymgtalgs**	iwpriv ath*N* keymgtalgs algs			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:
				Value / **Algorithm**
				0 / WPA_ASE_NONE
				1 / WPA_ASE_8021X_UNSPEC
				2 / WPA_ASE_8021X_PSK
				3 / The command combines the supported algorithms, so a value of 3 indicates both unspecified and PSK support
mcastcipher **get_mcastcipher**	iwpriv ath*N* mcastcipher *cipher*			Used mainly by the hostapd daemon; sets the cipher used for multi-cast. The iwpriv command sets the VAP cipher type, as required to support operation of the host_apd authenticator. Has no default value; has a corresponding get parameter. `#iwpriv ath0 mcastcipher 1` `#iwpriv ath0 get_mcastcipher` `ath0 get_mcastcipher:1` The cipher values include:
				Value / **Cipher type**
				0 / IEEE80211_CIPHER_WEP
				1 / IEEE80211_CIPHER_TKIP
				2 / IEEE80211_CIPHER_AES_OCB
				3 / IEEE80211_CIPHER_AES_CCM
				5 / IEEE80211_CIPHER_CKIP
				6 / IEEE80211_CIPHER_NONE
mcastkeylen **get_mcastkeylen**	iwpriv ath*N* mcastkeylen *length*	Y	Y	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. `#iwpriv ath0 mcastkeylen 5` `#iwpriv ath0 get_mcastkeylen` `ath0 get_mcastkeylen:5`

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

Parameter	Format	DA	OL	Description
privacy **get_privacy**	iwpriv ath*N* privacy 1|0			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. `#iwpriv ath0 privacy 1` `#iwpriv ath0 get_privacy` `ath0 get_privacy:1`
rsncaps **get_rsncaps**	iwpriv ath*N* rsncaps flags			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. `#iwpriv ath0 rsncaps 0x01` `#iwpriv ath0 get_rsncaps` `ath0 get_rsncaps:1`
setfilter	iwpriv ath*N* setfilter filter			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. `#iwpriv ath0 setfilter 0x24`
				Value **Algorithm**
				0x01 Beacon
				0x02 Probe request
				0x04 Probe response
				0x08 Association request
				0x10 Association response
				0x20 Authentication
				0x40 Deauthentication
				0x80 Disassociation
				0xFF ALL
setiebuf **getiebuf**	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: `struct ieee80211req_getset_appiebuf {` `u_int32_t app_frmtype; /*mgmt frame type for which buffer is added */` `u_int32_t app_buflen; /*application supplied buffer length */` `u_int8_t app_buf[];` `};`			

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 ath*N* 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 ath*N* 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 ath*N* 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 ath*N* 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.
bgscan **get_bgscan**	iwpriv ath*N* bgscan *1*|*0*	Enables or disables background scanning. Background scanning occurs on a specified interval to update the list of known APs. This command is only valid when a VAP is operating in STA mode. The default value is 1. The get parameter returns the current value. `#iwpriv ath0 bgscan 1` `#iwpriv ath0 get_bgscan` `ath0 get_bgscan:1`

Table 1-22 STA parameters (cont.)

Parameter	Format	Description
bgscanidle **get_bgscanidle**	iwpriv ath*N* bgscanidle *idlePeriod*	Sets the amount of time the background scan must be idle before it is restarted; it is different from the background scan interval, in that if the background scan is delayed for a long period, when it is complete it will be idle for this period even if the scan interval times out. This time is indicated in seconds. The default value is 250 seconds. The get parameter returns the current value. `#iwpriv ath0 bgscanidle 200` `#iwpriv ath0 get_bgscanidle` `ath0 get_bgscanidle:200`
bgscanintvl **get_bgscanintvl**	iwpriv ath*N* bgscanintvl *interval*	Sets the interval to perform background scans. A scan is started each time the interval times out, or if the idle interval is not timed out when the idle interval is complete. The interval timer is started when the scan is started, so a idle period timeout shifts all subsequent scan intervals. The interval value is specified in seconds. The default value is 300. The get parameter returns the current value. `#iwpriv ath0 bgscanintvl 250` `#iwpriv ath0 get_bgscanintvl` `ath0 get_bgscanintvl:250`
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
periodicScan **g_periodicScan**	iwpriv ath*N* 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 ath*N* 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 TDLS Parameters

