Qualcomm Technologies, Inc.

Wireless LAN Access Point (Driver Version 11.0) Command

Reference

80-YA728-5 Rev. YY

May 7, 2021

Confidential – May Contain Trade Secrets
2021-05-13 03:57:44 PDT
pan.yafeng@zte.com.cn

Confidential – Qualcomm Technologies, Inc. and/or its affiliated companies – May Contain Trade Secrets

NO PUBLIC DISCLOSURE PERMITTED: Please report postings of this document on public servers or websites to: DocCtrlAgent@qualcomm.com.

Confidential Distribution: Use or distribution of this item, in whole or in part, is prohibited except as expressly permitted by written agreement(s) and/or terms with Qualcomm Incorporated and/or its subsidiaries.

Not to be used, copied, reproduced, or modified in whole or in part, nor its contents revealed in any manner to others without the express written permission of Qualcomm Technologies, Inc.

All Qualcomm products mentioned herein are products of Qualcomm Technologies, Inc. and/or its subsidiaries.

Qualcomm is a trademark or registered trademark of Qualcomm Incorporated. Other product and brand names may be trademarks or registered trademarks of their respective owners.

This technical data may be subject to U.S. and international export, re-export, or transfer ("export") laws. Diversion contrary to U.S. and international law is strictly prohibited.

Qualcomm Technologies, Inc.
5775 Morehouse Drive
San Diego, CA 92121
U.S.A.

© 2017-2021 Qualcomm Technologies, Inc. and/or its subsidiaries. All rights reserved.

Revision history

Revision	Date	Description
A	June 2017	Initial release for WLAN AP Driver, Version 11.0. For 10.4 releases of AP software, refer to the *Wireless LAN Access Point Driver (Version 10.4) Command Reference Guide* (80-Y8052-1).
B	July 2017	The following sections are added/updated for **QCA_Networking_2017.SPF.7.0 ES release** and **QCA_Networking_2017.SPF.8.0 ED1 release**: ■ Added the following: □ "Set data path tracing" on page 257 □ "Tx and Rx statistics" on page 87 □ "Sample output of txrx_stats command" on page 87 □ "Packet log analysis for IPQ807x devices" on page 286 ■ Updated the following: □ "Applying channel and SSID parameters using cfg80211tool" on page 24 □ Table 3-5, "Beacon configuration parameters," on page 31 □ Table 3-15, "Physical layer parameters," on page 48 □ "Hy-Fi – WMM DSCP prioritization parameters" on page 136 □ "Disable selected legacy rates for an SSID (using iwpriv)" on page 150 □ "Mesh Support parameters" on page 154 □ Table 3-57, "11.0 WLAN driver parameters," on page 168 □ "Applying iwpriv commands using cfg80211tool" on page 178 □ "NAWDS configuration parameters" on page 266 □ "Using wlanconfig commands for cfg80211 mode: examples" on page 273 □ "Using miscellaneous commands for cfg80211 mode: examples" on page 285 □ "Example UCI configuration" on page 305
C	September 2017	The following sections are added/updated for **QCA_Networking_2017.SPF.7.0 ES3 release**: ■ Added the following: □ "Support for control frame Tx and Rx statistics" on page 115 □ "Supported Tx and Rx per-PPDU statistical details" on page 117 □ "Examples of console output for Tx and Rx per-PPDU statistics" on page 119 □ "Support for Rx statistics to contain MU-MIMO and OFDMA parameters" on page 121 □ "Support for Tx statistics to contain MU-MIMO and OFDMA parameters" on page 122 □ Table 3-63, "Extended NSS signaling parameters," on page 193

Revision	Date	Description
D	October 2017	The following sections are added/updated for **QCA_Networking_ 2017.SPF.7.0 ES4 release**: ■ Added the following: □ "Off-channel Tx/Rx without active WLAN" on page 274 □ "Channel change and CSA support for cfg80211" on page 288 ■ Updated the following: □ Table 3-57, "11.0 WLAN driver parameters," on page 168 □ Table 3-55, "Offload statistics parameters," on page 166
E	November 2017	The following sections are added/updated for **QCA_Networking_ 2017.SPF.7.0 ED5 release**: ■ Added the following: □ "Support for beacon count and beacon outage statistics" on page 290 □ "exttool parameters" on page 290 ■ Updated the following: □ "Applying channel and SSID parameters using cfg80211tool" on page 24 □ Table 3-52, "ACS/DCS/OBSS iwpriv parameters," on page 161 □ Table 3-57, "11.0 WLAN driver parameters," on page 168 □ Table 6-5, "ICM command line parameters," on page 282 □ "Channel change and CSA support for cfg80211" on page 288
F	December 2017	The following sections are added/updated for **QCA_Networking_ 2017.SPF.7.0 ED6 release**: ■ Updated the following: □ Table 3-57, "11.0 WLAN driver parameters," on page 168
G	January 2018	The following sections are added/updated for **QCA_Networking_ 2017.SPF.8.0 ED7 release:** ■ Added the following: □ "Enable 802.11v DMS" on page 379 ■ Updated the following: □ Table 3-38, "Target recovery parameters," on page 139 □ Table 3-57, "11.0 WLAN driver parameters," on page 168

Revision	Date	Description
H	March 5, 2018	The following sections are added/updated for **QCA_Networking_ 2017.SPF.8.0 FC release:** ■ Added the following: □ Table 3-69, “Neighbor report parameters,” on page 238 □ Table 3-70, “Target core dump parameter,” on page 238 □ Table 3-71, “CSL parameter,” on page 239 □ Table 3-72, “Tx data packet capture parameters,” on page 239 □ Table 3-73, “QCN broadcast probe response parameters,” on page 240 □ Table 3-74, “CE logging parameters,” on page 241 □ Table 3-75, “FILS parameter,” on page 243 □ Table 3-75, “Offload statistics parameters,” on page 244 □ Table 3-76, “Softblocking parameters,” on page 245 □ Table 3-77, “Protected management frames (PMF) parameters,” on page 246 □ “ACS channel score parameter” on page 258 □ “Softblocking wifitool parameters” on page 259 □ “Spectral debug enhancements” on page 280 □ “Manual VAP creation and deletion in cfg80211” on page 292 □ “Monitor mode setup” on page 293 □ “Configure best uplink hysteresis for Wi-Fi SON” on page 380 □ “Configure FILS capability using UCI commands” on page 380 □ “UCI commands to configure FILS Discovery Frames” on page 380 □ “UCI command to configure SA query timeout for PMF” on page 381 □ “Sample AP UCI configuration for 11ax” on page 382 ■ Updated the following: □ Table 3-1, “802.11ac interface aggregation parameters,” on page 25 □ Table 3-57, “11.0 WLAN driver parameters,” on page 168 □ Table 3-66, “Miscellaneous protocol layer parameters,” on page 211 □ Table 5-7, “Configure/show ATF parameters,” on page 270
J	March 12, 2018	Updated the following: ■ Table 3-1, “802.11ac interface aggregation parameters,” on page 25 ■ Table 3-2, “802.11na and 802.11ac interfaces specific statistics,” on page 28 ■ Table 3-5, “Beacon configuration parameters,” on page 31 ■ Table 3-15, “Physical layer parameters,” on page 48 ■ Table 3-17, “Radio-related parameters,” on page 53 ■ Table 3-57, “11.0 WLAN driver parameters,” on page 168 ■ Table 3-59, “Additional protocol and radio layer commands,” on page 178 ■ Table 3-66, “Miscellaneous protocol layer parameters,” on page 211 ■ Table 4-4, “Block Acknowledge,” on page 252

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Revision	Date	Description
K	April 2018	The following sections are added/updated for **QCA_Networking_ 2017.SPF.8.0 FCU2 release:** Updated the following: ■ Table 3-57, “11.0 WLAN driver parameters,” on page 168 ■ “exttool parameters” on page 290 ■ Table 6-4, “Spectral tool parameters,” on page 279
L	May 2018	The following sections are added/updated for **QCA_Networking_ 2017.SPF.8.0 CS release:** Added the following: ■ Table 3-79, “Tx chain-mask parameter,” on page 286 ■ Table 3-80, “ICM parameters,” on page 286 Updated the following: ■ Table 3-59, “11.0 WLAN driver parameters,” on page 196 ■ Table 3-74, “QCN broadcast probe response parameters,” on page 279 ■ “Enable or disable cfg80211 for UCI configuration” on page 420
M	June 2018	Updated the following: ■ Table 3-68, “Miscellaneous radio layer parameters in offload mode only,” on page 264 ■ Table 6-5, “Spectral tool parameters,” on page 337 ■ Table A-1, “Country code definitions,” on page 488
N	July 2018	The following sections are added/updated for **QCA_Networking_ 2017.SPF.8.0 CSU1 release:** Added the following: ■ “UCI commands to configure SAE” on page 453 ■ “UCI commands for OWE” on page 455 ■ “UCI commands for Suite B in WPA3” on page 457 Updated the following: ■ Table 3-59, “11.0 WLAN driver parameters,” on page 196 ■ Section 6.1, Athssd parameters ■ Section 6.5, Spectraltool parameters

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Revision	Date	Description
P	August 2018	The following sections are added/updated for **QCA_Networking_ 2019.SPF.9.0 ES release:** ■ Added the following: □ "TR-181 data elements for carrier networks" on page 286 □ "Wi-Fi medium utilization statistics parameters" on page 287 □ "Parameter APIs for carrier networks" on page 287 □ "CBS for VoW parameters" on page 288 □ "TDMA parameters" on page 290 □ "Retrieve Radio Resource Management utilities for parameter APIs in carrier networks" on page 305 □ "CBS for VoW parameters" on page 306 □ "Debug commands for AC-based ATF" on page 306 □ "Configure allowed channels and ACS-allowed channels" on page 307 □ "Extended VoW statistics" on page 309 □ "wifitool commands for distributed VoW" on page 311 □ "RSSI scan of NACs parameters" on page 331 □ "Configure the listing HW capabilities for parameter APIs in carrier networks" on page 332 □ "Display the hyd and lbd versions" on page 458 □ "UCI commands to configure AP steering for legacy clients" on page 458 □ "Best uplink algorithm for PLC interface metrics in a daisy chain Wi-Fi SON and PLC network" on page 460 □ "UCI commands for ETSI compliance for COT and CAT on QCA9880 and QCA9980 chipsets" on page 463 □ "UCI commands for Green AP power-saving in connected-idle status " on page 464 □ "UCI command to configure idle client detection for Tx path" on page 465 □ "UCI parameters for multi-AP" on page 465 ■ Updated the following: □ "Aggregation parameters" on page 30 □ "Green AP parameters" on page 50 □ "iQue parameters" on page 52 □ "Physical layer parameters" on page 54 □ "OCE parameters" on page 185 □ "QCN broadcast probe response parameters" on page 279 □ "Copy Engine logging parameters" on page 280 □ "ATF configuration commands" on page 326 □ "Thermal mitigation" on page 341 □ "Single AP Band Steering Daemon (lbd) parameters" on page 377 □ "Multi-AP Coordinated Steering and Adaptive Path Selection parameters" on page 392 □ Table A-1, "Country code definitions," on page 488

Revision	Date	Description
R	September 2018	The following sections are added/updated for **QCA_Networking_ 2017.SPF.8.0 CSU2 release:** Updated the following: ■ Section 6.1, Athssd parameters ■ Section 6.5, Spectraltool parameters
T	October 2018	The following sections are added/updated for **QCA_Networking_ 2019.SPF.9.0 REL3 (v1):** Added the following: ■ "Display RSSI for IPQ807x platforms" on page 95 ■ "Strict passive scan parameters" on page 290 ■ "wifitool commands to configure AP TWT" on page 312 ■ "UCI settings to preserve ATF configurations across a system reboot" on page 473 ■ "UCI parameters to configure AP TWT" on page 479 ■ "UCI commands to configure the legacy upgrade-steering mechanism for clients" on page 481 ■ "UCI commands for EAPOL minimum rate WMI resource configuration" on page 481 Updated the following: ■ "Beacon configuration parameters" on page 36 ■ "iQue parameters" on page 52 ■ "Dynamic Channel Selection for Interference Mitigation (DCS-IM) Parameters" on page 49 ■ "Radartool parameters" on page 335 ■ Table 8-4, "Multi-AP Coordinated Steering and Adaptive Path Selection Parameters," on page 394
U	December 10, 2018	The following sections are added/updated for **QCA_Networking_ 2019.SPF.9.0 REL4 (v1):** ■ Added the following: □ Section 2.2, Enable and disable Wi-Fi interfaces ■ Updated the following: □ Table 3-59, "11.0 WLAN driver parameters," on page 196
V	December 18, 2018	The following sections are added/updated for **QCA_Networking_ 2019.SPF.10.0 ES:** ■ Added the following: □ "Configure smart mesh enhancements based on monitor-direct feature for IPQ807xA and IPQ817x" on page 482 ■ Updated the following: □ Table 3-59, "11.0 WLAN driver parameters," on page 196

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Revision	Date	Description
W	January 21, 2019	The following sections are added/updated for **QCA_Networking_ 2019.SPF.9.0 CS (v1):** ■ Updated the following: □ Table 3-14, “iQue parameters,” on page 52 □ Table 3-59, “11.0 WLAN driver parameters,” on page 196 □ Table 3-60, “Additional protocol and radio layer commands,” on page 215, Table 3-65, “Uncategorized protocol layer commands,” on page 228, Table 3-67, “Miscellaneous protocol layer parameters,” on page 248, and Table 3-68, “Miscellaneous radio layer parameters in offload mode only,” on page 264 for removal of unsupported commands □ “UCI commands to configure SAE” on page 453 □ “UCI commands for OWE” on page 455
Y	February 12, 2019	The following sections are added/updated for **QCA_Networking_ 2019.SPF.10.0 ES2:** ■ Added the following: □ Section 3.92, Periodic channel frequency response dump capture parameters □ Section 4.23, Start or stop channel frequency response dump capture per peer ■ Updated the following: □ Table 3-59, “11.0 WLAN driver parameters,” on page 196 □ Table 3-22, “STA parameters,” on page 80, Table 3-67, “Miscellaneous protocol layer parameters,” on page 248, and Table 3-68, “Miscellaneous radio layer parameters in offload mode only,” on page 264 for removal of unsupported commands □ Table 6-4, “Zero CAC DFS Radartool parameters,” on page 335
YA	March 2019	The following sections are added/updated for **QCA_Networking_ 2019.SPF.10.0 FC:** ■ Added the following: □ “UCI commands to configure DPP” on page 482 ■ Updated the following: □ Table 3-59, “11.0 WLAN driver parameters,” on page 196
YB	May 2019	The following sections are added/updated for **QCA_Networking_ 2019.SPF.10.0 CS Preview:** ■ Added the following: □ “Configure off-channel Rx using wifitool command” on page 314 □ “UCI commands to configure Rx flow identification” on page 484 □ “UCI configuration of IPv4 over IPv6 (464) XLAT and CLAT” on page 485 ■ Updated the following: □ Table 3-14, “iQue parameters,” on page 52 for removal of multicast enhancement commands that are deprecated □ Table 3-59, “11.0 WLAN driver parameters,” on page 196 □ “Single AP Band Steering Daemon (lbd) parameters” on page 377 □ “Country Code Definitions” on page 488

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Revision	Date	Description
YC	June 2019	The following sections are added/updated for **QCA_Networking_ 2019.SPF.10.0 CS:** ■ Updated the following: □ Table 3-59, “11.0 WLAN driver parameters” on page 210 □ “Start or stop channel frequency response dump capture per peer for IPQ401x, IPQ807xA, and IPQ60xx” on page 367
YD	July 2019	Updated the document to indicate removal of ACFG support in WLAN driver, version 11.0
YE	July 2019	Updated the document to indicate the applicability of features for IPQ60xx platforms for the **IPQ6018.ILQ.11.0 ES1 r**elease.
YF	August 21, 2019	The following sections are added/updated for **QCA_Networking_ 2019.SPF.11.0 ED:** ■ Added the following: □ “Configure the live dump collection utility” on page 562 ■ Updated the following: □ Table 3-59, “11.0 WLAN driver parameters” on page 210 □ “Country code definitions” on page 578
YG	October 2019	The following sections are added/updated for **QCA_Networking_ 2019.SPF.11.0 ED1** and to indicate the applicability of features for IPQ60xx platforms for the **IPQ6018.ILQ.11.0 FC r**elease**:** ■ Added the following: □ “Configure Rx flow identification based on flow-type using user space commands” on page 368 □ “Global and target-specific INI files” on page 400 □ “Globally disable same SSID feature using UCI settings” on page 563 □ “Collection of Wi-Fi SON statistics using nl80211 vendor command” on page 563 ■ Updated the following: □ Table 3-59, “11.0 WLAN driver parameters” on page 210
YH	November 2019	The following sections are added/updated for **QCA_Networking_ 2019.SPF.11.0 CS r**elease**:** ■ Added the following: □ Section 5.13, Create DISA interface ■ Updated the following: □ Table 8-4, “Multi-AP Coordinated Steering and Adaptive Path Selection Parameters” on page 450
YJ	December 2019	The following sections are added/updated during **QCA_Networking_ 2019.SPF.11.0 CSU1 r**elease**:** ■ Updated the following: □ Table 8-4, “Multi-AP Coordinated Steering and Adaptive Path Selection Parameters” on page 450

Revision	Date	Description
YK	February 2020	Updated the document to indicate the applicability of features for QCN90xx platforms for the **IPQ8074.ILQ.11.1 ES** release. Also, the following sections are added/updated during **QCA_ Networking_2020.SPF.11.1 ED2 r**elease**:** ■ Added the following: □ Section 2.3, Sysfs enhancements to support 6 GHz □ Section 3.93, Changes to cfg80211tool commands for 6 GHz support □ Section 4.24, Changes to wifitool commands for 6 GHz □ Section 5.14, wlanconfig changes for 6 GHz □ Section 5.15, Configure repetitive channel capture using wlanconfig commands for IPQ60xx □ Section 6.12.1, Changes to existing exttool commands for 6 GHz □ Section 8.49.7, UCI parameters for legacy AP steering, backhaul steering, and channel selection in MAP □ Section 8.61, UCI script changes for 6 GHz support □ Section 8.62, Configure ACS optimization using target provided channel grading metrics □ Section 8.63, Configure DMA buffer debug for spectral scan □ Section 8.64, Support interoperation with dual-band and tri-band mesh nodes □ Section 8.65, Configure MAP 1905-only operation mode □ Section 8.66, UCI configuration for Wi-Fi SON static and dynamic memory computation ■ Updated the following: □ “Enable and disable Wi-Fi interfaces” on page 33 □ Table 3-59, “11.0 WLAN driver parameters” on page 210 □ “Spectral debug enhancements” on page 386
YL	April 2020	Updated the 6 GHz frequency support content to indicate applicability for QCN5064 platforms, starting with IPQ8074.11.1 ES release. The following sections are added/updated during **QCA_Networking_ 2020.SPF.11.1 CS r**elease and **IPQ6018.ILQ.11.1 ES** release**:** ■ Added the following: □ Section 6.16, athdiag over QMI for QCN90xx or QCN602x platforms □ Section 8.67, UCI configuration of Easy Mesh R2 commercialization algorithms □ Section 8.68, Configure single AP and multi-AP mode in data elements □ Section 8.69, Configure interoperation with dual-band and tri-band mesh nodes in Wi-Fi SON Updated the following: □ Table 4-6, “wifitool MBO parameters,” on page 412

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Revision	Date	Description
YM	May 2020	The following sections are added/updated during **QCA_Networking_ 2020.SPF.11.2 ED r**elease and **QCN90xx FC** release**:** ■ Added the following: □ Section 3.24, Enhanced NSS offloading statistics □ Section 4.25, Configure standalone sounding sequence □ Section 4.26, Disable probing in VHT20, MCS 9 rate in NSS 1 and 2 after a certain number of failures □ Section 8.70, Configure RCPI support for Wi-Fi SON networks □ Section 8.71, Configure advanced CAC implementation for MAP r1 □ Section 8.72, UCI support to configure individual VAPs independently □ Section 8.73, Configure zero BSS credential MAP agent onboarding support ■ Updated the following: □ Section 3.61, Offload statistics parameters □ Section 3.63, 11.0 WLAN driver parameters □ Section 6.1, Athssd parameters □ Section 6.5, Spectraltool parameters □ Section 6.6, Intelligent channel manager parameters
YN	July 2020	Updated the document to indicate the applicability of features for IPQ50xx platforms for the **IPQ5018.ILQ.11.3 ES** release. ■ Updated the following: □ Section 3.23, Tx and Rx statistics
YP	August 2020	The following sections are added/updated during **QCA_Networking_ 2020.SPF.11.2 CS r**elease and **QCN90xx CS** release**:** ■ Added the following: □ Section 5.16, Configure access control of clients based on a list of stations ■ Updated the following: □ Section 3.63, 11.0 WLAN driver parameters □ Section 4.11, MBO parameters □ Section 5.11, Start or stop channel frequency response dump capture per peer for IPQ401x, IPQ807xA, IPQ60xx, and QCN90xx /QCN602x (to include support for QCN90xx/QCN602x) □ Section 5.15, Configure repetitive channel capture using wlanconfig commands for IPQ60xx and QCN90xx/QCN602x (to include support for QCN90xx/QCN602x) □ Section 8.73, Configure zero BSS credential MAP agent onboarding support