Table 1-23 TDLS Parameters

Parameter	Format	Description
clr_tdls_rmac **set_tdls_rmac**	iwpriv ath*N* clr_tdls_rmac	
tdls **get_tdls**	iwpriv ath*N* tdls	
tdlsaction **gettdlsaction**	iwpriv ath*N* tdlsaction	
tdlsmacaddr1 **gettdlsmacaddr1**	iwpriv ath*N* tdlsmacaddr1	
tdlsmacaddr2 **gettdlsmacaddr2**	iwpriv ath*N* tdlsmacaddr2	

1.3.20 Turbo parameters

Table 1-24 Turbo parameters

Parameter	Format	Description	
burst **get_burst**	iwpriv ath*N* 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 ath*N* compression *1|0*	Enables/disables Data compression support Atheros supper G The get parameter returns the current value. `#iwpriv ath0 compression 0` `#iwpriv ath0 get_compression` `ath0 get_compression:0`	
		0	Disable
		1	Enable
ff **get_ff**	iwpriv ath*N* ff *1|0*	Enables/disables fast frames support of Atheros supper G. The get parameter returns the current value. #iwpriv ath0 ff 0 #iwpriv ath0 get_ff ath0 get_ff:0	
		0	Disable
		1	Enable

Table 1-24 Turbo parameters (cont.)

Parameter	Format	Description	
periodicScan **get_periodicScan**	iwpriv ath*N* 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
turbo **get_turbo**	iwpriv ath*N* turbo *1|0*	Enables/disables turbo prime support, which is related to Atheros super G dynamic turbo.	
		0	Disable
		1	Enable

1.3.21 Tx beamforming parameters

Table 1-25 Tx beamforming parameters

Parameter	Format	Description	
set_cvtimeout **get_cvtimeout**	`iwpriv wifi`*N* `set_cvtimeout`	Set CV update period in 1 ms resolution. The get parameter returns the current value. The macro "ATH_SUPPORT_TxBF" controls whether the commands are supported.	
TxBFCTL **GetTxBFCTL**	`iwpriv wifi`*N* `TxBFCTL`	Set TxBF control. The get parameter returns the current value. The macro "ATH_SUPPORT_TxBF" controls whether the commands are supported.	
		bit0	Reserved
		bit2	Enable Explicit Beamforming with no compressed report request (Bfer side)
		bit3	Enable Explicit Beamforming with compressed report request (Bfer side)
		bit4	Reserved
		bit5	Enable Explicit Beamforming with no compressed immediate report (Bfee side)
		bit6	Enable Explicit Beamforming with compressed immediate report (Bfee side)
		bit7	Enable Explicit Beamforming with no compressed delay report (Bfee side)
		bit8	Enable Explicit Beamforming with compressed delay report (Bfee side)
		bit9	Disable applying steering CV matrix

1.3.22 Unassociated power consumption improvement parameters

Table 1-26 Unassociated power consumption improvement parameters

Parameter	Format	Description	
ignore11d **get_ignore11d**	iwpriv `ath`*N* ignore11d *1|0*	Processes or ignores 11d beacon `#iwpriv ath0 ignore11d 0` `#iwpriv ath0 get_ignore11d` `ath0 get_ignore11d:0`	
		0	Process 11d beacon
		1	Ignore 11d beacon
scanprsleep **get_scanprsleep**	iwpriv `ath`*N* scanprsleep	Sets the value of scan pre-sleep. The get parameter returns the current value.	
sleepprscan **get_sleepprscan**	iwpriv `ath`*N* sleepprscan	Sets the value of sleep pre-scan. The get parameter returns the current value.	

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 1-27 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 that 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 1-28.	
				dword4: paramValue (required only for set_sa_param)	

Table 1-28 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.

Table 1-28 dword3 parameters (cont.)

ParamName	Param ID	Node or Radio param	Description
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	Goodput 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.
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.

Table 1-28 dword3 parameters (cont.)

ParamName	Param ID	Node or Radio param	Description
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 goodput 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.

1.3.24 WDS parameters

Table 1-29 WDS parameters

Parameter	Format	DA	OL	Description
nobeacon **get_nobeacon**	`iwpriv athN nobeacon`			Enables/disables VAP to transmit beacon and probe response. The get parameter returns the current value. 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 with DEBUG. Result is correct; ignore error message in console.
				3 Enable Extender AP support with DEBUG
vap-ind	iwpriv ath*N* vap-ind *{1|0}*	N	N	Enables (1) or disables (0) repeater independent mode. If this option is disabled, the AP VAP will wait for the STA VAP to connect before starting to transmit the beacons. If this option is enabled, the AP VAP will start transmitting beacons independently of the STA VAP status.
athnewind **get_athnewind**	iwpriv ath*N athnew*ind *{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 “WiFi 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 1-30 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 1-31 lists the parameters accessible in the Qualcomm Atheros driver.