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Revision	Date	Description
YR	September 2020	The following sections are added/updated during **QCA_Networking_2020.SPF.11.3 ED r**elease and **IPQ5018.ILQ.11.3 ES2** release**:** ■ Added the following: □ Section 3.93, Extending wifistats command for displaying control path statistics □ Section 5.6.1, ATF statistics commands □ Section 5.15.1, Additional CFR capture capabilities for QCN90xx/QCN602x □ Section 8.74, Enhancements to NSS offloading statistics □ Section 8.75, Overriding the default driver behavior for 6 GHz-only AP □ Section 8.76, Configure EMA AP using .ini and UCI settings □ Section 8.77, 6 GHz security compliance with 802.11 specification ■ Updated the following: □ Section 2.2, Enable and disable Wi-Fi interfaces □ Section 3.4, Beacon configuration parameters □ Section 3.8, Green AP parameters □ Section 3.39, ATF parameters □ Section 3.41, Dynamic Frequency Selection (DFS) parameters □ Section 3.59, Raw mode simulation parameters □ Section 3.63, 11.0 WLAN driver parameters □ Section 6.6, Intelligent channel manager parameters □ Section 6.10.1, Method 1: Using hostapd_cli □ Section 8.49.4, RE placement and auto-configuration daemon (repacd) parameters for multi-AP □ Section 8.49.5, Hy-Fi (hyd) parameters for multi-AP
YT	November 2020	Updated the document to indicate the applicability of features for QCN61xx platforms for the **IPQ5018.ILQ.11.4 ES** release. ■ Updated the following: □ Section 3.63, 11.0 WLAN driver parameters
YU	December 2020	The following sections are added/updated during **QCA_Networking_2020.SPF.11.3 CS r**elease and **QCN90xx/QCN60xx CS2** release**:** ■ Added the following: □ Section 4.27, Command to configure TPE □ Section 8.78, Configure WDS extended feature ■ Updated the following: □ Section 3.63, 11.0 WLAN driver parameters (for dcs_wideband_policy and set_triplets_bw parameters) □ “Country Code Definitions” on page 488 for addition of Curacao country code

Revision	Date	Description
YV	February 2021	The following sections are added/updated during **QCA_Networking_ 2021.SPF.11.4 ED r**elease**:** ■ Added the following: □ Section 4.28, Command for peer latency configuration □ Section 5.17, Wireless network management commands □ Section 5.18, Smart mesh wlanconfig commands □ Section 5.19, Support for RTT responder-only without LOWI tool □ Section 8.79, Configure Mesh 2.0 architecture □ Section 8.80, Implementation of mixed backhaul encryption across radios □ Section 8.81, Configuration changes for credential cloning over multiple SSIDs □ Section 8.82, Configure DPP R2 using UCI and ACFG commands □ Section 8.83, Configure service prioritization using UCI parameters ■ Updated the following: □ Section 3.70, Miscellaneous protocol layer parameters □ Section 3.63, 11.0 WLAN driver parameters □ Section 3.92, Changes to cfg80211tool commands for 6 GHz support □ Section 6.4, Radartool parameters □ Section 8.12, Range Extender Placement and Auto-configuration Daemon □ Section 8.49.4, RE placement and auto-configuration daemon (repacd) parameters for multi-AP □ Section 8.49.5, Hy-Fi (hyd) parameters for multi-AP □ “Country Code Definitions” on page 488
YW	March 2021	The following sections are added/updated during **QCA_Networking_ 2021.SPF.11.4 ED2 r**elease**:** ■ Added the following: □ Section 8.84, Configure dynamic role selection of CAP/RE (Controller/Agent) using UCI parameters □ Section 8.85, Configure optional IEEE 802.11 IE support for non-Tx VAP in MBSSID/EMA using .ini and UCI parameters ■ Updated the following: □ Section 3.63, 11.0 WLAN driver parameters (for removal of VAP-level and per-AC-level parameters for the spatial reuse feature and for addition of parameters for EMA enable, EMA for non Tx VAPs, user RNR frame, and upload pktlog) □ Section 8.40, UCI commands to configure SAE (for addition of WPS support for SAE in Wi-Fi SON) □ Section 8.49.4, RE placement and auto-configuration daemon (repacd) parameters for multi-AP (addition of EasyMesh R3 support for DPP parameters)

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Revision	Date	Description
YY	May 2021	The following sections are added/updated during **QCA_Networking_2021.SPF.11.4 CS r**elease**:** ■ Added the following: □ Section 8.84, Configure dynamic role selection of CAP/RE (Controller/Agent) using UCI parameters ■ Updated the following: □ Section 3.8, Green AP parameters (for addition of SMPS command for Green AP) □ Section 3.7, Dynamic Channel Selection for Interference Mitigation (DCS-IM) Parameters (for updates to the dcs_enable parameter) □ Section 3.63, 11.0 WLAN driver parameters (for addition of g_mbssid_ntx_max_pfl_size, vap_resource_pfl_config, g_vap_resource_pfl, dcs_random_chan_en, dcs_csa_tbtt, parameters and for updates to the dcs_sim parameter) □ Section 6.12, exttool parameters (for addition of list_chan_info, ist_chan_state, and force options)

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

Contents

1 WLAN AP driver CLI tools 33
1.1 Wireless tools 33
1.2 cfg80211 tool (iwpriv equivalent) 34

2 iwconfig commands 36
2.1 Applying channel and SSID parameters using cfg80211tool 39
2.2 Enable and disable Wi-Fi interfaces 40
2.3 Sysfs enhancements to support 6 GHz 42

3 iwpriv and cfg80211tool commands 43
3.1 Aggregation parameters 43
3.2 ANI parameters 47
3.3 Association/ACL parameters 48
3.4 Beacon configuration parameters 50
3.5 Channel width parameters 57
3.6 Debug parameters 60
3.7 Dynamic Channel Selection for Interference Mitigation (DCS-IM) Parameters 63
3.8 Green AP parameters 66
3.9 Hotspot 2.0 parameters 67
3.10 HT20/HT40 coexistence parameters 67
3.11 iQue parameters 68
3.12 Physical layer parameters 71
3.13 Protection mechanism parameters 76
3.14 Radio-related parameters 77
3.15 Radio resource management (802.11k) parameters 87
3.16 Radiotap monitor header parameters 92
3.17 Regulatory parameters 92
3.18 Security parameters 93
3.19 Station (STA) parameters 98
3.20 Turbo parameters 99
3.21 Tx beamforming parameters 100
3.21.1 TxBF configuration 101
3.21.2 TxBF statistics 101
3.22 Firmware statistics on legacy chipsets 102

Qualcomm
Confidential – May Contain Trade Secrets
2021-05-13 03:57:44 PDT
pan.yufeng@zte.com.cn

3.22.1 Target physical device stats 102
3.22.2 Rx reorder statistics 106
3.22.3 Rx rate info statistic 107
3.22.4 Tx rate info statistics 108
3.22.5 Copy engine and host statistics 108
3.22.6 Host multi-task enhance statistics 110
3.22.7 Tx beamforming data info statistics 110
3.22.8 Tx beamforming sounding info statistics 110
3.22.9 Packet error statistics 110
3.22.10 Tx self gen statistics 111
3.22.11 Tx multi-user info statistics 112
3.22.12 SIFS response statistics 112
3.22.13 Verify the resets due to different scenarios 113
3.22.14 Clear firmware statistics 114
3.22.15 Display RSSI for IPQ807x/IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx/QCN90xx or QCN602x + IPQ807xA/IPQ60xx/IPQ50xx platforms 114
3.23 Tx and Rx statistics 115
3.23.1 Sample output of txrx_stats command 116
3.23.2 Support for control frame Tx and Rx statistics 185
3.23.3 Supported Tx and Rx per-PPDU statistical details 187
3.23.4 Display statistics and packet logs for LTE-U 197
3.23.5 Enhanced client statistics for acknowledgment failures, modulation rates, authentication details 204
3.23.6 Extended VoW statistics 211
3.24 Enhanced NSS offloading statistics 213
3.24.1 athstats -i wifiX: This is per-radio level statistic 213
3.24.2 apstats -v -i athx: This is per-VAP level statistic 216
3.24.3 wlanconfig ath0 list: This is per client-level statistic 217
3.24.4 iwconfig ath0: This is per-VAP level statistic 217
3.24.5 apstats -s -m <mac_addr>: This is per-client level statistic 217
3.24.6 iwpriv ath0 fc_peer_stats <mac_addr>: This is per-client DP level statistic. . . . 219
3.25 Unassociated power consumption improvement parameters 221
3.26 Smart antenna parameters 221
3.27 WDS parameters 224
3.28 WMM parameters 225
3.29 256QAM rate support parameters 228
3.30 Hy-Fi – WMM DSCP prioritization parameters 229
3.31 Channel loading/Channel hopping parameters 230
3.32 802.11k parameters 232
3.33 Block channel list parameters 232
3.34 Aggregate size scaling parameters 233
3.35 Target recovery parameters 233

3.36 Uncategorized protocol layer parameters 234
3.37 2.4 GHz VHT 256-QAM Broadcom interoperability parameters 234
3.38 QWRAP debug parameters 235
3.39 ATF parameters 235
3.40 Wake on wireless – AP assist parameters 239
3.41 Dynamic Frequency Selection (DFS) parameters 239
3.42 NSS offload parameters 243
3.43 Disable selected MCS for a given SSID 245
3.44 Disable selected legacy rates for an SSID 246
3.45 Revised signaling for 160/80+80 MHz parameters 248
3.46 AP diagnostics for carrier parameters 248
3.47 Reject clients with low SNR parameters 249
3.48 VLAN tagging parameters 250
3.49 Mesh support parameters 250
3.50 DBDC repeater support parameters 252
3.51 TR-069 parameters 254
3.52 Rx monitor filter parameters 255
3.53 Bluetooth coexistence parameters 255
3.54 Interface Manager parameters 256
3.55 Override MU-MIMO capability parameter 257
3.56 OCE parameters 258
3.57 ACS/DCS/OBSS parameters 262
3.58 Dynamic encapsulation and decapsulation configuration 265
3.59 Raw mode simulation parameters 266
3.60 RDK-B HAL API parameters 267
3.61 Offload statistics parameters 269
3.62 Tx Ack timeout parameters 270
3.63 11.0 WLAN driver parameters 271
3.64 Applying iwpriv commands using cfg80211tool 315
3.65 Additional protocol and radio layer parameters 316
3.66 FTM debug parameters 327
3.67 Extended NSS signaling parameters 328
3.68 Uncategorized protocol and radio layer parameters 328
3.69 Miscellaneous radio layer parameters 339
3.70 Miscellaneous protocol layer parameters 348
3.71 Miscellaneous radio layer parameters in offload mode only 364
3.72 MBO parameters 374
3.72.1 MBO Information Element (IE) 375
3.72.2 MBO Attributes IE 375
3.72.3 MBO Capability Indication Attribute included in beacon, probe response, and reassociation frames 375
3.72.4 MBO Association Disallowed Attributes included in beacon, probe response, and reassociation frames 376

Qualcomm Confidential – May Contain Trade Secrets 2021-10-05 12:03:57.44 PDT jian.yafengdata.com.cn

3.73 Neighbor report parameters . . . 379
3.74 Target core dump parameter . . . 380
3.75 Connection state logging parameter . . . 380
3.76 Tx data packet capture parameters . . . 381
3.77 QCN broadcast probe response parameters . . . 382
3.78 Copy Engine logging parameters . . . 384
3.79 FILS parameter . . . 385
3.80 Target recovery parameter . . . 387
3.81 Soft-blocking parameters . . . 388
3.82 Protected management frames (PMF) parameters . . . 389
3.83 Tx chain-mask parameter . . . 390
3.84 ICM parameters . . . 390
3.85 TR-181 data elements for carrier networks . . . 391
3.86 Wi-Fi medium utilization statistics parameters . . . 391
3.87 Parameter APIs for carrier networks . . . 392
3.88 CBS for VoW parameters . . . 393
3.89 TDMA parameters . . . 395
3.90 Strict passive scan parameters . . . 395
3.91 Periodic channel frequency response dump capture parameters . . . 397
3.92 Changes to cfg80211tool commands for 6 GHz support . . . 397
3.92.1 Guidelines for 6 GHz out-of-band functionality . . . 398
3.92.2 CLI enhancements for in-band and OOB features . . . 399
3.93 Extending wifistats command for displaying control path statistics . . . 400

4 wifitool utility commands . . . 401
4.1 802.11k . . . 401
4.2 Channel loading . . . 402
4.3 Block channel list . . . 404
4.4 FIPS validation . . . 405
4.5 Chainmask per client . . . 406
4.6 Set antenna switch . . . 406
4.7 Set user control table . . . 407
4.8 Block acknowledge . . . 407
4.9 TR-69 parameters . . . 409
4.10 Preserve regulatory settings after HLOS replacement . . . 410
4.11 MBO parameters . . . 410
4.12 Using wifitool commands for cfg80211 mode: examples . . . 414
4.13 Set data path tracing . . . 415
4.14 ACS channel score parameter . . . 416
4.15 Softblocking wifitool parameters . . . 418
4.16 Retrieve Radio Resource Management utilities for parameter APIs in carrier networks 418
4.17 CBS for VoW parameters . . . 419
4.18 Debug commands for AC-based ATF . . . 420

Qualcomm Confidential – Qualcomm Contain Trade Secrets 2021-09-13 09:57:44 PDT pan.yefengchen@qualcomm.com

4.19 Configure allowed channels and ACS-allowed channels 421
4.19.1 Guidelines 421
4.19.2 Limitations 422
4.20 Extended VoW statistics 423
4.20.1 FW changes 423
4.20.2 FW debug changes 423
4.20.3 Wi-Fi firmware 424
4.21 wifitool commands for distributed VoW 425
4.21.1 Design enhancements 425
4.22 wifitool commands to configure AP TWT 426
4.23 Configure off-channel Rx using wifitool command 427
4.24 Changes to wifitool commands for 6 GHz 428
4.25 Configure standalone sounding sequence 430
4.26 Disable probing in VHT20, MCS 9 rate in NSS 1 and 2 after a certain number of failures . 432
4.27 Command to configure TPE 432
4.28 Command for peer latency configuration 433

5 wlanconfig utility commands 434
5.1 Create a VAP 434
5.2 List VAP parameters 435
5.2.1 AP list elements 435
5.2.2 STA list Elements 436
5.2.3 Channel list elements 439
5.2.4 Capabilities list elements 440
5.2.5 WME list elements 440
5.2.6 Keys list elements 441
5.3 Delete an interface 441
5.4 NAWDS configuration parameters 441
5.4.1 Configuration examples 443
5.5 HMWDS/HMMC commands 444
5.6 ATF configuration commands 445
5.6.1 ATF statistics commands 448
5.7 Using wlanconfig commands for cfg80211 mode: examples 449
5.7.1 Create and delete VAP using cfg80211tool 449
5.8 Off-channel Tx/Rx without active WLAN 450
5.9 RSSI scan of NACs parameters 451
5.10 Configure the listing HW capabilities for parameter APIs in carrier networks 452
5.11 Start or stop channel frequency response dump capture per peer for IPQ401x, IPQ807xA, IPQ60xx, and QCN90xx /QCN602x 453
5.12 Configure Rx flow identification based on flow-type using user space commands 454
5.13 Create DISA interface 455
5.14 wlanconfig changes for 6 GHz 456

5.15 Configure repetitive channel capture using wlanconfig commands for IPQ60xx and QCN90xx/QCN602x 458
5.15.1 Additional CFR capture capabilities for QCN90xx/QCN602x 465
5.16 Configure access control of clients based on a list of stations 466
5.16.1 Configure access control of clients based on a list 467
5.17 Wireless network management commands 467
5.18 Smart mesh wlanconfig commands 468
5.19 Support for RTT responder-only without LOWI tool 469

6 Miscellaneous commands 471
6.1 Athssd parameters 471
6.2 DFS parameters 473
6.3 NAT parameters 473
6.4 Radartool parameters 474
6.5 Spectraltool parameters 477
6.5.1 Spectraltool options 477
6.5.2 spectraltool commands 479
6.5.3 Spectral debug enhancements 480
6.6 Intelligent channel manager parameters 481
6.6.1 Enabling selection debug information 482
6.6.2 ICM command line parameters (debugging only) 483
6.7 Thermal mitigation 484
6.7.1 Thermal tool 484
6.7.2 SYSFS entries 485
6.8 Using miscellaneous commands for cfg80211 mode: examples 486
6.9 Packet log analysis for IPQ807x/IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx/QCN90xx or QCN602x + IPQ807xA/IPQ60xx/IPQ50xx devices 487
6.9.1 Prerequisites 488
6.9.2 pktlog-lite 488
6.9.3 pktlog-full 489
6.10 Channel change and CSA support for cfg80211 489
6.10.1 Method 1: Using hostapd_cli 490
6.10.2 Method 2: Using vendor command (for channel change only and without CSA) 491
6.11 Support for beacon count and beacon outage statistics 492
6.12 exttool parameters 492
6.12.1 Changes to existing exttool commands for 6 GHz 494
6.13 Manual VAP creation and deletion in cfg80211 495
6.13.1 Create interface 495
6.13.2 Delete interface 495
6.14 Monitor mode setup 496
6.15 Global and target-specific INI files 497
6.15.1 Sample INI file 498
6.16 athdiag over QMI for QCN90xx or QCN602x platforms 500

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:27:24 PDT 8 yxfeng@zte.com.cn

6.16.1 Word read 500
6.16.2 Word write 500
6.16.3 Block read 501
6.16.4 Block read and save into file 501
6.16.5 Examples of athdiag over QMI commands 501

7 LOWI commands 503
7.1 Ranging measurements 504
7.1.1 Command line options 504
7.1.2 User input 504
7.1.3 Output files 505
7.1.4 Example and test result 506
7.1.5 Single sided RTT 506
7.1.6 Double sided RTT 507
7.2 LCR configuration 508
7.2.1 Command line options 508
7.2.2 Output files 509
7.2.3 Example and test result 509
7.3 LCI configuration 510
7.3.1 Command line options 510
7.3.2 Output files 511
7.3.3 Example and test result 511
7.4 Where are you (LCI request) 512
7.4.1 Command line options 512
7.4.2 Output files 512
7.4.3 Example and test results 512
7.5 Fine Timing Measurement Range Request (FTMRR) 513
7.5.1 Command line options 513
7.5.2 Output files 514
7.5.3 Example and test result 514
7.5.4 References 515

8 UCI commands 516
8.1 Per-VAP configuration parameters 517
8.2 Example UCI configuration for WLAN AP modes 518
8.3 QWRAP configuration (basic) 521
8.3.1 QWRAP per radio configuration 521
8.3.2 QWRAP 'wrap' interface 521
8.3.3 QWRAP 'sta' interface 522
8.3.4 Automatic addition of proxystas 522
8.3.5 wrapd_vma configuration 523
8.4 QWRAP configuration (DBDC) 523
8.4.1 QWRAP DBDC configuration 1 523
8.4.2 QWRAP DBDC configuration 2 524

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT peng.yefeng@zte.com.cn

8.4.3 QWRAP DBDC configuration 3 525
8.4.4 QWRAP TBTC configuration 1 526
8.4.5 QWRAP TBTC configuration 2 527
8.5 DBDC repeater mode 529
8.6 Enable NSS Wi-Fi offload 530
8.7 Enable WPS enhancement for repeaters with a single push button 531
8.8 Enable Wi-Fi memory pre-allocation 532
8.9 Enable ATF using UCI commands 532
8.9.1 UCI command to view airtime statistics and logs 533
8.10 Single AP Band Steering Daemon (lbd) parameters 533
8.11 Multi-AP Coordinated Steering and Adaptive Path Selection parameters 549
8.12 Range Extender Placement and Auto-configuration Daemon 562
8.13 Wi-Fi SON with PLC: UCI commands to enable Dynamic Link Metrics 568
8.14 Co-ordinated ATF between root AP and repeater 569
8.14.1 Enable coordinated ATF feature in wsplcd config file 569
8.14.2 Config section 'atf-config' 570
8.14.3 Config section 'atf-re-config' 570
8.14.4 Config section 'atf-re-radioparams-config' 572
8.15 Enable LOWI (Locationing Service) 574
8.16 Avoid module reload during wifi up/down 574
8.17 Content aware routing 575
8.18 Support for SFE on IPQ8064 platforms-Coexistence between offload engines 575
8.18.1 Select acceleration engine using UCI command 576
8.18.2 Select acceleration engine automatically 576
8.18.3 Select NSS acceleration engine explicitly 576
8.18.4 Select SFE acceleration engine explicitly 577
8.18.5 Select NSS acceleration engine in IPQ40xx platform 577
8.18.6 Select unknown acceleration engine 578
8.19 Disable Wi-Fi SON blacklist functionality in band steering 578
8.20 UCI commands to specify separate Wi-FI SON AP steering threshold values for 2.4 GHz and 5 GHz bands 579
8.21 UCI command to specify backhaul for root AP and RE for enhanced WPS in Wi-Fi SON . 581
8.22 Disable satellite fronthaul during backhaul disconnection 582
8.22.1 Network socket connection with wpa_supplicant 583
8.23 UCI command to configure fast_lane mode in tri-radio boards 583
8.24 Configure memory footprint reduction on QCA9880, QCA9886, and QCA9889 585
8.25 Suspension of probes in 2.4 GHz band 587
8.26 Configure round-trip time on AP using UCI commands 587
8.27 Enable or disable cfg80211 for UCI configuration 588
8.28 UCI configuration of the multiwan 589
8.28.1 Sample multiwan connection 590
8.28.2 User interface for configuring mwan3 591

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

8.28.3 Configure mwan3 592
8.28.4 Configure policy 594
8.28.5 Sample mwan3 settings on IPQ40x 595
8.29 User interface of link aggregation 600
8.30 UCI configuration for ACL 602
8.31 UCI configuration for multiple PPPoE sessions 610
8.31.1 Create multiple PPPoE sessions 610
8.31.2 Create dual WAN port 610
8.31.3 Display PPPoE statistics 611
8.31.4 UCI commands 611
8.32 UCI commands to configure RADIUS parameters for retries 614
8.33 Disable hyd IGMP snooping and enable hardware snooping 616
8.34 Enable 802.11v DMS 617
8.35 Configure best uplink hysteresis for Wi-Fi SON 619
8.36 Configure FILS capability using UCI commands 619
8.37 UCI commands to configure FILS Discovery Frames 620
8.38 UCI command to configure SA query timeout for PMF 620
8.39 Sample AP UCI configuration for 11ax 622
8.40 UCI commands to configure SAE 624
8.41 UCI commands for OWE 627
8.42 UCI commands for Suite B in WPA3 629
8.43 Display the hyd and lbd versions 630
8.44 UCI commands to configure AP steering for legacy clients 631
8.45 Best uplink algorithm for PLC interface metrics in a daisy chain Wi-Fi SON and PLC network 633
8.46 UCI commands for ETSI compliance for COT and CAT on QCA9880 and QCA9980 chipsets 636
8.47 UCI commands for Green AP power-saving in connected-idle status 637
8.48 UCI command to configure idle client detection for Tx path 638
8.49 UCI parameters for multi-AP 638
8.49.1 Wi-Fi SON messages and multi AP equivalents 639
8.49.2 Load balancing daemon (lbd) parameters for multi-AP 640
8.49.3 Wi-Fi security and PLC daemon (wsplcd) parameters for multi-AP 642
8.49.4 RE placement and auto-configuration daemon (repacd) parameters for multi-AP 642
8.49.5 Hy-Fi (hyd) parameters for multi-AP 646
8.49.6 Wi-Fi parameters for multi-AP 648
8.49.7 UCI parameters for legacy AP steering, backhaul steering, and channel selection in MAP 649
8.50 UCI settings to preserve ATF configurations across a system reboot 651
8.50.1 UCI configuration sections for ATF 651
8.51 UCI parameters to configure AP TWT 656
8.52 UCI commands to configure the legacy upgrade-steering mechanism for clients 657
8.53 UCI commands for EAPOL minimum rate WMI resource configuration 658

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

8.54 Configure smart mesh enhancements based on monitor-direct feature for IPQ807xA and IPQ817x 659
8.55 UCI commands to configure DPP 659
8.56 UCI commands to configure Rx flow identification based on protocol type 662
8.57 UCI configuration of IPv4 over IPv6 (464) XLAT and CLAT 663
8.58 Configure the live dump collection utility 667
8.59 Globally disable same SSID feature using UCI settings 668
8.60 Collection of Wi-Fi SON statistics using nl80211 vendor command 668
8.61 UCI script changes for 6 GHz support 670
8.61.1 Guidelines for 6 GHz out-of-band functionality 670
8.62 Configure ACS optimization using target provided channel grading metrics 671
8.62.1 ICM with target-provided channel grading 671
8.63 Configure DMA buffer debug for spectral scan 673
8.64 Support interoperation with dual-band and tri-band mesh nodes 673
8.65 Configure MAP 1905-only operation mode 675
8.66 UCI configuration for Wi-Fi SON static and dynamic memory computation 676
8.66.1 Enable and configure memory debug feature 683
8.67 UCI configuration of Easy Mesh R2 commercialization algorithms 683
8.67.1 Traffic separation configuration 684
8.67.2 Channel scan configuration 685
8.67.3 WPA3 support configuration 685
8.67.4 Coordinated CAC configuration 686
8.68 Configure single AP and multi-AP mode in data elements 687
8.68.1 UCI configuration of data elements 688
8.69 Configure interoperation with dual-band and tri-band mesh nodes in Wi-Fi SON 690
8.70 Configure RCPI support for Wi-Fi SON networks 690
8.71 Configure advanced CAC implementation for MAP r1 691
8.71.1 Configuration of fallback frequency 692
8.72 UCI support to configure individual VAPs independently 692
8.73 Configure zero BSS credential MAP agent onboarding support 693
8.74 Enhancements to NSS offloading statistics 695
8.75 Overriding the default driver behavior for 6 GHz-only AP 696
8.76 Configure EMA AP using .ini and UCI settings 697
8.77 6 GHz security compliance with 802.11 specification 698
8.78 Configure WDS extended feature 699
8.79 Configure Mesh 2.0 architecture 699
8.80 Implementation of mixed backhaul encryption across radios 702
8.81 Configuration changes for credential cloning over multiple SSIDs 703
8.82 Configure DPP R2 using UCI and ACFG commands 704
8.83 Configure service prioritization using UCI parameters 705
8.84 Configure dynamic role selection of CAP/RE (Controller/Agent) using UCI parameters . . 706

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

8.85 Configure optional IEEE 802.11 IE support for non-Tx VAP in MBSSID/EMA using .ini and UCI parameters 708

A Country code definitions 711

Qualcomm
Confidential – May Contain Trade Secrets
2021-05-13 03:57:44 PDT
pan.yafeng@zte.com.cn

Tables

Table 2-1 iwconfig parameters 36
Table 3-1 802.11ac interface aggregation parameters 44
Table 3-2 802.11na and 802.11ac interfaces specific statistics 46
Table 3-3 ANI parameters 48
Table 3-4 Association/ACL parameters 48
Table 3-5 Beacon configuration parameters 51
Table 3-6 Channel width parameters 57
Table 3-7 Debug parameters 60
Table 3-8 802.11 Protocol layer debug bitmask 61
Table 3-9 HAL debug flags 62
Table 3-10 DCS-IM parameters 64
Table 3-11 Green AP Parameters 66
Table 3-12 Hotspot 2.0 parameters 67
Table 3-13 HT20/HT40 coexistence parameters 67
Table 3-14 iQue parameters 68
Table 3-15 Physical layer parameters 71
Table 3-16 Protection mechanism parameters 77
Table 3-17 Radio-related parameters 78
Table 3-18 Radio resource management (802.11k) parameters 88
Table 3-19 Radiotap monitor header parameters 92
Table 3-20 Regulatory parameters 93
Table 3-21 Security-related parameters 94
Table 3-22 STA parameters 99
Table 3-23 Turbo parameters 100
Table 3-24 Tx beamforming parameters 101
Table 3-25 Tx and Rx statistics parameters 116
Table 3-26 Unassociated power consumption improvement parameters 221
Table 3-27 Smart antenna parameters 222
Table 3-28 dword3 parameters 222
Table 3-29 WDS parameters 225
Table 3-30 Access categories and modes 225
Table 3-31 WMM parameters 226
Table 3-32 256QAM parameters 228
Table 3-33 Hy-Fi parameters 229
Table 3-34 Channel loading/Channel hopping parameters 231
Table 3-35 802.11k Parameters 232
Table 3-36 Block Channel List Parameters 233
Table 3-37 Aggregate Size Parameters 233
Table 3-38 Target recovery parameters 233
Table 3-39 Uncategorized protocol layer parameters 234
Table 3-40 2.4 GHz VHT 256-QAM Broadcom interoperability support 235

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

Table 3-41 QWRAP debug 235
Table 3-42 ATF parameters 236
Table 3-43 Wake-on-wireless AP assist parameters 239
Table 3-44 DFS parameters 240
Table 3-45 Repeater DFS channel switch options 241
Table 3-46 Zero CAC DFS options 242
Table 3-47 NSS Offload - Tx Flow Control buffer pool size 244
Table 3-48 TR-069 commands 255
Table 3-49 List of monitor filters 255
Table 3-50 Bluetooth coexistence parameters 256
Table 3-51 List of Interface Manager parameters 257
Table 3-52 OCE parameters 259
Table 3-53 ACS/DCS/OBSS iwpriv parameters 263
Table 3-54 Dynamic encapsulation and decapsulation configuration 266
Table 3-55 Raw mode simulation parameters 267
Table 3-56 RDK-B HAL API parameters 268
Table 3-57 Offload statistics parameters 269
Table 3-58 Tx Ack timeout parameters 271
Table 3-59 11.0 WLAN driver parameters 272
Table 3-60 Additional protocol and radio layer commands 316
Table 3-61 FTM debug parameters 327
Table 3-62 Extended NSS signaling parameters 328
Table 3-63 Uncategorized protocol and radio layer commands 328
Table 3-64 Miscellaneous radio layer parameters 339
Table 3-65 Miscellaneous protocol layer parameters 348
Table 3-66 Miscellaneous radio layer parameters in offload mode only 364
Table 3-67 MBO parameters 376
Table 3-68 Neighbor report parameters 380
Table 3-69 Target core dump parameter 380
Table 3-70 CSL parameter 381
Table 3-71 Tx data packet capture parameters 382
Table 3-72 QCN broadcast probe response parameters 383
Table 3-73 CE logging parameters 384
Table 3-74 FILS parameter 386
Table 3-75 Soft-blocking parameters 388
Table 3-76 Protected management frames (PMF) parameters 389
Table 3-77 Tx chain-mask parameter 390
Table 3-78 ICM parameters 391
Table 3-79 Wi-Fi medium utilization statistics parameters 391
Table 3-80 Parameter APIs for carrier networks 393
Table 3-81 CBS for VoW parameters 394
Table 3-82 TDMA parameter 395
Table 3-83 Strict passive scan parameter 396

Qualcomm Confidential – May Contain Trade Secrets 2021-09-13 03:57:44 PDT (ann.yafengzhe.com.cn)

Table 3-84 Periodic CFR dump generation parameter 397
Table 4-1 Wifitool 802.11k parameters 401
Table 4-2 Wifitool channel loading parameters 402
Table 4-3 Block channel list 404
Table 4-4 Block acknowledge 408
Table 4-5 wifitool commands 409
Table 4-6 wifitool MBO parameters 412
Table 4-7 ACS channel score parameter 416
Table 4-8 Softblocking parameters 418
Table 4-9 CBS for VoW parameters 420
Table 5-1 AP list elements 436
Table 5-2 STA list elements 436
Table 5-3 Channel list elements 439
Table 5-4 Capabilities list elements 440
Table 5-5 Configure NAWDS parameters 441
Table 5-6 Configure HMWDS/HMMC parameters 444
Table 5-7 Configure/show ATF parameters 445
Table 5-1 RSSI scan of NACs parameters 452
Table 5-1 RCC mode config commands 459
Table 5-2 TA_RA configuration commands 460
Table 5-3 Other CFR config commands 463
Table 5-4 Sample commands 464
Table 5-5 Configure/show ATF parameters 467
Table 6-1 Athssd parameters 471
Table 6-2 Athssd debug parameters 471
Table 6-3 Radartool parameters 474
Table 6-4 Zero CAC DFS Radartool parameters 475
Table 6-5 Spectral tool parameters 477
Table 6-6 ICM command line parameters 483
Table 8-1 Per VAP configuration parameters 517
Table 8-2 Maximum number of wired and wireless clients 522
Table 8-3 Band and AP Steering Configurable Parameters 534
Table 8-4 Multi-AP Coordinated Steering and Adaptive Path Selection Parameters 550
Table 8-5 Replacement and Auto-Configuration Daemon Parameters 564
Table 8-6 Wireless configuration file parameters 567
Table 8-7 Acceleration engine selection parameters 576
Table 8-8 UDF profile configuration format 602
Table 8-9 ACL rule confirmation format 603
Table 8-10 ARP configuration format 607
Table 8-11 Interface entry configuration format 609
Table 8-12 UCI parameters for legacy AP steering, backhaul steering, and channel selection in MAP 649
Table 8-13 INI settings for EMA AP 697
Table 8-14 UCI parameters to configure DPP R2 704

Qualcomm Confidential – May Contain Trade Secrets 2021-05-12 03:57:44 PDT jan.yafeng@zte.com.cn

Table 8-15 UCI parameters to configure DPP R2 704
Table A-1 Country code definitions 711

Qualcomm
Confidential – May Contain Trade Secrets
2021-05-13 03:57:44 PDT
pan.yafeng@zte.com.cn

Preface

This document provides information on the Qualcomm Technologies Wireless LAN (WLAN) Access Point (AP) Driver Command Line Interface, which is a part of the Qualcomm Technologies AP system. This system consists of the OS kernel, utility functions, and the WLAN AP driver.

About this document

The document consists of the following chapters and appendixes:

Chapter 1	WLAN AP driver CLI tools
Chapter 2	iwconfig commands
Chapter 3	iwpriv and cfg80211tool commands
Chapter 4	wifitool utility commands
Chapter 5	wlanconfig utility commands
Chapter 6	Miscellaneous commands
Chapter 7	LOWI commands
Chapter 8	UCI commands
Appendix A	Country code definitions

NOTE: Starting with the QCA_Networking_2019.SPF.9.0 release, parity and uniformity of support is established in WLAN driver, version 11.0, for all WLAN host features that were supported in WLAN driver, version 10.4. Several WLAN host capabilities that were previously available in the WLAN 10.4 driver are now supported in the WLAN 11.0 driver.

NOTE: Support is not available for direct attach radios and the QCA9531, QCA9558, and QCA9563 family of chipsets in the QCA_Networking_2019.SPF.9.0 release and later.

NOTE Starting with the IPQ6018.ILQ.11.0 ES1 release, the WEXT mode (iwpriv) is not supported. Use the cfg80211tool command equivalent, instead. In this document, replace `iwpriv` in the command syntax instances with `cfg80211tool`, while configuring the different parameters. Parameters that are newly introduced, starting

with the WLAN driver, version 11.0, can be configured only with cfg80211tool mode; only parameters that existed in WLAN driver, version 10.4 and earlier continue to be supported in WLAN driver, version 11.0, have iwpriv equivalent syntaxes.

NOTE All 160/80+80 MHz related information is applicable for the QCA9994, QCA9886, QCA9888, QCA9984 platforms and all 160 MHz related information is applicable QCN90xx/QCN602x and IPQ807x platforms only.

Additional resources

Qualcomm Technologies reference design hardware, software, and documentation contain proprietary information of Qualcomm Technologies, 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, see the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

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.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

…	An ellipsis indicates a repetition of the preceding parameter.
>	The right angle bracket separates successive menu selections. Example: Start > Programs > DK > wdreg_install.

Qualcomm
Confidential – May Contain Trade Secrets
2021-05-13 03:57:44 PDT
pan.yafeng@zte.com.cn

1 WLAN AP driver CLI tools

The access point (AP) driver command line interface (CLI) 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 Netlink interface for the driver. The Qualcomm Technologies WLAN driver supports these tools out of the box without modification.

The wireless tools use device names to determine the device to configure. In the Qualcomm Technologies 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 AP software — Direct Attach (DA) and Offload (OL). Support is not available for direct attach radios and the QCA9531, QCA9558, and QCA9563 family of chipsets in WLAN driver, version 11.0. Information about direct attach architecture that is presented in this guide is for representative purposes only and is not meant for implementation.

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

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

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

1.2 cfg80211 tool (iwpriv equivalent)

WEXT-based configuration tools (such as iwpriv or iwconfig) cannot be used on cfg80211-based systems. Generic cfg80211 based drivers contain no applications such as iwpriv or iwlist to configure private commands. Therefore, Qualcomm Technologies developed an nl80211-based cfg80211tool to configure private commands. Input to the cfg80211tool is aligned with existing iwpriv commands; for example, command form and the number of arguments are as same as existing iwpriv commands.

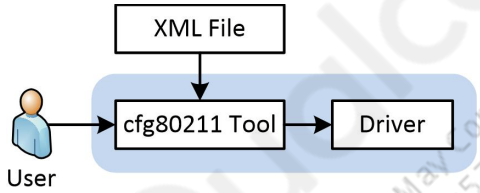


Figure 1-1 cfg80211 tool block diagram

The cfg80211tool is easily configurable and includes many options. The cfg80211tool sends and received commands from the driver. The XML file is populated with command/event details and help information. To add new commands, the developer populates the XML file with the required details and cfg80211toll to handle sending (set) and receiving (get) data from driver. XML format supports these tags:

Tag	Properties	
VendorCmd	*name*	Map the command with user option
	id	Get corresponding id for the command
VendorRsp	*name*	Map the ID with the command to display the details to user
	id	Get corresponding command name for the ID
	max_attr	Parse nlmsg
VendorEvent	*name*	Map the ID with command to display the details to user
	id	Get corresponding command name for the ID
	max_attr	Parse nlmsg

Attribute	*name*	Map attribute ID to a name
	id	Prepare/parse the attribute
	type	Type of attribute to be filled / parsed (Supported: s8, s16, s32, u8, u16, u32, u64, mac_addr, string, nested, BLOB…)
	max_attr	Mandatory for nested, not required for other attributes
Switch	*name*	Name to display; different values may carry different data (obtainable by parsing)
Case	*value*	Evaluate switch values
	name	Name to display
HELP	*data*	Data to display
END_CMD	Indicates end of the command	

For more information, see Section 8.27, Enable or disable cfg80211 for UCI configuration.

NOTE Starting with WLAN driver, version 11.0, support for the configuration API (ACFG) utility is discontinued. It is recommended to use the Linux 802.11 configuration API, cfg80211tool mode, for configuring settings.

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 pan.yafeng@zte.com.cn

2 iwconfig commands

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 2-1 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`

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 2-1 iwconfig parameters (cont.)

Parameter	Format	DA	OL	Description

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

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

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 2-1 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** ■ For 802ax rates – 1 to 54 Mbps (11b and 11a/g rates) can be configured. For other rates, other commands can be used 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 beyond 802.11n rates. The threshold should be more than 256 B (as defined by iwconfig). `#iwconfig ath0 rts 256`

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 2-1 iwconfig parameters (cont.)

Parameter	Format	DA	OL	Description
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. When the power_setting value is not configured with this paramter, it 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`

2.1 Applying channel and SSID parameters using cfg80211tool

For cfg80211-based drivers, no generic applications, such as iwpriv or iwlist, are available to configure private commands. Therefore, Qualcomm Technologies developed an nl80211-based cfg80211tool to configure private commands. Data that is sent as input to the cfg80211tool is aligned with existing iwpriv commands; for example, the command form and the number of arguments are the same as existing iwpriv commands.

To configure the channel and SSID parameters with cfg80211tool, use the `cfg80211tool athN <command> <arguments>` command.

The **iw** is a generic-purpose tool that can be used for other generic configurations such as RTS and fragmentation.

1. Use `cfg8021tool athX channel <channel number>` command (same as iwconfig) to configure channel.
2. Enter the `hostapd_cli -i ath0 -p /var/run/hostapd-wifi0/ update_beacon` command to update channel information in beacon.

NOTE Starting with WLAN driver, version 11.0, both WEXT mode and the Linux 802.11 configuration API, cfg80211tool mode, are supported. If a certain proprietary wireless tool command, such as `wlanconfig`, `wifitool`, or `apstats`, is entered, the driver checks the mode that is being currently used—WEXT or cfg80211—on the system, and processes the command accordingly.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

2.2 Enable and disable Wi-Fi interfaces

Typically, Linux interfaces are brought down using the ifconfig command. While this mechanism is applicable for non-wireless network interfaces, certain additional commands must be issued for all Wi-Fi interfaces. There are dependent applications on top of the wireless interface and it is necessary to bring them down properly, at first. hapd and wpsd daemons are used, instead of hostapd_cli and wpa_cli daemons, to bringing down/up/reconfigure AP and STA interfaces respectively.

Commands to disable Wi-Fi interface

The following are the commands to disable Wi-Fi interface:

- In AP:
  - hapd <interface name> disable
  - Example: hapd ath0 disable
- In STA:
  - wpsd <interface name> disable
  - Example: wpsd ath0 disable

The hapd and wpsd commands will use the default control path while bringing down the interface.

The following are the commands for user-specified control path

- In AP:
  - hostapd_cli -i <interface name> -p <control path> disable
  - Example: hostapd_cli -i ath0 -p /var/run/hostapd-wifi0 disable
- In STA:
  - wpa_cli -p <control path> disconnect
  - Example: wpa_cli -p /var/run/wpa_supplicant-ath0/ disconnect

Commands to enable Wi-Fi interface

The following are the commands to enable Wi-Fi interface:

- In AP:
  - hapd <interface name> enable
  - Example: hapd ath0 enable
- In STA:
  - wpsd <interface name> enable
  - Example: wpsd ath0 enable

The hapd and wpsd commands use the default control path while bringing up the interface.

The following are the commands to use the user-specified control path:

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yifeng@zte.com.cn

- In AP:
  - hostapd_cli -i <interface name> -p <control path> enable
  - Example: hostapd_cli -i ath0 -p /var/run/hostapd-wifi0 enable
- In STA:
  - wpa_cli -p <control path> disconnect
  - Example: wpa_cli -p /var/run/wpa_supplicant-ath0/ reconnect

Commands to reconfigure Wi-Fi interface

- Interface must be up before using this <interface> reconfig command.
- It is not possible to use user-specific configuration file to bring up STA interface. Therefore, the default configuration file is used to bring up STA interface.
- The reconfigure command brings up the interface with the default configuration file. Therefore, a user must change the default configure file according to their r requirement before issuing this command.

The following are the commands to reconfigure interface:

- In AP:
  - hapd <interface> reconfig
  - Example: hapd ath0 reconfig
- In STA:
  - wpsd <interface name> reconfig
  - Example: wpsd ath0 reconfig

The following are the commands to use user-specified control path:

- In AP:
  - wpa_cli -g /var/run/hostapd/global raw REMOVE <interface name>
  - wpa_cli -g /var/run/hostapd/global raw ADD bss_config=<interface name>:<user defined configuration file>

  Example:
  - wpa_cli -g /var/run/hostapd/global raw REMOVE ath0
  - wpa_cli -g /var/run/hostapd/global raw ADD bss_config=ath0:/var/run/hostapd-ath0.conf
- In STA:
  - wpa_cli -p <control path> reconfigure
  - Example: wpa_cli -p /var/run/wpa_supplicant-ath0/ reconfigure

NOTE Starting with QCA_Networking_2020.SPF.11.3 release, due to a hostpad upgrade, the “sec_channel_offset” option for “hostapd_cli chan_switch” command is mandatory for BW greater than 20 MHz.