Table 1-31 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 Table 1-30. *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.
qosnull	iwpriv `ath`*N* qosnull *0/1*			Force sends QoS null data and sets power management bit on (1) or off (0). This is an action command, thus does not have any get parameter or default value. `#iwpriv ath0 qosnull 0`

Table 1-31 WMM parameters (cont.)

Parameter	Format	DA	OL	Description
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`
				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 1-31 WMM parameters (cont.)

Parameter	Format	DA	OL	Description
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 0. 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 1-32 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 1-33 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 s_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 s_dsco_ovride 1` `#iwpriv wifi0 g_dsco_ovride` `g_dsco_ovride:1`

Table 1-33 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_tidmap **get_dscp_tidmap**	iwpriv athX set_dscp_tidmap <tid><map> iwpriv athX get_dscp_tidmap <tid>	Y	Y	To configure a specific tid for specific map value. Iwpriv option set_dscp_tidmap should be set to <tid> <map> value `#iwpriv ath0 set_dscp_tidmap 1 2` `#iwpriv ath0 get_dscp_tidmap 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 reportflooding. 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 jumpstart button gets pushed. `#iwpriv ath0 nopbn 1` `#iwpriv ath0 get_nopbn 1` `get_nopbn:1`

1.3.28 Channel loading/Channel hopping parameters

Table 1-34 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. `#iwpriv ath0 acsmaxdwell 100` `#iwpriv ath0 get_acsmaxwell` `ath0 get_acsmindwell:100`
acsreprt	Iwpriv ath*N* acsreport *value_in_ms*	Y	Y	Enable (1) or disable (0) channel loading. `#iwpriv ath0 acsreport 100`
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`
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 get_ch_cntwn_dur` `get_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`

1.3.29 802.11k parameters

Table 1-35 802.11k 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 Aggregate size scaling parameters

Table 1-36 Aggregate Size Parameters

Parameter	Format	DA	OL	Description
acparams	iwpriv ath*N* 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`

1.3.31 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.

1.3.31.1 802.11k

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.

Table 1-37 Wifitool 802.11k parameters (cont.)

Parameter	Format	Description
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.

1.3.31.2 Channel loading

Table 1-38 Wifitool channel loading parameters

Parameter	Format	Description
acsreport	wifitool ath*N* acs_report	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

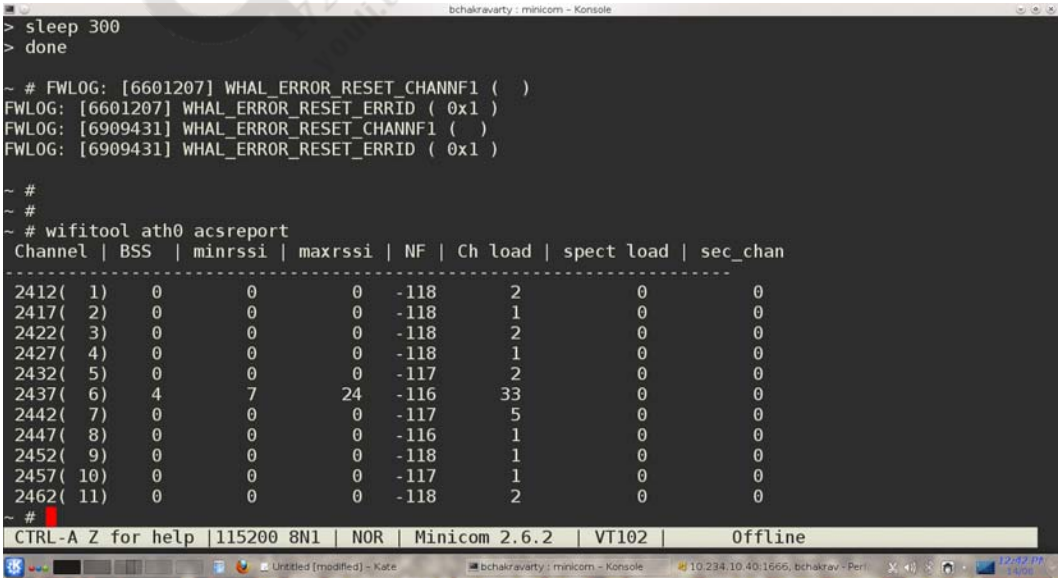

Figure 1-1 Channel loading

1.3.31.3 Block channel list

To block any set of channel from participating in 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,channel 2,channel3 ……channel N*	Set list of channels to be blocked from 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 command will amend previously stored list in driver. If the user want to flush 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 4,5,6

1.3.32 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 Peregrine 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.