Qualcomm Confidential – May Contain Trade Secrets 2021-06-13 03:57:44 PDT pan.yafeng@zte.com.cn

Command syntax until SPF.11.2:

```
hostapd_cli -i ath0 -p /var/run/hostapd-wifi0/ chan_switch 10 5320
bandwidth=40 vht center_freq1=5310 center_freq2=0
```

Command syntax starting with SPF.11.3:

```
hostapd_cli -i ath0 -p /var/run/hostapd-wifi0/ chan_switch 10 5320
bandwidth=40 vht center_freq1=5310 center_freq2=0 sec_channel_offset=-1
```

For 20 MHz, sec_channel_offset is not expected; +1/-1 provided for 20MHz will fail.

For 40 MHz, HT40+ requires sec_channel_offset=1, for 40- it is -1.

For 80 MHz and 160 MHz, sec_channel_offset should be 1.

2.3 Sysfs enhancements to support 6 GHz

- Sysfs entry named “supported_bands” added for reading supported bands on underlying pdev

  The following is a sample output:

```
root@OpenWrt:/# cat ./sys/devices/*/net/wifi0/supported_bands
5GHz/root@OpenWrt:/#
root@OpenWrt:/# cat ./sys/devices/*/net/wifi1/supported_bands
2GHz/root@OpenWrt:/#
root@OpenWrt:/# cat ./sys/devices/*/net/wifi2/supported_bands
2GHz/5GHz/root@OpenWrt:/#
```

- Sysfs entry named “6g_maxchwidth” added for reading max 6G supported chan width on underlying pdev

  The following is a sample output:

```
root@OpenWrt:/# cat ./sys/devices/*/net/wifi0/2g_maxchwidth
root@OpenWrt:/# cat ./sys/devices/*/net/wifi0/5g_maxchwidth
160root@OpenWrt/#
root@OpenWrt:/# cat ./sys/devices/*/net/wifi0/6g_maxchwidth
160root@OpenWrt/#
```

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

3 iwpriv and cfg80211tool commands

This chapter defines all of the **iwpriv** parameters available for each layer. This chapter also describes the **cfg80211tool** parameters that are available for configuration, instead of the iwpriv parameters, starting with the WLAN driver, version 11.0. For cfg80211 based drivers, no generic applications such as iwpriv or iwlist are present to configure private commands. Therefore Qualcomm Technologies developed an nl80211-based cfg80211tool to configure private commands.

NOTE Starting with the IPQ6018.ILQ.11.0 ES1 release, the WEXT mode (iwpriv) is not supported. Use the cfg80211tool command equivalent, instead. In this chapter, replace `iwpriv` in the command syntaxes with `cfg80211tool`, while configuring the different parameters. Parameters that are newly introduced, starting with the WLAN driver, version 11.0, can be configured only with cfg80211tool mode; only parameters that existed in WLAN driver, version 10.4 continue to be supported in WLAN driver, version 11.0s, have iwpriv equivalent syntaxes.

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 virtual access points (VAPs) attached to the specific radio are affected by the configurations made to the radio layer.

For all parameters that have a corresponding “get” parameter, the current default or configured values are returned.

3.1 Aggregation parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *802.11n features, Configure AMPDU per VAP for OL modes, and Configure AMSDU per VAP for OL modes* sections in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-1 802.11ac interface aggregation parameters

Parameter	Command	DA	OL	Description

AMPDU
getAMPDU
	iwpriv wifi*N* AMPDU
	{1|0}
		Y	N	Enables (1) or disables (0) Tx aggregated MAC protocol data unit (AMPDU) aggregation for the entire interface. This configuration is per radio and not per VAP. 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`
				In DA mode, enter the iwpriv wifiX AMPDU <0/1> command, which is an equivalent command on wifiX interface to disable/enable AMPDU. This command calls ath_set_config with parameter ATH_PARAM_AMPDU to set sc->sc_txaggr. Then, sc->sc_txaggr is used in multiple places.
				In DA mode, enter the iwpriv wifiX AMPDUFrames {1-64} command to set AMPDU maximum frames limit per radio.

ampdu get_ ampdu	iwpriv ath*N* ampdu
	{1...64}
		Y	Y	Sets maximum number of MAC PDUs (MPDUs) that are aggregated in a single AMPDU. This maximum number is per VAP and not per radio. This parameter is specific to 802.11ac.
				`#iwpriv ath0 ampdu 1`
				`#iwpriv ath0 get_ampdu`
				`ath0 get_ampdu:1`
				Because several vendors configure AMPDU per VAP, it is necessary to make this maximum-count to be compatible and consistent. As a result, the count is performed per VAP and not per radio.
				In offload mode, enter the iwpriv athX ampdu {0-64} command to set the AMPDU parameter per VAP. The value 0 indicates AMPDU must be disabled. The values 1-64 denote the AMPDU subframes limit. The value of 1 signifies the presence of only one frame in an AMPDU packet.
				Enter the iwpriv athX get_ampdu command to retrieve the AMPDU parameters per VAP.
				For 11ax devices, the syntax of this command is iwpriv athN ampdu {0...255}.

Table 3-1 802.11ac interface aggregation parameters (cont.)

Parameter	Command	DA	OL	Description

amsdu get_ amsdu	iwpriv ath*N* amsdu {1...31}	Y	Y	Sets maximum number of AMSDU subframes. This maximum number is per VAP and not per radio. Specific to 802.11ac.
				`#iwpriv ath0 amsdu 1`
				`#iwpriv ath0 get_amsdu`
				`ath0 get_amsdu:1`
				Because several vendors configure AMSDU per VAP, it is necessary to make this maximum-count to be compatible and consistent. As a result, the count is performed per VAP and not per radio.
				In both DA and OL modes, for AMSDU support per VAP, an effective approach is to use “iwpriv athX amsdu {0-31}” to enable/disable AMSDU and set AMSDU subframes limit. The value 0 indicates that AMSDU is disabled and the values 1-31 indicate the AMSDU subframes limit. The radio-level AMSDU setting for DA path that is available using the iwpriv wifiX amsdu <0 |1> command remains unchanged. The VAP-level setting and radio-level setting are independent of each other. Only when both are enabled in DA mode, AMSDU is activated.
				For 11ax devices, the syntax of this command is iwpriv athN amsdu {0...7}.
				This command does not have a cfg80211 equivalent.

AMSDU get_AMSDU	iwpriv wifi*N* amsdu {0 | 1}	Y	N	Enables (1) or disables (0) Tx aggregated MAC service data unit (AMSDU) aggregation for the entire interface. This configuration is per radio and not per VAP. 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 802.11ac.
				`#iwpriv wifi0 AMSDU 1`
				`#iwpriv wifi0 getAMSDU`
				`wifi0 getAMSDU:1`
				In DA mode, enter the iwpriv wifiX amsdu <0/1> command, which is an equivalent command on wifiX interface to disable/enable AMSDU. This command calls ath_set_config with parameter ATH_PARAM_AMSDU to set sc to sc_txamsdu.
				In DA mode, enter the iwpriv wifiX amsdu {1-31} command to set AMSDU maximum frames limit per radio. A known limitation exists with the DA implementation. It only supports small packet AMSDU, if the packet length is greater than 100, aggregation to AMSDU is not performed.

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

Table 3-1 802.11ac interface aggregation parameters (cont.)

Parameter	Command	DA	OL	Description

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

aggr_burst get_ aggr_burst	iwpriv wifiN aggr_ burst 0|1|2|3 duration	N	Y	Sets the aggregate burst duration for a particular traffic class. 0 – BE 1 – BK;2 – VI;3 – VO `#iwpriv wifi0 aggr_burst 0 800` Use the `get` parameter to retrieve the configured aggregate burst duration. UDP downlink throughput is approximately 30 Mbps less when compared with the throughput in QCA_Networking_ 2016.SPF.3.0 release for 100-500 clients. For multi-client test, it is expected to test with aggr_burst set to '0'.

Table 3-2 802.11na and 802.11ac interfaces specific statistics

Parameter	Command	DA	OL	Description

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

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-2 802.11na and 802.11ac interfaces specific statistics (cont.)

Parameter	Command	DA	OL	Description

txrx_fw_stats	iwpriv ath*N* txrx_fw_stats *{1,2,3,6,13,14,16, 17, 18, 19}*	N	Y	Tx and Rx related statistics from target
				`#iwpriv wifiN enable_ol_stats 1 (When you want to run some statistics, you need to run the this command before entering the command to collect and display statistics)`
				`#iwpriv ath0 txrx_fw_stats 1`
				NOTE This section applies only to legacy chipsets and not the IPQ807x/IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx/QCN90xx or QCN602x + IPQ807xA/IPQ60xx/IPQ50xx chipset.

setaddbaoper	`iwpriv ath`*N* `setaddbaoper` *1|0*	N	Y	Enables/disables automatic processing for aggregation/block ACK setup frames. More simplistically, this command might be interpreted as setting both Tx(FW)/Rx(Host) BA modules to manual/auto mode. 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` This command can also set the Tx (FW) module to manual mode which will disable FW from sending out ADD BA Req by default after association. All the following wifitool commands to set the BA module to certain user expectations must be issues only after setting the BA module mode manually: ■ wifitool athN refusealladdbas X ■ wifitool athN sendaddba AID TID BufSize ■ wifitool athN senddelba AID TID initiator reason ■ wifitool athN setaddbaresp AID TID status
dyn_burst get_dyn_burst	iwpriv athX dyn_burst <1 | 0> iwpriv athX get_dyn_burst	N	Y	Enables (1) or disables (0) burst dynamically per interface. With burst enabled, it is observed in certain network deployments with multi-clients, such as more than 10 clients, key performance indicator (KPI) degradation occurs with client phones. These drops in KPIs are not observed with burst disabled. Enter the `iwpriv athX get_dyn_burst` command to retrieve the configured setting of dynamic burst.

3.2 ANI parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Adaptive noise immunity (ANI)* section in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-3 ANI parameters

Parameter	Command	DA	OL	Description

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 and 802.11ax. In the QCA9980 and iPQ807x family of chipsets, ANI is enabled by default for all WLAN modes.
				`#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`

3.3 Association/ACL parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Access control lists functionality* section in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-4 Association/ACL parameters

Parameter	Command	DA	OL	Description

addmac delmac getmac
 	
maccmd get_maccmd
 	
acl_notify
get_acl_notify
	iwpriv ath*N* addmac *macaddr*
	
	iwpriv ath*N* delmac *macaddr*
	
	iwpriv ath*N* maccmd *cmd*
	
	iwpriv athN acl_notify 1|0
	
	iwpriv athN get_acl_notify
		Y	Y	These parameters set up and modify the primary 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 primary access control list (ACL). **delmac** deletes addresses from the primary ACL. These parameters have no get equivalents. **getmac** displays the list of MAC addresses monitored by the primary 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:
				Refer valid cmd values below.
				The configuration to support association denial notification can be enabled/disabled and queried with following iwpriv commands:
				iwpriv athX acl_notify 1|0
				iwpriv athX get_acl_notify.

				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

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-4 Association/ACL parameters (cont.)

				3	Flush the current ACL list
				4	Suspend current ACL policies. Re-enable with a 1 or 2 command.

addmac_sec
 	
delmac_sec
 	
getmac_sec
 	
get_maccmd_sec
 	
maccmd_sec
	iwpriv athN addmac_sec macaddr
	
	iwpriv athN delmac_sec macaddr
	
	iwpriv athN getmac_sec
	
	iwpriv athN get_maccmd_sec
	
	iwpriv athN maccmd_sec
		Y	Y	These parameters set up and modify the secondary 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_sec adds specified MAC addresses to the secondary access control list (ACL). delmac_sec deletes addresses from the secondary ACL. These parameters have no get equivalents. getmac_sec displays the list of MAC addresses monitored by the secondary ACL.
				`#iwpriv ath0 addmac_sec 00:03:7f:00:00:20`
				`#iwpriv ath0 delmac_sec 00:03:7f:00:12:34`
				`#iwpriv ath0 getmac_sec`
				`ath0 getmac_sec:00:03:7f:00:00:20`
				maccmd_sec instructs how the ACL is used to limit access the AP. The default is 0. The get parameter returns the current value.
				Refer valid cmd values below.	

					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`
					For the secondary ACL, use the following command
					`#iwpriv ath0 maccmd_sec 1`
					`#iwpriv ath0 get_maccmd_sec`
					`ath0 get_maccmd:1`
					acl_notify configures if denial of association should be notified on the interface. Enable: 1 Disable: 0
					The default value is 1. The get parameter **get_acl_notify** returns the current configuration
					`#iwpriv ath0 acl_notify 1`
					`#iwpriv ath0 get_acl_notify`
					`ath0 get_acl_notify:1`

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-4 Association/ACL parameters (cont.)

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

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

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

block_interbss g_blk_interbss	iwpriv wifiN block_interbss 0|1 iwpriv wifiN g_blk_interbss	N	Y	Allow or disallow forwarding the traffic between stations of two different VAPs. 0 – Allow traffic to switch between stations of two vaps 1 – Disallow the traffic to switch between stations The get command retrieves the configured blocking of inter-BSS capability.

3.4 Beacon configuration parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Transmission of frames and beacons* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-5 Beacon configuration parameters

Parameter	Command	DA	OL	Description

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

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`

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-5 Beacon configuration parameters (cont.)

Parameter	Command	DA	OL	Description

doth_ch_chwidth	iwpriv ath*N* doth_ch_ch_width {*home-channel* | *foreign-channel*} *<number of beacons before change>* *<bandwidth>*	Y	Y	Configures dynamic bandwidth switching within the operating channel or different channel. This parameter applies for 5 GHz radios only; it is not supported for 2 GHz radios. To switch only the bandwidth and retain the current operating channel, define the same operating channel and enter the new bandwidth. To switch the bandwidth and also switch to a different channel, enter both the new channel and the new bandwidth to be configured.
				The following is the channel-width change command for the same channel:
				`iwpriv ath0 doth_ch_chwidth <Home Channel> <number of beacons before change> <new b/w>`
				`iwpriv ath0 doth_ch_chwidth 40 5 20`
				The following is the channel-width change command for a different channel:
				`iwpriv ath0 doth_ch_chwidth <Foreign Channel> <number of beacons before change> <new b/w>`
				`iwpriv ath0 doth_ch_chwidth 149 5 20`
				For this command to work, the iwpriv athN doth 1 command must be issued first. The command does not work if iwpriv athN get_doth returns 0.
				An AP's channel switching ability is enhanced to allow connected peers to switch AP's operating bandwidth, if such a new bandwidth is supported.
				The following existing command is used to switch AP's operating bandwidth:
				iwpriv athX doth_ch_chwidth <chan_num> <beacon_count> <new_bandwidth>
				This command is enhanced to allow connected peers which have the necessary support to update their operating bandwidth. This feature is intended to work with all existing configurations of the APs. Previously, this command only changes the operating bandwidth of the AP and not the connected peers/STAs.

doth_chanswitch	`iwpriv ath`*N* `doth_chanswitch` *channel tbtt*	Y	Y	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`
		Y	Y	channel: Specifies channel to which AP will switch
		Y	Y	tbtt: Number of beacons to wait before doing the switch

Table 3-5 Beacon configuration parameters (cont.)

Parameter	Command	DA	OL	Description

strict_doth	iwpriv wifiN strict_doth {0/1}
	
	iwpriv wifiN g_strict_doth
		Y	Y	Impacts channel switch triggered by iwpriv commands doth_chanswitch and doth_ch_chwidth.
				If enabled (set to 1) channel switch will happen if and only if all associated stations over all VAP's on this radio interface support spectrum management else channel switch triggered by user space applications by means of iwpriv command will not be honored.
				If disabled (set to 0), spectrum management capability of associated stations is not checked and channel switch will happen after doing channel switch announcement.
				By default strict_doth is disabled. g_strict_doth is corresponding getter command.
				`#iwpriv wifi0 strict_doth 1`
				`#iwpriv wifi0 g_strict_doth`

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

set_bcn_rate <rate> **get_bcn_rate**	iwpriv athX set_bcn_rate <rate in Kbps>	Y	Y	Specify the beacon rate and retrieve the beacon rate. By default for 2G radio : iwpriv ath1 get_bcn_rate will give 1000 By default for 5G radio : iwpriv ath1 get_bcn_rate will give 6000 Apart from the supported operating rates for the current operating band, following rates will be supported for AP operating in 6 GHz band: 8600 Mbps - HE MCS0 NSS1 17200 Mbps - HE MCS1 NSS1 25800 Mbps - HE MCS2 NSS1 For instance, to set beacon rate to HE MCS0 NSS1, the user will need to issue the cfg80211tool athX set_bcn_rate 8600 command. Issuing this command should update only the beacon rates and rates for other management frames should be unaffected.

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`

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-5 Beacon configuration parameters (cont.)

Parameter	Command	DA	OL	Description

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. This command is not supported on IPQ807x/IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx/QCN90xx or QCN602x + IPQ807xA/IPQ60xx/IPQ50xx platforms.
				`#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
		Y	Y	Sets/gets application specific optional IE buffer.
				`#iwpriv ath0 setoptie`
				`#iwpriv ath0 getoptie`
				`ath0 getoptie:`
				This command does not have a cfg80211 equivalent.

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 09:25:55 pal.yafeng@ste-cdma

Table 3-5 Beacon configuration parameters (cont.)

Parameter	Command	DA	OL	Description

shortgi
get_shortgi
	iwpriv ath*N* shortgi *{X}*	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.
				In HE mode, sets the user selected short guard interval (shortgi) for fixed rate (not auto rate) only. It is important to set fixed rate values using the iwpriv athX he_mcs command before issuing this command.
				`#iwpriv ath0 shortgi X`
				`#iwpriv ath0 get_shortgi`
				`ath0 get_shortgi:X`
				Where *X* is:
				■ 0 - 800 ns
				■ 1 - 400 ns (supported in VHT only)
				■ 2 - 1600 ns (supported for 11ax devices only)
				■ 3 - 3200 ns (supported for 11ax devices only)
				The max PHY rate displayed by default will be with 800ns SGI. For both auto-rate and fixed rate the FW will use 800ns SGI by default. If the user changes the auto-rate (he_ar_gi_ltf)/fixed rate (shortgi) SGI, the max PHY rate will reflect bit rate corresponding to the SGI setting.
				Max PHY rate is displayed according to 800ns SGI(4.8039 Gb/s | 1.1471 Gb/s). Because 400ns SGI is not used by default for HE., it will only be used when the user explicitly sets SGI to 400ns (in which case max PHY rate is displayed based on SGI 400ns. The following list describes the differences in the max PHY rate values between SPF 10.0 and SPF 11.0.
				Until SPF 10.0, the values corresponded to QCA proprietary HE 400 ns GI. Starting with SPF 11.0, the values correspond to 802.11ax standard based 800 ns GI.
				On IPQ807x:
				■ For 5G 11axa HT80 ath0, 4.949 Gb/s in SPF 10.0 CS and 4.8039 Gb/s in SPF 11 CS
				■ For 2.4G 11axg HT40 ath1, 1.1818 Gb/s in SPF 10.0 CS 1.1471 Gb/s in SPF 11.0 CS

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. Specifying this command has no effect.
				`#iwpriv ath0 vap_contryie 1`
				`#iwpriv ath0 get_vapcontryie`
				`ath0 get_vapcontryie:1`
				1 — Enable Country IE support
				0 — Disable Country IE support

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-5 Beacon configuration parameters (cont.)

Parameter	Command	DA	OL	Description

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

dynamicbeacon
g_dynamicbeacon
	iwpriv athN dynamicbeacon {1|0}	Y	Y	Enable or disable the Dynamic Beacon feature. This is to create a dedicated AP VAP with hidden SSID enabled. This AP vap will not send beacons always, it sends only when certain condition met. The default this feature is disabled (value is 0). The get parameter returns the current value.
				`#iwpriv ath0 dynamicbeacon 1`
				`#iwpriv ath0 g_dynamicbeacon`

db_rsssi_thr g_db_rssi_thr	iwpriv athN db_rssi_thr	Y	Y	Set /Get the RSSI threshold value. This is to check the RSSI of received probe request when dynamic beacon feature is enabled to start the beacon. The default value is set to 60 dBm.
				This value can be configured from 10dBm to 90dBm.
				`#iwpriv ath0 db_rssi_thr 50`
				`#iwpriv ath0 g_db_rssi_thr`

mgmt_rsssi_thr
g_mgmt_rssi_thr
	iwpriv athN mgmt_rssi_thr	Y	Y	Set/get the rule to discard management frames below a configured RSSI threshold.
				The capability to enable the vendor driver to support a rule to discard the reception of a management frame type when the RSSI is lower than a specified dBm is available. The threshold in dBM is programmable by customer stack.
				A check is implemented for setting RSSI in AP's management frame receive path. If the Rx RSSI for this management frame is lower than the RSSI set using the ioctl from user application, then the received management frame is dropped. A counter is incremented for every management frame that is dropped.
				This value can be configured from 10dBm to 90dBm.
				`#iwpriv ath0 mgmt_rssi_thr 50`
				`#iwpriv ath0 g_mgmt_rssi_thr`

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:54:44 PDT pan.yafeng@zte.com.cn

Table 3-5 Beacon configuration parameters (cont.)

Parameter	Command	DA	OL	Description

db_timeout g_db_timeout	iwpriv athN db_timeout	Y	Y	Set/Get the duration for the AP vap to send the beacons when dynamic beacon feature is enabled. The default value is set to 30 secs. This value can be configured from 30 secs to 3600 secs (1 hour).
				`#iwpriv ath0 db_timeout 40`
				`#iwpriv ath0 g_db_timeout`

random_gen_mode <0-2> **get_random_gen**	iwpriv athx random_gen_mode <0-2> iwpriv athx get_random_gen	Y	N	Set the mode for improving random number generation; supported on all direct-attach chipsets. This parameter is not applicable for offload chipsets. Description of modes: Mode0—Random is updated by rssi on interrupt. Mode1—Random is updated by rssi every 10 ms, and if the same rssi is read for 5 times, it is read from ADC regs and a sleep of 100 ms occurs. Mode2—Random is updated by analog-to-digital (ADC) registers alone every 100 ms. If a mode value that it outside of the prescribed range of mode values is entered, the entry is not validated successfully and the setting is not saved.

3.5 Channel width parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Channel width management (CWM 20/40)* and *Modify channel width in CSA for 5 GHz* sections in the *Core channel selection capabilities* chapter of the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-6 Channel width parameters

Parameter	Command	DA	OL	Description	

chextoffset
get_chextoffset
	iwpriv ath*N* chextoffset *channeloffset*	Y	Y	Sets the extension (Secondary) channel offset field in the AP beacon High Throughput Information Element (HT IE). If this parameter is not executed, then the extension channel offset is taken from the device settings. This parameter has a corresponding get parameter. The default value is 0.
				`#iwpriv ath0 chextoffset 0`
				`#iwpriv ath0 get_chextoffset`
				`ath0 get_chextoffset:0`	

				0	Use the device settings as default
				1	Extension (Secondary) channel is above the control (Primary) channel
				-1	Extension (Secondary) channel is below the control (Primary) channel

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-6 Channel width parameters (cont.)

Parameter	Command	DA	OL	Description

chwidth
get_chwidth
	iwpriv ath*N* chwidth *channelwidth*	Y	Y	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.
				`#iwpriv ath0 chwidth 0`
				`#iwpriv ath0 get_chwidth`
				`ath0 get_chwidth:0`

				0 (HT)20 MHz
				1 40 MHz
				2 80 MHz
				3 80_80/160 MHz
				>3 Invalid

cwmenable
get_ cwmenable
	iwpriv ath*N* cwmenable *{1|0}*	Y	Y	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.
				`#iwpriv ath0 cwmenable 1`
				`#iwpriv ath0 get_cwmenable`
				`ath0 get_cwmenable:1`

Qualcomm
Confidential – May Contain Trade Secrets
2021-05-13 03:57:44 GMT
pan.yafeng@zte.com.cn

Table 3-6 Channel width parameters (cont.)

Parameter	Command	DA	OL	Description	

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

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

Table 3-6 Channel width parameters (cont.)

Parameter	Command	DA	OL	Description	
mode **get_mode**	iwpriv ath*N* mode *operatingmode*	Y	Y	11AHE20	A-band 20 MHz channels
				11GHE20	G-band 20 MHz channels
				11AHE40PLUS	A-band 40 MHz channels. Select frequency channels higher that control channel as the extension channel.
				11AHE40MINUS	HE G-band 40 MHz channels. Select frequency channels higher that control channel as the extension channel.
				11GHE40PLUS	HE A-band 40 MHz channels. Select frequency channels lower that control channel as the extension channel.
				11GHE40MINUS	HE G-band 40 MHz channels. Select frequency channels lower that control channel as the extension channel.
				11AHE80	HE A-band 80 MHz channels.
				11AHE160	HE A-band continuous 160 MHz channels.
				11AHE80_80	HE A-band discontinuous 80+80 MHz channels.

cfreq2
get_cfreq2
	iwpriv athN cfreq2 center_ freq	N	Y	This is only applicable when operating mode is 11ACVHT80_80. This sets center frequency for 2nd 80MHz band. The argument for cfreq2 is as provided as integer. This integer can be channel center frequency index (IEEE channel) or center frequency in MHz.
				`#iwpriv ath0 cfreq2 106 or`
				`#iwpriv ath0 cfreq2 5530`	

3.6 Debug parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Transmit power control (TPC) section* and the *Using statistics for debugging* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-7 Debug parameters

Parameter	Command	DA	OL	Description	

dbgLVL
getdbgLVL
	iwpriv ath*N* dbgLVL *{1|0}*	Y	Y	Controls the debug level of the VAP-based debug print statements. It is normally set to zero, eliminating all prints. The input value should be a hexadecimal value. See Table 3-8.
				`#iwpriv ath0 dbgLVL 0xffffffff`
				`# iwpriv ath0 getdbgLVL`
				`ath0 getdbgLVL:0xffffffff`	

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

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-7 Debug parameters (cont.)

Parameter	Command	DA	OL	Description

HALDbg GetHALDbg	iwpriv wifi*N* HALDbg *{1|0}*	Y	N	Not applicable for OL architecture. 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 3-8 on page 61. 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

Qualcomm Confidential – May Contain Trade Secrets 2021-05-12 09:57:14 PDT jerman.yafeng@21cn.com

Table 3-8 802.11 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
IEEE80211_MSG_CRYPTO	0x10000000	Crypto work
IEEE80211_MSG_INPUT	0x08000000	Input handling
IEEE80211_MSG_XRATE	0x04000000	Rate set handling

Table 3-8 802.11 Protocol layer debug bitmask (cont.)

Symbolic name	Bit value	Description
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 3-9 HAL debug flags

Symbolic name	Enable Bit	Description
HAL_DBG_RESET	0x00000001	Information pertaining to reset processing and initialization
HAL_DBG_PHY_IO	0x00000002	PHY read/write states
HAL_DBG_REG_IO	0x00000004	Register I/O, including all register values. Use with caution.
HAL_DBG_RF_PARAM	0x00000008	RF parameter information and table settings
HAL_DBG_QUEUE	0x00000010	Queue management for WMM support

Table 3-9 HAL debug flags (cont.)

Symbolic name	Enable Bit	Description
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

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

For more information on usage guidelines and the functionality regarding these parameters, see the *Dynamic channel selection—Interference mitigation (DCS-IM) for 802.11n and 802.11a* section in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-10 DCS-IM parameters

Parameter	Command	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`
				The values for dcs_enable parameter can be entered either in decimal format or hexadecimal format (by prefixing the 0x notation before the number).

				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). **NOTE** This value is supported only in 5G and 6 G.
				3 — Enable both DCS for CW_IM and DCS for WLAN_IM **NOTE** This value is supported only in 5G and 6G.
		N		4 — Enable DCS for AWGN interference mitigation. This value is introduced in the QCA_Networking_2021.SPF.11.4 release. This value is supported only in 6G.
				5 — Enable DCS for CW_IM and AWGN_IM. This value is introduced in the QCA_Networking_2021.SPF.11.4 release. This value is supported only in 6G.
				6 — Enable DCS for WLAN_IM and AWGN_IM. This value is introduced in the QCA_Networking_2021.SPF.11.4 release. This value is supported only in 6G.
				7 — Enable DCS for CW_IM, WLAN_IM, and AWGN_IM. This value is introduced in the QCA_Networking_2021.SPF.11.4 release. This value is supported only in 6G.

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	N	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`

Qualcomm Confidential – May Contain Trade Secrets 2021-05-19 03:57:44 PDT pan.yafeng@qc.com

Table 3-10 DCS-IM parameters (cont.)

Parameter	Command	DA	OL	Description

s_dcs_phyerrth
g_**dcs_phyerrth**
	iwpriv wifi*N* s_dcs_phyerrth *value*	Y	Y	Configures PHY error threshold. Default value of PHY error penalty is set as 300.
				`#iwpriv wifi0 s_dcs_phyerrth 500`
				`#iwpriv wifi0 g_dcs_phyerrth`
				`wifi0 get_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`

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

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

NOTE The valid values for disabling/enabling DCS with CW detection on 2.4 GHz radios are either 0 or 1.

Although the dcs_enable parameter can be enabled through the console by entering the `cfg80211tool wifiX dcs_enable <val>` command, this parameter can be alternatively set in /etc/config/wireless also as follows:

option dcs_enable <val> (if enabled through /etc/config/wireless)

Also, this command can also be entered using UCI as follows:

wireless.wifi0.dcs_enable='3'

If the dcs_enable parameter is used as part of radio/VAP bring-up scripts using UCI or /etc/config/wireless addition, DCS WLAN interference management is not supported (value: 0x2) for fixed channel configuration. This behavior ensures that a movement away from the preset channel does not occur. If WLAN interference management is required even for fixed channel configuration, then it needs to be explicitly enabled using cfg80211tool after the bring-up is complete. This restriction is not applied on channels selected using ACS.

3.8 Green AP parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Green AP power-saving in connected-idle status* and *Green AP componentization* sections in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-11 Green AP Parameters

Parameter	Command	DA	OL	Description

ant_ps_on
get_ant_ps_on
	iwpriv ath*N* ant_ps_on *{0 | 1 |2}*	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`
				A value of 1 indicates legacy Green AP mode, NO STA mode.
				A value of 2 indicates NUM STREAM mode.
				NO STA mode is the default Green AP mode, where antenna power save is enabled when no clients are connected to the AP. This mode is enabled by iwpriv command “ant_ps_on” 1.
				NUM STREAM mode is the new feature added to the Green AP where if only 1x1 clients are connected to the AP then Antenna power save is turned ON. If a client capable of 2x2 and above connects to the AP then Antenna power save is turned OFF. There is a counter maintained in the driver to keep track of number of multistream clients connected, if the count goes to zero then antenna power save is turned on after the wait period. This mode is enabled by iwpriv command “ant_ps_on” 2.
				Starting with QCA_Networking_2020.SPF.11.3 release, Advanced Green AP mode is introduced, which supports reduced PHY power consumption in beaconing-only-mode, is introduced over legacy Green AP mode. This feature is implemented on top of legacy Green AP, so the feature is supported only if legacy Green AP mode is supported.

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`

athX qdf_cv_lvl 0x5d0005	iwpriv athX qdf_cv_lvl 0x5d0005	Y	Y	Enables the display of green AP state machine transitions.
sm_gap_ps <value> **g_sm_gap_ps**	**cfg80211tool athX sm_gap_ps <value>** **cfg80211tool athX g_sm_gap_ps**			Enable (1) or disable (0) Spatial Multiplexing Power Save (SMPS) used for further enabling Green AP in the STA VAP of root AP or repeater setup. The command needs to be given to the STA VAP of RE and root AP VAP, before giving the green AP command. By default, this SMPS for Green AP is disabled. Enter the get command to show if SMPS is enabled or not.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

3.9 Hotspot 2.0 parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *HotSpot 2.0* section in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-12 Hotspot 2.0 parameters

Parameter	Command	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`

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`

3.10 HT20/HT40 coexistence parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Channel width management (CWM 20/40)* section in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-13 HT20/HT40 coexistence parameters

Parameter	Command	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

Table 3-13 HT20/HT40 coexistence parameters (cont.)

Parameter	Command	DA	OL	Description

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

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

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

3.11 iQue parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *iQue (Intelligent QoS for User Experience)* section in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-14 iQue parameters

Parameter	Command	DA	OL	Description
get_hbrstate	iwpriv ath*N* *get_hbrstate*	N	N	Displays 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*	Y	N	Sets HBR mitigation. See Table 3-30 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%*	Y	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%*	Y	N	Sets the lower bound PER. If PER is better than this value while probing, HBR unblocks the node (UDP video traffic to this node gets resumed). The PER is expressed as a percentage; for example, 25 means a 25% packet error rate. The get parameter returns the current value.
get_hbrtimer	`iwpriv ath`*N* `get_hbrtimer`	Y	N	Disabled internally.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-14 iQue parameters (cont.)

Parameter	Command	DA	OL	Description

hbrtimer
get_hbrtimer
	iwpriv ath*N* hbrtimer *timeout*	Y	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`

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

Table 3-14 iQue parameters (cont.)

Parameter	Command	DA	OL	Description

mcastenhance
g_
mcastenhance
	iwpriv ath*N* mcastenhance *mode*	Y	Y	Set multicast 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 multicast enhancement
				1 — Enable multicast enhancement; use tunneling mode. In OL chipsets, tunneling is not supported. Multicast enhancement in tunneling mode is not supported for DA chipsets; this option is deprecated.
				2 — Enable multicast enhancement; use translating mode.
				AP software versions 9.5/9.5.1/9.5.2/9.5.3/10.x
				Value | Snooping | Multicast enhancement
				0 | Enabled | True multicast packet is send if any interested member is present.
				1 | Enabled | Tunneled 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 multicast functionality.
				AP software versions 11.x **NOTE** Only the values of 0, 5, and 6 are supported; other values are not supported. 0 denotes that multicast enhancement be disabled.
				Value | Snooping | Multicast enhancement
				5 | Disabled | Enables Hy-Fi managed multicast functionality.
				6 | Disabled | Enables APs to allow multicast frames to be sent as unicast frames to each individual member of the mentioned multicast group to improve the QoS of the transmission between the STA and the AP. After setting this command to 6, the AMSDU path will be enabled in the multicast enhancement path, which sends frames over the air in the native Wi-Fi format (802.11).The feature will need to be disabled explicitly to revert to transmission of frames in the default Ethernet (802.3) mode.

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

3.12 Physical layer parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Power management techniques, Regulatory compliance of WLAN APs,* and *IEEE802.11 features* chapters in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-15 Physical layer parameters

Parameter	Command	DA	OL	Description	

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 711 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) This parameter is applicable to mobile hostposts only and is not supported for WLAN APs.

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:49 PDT pan.yafeng@zte.com.cn

Table 3-15 Physical layer parameters (cont.)

Parameter	Command	DA	OL	Description

LDPC
getLDPC
	iwpriv athN ldpc
	{0/1/2/3}
		Y	N	The default value is 3.
				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.
				Below are the options for Low-density parity check.
				Disable LDPC:-
				`#iwpriv athN0 LDPC 0`
				LDPC RX:-
				`#iwpriv athN0 LDPC 0x1`
				LDPC TX:-
				`#iwpriv athN0 LDPC 0x2`
				
				LDPC TXRX:-
				`#iwpriv athN0 LDPC 0x3`
				Enables (1) or disables (0) the Low-density parity check feature, as described in 802.11n specification. The default value is 1.
				The iwpriv athN ldpc command sets/unsets B13 of HE PHY CAP indicating support for the transmission and reception of LDPC encoded packets.
				where LDPC command value is:
				0 - Disable (which also indicates enable 'bcc')
				3 - Enable Tx/Rx LDPC
				Note: values 1 and 2 are not allowed in HE mode where 1 means Rx ldpc and 2 means Tx ldpc for modes lower than HE such as VHT.
				This option will have an effect only on chips supporting the LDPC feature. On other chips, this option will have no effect. Specific to 802.11n.
				`# iwpriv athN0 LDPC 1`
				Get LDPC:-
				`# iwpriv athN0 getLDPC ath0 getLDPC:1`
				`Enable/disable ldpc.`
				
				For 11ax devices:
				Sets/resets B13 of HE PHY CAP indicating support for the transmission and reception of LDPC encoded packets.
				Where X is:
				■ 0 - disable (which also indicates enable 'bcc')
				■ 3 - enable tx/rx ldpc
				Note: Values of 1 and 2 are not allowed in HE mode where 1 means Rx LDPC, and 2 means Tx LDPC for modes lower than HE such as VHT.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-15 Physical layer parameters (cont.)

Parameter	Command	DA	OL	Description

txchainmask
rxchainmask
get_ txchainmask
get_ rxchainmask
	iwpriv wifi*N* txchainmask *mask*
	iwpriv wifi*N* rxchainmask *mask*
		Y	Y	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 Technologies devices, up to 3 chains can be used, others are restricted to 3, 2 or 1.
				The maximum number of chains available for the device. For dual chain devices, chain 2 is not available. Single chain devices only support chain 0. The chains are represented in the bit mask as:
				Chain 0: 0x01
				Chain 1: 0x02
				Chain 2: 0x04
				Chain 4: 0x08
				Chainmask selection can affect several performance factors. For a 3-chain device, an Rx chainmask of 0x05 (or 0x03) is used for 2x2 stream reception. For near range operations, a Tx chainmask of 0x05 (or 0x03) minimizes near range effects. For far range, a mask of 0x07 is used for Tx. The default chainmask values are stored in EEPROM. This iwpriv command overrides the chainmask settings. The get parameters returns the current values.
				`#iwpriv wifi0 txchainmask 0x05`
				`#iwpriv wifiN rxchainmask 0x05`
				`#iwpriv wifiN get_txchainmask`
				`wifi0 get_txchainmask:5`
				`#iwpriv wifiN get_rxchainmask mask`
				`wifi0 get_rxchainmask:5`
				For 11ax devices, the `txchainmask` and `rxchainmask` parameters set the transmit chainmask and receive chainmask values, respectively, where mask is:
				■ 0x0f for NSS 1-4.
				■ 0xff for NSS 5 and higher
				The txchainmask indicates the number of Tx streams, and the chains that are used for MIMO devices. The rxchainmask indicates the number of Rx streams, and the chains that are used for MIMO devices.
				To support 1x1 160 MHz with 4x4 160 MHz, such as IPQ807xA and QCN90xx/QCN61xx, the Tx and Rx chainmasks can be configured appropriately.
				To change chainmask for 1x1 160 MHz:
				`cfg80211tool wifiX txchainmask 0x11`
				`cfg80211tool wifiX rxchainmask 0x11`
				To change chainmask for 4x4 160 MHz:
				`cfg80211tool wifiX txchainmask 0xff`
				`cfg80211tool wifiX rxchainmask 0xff`

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-15 Physical layer parameters (cont.)

Parameter	Command	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 20`
				`#iwpriv wifi0 getTxPowLim2G`
				`wifi0 getTxPowLim2G:20`

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

antgain_2g antgain_5g	**iwpriv wifiX antgain_2g**
	iwpriv wifiX antgain_5g
		Y	Y	Set antenna gain for 2 GHz band and 5 GHz band. The allowed values are between 0 and 30, which are measured in dBm.
				Example:
				`iwpriv wifi0 antgain_2g 10`
				`iwpriv wifi0 antgain_5g 10`

disp_tpc	**iwpriv wifi0 disp_tpc**	Y	Y	Displays the tpc table. For each available transmission rate, it shows whether it supports TxBF/STBC/1-chain/2-chain/3- chain. In case of TxBF or STBC, it also shows the Tx power limit in txpower_txbf[] and txpower_ stbc[] respectively. Example: `iwpriv wifi0 disp_tpc 1` This is not currently available on IPQ8074.
get_minpower	**iwpriv ath0 get_minpower**	Y	Y	Get the minium tx power of radio.
get_maxpower	**iwpriv ath0 get_maxpower**	Y	Y	Get the max tx power of radio.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-15 Physical layer parameters (cont.)

Parameter	Command	DA	OL	Description

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

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

s_txpow_ mgmt
g_txpow_ mgmt
	iwpriv athN s_ txpow_ mgmt *frame_ subtype transmit_ power*
	iwpriv athN g_ txpow_ mgmt frame_subtype
		Y	Y	Used to configure transmit power for beacon, probe response, (re-)association request,(re-)association response, auth, disassociation and de-auth frames for each SSID.The power value configured for a frame can be altered dynamically without any need for restart. The frame subtype is set as per standard IEEE conventions.
				For example, to set the transmit power of beacon, enter the following:
				`iwpriv ath0 s_txpow_mgmt 0x80 8`
				For example, to obtain the tx power set for beacon, enter the following:
				`iwpriv ath0 g_txpow_mgmt 0x80 wifi0 get_ txpow_mgmt:8`
				The transmit power set for a particular frame type can be undone by setting the power to 255. The transmit power value is an 8 bit integer for both direct attach and offload radios.
				If the txpower is configured for both the radio and the SSID, then the txpower of the SSID overrides the txpower of the radio. To undo the effects of tx power set for a particular management frame type, use the following command: "iwpriv athN set_txpow_ mgmt <frame-subtype> 255".

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-15 Physical layer parameters (cont.)