1.3.33 2.4 GHz VHT 256-QAM Broadcom interoperability support

Table 1-40 lists the parameters for 2.4 GHz VHT 256-QAM Broadcomm interoperability support.

Table 1-40 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`

1.4 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]
  [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

Argument	Description	
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
bssid *-bssid*	Optional parameter indicating that the MAC address should be cloned from the first VAP for this interface. Not normally specified. Note that -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.	

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

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|keys|caps|wme]
```

1.4.2.1 AP list elements

Table 1-41 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 1-41 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					
(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 1-42 describes the list elements for each STA associated with the indicated VAP. This listing is produced:

```
# wlanconfig ath0 list sta
ADDR              AID CHAN RATE RSSI IDLE  TXSEQ RXSEQ  CAPS ACAPS ERP   STATE  HTCAPS
00:03:7f:08:62:23 1   36   6M   59   135   13    12128  E    0     33    Q      WME
```

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 1-42.

Table 1-42 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		

Table 1-42 STA list elements (cont.)

Element	Description					
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.					
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 1-43 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   Channel 120: 5600 *~ Mhz 11na C CL V VL V80-122
Channel 40:  5200    Mhz 11na C CL V VL V80-42   Channel 124: 5620 *~ Mhz 11na C CU V VU V80-122
Channel 44:  5220    Mhz 11na C CU V VU V80-42   Channel 128: 5640 *~ Mhz 11na C CL V VL V80-122
Channel 48:  5240    Mhz 11na C CL V VL V80-42   Channel 132: 5660 *~ Mhz 11na C CU V VU V80-138
Channel 52:  5260 *~ Mhz 11na C CU V VU V80-58   Channel 136: 5680 *~ Mhz 11na C CL V VL V80-138
Channel 56:  5280 *~ Mhz 11na C CL V VL V80-58   Channel 140: 5700 *~ Mhz 11na C CU V VU V80-138
Channel 60:  5300 *~ Mhz 11na C CU V VU V80-58   Channel 144: 5720 *~ Mhz 11na C CL V VL V80-138
Channel 64:  5320 *~ Mhz 11na C CL V VL V80-58   Channel 149: 5745 Mhz    11na C CU V VU V80-155
Channel 100: 5500 *~ Mhz 11na C CU V VU V80-106  Channel 153: 5765 Mhz    11na C CL V VL V80-155
Channel 104: 5520 *~ Mhz 11na C CL V VL V80-106  Channel 157: 5785 Mhz    11na C CU V VU V80-155
```

```
Channel 108: 5540 *~ Mhz 11na C CU V VU V80-106  Channel 161: 5805 Mhz    11na C CL V VL V80-155
Channel 112: 5560 *~ Mhz 11na C CL V VL V80-106  Channel 165: 5825 Mhz    11na C V
Channel 116: 5580 *~ Mhz 11na C CU V VU V80-122
```

Table 1-43 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
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

1.4.2.4 Capabilities list elements

Table 1-44 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 1-44 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
```

1.4.3 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 1-45 as a separate entry.

Table 1-45 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 1-45 Configure NAWDS parameters (cont.)

Parameter	Format	DA	OL	Description
defcaps	wlanconfig ath*N* nawds *defcaps caps*	Y	Y	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_HT20 0x01` `#define NAWDS_REPEATER_CAP_HT2040 0x02` `#define NAWDS_REPEATER_CAP_DS 0x04` 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: 0x0 HT Rate Disabled 0x1 Single stream HT20 Rates 0x2 Single Stream HT40 Rates 0x5 Double Stream HT20 Rates 0x6 Double Stream HT40 Rates 0x9 Triple Stream HT20 Rates 0xa Triple Stream HT40 Rates
del-repeater	wlanconfig ath*N* nawds *del-repeater mac_addr*	Y	Y	Delete a NAWDS AP with the specified MAC address. mac_addr MAC address
list	wlanconfig ath*N* nawds *list*	Y	Y	Display current NAWDS configurations.
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

```
Cfg -a AP_PRIMARY_CH=149
Cfg -a AP_CHMODE=11NAHT40PLUS
Cfg -a AP_SSID=Anish_test
Apup
Iwpriv ath0 wds 1
Wlanconfig ath0 nawds mode 2
Wlanconfig ath0 nawds add-repeater 00:03:7f:xx:xx:xx:xx 0x1
```

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

```
Cfg -a AP_PRIMARY_CH=149
Cfg -a AP_CHMODE=11ACVHT80
Cfg -a AP_SSID=Anish_test
Apup
Iwpriv ath0 wds 1
Wlanconfig ath0 nawds mode 4
Wlanconfig ath0 nawds defcaps 0x6
```

Static bridge and peer node supports VHT rates