Parameter	Command	DA	OL	Description
ath0 disable_ cabq <0 | 1>	iwpriv ath0 disable_cabq <0 | 1>	Y	Y	In the Wi-Fi driver, the `iwpriv ath0 disable_cabq 1` command is implemented to disable the buffering multicast frames when STA is in power-save mode for both OL and DA modes. For OL mode, this command triggers a WMI message to target to enable this do-not-buffer or disable-CABQ feature. After enabling this feature, on the multicast client, it is noticed that certain packets are missing because the multicast packets are delivered instantaneously as they are received, regardless of whether any of the STAs are in power-save mode. Also, in the sniffer capture, it is observed that multicast packets are delivered at different beacon intervals, unlike the behavior in power save mode. Alternatively, enter the `iwriv ath0 disable_cabq 0` command to disable this feature to avoid buffering multicast frames when STA is in power-save mode for both OL and DA modes

set_txpow_ctl
get_txpow_ctl
	iwpriv wifiN set_ txpow_ctl *frame_ subtype* transmit_power
	iwpriv wifiN get_ txpow_ctl frame_subtype
		Y	Y	Configure transmit power for control frames for each SSID. The power value configured for a frame can be altered dynamically without any need for restart. The frame subtype is set according to the standard IEEE conventions. For example, to set the transmit power of beacon, enter the following:
				`iwpriv wifi0 s_txpow_ctl 0x040 8`
				For example, to obtain the Tx power set for beacon, enter the following command:
				`iwpriv wifi0 g_txpow_ctl 0x040 wifi0 get_txpow_ctl:4`
				The transmit power set for a particular frame type can be undone by setting the power to 255. The transmit power value is an 8-bit integer for both direct attach and offload radios. To undo the effects of tx power set for a particular control frame type, use the following command:
				`iwpriv athN set_txpow_ctl <frame-subtype> 255`

3.13 Protection mechanism parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *802.11w protected management frames (PMF)* section in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-16 Protection mechanism parameters

Parameter	Command	DA	OL	Description	

protmode
get_protmode
	iwpriv ath*N* protmode *{2|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

3.14 Radio-related parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Power management techniques* section in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-17 Radio-related parameters

Parameter	Command	Description	

6MBAck
Get6MBack
	`iwpriv wifi`*N*
	`6MBAck` *1|0*
		Not applicable for offload mode. This command enables (1) or disables (0) the use of the 6 MBps (OFDM) data rate for ACK frames. If disabled, ACK frames will be sent at the CCK rate. The default value is 0. The get parameter returns the current value. Not applicable for OL.
		`#iwpriv wifi0 6MBAck 1`
		`#iwpriv wifi0 Get6MBAck`
		`wifi0 Get6MBAck:1`	

AddSWBbo
SWBcnRespT
DMABcnRespT
GetAddSWBbo
GetSWBcnRespT
GetDMABcnRespT
	`iwpriv wifi`*N*
	`SWBcnRespT`
	`iwpriv wifi`*N*
	`DMABcnRespT`
	`iwpriv wifi`*N*
	`AddSWBb0`
		Not applicable for offload mode. Adjust the calculation of the ready time for the QoS queues to adjust the QoS queue performance for optimal timing. These parameters are used for experimental adjustment of queue performance. In the AP application they are not relevant, so they should not be modified. Their default value is 0. Each get parameter returns the current value for its parameter. Not applicable for OL.
		`#iwpriv wifi0 SWBcnRespT 1`
		`#iwpriv wifi0 DMABcnRespT 2`
		`#iwpriv wifi0 AddSWBbo 10`
		`#iwpriv wifi0 GetSWBcnRespT`
		`wifi0 GetSWBcnRespT:1`
		`#iwpriv wifi0 GetDMABcnRespT`
		`wifi0 GetDMABcnRespT:2`
		`#iwpriv wifi0 GetAddSWBbo`
		`wifi0 GetAddSWBbo:10`	

		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

AggrProt
AggrProtDur
AggrProtMax
getAggrProt
getAggrProtDur
getAggrProtMax
	`iwpriv wifi`*N*
	`AggrProt` *1|0*
	`iwpriv wifi`*N*
	`AggrProtDur`
	duration
	`iwpriv wifi`*N*
	`AggrProtMax`
	size
		Not applicable for offload mode. Enable RTS/CTS protection on aggregate frames and control the size of the frames receiving RTS/CTS protection. Typically used as a test commands to set a specific condition in the driver. Each get parameter returns the current value for its parameter. Not applicable for OL.
		`#iwpriv wifi0 AggrProt 1`
		`#iwpriv wifi0 AggrProtDuration 8192`
		`#iwpriv wifi0 AggrProtMax 8192`
		`#iwpriv wifi0 getAggrProt`
		`wifi0 getAggrProt:1`
		`#iwpriv wifi0 getAggrProtDur`
		`wifi0 getAggrProtDur:8192`
		`#iwpriv wifi0 getAggrProtMax`
		`wifi0 getAggrProtMax:8192`	

		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.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-17 Radio-related parameters (cont.)

Parameter	Command	Description

ANIEna
GetANIEna
	iwpriv wifi*N*
	ANIEna *0|1*
		Not applicable for offload mode. Enables the automatic noise immunity (ANI) processing in both the driver and the baseband unit. ANI mitigates unpredictable noise spurs in Rx channels that are due to the host system the device is installed in. This feature was added for CardBus and PCIE devices sold in the retail market not pre-installed in host systems. Most AP implementations do not enable ANI, preferring to limit noise spurs by design. The get parameter returns the current value. Not applicable for OL.
		`#iwpriv wifi0 ANIEna 1`
		`#iwpriv wifi0 GetANIEna`
		`wifi0 GetANIEna:1`

AntSwap
DivtyCtl
GetAntSwap
GetDivtyCtl
	iwpriv wifi*N*
	AntSwap *1|0*
	iwpriv wifi*N*
	DivtyCtl *AntSel*
		Not applicable for offload mode. Control antenna switching behavior. For 802.11n devices, these control which chains are used for Tx. For legacy devices, used to determine if diversity switching is enabled or disabled. The get parameters return the current values. Not applicable for OL.
		`#iwpriv wifi0 AntSwap 1`
		`#iwpriv wifi0 DivtyCtl 2`
		`#iwpriv wifi0 GetAntSwap`
		`wifi0 GetAntSwap:0`
		`#iwpriv wifi0 GetDivtyCtl`
		`wifi0 GetDivtyCtl:0`

		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*
		Not applicable for offload mode. Controls a debug flag that will either reset the chip or not when a stuck beacon is detected. If enabled (1), the system will NOT reset the chip upon detecting a stuck beacon, but will dump several registers to the console. Additional debug messages will be output if enabled, also. The default value is 0. The get parameter returns the current value. Not applicable for OL.
		`#iwpriv wifi0 BcnNoReset 1`
		`#iwpriv wifi0 getBcnNoReset`
		`wifi0 getBcnNoReset:1`

CABlevel
getCABlevel
	iwpriv wifi*N*
	CABlevel *%Multicast*
		Not applicable for offload mode. Sets the amount of space that can be used by Multicast 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 multicast 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 Multicast traffic, freeing the remainder for BE traffic. The default value of this parameter is 80 (80% Multicast). The get parameter returns the current value. Not applicable for OL.
		`#iwpriv wifi0 CABlevel 50`
		`#iwpriv wifi0 getCABlevel`
		`wifi0 getCABlevel:50`

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-17 Radio-related parameters (cont.)

Parameter	Command	Description

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

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

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-17 Radio-related parameters (cont.)

Parameter	Command	Description

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

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

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

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

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-17 Radio-related parameters (cont.)

Parameter	Command	Description	

ForceBias
ForBiasAuto
GetForceBias
GetForBiasAuto
	iwpriv wifi*N* ForBiasAuto *1|0*
	iwpriv wifi*N* ForceBias *Bias*
		Not applicable for OL architecture. This command activates the force bias feature; used as a workaround to a directional sensitivity issue in the AR5133 PHY chip in 2.4 GHz bands. The get parameters return the current values.
		`#iwpriv wifi0 ForBiasAuto 1`
		`#iwpriv wifi0 ForceBias 2`
		`#iwpriv wifi0 GetForBiasAuto`
		`wifi0 GetForBiasAuto:0`
		`#iwpriv wifi0 GetForceBias`
		`wifi0 GetForceBias:1`	

		ForBiasAuto	Automatically selects the bias level depending on the selected frequency.
		ForceBias	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.

getchaninfo	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 Qualcomm driver specific data structure. This command has no command line equivalent interface. The data structures used are defined as:
	`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 */`
	`};`
	
	`iwpriv athX getchaninfo <value>`		

HTEna
GetHTEna
	iwpriv wifi*N* HTEna *1|0*	Not applicable for OL architecture. Enables (1) or disables (0) 802.11n (HT) data rates. Normally, only used as a test command. The parameter is set to 1 (enabled) by default. The get parameter returns the current value.
		`#iwpriv wifi0 HTEna 1`
		`#iwpriv wifi0 GetHTEna`
		`wifi0 GetHTEna:1`	

mcast_rate
get_mcast_rate
	iwpriv ath*N* mcast_rate *rate*	Sets multicast 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 multicast on the overall performance of the system. Default is 11 Mbps in 2.4 GHz mode and 6 Mbps in 5 GHz mode. The get parameter returns the current value. For 5 GHz OFDM rates should be used.
		`#iwpriv ath0 mcast_rate 12000`
		`#iwpriv ath0 get_mcast_rate`
		`ath0 get_mcast_rate: 12000`	

Table 3-17 Radio-related parameters (cont.)

Parameter	Command	Description

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

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

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-17 Radio-related parameters (cont.)

Parameter	Command	Description

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

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

QUALCOMM Confidential – Qualcomm Technologies, Inc. and/or its affiliated companies – May Contain Trade Secrets 2021-05-13 pan.yafeng@zte.com.cn

Table 3-17 Radio-related parameters (cont.)

Parameter	Command	Description

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`
		For 11ax devices:
		For 11ax devices, the rate_series parameter takes the following values:
		11N Rates: 0x80-0x97 for HT MCS0..23
		CCK Rates:
		■ 0x1b - 1 Mbps
		■ 0x1a - 2 Mbps
		■ 0x19 - 5.5 Mbps
		■ 0x18 - 11 Mbps
		OFDM:
		■ 0x0b - 6 Mbps
		■ 0x0f - 9 Mbps
		■ 0x0a - 12 Mbps
		■ 0x0e - 18 Mbps
		■ 0x09 - 24 Mbps
		■ 0x0d - 36 Mbps
		■ 0x08 - 48 Mbps
		■ 0x0c - 54 Mbps

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

setchanlist
getchanlist
	Used by an application to set the channel list manually. Channels that are not valid from a regulatory perspective will be ignored. This command is passed a byte array 255 bytes long that contains the list of channels required. A value of 0 indicates no channel, but all 255 bytes must be provided. getchanlist receives this array from the driver in a 255 byte array that contains the valid channel list. The response is a binary array that WLAN tools cannot parse; therefore this cannot be used on the command line.
	Example:
	`iwpriv athN setchanlist ch1 ch2 .....n`
	`iwpriv athN getchanlist`
	`Following list used in channel load report`
	`-band 2 36 40 44 100 104 108`	

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-17 Radio-related parameters (cont.)

Parameter	Command	Description

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

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

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

CCAThEna
GetCCAThEna
	iwpriv wifiX CCAThEna 1/0
	iwpriv wifiX GetCCAThEna
		Not applicable for offload mode. Enables/Disables the Adaptive CCA threshold algorithm. The default value is 0 which means disabled.
		`Along with enabling this flag need to set the values of CCA detection level and CCA detection margin. The CCA threshold value is calculated from CCA detection and CCA margin value.`
		`Disable CCA threshold:`
		`#iwpriv wifi0 CCAThEna 0`
		Enable CCA threshold:
		`#iwpriv wifi0 CCAThEna 1`
		Get CCA threshold value:
		`#iwpriv wifi0 GetCCAThEna`

CCADetLevel
GetCCADetLevel
	iwpriv wifiX CCADetLevel <Value>
	iwpriv wifiX GetCCADetLevel
		Not applicable for offload mode. Set the CCA Detection Level (default value is -70dBm)
		Set CCA detection Level
		`#iwpriv wifi0 CCADetLevel -50`
		Get CCA Detection Level:
		`#iwpriv wifi0 GetCCADetLevel`

CCADetMargin
GetCCADetMargin
	iwpriv wifiX CCADetMargin <Value>
	iwpriv wifiX GetCCADetMargin
		Not applicable for offload mode. Set the CCA Detection Margin (default value is 3)
		Set CCA Detection Margin Level:
		`#iwpriv wifi0 CCADetMargin 5`
		Get CCA Detection Margin Level:
		`#iwpriv wifi0 GETCCADetMargin`

Table 3-17 Radio-related parameters (cont.)

Parameter	Command	Description
pause_scan **get_pause_scan**	iwpriv athX pause_scan <0 | 1> iwpriv athX get_pause_scan	When a client is connected with single band root AP in the 2 GHz band, STA VAPs off the client are on continuous background scan to find the root AP. Therefore, ping timeouts are expected on clients connected with upper 5 GHz and lower 5 GHz AP VAPs. To avoid continuous background scan, pause scan on upper 5 GHz and lower 5 GHz STA VAPs using the `iwpriv athX pause_scan 1` command. To disable the pausing of scan, enter the `iwpriv athX pause_scan 0` command. To retrieve the status of pause scan, enter the `iwpriv ath1 get_pause_scan` command.
vht_sgimask <value> **get_vht_sgimask**	`iwpriv athX vht_sgimask <value>` `iwpriv athX get_vht_sgimask`	By default, the very high throughput (VHT) short guard interval (SGI) mask is enabled for all modulation coding schemes (MCSs). Enter the `iwpriv athX vht_sgimask <value>` command to specify the VHT SGI mask for a specific MCS. For example: ■ A value of 0x4 indicates VHT SGI mask set for only MCS2 ■ A value of 0xFF indicates VHT SGI mask set for MCS0-MCS7 ■ A value of 0x3FF indicates VHT SGI mask set for MCS0-MCS9 Enter the `iwpriv athX get_vht_sgimask` command to retrieve and display the configured SGI mask for the corresponding MCSs.
dyn_bw_rts <0 | 1> **get_dyn_bw_rts**	`iwpriv athX dyn_bw_rts <0 | 1>` `iwpriv athX get_dyn_bw_rts`	Enable (0) or disable (1) dynamic bandwidth for receive to send (RTS) cases only. By default, RTS is disabled. Use this parameter to enable it while setting RTS threshold or for retry frames. If RTS is enabled (using RTS threshold), the bandwidth signaling is enabled (multicast bit set in TA of RTS frame) and RTS is sent in both primary and secondary 20 MHz channels. For 80 MHz, 4 RTS frames are sent in each 20 MHz channel. If client sends CTS back for all 4 RTS frames, then the whole 80 MHz bandwidth can be used for data transmission. If CTS received only in two 20 MHz channels, fallback is performed and only 40 MHz bandwidth is used for data transmission.

3.15 Radio resource management (802.11k) parameters

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.

For more information on usage guidelines and the functionality regarding these parameters, see the *Voice Enterprise* section in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

All of these parameters are supported in both non-offload and offload modes.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-18 Radio resource management (802.11k) parameters

Parameter	Command	Description	

quiet
get_quiet
	iwpriv ath*N* quiet <value>
	iwpriv ath*N* get_quiet
		Configure Quiet Period functions, which are part of the 802.11k specification. The default quiet period parameters are used when this feature is turned on.The *get_quiet* command returns the current status.
		For beacon offload. the value being passed is a 3-bit equivalent where bit0 - enable/disable, bit1- single shot/continuous, and bit2- include /skip Quiet IE in swab.
		For non-beacon offload, the valid value is 0/1, which denotes disable/enable.
		`#iwpriv ath0 quiet 1`
		`#iwpriv ath0 get_quiet`
		`ath0 get_quiet:1`	

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

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

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-18 Radio resource management (802.11k) parameters (cont.)

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

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:00 pan.xifeng@zte.com.cn

Table 3-18 Radio resource management (802.11k) parameters (cont.)

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

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-18 Radio resource management (802.11k) parameters (cont.)

Parameter	Command	Description	
sendnhist	wifitool ath*N* sendnhist *dstmac n_rpts regclass chnum rand_ivl mandatory_duration optional_condtion condition_val*	Transmits a noise histogram report	
		mac_addr	Destination MAC address
		n_rpts	Number of repetitions client should perform. Refer to 802.11k specification for details.
		regclass	Regulatory class.
		chnum	Channel number.
		rand_ivl	Random interval. Refer to 802.11k specification for details.
		mandatory_ duration	Measurement duration. Refer to 802.11k specification for definition.
		optional_condtion	Set optional condition to (1) if desired as part of request. Default is (0).
		condition_val	Condition value if optional condition is true. Refer to 802.11k specification for details.
sendlcireq	wifitool ath*N* sendlcireq *dstmac location latitude_res longitude_res altitude_res azimuth_res optional_condtion condition_val*	Transmits Location Configuration Information for requested station.	
		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	iwpriv 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	iwpriv ath*N* bcnrpt	Gets a beacon report in user space. Will provide most information received in a beacon report.	

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

3.16 Radiotap monitor header parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Support for radiotap header* section in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-19 Radiotap monitor header parameters

Parameter	Command	Description
mon_decoder {0 | 1} **get_mon_decoder** **g_mon_decoder**	*iwpriv athX mon_decoder {0 | 1}* *iwpriv athX get_mon_decoder* *cfg80211tool athX g_mon_decoder*	Supported on both non-offload and offload modes. Switch between radiotap (default) and prism header for monitor VAP: ■ "iwpriv ath01 mon_decoder 1 to use prism ■ "iwpriv ath01 mon_decoder 0 to use radiotap Radiotap is planned to be the primary supported header in the future. The prism header will be deprecated gradually. Currently, radiotap is considered as an industry standard. Linux kernel and other chip manufacturers migrated from PRISM to radiotap many years back. In the future, only radiotap is planned to be supported. However, because of the present infrastructure requiring support for PRISM, the transformation to radiotap is not entirely completed. During this period of migration from PRISM to radiotap, both PRISM and radiotap are continued to be supported with an early stage of radiotap format. The radiotap header format provides more flexibility than the Prism or AVS header formats and allows the driver developer to specify an arbitrary number of fields based on a bitmask presence field in the radiotap header. The get command retrieves the configured value of the radiotap monitor header.

3.17 Regulatory parameters

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

For more information on usage guidelines and the functionality regarding these parameters, see the *Power management techniques* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-20 Regulatory parameters

Parameter	Command	Description

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

3.18 Security parameters

The security-related parameters relate to the security subsystem, and are specific interfaces required by the hostapd and wpa_supplicant programs. Table 3-21 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).

For more information on usage guidelines and the functionality regarding these parameters, see the *WLAN AP modes* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

Table 3-21 Security-related parameters

Parameter	Command	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.

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

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

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

Table 3-21 Security-related parameters (cont.)

Parameter	Command	DA	OL	Description

driver_caps
get_driver_caps
	`iwpriv athN driver_caps caps`	Y	Y	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		

Parameter	Command	DA	OL	Description

dropunencrypted
get_dropunencry
	`iwpriv athN dropunencrypted 0|1`	Y	Y	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`

keymgtalgs
get_keymgtalgs
	`iwpriv athN keymgtalgs algs`	Y	Y	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

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 09:57:44 PDT pan.qualcomm.com.cn

Table 3-21 Security-related parameters (cont.)

Parameter	Command	DA	OL	Description

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`

privacy
get_privacy
	`iwpriv athN privacy 1|0`	Y	Y	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 athN rsncaps flags`	Y	Y	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 athN setfilter filter`	Y	Y	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 0x08`

Value	Algorithm
0x01	Beacon
0x02	Probe request
0x04	Probe response
0x08	Association request
0x10	Association response
0x20	Authentication
0x40	De-authentication
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[];`
	`};`

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-21 Security-related parameters (cont.)

Parameter	Command	DA	OL	Description

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

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

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

ucastciphers **get_ucastciphers**	iwpriv athN ucastciphers cipher_types	Y	Y	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 3-21 Security-related parameters (cont.)

Parameter	Command	DA	OL	Description	

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

wpa
get_wpa
	`iwpriv athN wpa WPA Mode`	Y	Y	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 and it cannot be set explicitly. 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:	

		Y	Y	0	No WPA support (default value; this value is not available for explicit user configuration using the `wpa` parameter)
		Y	Y	1	WPA support
		Y	Y	2	WPA2 support
		Y	Y	3	Both WPA and WPA2 support

wps
get_wps
	`iwpriv athN wps WPS Mode`	Y	Y	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`	

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

3.19 Station (STA) parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *WLAN AP modes* and *Power management techniques* chapters in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-22 STA parameters

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

periodicScan
g_periodicScan
	iwpriv `ath`*N* *periodicScan enable and set*	Sets STA periodic scan support in milliseconds. 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

conf_bssid
get_conf_bssid
	iwpriv athN conf_bssid set and get	Configure the BSSID of hidden SSID AP for which Client/STA wants to connect when strict passive scan is enabled.
		`# iwpriv athx conf_bssid 'bssid of the hidden AP'`
		`Example: iwpriv ath1 conf_bssid 8C:FD:F0:06:B2:2F`
		`# iwpriv ath1 get_conf_bssid`
		`ath1 get_conf_bssid: 8C:FD:F0:06:B2:2F`	

3.20 Turbo parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Memory and bandwidth optimization features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

These parameters are supported in offload and non-offload modes.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-23 Turbo parameters

Parameter	Command	Description	

burst
get_burst
	iwpriv `ath`*N burst 1|0*	Enables (1) or disables (0) Qualcomm 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 Qualcomm super G The get parameter returns the current value. Not valid for partial offload.
		`#iwpriv ath0 compression 0`
		`#iwpriv ath0 get_compression`
		`ath0 get_compression:0`	

		0	Disable
		1	Enable

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

		0	Disable
		1	Enable

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

3.21 Tx beamforming parameters

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

NOTE The devices that are running the WLAN driver, version 11.0 architecture do not support any of the 802.11n beamforming settings. This section describes the Tx beamforming parameters that apply to 802.11ac. For beamforming parameters that apply to 802.11ax, see "11.0 WLAN driver parameters" on page 271.

For more information on usage guidelines and the functionality regarding these parameters, see the *Transmission of frames and beacons* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-24 Tx beamforming parameters

Parameter	Command	Description	
vhtsubfer	iwpriv ath***N*** vhtsubfer *{0|1}*	Single-user beam former	
		0	Disable single-user beam former
		1	Enable single-user beam former
vhtsubfee	iwpriv ath***N*** vhtsubfee *{0|1}*	Single-user beam formee	
		0	Disable single-user beam formee
		1	Enable single-user beam formee
vhtmubfer	iwpriv ath***N*** vhtmubfer *{0|1}*	Multiple-user beam former	
		0	Disable multiple-user beam former
		1	Enable multiple-user beam former
vhtmubfee	iwpriv ath***N*** vhtmubfee *{0|1}*	Multiple-user beam formee	
		0	Disable multiple-user beam formee
		1	Enable multiple-user beam formee
get_vhtsubfee	iwpriv ath*N* get_vhtsubfee	Retrieves the VHT SU beamformee configuration.	
get_vhtmubfee	iwpriv ath*N* get_vhtmubfee	Retrieves the VHT MU beamformee configuration.	
get_vhtsubfer	iwpriv ath*N* get_vhtsubfer	Retrieves the VHT SU beamformer configuration.	
get_vhtmubfer	iwpriv ath*N* get_vhtmubfer	Retrieves the VHT MU beamformer configuration.	

3.21.1 TxBF configuration

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

Beamformer (AP):

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

Beamformee (STA):

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

3.21.2 TxBF statistics

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

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

- See “Tx beamforming data info statistics” on page 110.
- See “Tx self gen statistics” on page 111.

3.22 Firmware statistics on legacy chipsets

NOTE The txrx_fw_stats parameter is available only on legacy chipsets and not on the IPQ807x/IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx/QCN90xx or QCN602x + IPQ807xA/IPQ60xx/IPQ50xx chipsets. Instead, use the txrx_stats parameter to view the Tx and Rx statistics on such chipsets.

For more information on usage guidelines and the functionality regarding these parameters, see the *Using statistics for debugging* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

3.22.1 Target physical device stats

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

Output (STA)

```
root@OpenWrt:/# iwpriv ath0 txrx_fw_stats 1
[ 1277.221252] ### Tx ###
[ 1277.221256] comp_queued       :      128
[ 1277.221258] comp_delivered    :      180
[ 1277.221261] msdu_enqued       :      12218
[ 1277.221264] wmm_drop          :      0
[ 1277.221267] local_enqued      :      12218
[ 1277.221269] local_freed       :      12218
[ 1277.221273] hw_queued         :      12457
[ 1277.221276] hw_reaped         :      12457
[ 1277.221278] mac underrun      :      0
[ 1277.221282] phy underrun      :      0
[ 1277.221284] hw_paused         :      8
[ 1277.221287] seq_posted        :      12221
[ 1277.221291] mu_seq_posted     :      0
[ 1277.221293] seq_failed        :      0
[ 1277.221296] seq_restarted     :      236
[ 1277.221300] tx_abort          :      0
[ 1277.221302] mpdus_requed      :      239
[ 1277.221304] mpdus_sw_flush    :      0
[ 1277.221307] mpdus_hw_filter   :      8
[ 1277.221309] mpdus_truncated   :      0
[ 1277.221312] mpdus_ack_failed  :      318
[ 1277.221314] mpdus_expired     :      0
[ 1277.221317] excess retries    :      318
[ 1277.221319] last rc           :      0
[ 1277.221321] sched self trig   :      0
[ 1277.221325] ampdu retry failed:      0
[ 1277.221328] illegal rate errs :      0
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
[ 1277.221330] pdev cont xretry  :      0
[ 1277.221334] pdev tx timeout   :      0
[ 1277.221336] pdev resets       :      1
[ 1277.221339] ppdu txop ovf     :      0
[ 1277.221342] mcast Drop        :      0
[ 1277.221344]
[ 1277.221344] ### Rx ###
[ 1277.221346] ppdu_route_change :      0
[ 1277.221349] status_rcvd       :      255
[ 1277.221351] r0_frags          :      0
[ 1277.221354] r1_frags          :      0
[ 1277.221356] r2_frags          :      0
[ 1277.221360] htt_msdus         :      0
[ 1277.221362] htt_mpdus         :      0
[ 1277.221365] loc_msdus         :      243
[ 1277.221369] loc_mpdus         :      243
[ 1277.221371] oversize_amsdu    :      0
[ 1277.221373] phy_errs          :      0
[ 1277.221376] phy_errs dropped  :      0
[ 1277.221378] mpdu_errs         :      12
[ 1277.221381] pdev_rx_timeout   :      0
[ 1277.221385] ovfl_mpdu_errs    :      0
[ 1277.221392]
[ 1277.221392]  ### TX extended stats ###
[ 1277.221395] tx_de_cache_miss          128
[ 1277.221398] tx_enqueue_peer_invalid   0
[ 1277.221400] tx_burst_disable_cong_ctrl 0
[ 1277.221403] tx_de_lookup_failed       128
[ 1277.221407] tx_delay_proc             0
[ 1277.221410] tt_prefetch_suspend       0
[ 1277.221412] short_pkt                 0
[ 1277.221416] total_enqueue             12218
[ 1277.221418] peer_id_invalid           0
[ 1277.221421] host_inspect              0
[ 1277.221424] pdev_zero_discard         0
[ 1277.221427] tickle_proc_sched         0
[ 1277.221430] residue_cleanup           0
```

Output interpretation

Tx: Transmission-related statistical information

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

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

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

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

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

local_freed: # of local MSDUs completed

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

hw_queued: # of PPDUs queued to hardware

hw_reaped: # of PPDUs completed from hardware

underrun: # of times Tx under run occurred

tx_abort: N/A

mpdus_requed: # of MPDUs retried

excess_retries: # of times excess tries happened

last_rc: the last hardware rate code used for transmission.

The rate code is encoded as follows:

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

sched_self_trig:

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

ampdu retry failed:

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

illegal rate errs:

Number of times hardware encountered illegal VHT rate PHY errors.

pdev cont xretry:

Number of times firmware encountered persistent excess retries

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

pdev reset:

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

Rx: Reception-related statistical details

ppdu_route_change:

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

status_rcvd:

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

r0_frags:

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

r1_frags: # of buffer fragmentation happened in Ring 1

r2_frags: # of buffer fragmentation happened in Ring 2

r3_frags: # of buffer fragmentation happened in Ring 3

htt_msdus: # of data MSDUs received

htt_mpdus: # of data MPDUs received

loc_msdus: # of non-data MSDUs received

loc_mpdus: # of non-data MPDUs received

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

TX extended statistics

t**x_de_cache_miss:** # of times de_cache missed

t**x_enqueue_peer_invalid:** Data packet is dropped because peer is invalid

tx_burst_disable_cong_ctrl: # of times burst is disabled

tx_de_lookup_failed: Data packets are dropped in tx_de_input_ext()

tx_delay_proc: Pre fetch delay schedule

tt_prefetch_suspend: # of times pre fetch got suspended

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

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

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

Qualcomm Confidential – May Contain Trade Secrets 2021-03-13 03:57:44 PDT yan.yufeng@zte.com.cn

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

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

tickle_proc_sched: Pre-fetch schedule based on tickle

residue_cleanup: Pre-fetch schedule based on waitq residue cleanup

3.22.2 Rx reorder statistics

The iwpriv ath0 txrx_fw_stats 2 command is used to check how frames are discarded due to FCS error.

```
root@OpenWrt:/# iwpriv ath0 txrx_fw_stats 2
```

Output (STA)

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

Output interpretation

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

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

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

NB: those frames are discarded

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

NB: These frames are delivered to upper stack

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

NB: These frames are discarded

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

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

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

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

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

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

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

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

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

3.22.3 Rx rate info statistic

The iwpriv ath0 txrx_fw_stats 3 command is used toe verify RX rate => MCS count, NSS count frames went in 1x1,2x2,3x3,4x4, BW whether used 20/40/80/160,LDPC count and RSSI (comb_ht)s.

```
root@OpenWrt:/# iwpriv ath0 txrx_fw_stats 3
```

Output (STA)

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

Output interpretation

- **MCS counts:** Counters for each MCSs in the range of 0..9 in case of VHT, and MCSs in the range of 0..7 in the case of the HT association. For 802.11n MCSs in the range of 8..23 combine this field with NSS field; for example, MCS8 is NSS 2 MCS0.

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

- **SGI counts**: Counters for each SGI-enabled MCS
- **NSS counts**: Number of spatial streams. This field indicates whether 1x1, 2x2, or 3x3 rate is being used. Combined with MCS, this value provides the actual (802.11n) MCS in case of HT

- **NSTS count**: Specifies whether the frames are being sent with STBC enabled and the transmission is at a 1x1 rate. The NSTS count can be seen to be equal to the sum of STBC counts
- **BW counts**: Number of received frames on 20, 40, and 80 MHz. Useful to debug all the BWs that are being used currently by the transmitter STA
- **Preamble counts**: Index 0 counts legacy (CCK/OFDM) PPDUs, 1 HT, 2 HT with BF (on QCA9880 always 0), three VHT and four VHT with BF (on QCA9880 always zero), five all other, e.g., PHY error
- **STBC rate counts**: Similar to MCS counts, this field displays all MCSs that are enabled with STBC
- **LDPC TXBF counts**: The first counter increments for each received LDPC PDU, whereas the second counter increments for each received TxBF frame (which is not supported on QCA9880)
- **RSSI (data, management)**: Absolute RSSI value as observed in received MAC descriptor for data and management frames, respectively
- **RSSI chain 0**: (sec80, sec40, sec20, pri20) displays the RSSI seen in MAC descriptor for a particular chain across primary or secondary channels. This statistic is useful to ensure that all chains are balanced.

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

3.22.4 Tx rate info statistics

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

3.22.5 Copy engine and host statistics

```
root@OpenWrt:/# iwpriv ath0 txrx_fw_stats 8
++++++++++ CE STATISTICS +++++++++++
CE0 Host sw_index (dst_ring):     0
CE0 Host write_index (dst_ring):  0
CE1 Host sw_index (dst_ring):     9
CE1 Host write_index (dst_ring):  8
CE2 Host sw_index (dst_ring):     54
CE2 Host write_index (dst_ring):  53
CE3 Host sw_index (dst_ring):     0
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
CE3 Host write_index (dst_ring):  0
CE4 Host sw_index (dst_ring):     0
CE4 Host write_index (dst_ring):  0
CE5 Host sw_index (dst_ring):     190
CE5 Host write_index (dst_ring):  189
CE6 Host sw_index (dst_ring):     0
CE6 Host write_index (dst_ring):  0
CE7 Host sw_index (dst_ring):     0
CE7 Host write_index (dst_ring):  1
CE8 Host sw_index (dst_ring):     0
CE8 Host write_index (dst_ring):  127
CE9 Host sw_index (dst_ring):     0
CE9 Host write_index (dst_ring):  0
CE10 Host sw_index (dst_ring):     0
CE10 Host write_index (dst_ring):  0
CE11 Host sw_index (dst_ring):     0
CE11 Host write_index (dst_ring):  0
++++++++++ HOST TX STATISTICS +++++++++++
Ol Tx Desc In Use:  0
Ol Tx Desc Failed:  0
CE Ring (4) Full :  0
DMA Map Error    :  0
Tx pkts completed:  0
Tx bytes completed:  0
Tx pkts from stack:  0
++++++++++ HOST RX STATISTICS +++++++++++
Rx pkts completed:  0
Rx bytes completed:  0
++++++++++ HOST GENERIC STATISTICS +++++++++++
Fast Path on CPU[0]:  8405
Fast Path on CPU[1]:  59466
Fast Path on CPU[2]:  0
Fast Path on CPU[3]:  0
Fast Path on CPU[4]:  0
Fast Path on CPU[5]:  0
Fast Path on CPU[6]:  0
Fast Path on CPU[7]:  0
Non Fast Path on CPU[0]:  4023668
Non Fast Path on CPU[1]:  6475825
Non Fast Path on CPU[2]:  0
Non Fast Path on CPU[3]:  0
Non Fast Path on CPU[4]:  0
Non Fast Path on CPU[5]:  0
Non Fast Path on CPU[6]:  0
Non Fast Path on CPU[7]:  0
Fast Tasklet on CPU[0]:  6108975
Fast Tasklet on CPU[1]:  7891404
Fast Tasklet on CPU[2]:  0
Fast Tasklet on CPU[3]:  0
Fast Tasklet on CPU[4]:  0
Fast Tasklet on CPU[5]:  0
Fast Tasklet on CPU[6]:  0
Fast Tasklet on CPU[7]:  0
Reg. Tasklet on CPU[0]:  3082
Reg. Tasklet on CPU[1]:  0
Reg. Tasklet on CPU[2]:  0
```

```
Reg. Tasklet on CPU[3]:  0
Reg. Tasklet on CPU[4]:  0
Reg. Tasklet on CPU[5]:  0
Reg. Tasklet on CPU[6]:  0
Reg. Tasklet on CPU[7]:  0
++++++++++ HOST FLOW CONTROL STATISTICS +++++++++++
Receive from stack count: 0
non queued pkt count: 33961
queued pkt count: 0
queue overflow count: 0
```

3.22.6 Host multi-task enhance statistics

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

3.22.7 Tx beamforming data info statistics

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

3.22.8 Tx beamforming sounding info statistics

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

3.22.9 Packet error statistics

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

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

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


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

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

3.22.10 Tx self gen statistics

```
root@OpenWrt:/# iwpriv ath0 txrx_fw_stats 16
TX_SELFGEN Info:
su_ndpa           :0
su_ndp            :0
su_bar            :0
SU-BAR is typically used to reset the BA window state on the peer.
su_cts2self       :0
su_ndpa_err       :0
su_ndp_err        :0
mu_ndpa           :0
mu_ndp            :0
mu_brpoll_1       :0
mu_brpoll_2       :0
mu_bar_1          :0
mu_bar_2          :0
mu_cts2self       :0
mu_ndpa_err        :0
mu_ndp_err        :0
mu_brp1_err       :0
mu_brp2_err       :0
```

Output interpretation

- **su_ndpa:** Number of times SU-NDPA frames transmitted
- **su_ndp:** Number of times SU-NDP frames transmitted
- **su_bar:** Number of times the BAR frames transmitted in SU seq to flush BA state; SU-BAR is typically used to reset the BA window state on the peer
- **su_cts2self:** Number of times the CTS2SELf frames to extend SU-data burst
- **su_ndpa_err:** Number of times the SU-NDPA frames not transmitted due to hardware pause
- **su_ndp_err:** Number of times the SU-NDP frames that didn't receive correct CBF
- **mu_ndpa:** Number of times the MU-NDPA frames transmitted
- **mu_ndp:** Number of times the MU-NDP frames transmitted
- **mu_brpoll_1:** Number of times the BRPOLL frames transmitted to second user
- **mu_brpoll_2:** Number of times the BRPOLL frames transmitted to third user
- **mu_bar_1:** Number of times the BAR frames to second user in MU-PPDU
- **mu_bar_2:** Number of times the BAR frames to third user in MU-PPDU
- **mu_cts2self :** Number of times the CTS2SELF frames to extend MU-data burst
- **mu_ndpa_err:** Number of times the MU-NDPA frames not transmitted due to hardware pause
- **mu_ndp_err:** Number of times the MU-NDP frames that did not receive correct CBF

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT hai.yafeng@ute.com.cn

- **mu_brp1_err:** Number of times the BRPOLL frames to second user that did not receive correct CBF
- **mu_brp2_err:** Number of times the BRPOLL frames to third user that did not receive correct CBF

3.22.11 Tx multi-user info statistics

```
root@OpenWrt:/# iwpriv ath0 txrx_fw_stats 17
TX_MU Info:
mu_sch_nusers_2         :0
mu_sch_nusers_3         :0
mu_mpdus_queued_usr0    :0
mu_mpdus_queued_usr1    :0
mu_mpdus_queued_usr2    :0
mu_mpdus_queued_usr3    :0
mu_mpdus_tried_usr0     :0
mu_mpdus_tried_usr1     :0
mu_mpdus_tried_usr2     :0
mu_mpdus_tried_usr3     :0
mu_mpdus_failed_usr0    :0
mu_mpdus_failed_usr1    :0
mu_mpdus_failed_usr2    :0
mu_mpdus_failed_usr3    :0
mu_mpdus_requeued_usr0  :0
mu_mpdus_requeued_usr1  :0
mu_mpdus_requeued_usr2  :0
mu_mpdus_requeued_usr3  :0
mu_err_no_ba_usr0       :0
mu_err_no_ba_usr1       :0
mu_err_no_ba_usr2       :0
mu_err_no_ba_usr3       :0
mu_mpdu_underrun_usr0   :0
mu_mpdu_underrun_usr1   :0
mu_mpdu_underrun_usr2   :0
mu_mpdu_underrun_usr3   :0
mu_ampdu_underrun_usr0  :0
mu_ampdu_underrun_usr1  :0
mu_ampdu_underrun_usr2  :0
mu_ampdu_underrun_usr3  :0
```

3.22.12 SIFS response statistics

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

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

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

```
SIFS RESP TX stats:

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

Output interpretation:

- **s-poll trigger:**0 Number of times the s-poll trigger frames
- **u-apsd trigger:0** Number of times the uapsd trigger frames
- **qboost trigger data [exp]:**0 Number of times the data trigger frames; idx 0: explicit and idx 1: implicit
- **qboost trigger bar [exp]:**0 Number of times the bar trigger frames; idx 0: explicit and idx p1: implicit SIFS RESP Tx stats:
- **SIFS response data:**0 Number of times the ppdus transmitted at SIFS interval
- **SIFS response timing err:** 0 Number of times the ppdus failed to meet SIFS resp timing

3.22.13 Verify the resets due to different scenarios

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

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

Output interpretation

- **warm reset**: Number of warm resets from reboot
- **cold reset:** Number of cold resets from reboot
- **tx flush:** Number of resets to recover from Tx hang
- **tx glb reset:** Number of resets because of Tx queue timeout
- **tx txq reset:** Number of resets because of Tx hw stuck
- **rx timeout reset:** Number of resets because of Rx hardware timeout
- **hw status mismatch:** Number of mismatches between status and schedule id in hardware scheduler
- **hw status multi mismatch:** Number of resets because status and schedule id out of sync in hardware scheduler

Qualcomm Confidential - May Contain Trade Secrets 2021-06-13 03:57:44 PDT zan.yafeng@zte.com.cn

3.22.14 Clear firmware statistics

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

Execute the following command:

```
iwpriv ath0 txrx_fw_st_rst < mask>
```

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

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

3.22.15 Display RSSI for IPQ807x/IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx/QCN90xx or QCN602x + IPQ807xA/IPQ60xx/IPQ50xx platforms

For IPQ807x/IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx/QCN90xx or QCN602x + IPQ807xA/IPQ60xx/IPQ50xx platforms, the real signal strength, in dBm units, is displayed when a CLI command, wifistats wifiX 10, is issued by the user in the mission mode.

In FTM mode, the computed RSSI (dBm) value is displayed as part of the Rx report. The computation is based on the RxGain Calibration and noise floor (NF) categorization. NF categorization gives the correct NF value in dBr units and RxGain Calibration computes NF in dBm units. This data is used in the calculation of RSSI (dBm) in FTM mode. The same data can be used in mission mode also for the signal strength in dBm conversion.

Calculate the RSSI (dBm) for the current packet (either data/management) and send it to the host as part of htt_rx_pdev_rate_stats_tlv. Rssi (dBm) using the following formula:

RSSI (dBm) = RSSI (rx Descriptor value) + NF dBm + rxGainOffset(BDF)

+ Bandwidth Offset - Chainmask Offset

rssi_in_dbm is displayed when the `wifistats wifiX 10` command is entered.

Configure promiscuous mode

To set promiscuous mode, enter the iwpriv wifi0 promisc 1 command.

To display the status of promiscuous mode, enter the iwpriv wifi0 get_promisc command.

To measure RSSI in AP being promiscuous mode and generate ISS using LitePoint,, do the following:

1. Bring up the AP.
2. Set promiscuous mode in AP using the iwpriv wifi0 promisc 1 command.
3. Use LP/CMW500 to generate Waveform with AP center frequency.
4. Mention negative power level in the VSG settings (as no attenuator is added in this setup)
5. Enter the wifistats wifiX 10 command to display the measured RSSI (dBm)stats

   The generated ISS and measured RSSI(dBm) power will be approximation of +or -3 dbm

Run traffic in WDS setup

To run traffic in WDS setup and capture the power for various rates and chainmask, do the following:

1. Bring up one board as AP, and another board as STA with previously mentioned configuration
2. After the interface is associated, run the traffic by forcing various rates and chainmasks and measure the RSSI in dBm varying across different MCS

   RSSI in dBm must vary across the MCS, it should not be same for all the MCS rates

Sample AP configuration

```
config wifi-device  wifi0
        option type     qcawificfg80211
        option channel  36
        option macaddr  00:03:7f:12:f4:a7
        option hwmode   11axa
        option htmode   HT80
        # REMOVE THIS LINE TO ENABLE WIFI:
        option disabled 0

config wifi-iface
        option device   wifi0
        option network  lan
        option mode     ap
        option ssid     OpenWrt
       option encryption none
```

Sample STA configuration