```
Cfg -a AP_PRIMARY_CH=149
Cfg -a AP_CHMODE=11ACVHT80
Cfg -a AP_SSID=Anish_test
Apup
Iwpriv ath0 wds 1
Wlanconfig ath0 nawds mode 2
Wlanconfig ath0 nawds add-repeater 00:03:7f:xx:xx:xx:xx 0x88
```

VHT Example Rates: 3x3

```
wlanconfig ath0 nawds add-repeater <mac> 0x88 - 3x3 HT80
wlanconfig ath0 nawds add-repeater <mac> 0x48 - 3x3 HT40
wlanconfig ath0 nawds add-repeater <mac> 0x28 - 3x3 HT20
```

VHT Example Rates: 2x2

```
wlanconfig ath0 nawds add-repeater <mac> 0x84 - 2x2 HT80
wlanconfig ath0 nawds add-repeater <mac> 0x44 - 2x2 HT40
wlanconfig ath0 nawds add-repeater <mac> 0x24 - 2x2 Ht20
```

VHT Example Rates: 1x1

```
wlanconfig ath0 nawds add-repeater <mac> 0x80 - 1x1 HT80
wlanconfig ath0 nawds add-repeater <mac> 0x40 - 1x1 HT40
wlanconfig ath0 nawds add-repeater <mac> 0x20 - 1x1 HT20
```

1.4.5 HMWDS/HMMC commands

Table 1-46 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 specfied.
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.

1.5 Other commands

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

1.5.1 Athssd parameters

Table 1-47 Athssd Parameters

Configuration	Format	DA	OL	Description
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 using cfg -a command or appropriate iwpriv commands. Please refer to iwpriv command reference for further details.

1. Check the current set up using the command, cfg -e.
2. Use the command, cfg -x to set it up in default mode
3. 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
   d. Select the appropriate channel
   e. For FCC testing following extra set up is necessary:
      i. Set rate control to manual mode (cfg -a RATECTL=manual)

ii. Set manual rate to 9 Mbps (cfg -a MANRATE=0x0f0f0f0f)

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

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 1-48 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
shownol	radartool -i wifi [*0*|*1*] shownol *debug_level*	Y	Y	Displays the NOL list. Set dfsdebuglevel to 0x2000 before using command	

1.5.5 Spectraltool parameters

Table 1-49 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)*
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)

Table 1-49 Spectraltool parameters (cont.)

Parameter	Format	DA	OL	Description	
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
				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. As at present, ICM is supported only for the AP135 platform from 10.2 onwards.

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
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 that 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 1-50 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

Table 1-50 ICM command line parameters

Parameter	Format	Description	
-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 general use. It is mentioned here for completeness. It is disabled	
-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
-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 1-51 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 g_acs_bkscanen	ACS/OBSS background scan value

Table 1-51 ACS/DCS/OBSS iwpriv commands

Command	Format	Description
acs_bkscanintvl	iwpriv wifi1 acs_bkscanintvl <*value*>	Set the background scan value default is one minute
get_acsscanintvl	iwpriv wifi1 get_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: 30
g_acs_rssivar	g_acs_rssivar	Display RSSI variance value
acs_chloadvar	iwpriv wifi1 acs_chloadvar <*value*>	Set the channel load variance Iftwo 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
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
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

1.5.7 UCI Wireless configuration (qca-wifi)

The UCI commands listed in this section are applicable only to AP 10.2.3 and later software versions. For details of UCI, visit http://wiki.openwrt.org/doc/uci.

QSDK supports qca-wifi driver natively. The UCI database section to configure 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-section contains configuration parameters such as mode (11n, 11ac...), channel (1, 6 11, 36...).
- **wifi-iface[N]:** a wifi-iface section represents a Wi-Fi VAP. It supports configuration parameters such as SSID, shortgi, and so on. 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.

Table 1-52 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.

Table 1-52 Per-VAP configuration parameters (cont.)

Parameter	Format	Description
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.

1.5.7.1 Example UCI Configurations

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
```

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
```

1.5.7.2 QWRAP configuration (basic)

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
```

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
uci set wireless.@wifi-iface[0].vap_ind=1
```

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
```

1.5.7.3 QWRAP configuration (DBDC)

QWRAP DBDC configuration 1

```
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
```

QWRAP DBDC configuration 2

```
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=ap
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 set wireless.@wifi-iface[0].qwrap_ap=1

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=wrap
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
```

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