```
config wifi-device  wifi0
        option type     qcawificfg80211
        option channel  auto
        option macaddr  00:03:7f:12:a8:6f
        option hwmode   11axa
        option htmode   HT80
        # REMOVE THIS LINE TO ENABLE WIFI:
        option disabled 0

config wifi-iface
        option device   wifi0
        option network  lan
        option mode     sta
        option wds      1
        option ssid     OpenWrt
        option encryption none
```

3.23 Tx and Rx statistics

The *iwpriv txrx_stats* command provides extended set of statistics related to Tx and Rx data path functionalities, and also serves as a debugging tool for engineers (Host SW, FW, and CEs) to analyze and troubleshoot data path-related problems (such as packet drops, low throughput, rate, and high PER).

The iwpriv txrx_stats command in the WLAN driver, version 11.0, replaces the iwpriv txrx_fw_stats command that was available in the WLAN driver, version 10.4. The txrx_stats parameter provides support for following statistics

- Data path statistics from firmware
- Data path statistics that Host extracts from per-MSDU H/W data structures.

As a general guideline for debug, firmware statistics can be used to debug or get any statistics at per-PPDU level (802.11 frames), and Host statistics can be used for debug issues at per-MSDU level.

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Using statistics for debugging* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

The wifistats wifiN <stats#> commands are an alternative for existing DP statistics commands, “iwpriv athX txrx_stats <1-19>” and “iwpriv wifiX dp_peer_stats”, with the difference being that the statistics are displayed in user space instead of being logged in the kernel.

The following table lists the parameters available with the Tx and Rx statistics-related commands:

Table 3-25 Tx and Rx statistics parameters

iwpriv athX txrx_stats <1-16>	Firmware radio level Statistics
iwpriv athX txrx_stats <257-262>	Host radio level Statistics
iwpriv wifiX dp_peer_stats <mac_addr>:0x3	Firmware per-peer Statistics
iwpriv wifiX fc_peer_stats <mac_addr>	Host per-peer Statistics

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:41 PDT pan.yafeng@zte.com.cn

3.23.1 Sample output of txrx_stats command

3.23.1.1 Host Radio level/PDEV Tx statistics

```
root@OpenWrt:/# iwpriv ath0 txrx_stats 261

[  958.432318] wlan: [3199: F:DP] PDEV Tx Stats:
[  958.432318]
[  958.432347] wlan: [3199: F:DP] Received From HW (Per Rx Ring):
[  958.437086] wlan: [3199: F:DP] Packets = 15
[  958.441858] wlan: [3199: F:DP] Bytes = 1290
[  958.446368] wlan: [3199: F:DP] Processed:
[  958.450599] wlan: [3199: F:DP] Packets = 15
[  958.454634] wlan: [3199: F:DP] Bytes = 1290
[  958.458848] wlan: [3199: F:DP] Completions:
[  958.463135] wlan: [3199: F:DP] Packets = 15
[  958.467001] wlan: [3199: F:DP] Bytes = 1320
[  958.471510] wlan: [3199: F:DP] Dropped:
[  958.475803] wlan: [3199: F:DP] Total = 0
[  958.479316] wlan: [3199: F:DP] Dma_map_error = 0
[  958.483533] wlan: [3199: F:DP] Ring Full = 0
[  958.488192] wlan: [3199: F:DP] Descriptor Not available = 0
[  958.492559] wlan: [3199: F:DP] HW enqueue failed= 0
[  958.498242] wlan: [3199: F:DP] Resources Full = 0
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
[  958.503215] wlan: [3199: F:DP] FW removed = 0
[  958.507823] wlan: [3199: F:DP] FW removed transmitted = 0
[  958.512244] wlan: [3199: F:DP] FW removed untransmitted = 0
[  958.517703] wlan: [3199: F:DP] Aged Out from msdu/mpdu queues = 0
[  958.523513] wlan: [3199: F:DP] Scatter Gather:
[  958.529504] wlan: [3199: F:DP] Packets = 0
[  958.533947] wlan: [3199: F:DP] Bytes = 0
[  958.538011] wlan: [3199: F:DP] Dropped By Host = 0
[  958.542204] wlan: [3199: F:DP] Dropped By Target = 0
[  958.546852] wlan: [3199: F:DP] TSO:
[  958.552082] wlan: [3199: F:DP] Number of Segments = 0
[  958.555419] wlan: [3199: F:DP] Packets = 0
[  958.560625] wlan: [3199: F:DP] Bytes = 0
[  958.564973] wlan: [3199: F:DP] Dropped By Host = 0
[  958.568835] wlan: [3199: F:DP] Mcast Enhancement:
[  958.573516] wlan: [3199: F:DP] Packets = 0
[  958.578379] wlan: [3199: F:DP] Bytes = 0
[  958.582383] wlan: [3199: F:DP] Dropped: Map Errors = 0
[  958.586543] wlan: [3199: F:DP] Dropped: Self Mac = 0
[  958.591589] wlan: [3199: F:DP] Dropped: Send Fail = 0
[  958.596780] wlan: [3199: F:DP] Unicast sent = 0
[  958.601825] wlan: [3199: F:DP] Raw:
[  958.606607] wlan: [3199: F:DP] Packets = 0
[  958.609800] wlan: [3199: F:DP] Bytes = 0
[  958.613963] wlan: [3199: F:DP] DMA map error = 0
[  958.618134] wlan: [3199: F:DP] Reinjected:
[  958.622813] wlan: [3199: F:DP] Packets = 0
[  958.626815] wlan: [3199: F:DP] Bytes = 0
[  958.626815]
[  958.630889] wlan: [3199: F:DP] Inspected:
[  958.636579] wlan: [3199: F:DP] Packets = 0
[  958.640444] wlan: [3199: F:DP] Bytes = 0
[  958.644429] wlan: [3199: F:DP] SOC Tx Stats:
[  958.644429]
[  958.648606] wlan: [3199: F:DP] Tx Descriptors In Use = 0
[  958.654356] wlan: [3199: F:DP] Invalid peer:
[  958.659831] wlan: [3199: F:DP] Packets = 0
[  958.663961] wlan: [3199: F:DP] Bytes = 0
[  958.667874] wlan: [3199: F:DP] Packets dropped due to TCL ring full = 0 0 0
```

3.23.1.2 Host Radio level/PDEV Rx statistics

```
root@OpenWrt:/# iwpriv ath0 txrx_stats 260
[  963.283152] wlan: [3200: F:DP] PDEV Rx Stats:
[  963.283182] wlan: [3200: F:DP] Received From HW (Per Rx Ring):
[  963.287913] wlan: [3200: F:DP] Packets = 0 0 0 0
[  963.293770] wlan: [3200: F:DP] Bytes = 0 0 0 0
[  963.298485] wlan: [3200: F:DP] Replenished:
[  963.302825] wlan: [3200: F:DP] Packets = 1078
[  963.306988] wlan: [3200: F:DP] Bytes = 2207744
[  963.311528] wlan: [3200: F:DP] Buffers Added To Freelist = 1
[  963.315943] wlan: [3200: F:DP] Dropped:
[  963.321826] wlan: [3200: F:DP] msdu_not_done = 0
[  963.325479] wlan: [3200: F:DP] Sent To Stack:
[  963.330324] wlan: [3200: F:DP] Packets = 0
[  963.334695] wlan: [3200: F:DP] Bytes = 0
[  963.338672] wlan: [3200: F:DP] Multicast/Broadcast:
```

```
[  963.342867] wlan: [3200: F:DP] Packets = 0
[  963.347527] wlan: [3200: F:DP] Bytes = 0
[  963.351678] wlan: [3200: F:DP] Errors:
[  963.355902] wlan: [3200: F:DP] Rxdma Ring Un-inititalized = 0
[  963.359551] wlan: [3200: F:DP] Desc Alloc Failed: = 0
[  963.365332] wlan: [3200: F:DP] SOC Rx Stats:
[  963.365332]
[  963.370543] wlan: [3200: F:DP] Errors:
[  963.370543]
[  963.376274] wlan: [3200: F:DP] Rx Decrypt Errors = 0
[  963.381303] wlan: [3200: F:DP] Invalid RBM = 0
[  963.386510] wlan: [3200: F:DP] Invalid Vdev = 0
[  963.390773] wlan: [3200: F:DP] Invalid Pdev = 0
[  963.395177] wlan: [3200: F:DP] Invalid Peer = 0
[  963.399698] wlan: [3200: F:DP] HAL Ring Access Fail = 0
[  963.404247] wlan: [3200: F:DP] RXDMA Error (0-31): 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
[  963.409436] wlan: [3200: F:DP] REO Error(0-14): 55 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

3.23.1.3 Host Tx Rate statistics

```
root@OpenWrt:/# iwpriv ath0 txrx_stats 259
[  953.249958] wlan: [3198: F:DP] Tx Rate Info:
[  953.249958]
[  953.249992] wlan: [3198: F:DP] CCK 11 Mbps Long   = 0
[  953.254733] wlan: [3198: F:DP] CCK 5.5 Mbps Long  = 0
[  953.259769] wlan: [3198: F:DP] CCK 2 Mbps Long    = 0
[  953.264861] wlan: [3198: F:DP] CCK 1 Mbps Long    = 0
[  953.269978] wlan: [3198: F:DP] CCK 11 Mbps Short  = 0
[  953.275107] wlan: [3198: F:DP] CCK 5.5 Mbps Short = 0
[  953.280262] wlan: [3198: F:DP] CCK 2 Mbps Short   = 0
[  953.285354] wlan: [3198: F:DP]
[  953.285354]
[  953.290462] wlan: [3198: F:DP] OFDM 48 Mbps = 0
[  953.295232] wlan: [3198: F:DP] OFDM 24 Mbps = 0
[  953.299856] wlan: [3198: F:DP] OFDM 12 Mbps = 0
[  953.304446] wlan: [3198: F:DP] OFDM 6 Mbps  = 0
[  953.309055] wlan: [3198: F:DP] OFDM 54 Mbps = 0
[  953.313634] wlan: [3198: F:DP] OFDM 36 Mbps = 0
[  953.318240] wlan: [3198: F:DP] OFDM 18 Mbps = 0
[  953.322834] wlan: [3198: F:DP] OFDM 9 Mbps  = 0
[  953.327473] wlan: [3198: F:DP]
[  953.327473]
[  953.332250] wlan: [3198: F:DP] HT MCS 0 (BPSK 1/2)   = 0
[  953.336629] wlan: [3198: F:DP] HT MCS 1 (QPSK 1/2)   = 0
[  953.341949] wlan: [3198: F:DP] HT MCS 2 (QPSK 3/4)   = 0
[  953.347306] wlan: [3198: F:DP] HT MCS 3 (16-QAM 1/2) = 0
[  953.352745] wlan: [3198: F:DP] HT MCS 4 (16-QAM 3/4) = 0
[  953.358102] wlan: [3198: F:DP] HT MCS 5 (64-QAM 2/3) = 0
[  953.363462] wlan: [3198: F:DP] HT MCS 6 (64-QAM 3/4) = 0
[  953.368851] wlan: [3198: F:DP] HT MCS 7 (64-QAM 5/6) = 0
[  953.374272] wlan: [3198: F:DP]
[  953.374272]
[  953.379596] wlan: [3198: F:DP] VHT MCS 0 (BPSK 1/2)      = 0
[  953.384309] wlan: [3198: F:DP] VHT MCS 1 (QPSK 1/2)      = 0
[  953.389943] wlan: [3198: F:DP] VHT MCS 2 (QPSK 3/4)      = 0
[  953.395672] wlan: [3198: F:DP] VHT MCS 3 (16-QAM 1/2)    = 0
```

```
[  953.401415] wlan: [3198: F:DP] VHT MCS 4 (16-QAM 3/4)    = 0
[  953.407200] wlan: [3198: F:DP] VHT MCS 5 (64-QAM 2/3)    = 6
[  953.412854] wlan: [3198: F:DP] VHT MCS 6 (64-QAM 3/4)    = 0
[  953.418590] wlan: [3198: F:DP] VHT MCS 7 (64-QAM 5/6)    = 9
[  953.424359] wlan: [3198: F:DP] VHT MCS 8 (256-QAM 3/4)   = 0
[  953.430033] wlan: [3198: F:DP] VHT MCS 9 (256-QAM 5/6)   = 0
[  953.435837] wlan: [3198: F:DP] VHT MCS 10 (1024-QAM 3/4) = 0
[  953.441517] wlan: [3198: F:DP] VHT MCS 10 (1024-QAM 5/6) = 0
[  953.447252] wlan: [3198: F:DP]
[  953.447252]
[  953.452981] wlan: [3198: F:DP] HE MCS 0 (BPSK 1/2)      = 0
[  953.457652] wlan: [3198: F:DP] HE MCS 1 (QPSK 1/2)      = 0
[  953.463303] wlan: [3198: F:DP] HE MCS 2 (QPSK 3/4)      = 0
[  953.468955] wlan: [3198: F:DP] HE MCS 3 (16-QAM 1/2)    = 0
[  953.474587] wlan: [3198: F:DP] HE MCS 4 (16-QAM 3/4)    = 0
[  953.480221] wlan: [3198: F:DP] HE MCS 5 (64-QAM 2/3)    = 0
[  953.485883] wlan: [3198: F:DP] HE MCS 6 (64-QAM 3/4)    = 0
[  953.491508] wlan: [3198: F:DP] HE MCS 7 (64-QAM 5/6)    = 0
[  953.497176] wlan: [3198: F:DP] HE MCS 8 (256-QAM 3/4)   = 0
[  953.502802] wlan: [3198: F:DP] HE MCS 9 (256-QAM 5/6)   = 0
[  953.508483] wlan: [3198: F:DP] HE MCS 10 (1024-QAM 3/4) = 0
[  953.514084] wlan: [3198: F:DP] HE MCS 10 (1024-QAM 5/6) = 0
[  953.519741] wlan: [3198: F:DP]
[  953.519741]
[  953.525368] wlan: [3198: F:DP] SGI = 0.8us 0 0.4us 15 1.6us 0 3.2us 0
[  953.529973] wlan: [3198: F:DP] BW Counts = 20MHZ 0, 40MHZ 0, 80MHZ 15, 160MHZ
0
[  953.536295] wlan: [3198: F:DP] OFDMA = 0
[  953.543465] wlan: [3198: F:DP] STBC = 0
[  953.547592] wlan: [3198: F:DP] LDPC = 15
[  953.551138] wlan: [3198: F:DP] Retries = 3
[  953.555314] wlan: [3198: F:DP] Last ack rssi = 0
[  953.555314]
[  953.559200] wlan: [3198: F:DP] Aggregation:
[  953.559200]
[  953.565510] wlan: [3198: F:DP] Number of Msdu's Part of Amsdu = 0
[  953.571108] wlan: [3198: F:DP] Number of Msdu's With No Msdu Level Aggregation
= 0
```

3.23.1.4 Host Rx Rate statistics

```
root@OpenWrt:/# iwpriv ath0 txrx_stats 258
[  947.451829] wlan: [3197: F:DP] Rx Rate Info:
[  947.451829]
[  947.451861] wlan: [3197: F:DP] CCK 11 Mbps Long   = 0
[  947.456549] wlan: [3197: F:DP] CCK 5.5 Mbps Long  = 0
[  947.461583] wlan: [3197: F:DP] CCK 2 Mbps Long    = 0
[  947.466717] wlan: [3197: F:DP] CCK 1 Mbps Long    = 0
[  947.471836] wlan: [3197: F:DP] CCK 11 Mbps Short  = 0
[  947.476949] wlan: [3197: F:DP] CCK 5.5 Mbps Short = 0
[  947.482079] wlan: [3197: F:DP] CCK 2 Mbps Short   = 0
[  947.487190] wlan: [3197: F:DP]
[  947.487190]
[  947.492321] wlan: [3197: F:DP] OFDM 48 Mbps = 0
[  947.497086] wlan: [3197: F:DP] OFDM 24 Mbps = 0
[  947.501687] wlan: [3197: F:DP] OFDM 12 Mbps = 0
[  947.506297] wlan: [3197: F:DP] OFDM 6 Mbps  = 0
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
[  947.510889] wlan: [3197: F:DP] OFDM 54 Mbps = 0
[  947.515496] wlan: [3197: F:DP] OFDM 36 Mbps = 0
[  947.520090] wlan: [3197: F:DP] OFDM 18 Mbps = 0
[  947.524697] wlan: [3197: F:DP] OFDM 9 Mbps  = 0
[  947.529291] wlan: [3197: F:DP]
[  947.529291]
[  947.533900] wlan: [3197: F:DP] HT MCS 0 (BPSK 1/2)   = 0
[  947.538495] wlan: [3197: F:DP] HT MCS 1 (QPSK 1/2)   = 0
[  947.543799] wlan: [3197: F:DP] HT MCS 2 (QPSK 3/4)   = 0
[  947.549172] wlan: [3197: F:DP] HT MCS 3 (16-QAM 1/2) = 0
[  947.554561] wlan: [3197: F:DP] HT MCS 4 (16-QAM 3/4) = 0
[  947.559935] wlan: [3197: F:DP] HT MCS 5 (64-QAM 2/3) = 0
[  947.565323] wlan: [3197: F:DP] HT MCS 6 (64-QAM 3/4) = 0
[  947.570700] wlan: [3197: F:DP] HT MCS 7 (64-QAM 5/6) = 0
[  947.576087] wlan: [3197: F:DP]
[  947.576087]
[  947.581462] wlan: [3197: F:DP] VHT MCS 0 (BPSK 1/2)      = 0
[  947.586157] wlan: [3197: F:DP] VHT MCS 1 (QPSK 1/2)      = 0
[  947.591801] wlan: [3197: F:DP] VHT MCS 2 (QPSK 3/4)      = 0
[  947.597523] wlan: [3197: F:DP] VHT MCS 3 (16-QAM 1/2)    = 0
[  947.603259] wlan: [3197: F:DP] VHT MCS 4 (16-QAM 3/4)    = 0
[  947.608981] wlan: [3197: F:DP] VHT MCS 5 (64-QAM 2/3)    = 0
[  947.614716] wlan: [3197: F:DP] VHT MCS 6 (64-QAM 3/4)    = 0
[  947.620440] wlan: [3197: F:DP] VHT MCS 7 (64-QAM 5/6)    = 0
[  947.626176] wlan: [3197: F:DP] VHT MCS 8 (256-QAM 3/4)   = 0
[  947.631905] wlan: [3197: F:DP] VHT MCS 9 (256-QAM 5/6)   = 0
[  947.637627] wlan: [3197: F:DP] VHT MCS 10 (1024-QAM 3/4) = 0
[  947.643363] wlan: [3197: F:DP] VHT MCS 10 (1024-QAM 5/6) = 0
[  947.649085] wlan: [3197: F:DP]
[  947.649085]
[  947.654820] wlan: [3197: F:DP] HE MCS 0 (BPSK 1/2)      = 0
[  947.659502] wlan: [3197: F:DP] HE MCS 1 (QPSK 1/2)      = 0
[  947.665151] wlan: [3197: F:DP] HE MCS 2 (QPSK 3/4)      = 0
[  947.670786] wlan: [3197: F:DP] HE MCS 3 (16-QAM 1/2)    = 0
[  947.676437] wlan: [3197: F:DP] HE MCS 4 (16-QAM 3/4)    = 0
[  947.682078] wlan: [3197: F:DP] HE MCS 5 (64-QAM 2/3)    = 0
[  947.687714] wlan: [3197: F:DP] HE MCS 6 (64-QAM 3/4)    = 0
[  947.693363] wlan: [3197: F:DP] HE MCS 7 (64-QAM 5/6)    = 0
[  947.698998] wlan: [3197: F:DP] HE MCS 8 (256-QAM 3/4)   = 0
[  947.704648] wlan: [3197: F:DP] HE MCS 9 (256-QAM 5/6)   = 0
[  947.710284] wlan: [3197: F:DP] HE MCS 10 (1024-QAM 3/4) = 0
[  947.715934] wlan: [3197: F:DP] HE MCS 10 (1024-QAM 5/6) = 0
[  947.721567] wlan: [3197: F:DP]
[  947.721567]
[  947.727216] wlan: [3197: F:DP] NSS(0-7) =  0 0 0 0 0 0 0 0
[  947.731819] wlan: [3197: F:DP] SGI = 0.8us 0, 0.4us 0, 1.6us 0, 3.2us 0,
[  947.737110] wlan: [3197: F:DP] BW Counts = 20MHZ 0, 40MHZ 0, 80MHZ 0, 160MHZ 0
[  947.743975] wlan: [3197: F:DP] Reception Type = SU: 0, MU_MIMO:0, MU_OFDMA:0,
MU_OFDMA_MIMO:0
[  947.743975]
[  947.751003] wlan: [3197: F:DP] Aggregation:
[  947.751003]
[  947.760985] wlan: [3197: F:DP] Number of Msdu's Part of Ampdus = 0
[  947.766801] wlan: [3197: F:DP] Number of Msdu's With No Mpdu Level Aggregation
: 0
[  947.772793] wlan: [3197: F:DP] Number of Msdu's Part of Amsdu: 0
[  947.780336] wlan: [3197: F:DP] Number of Msdu's With No Msdu Level
Aggregation: 0
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
root@OpenWrt:/#
```

3.23.1.5 Host peer/node statistics

```
root@OpenWrt:/# iwpriv wifi2 fc_peer_stats 00:03:7f:12:fe:6b
[1446967.025921] wlan: [26093:F:DP] Node Tx Stats:
[1446967.025921]
[1446967.025956] wlan: [26093:F:DP] Total Packet Completions = 0
[1446967.031087] wlan: [26093:F:DP] Total Bytes Completions = 0
[1446967.036817] wlan: [26093:F:DP] Success Packets = 0
[1446967.042579] wlan: [26093:F:DP] Success Bytes = 0
[1446967.047492] wlan: [26093:F:DP] Unicast Success Packets = 0
[1446967.052132] wlan: [26093:F:DP] Unicast Success Bytes = 0
[1446967.057922] wlan: [26093:F:DP] Multicast Success Packets = 0
[1446967.063233] wlan: [26093:F:DP] Multicast Success Bytes = 0
[1446967.069042] wlan: [26093:F:DP] Broadcast Success Packets = 0
[1446967.074865] wlan: [26093:F:DP] Broadcast Success Bytes = 0
[1446967.080507] wlan: [26093:F:DP] Packets Failed = 0
[1446967.086319] wlan: [26093:F:DP] Packets In OFDMA = 0
[1446967.090911] wlan: [26093:F:DP] Packets In STBC = 0
[1446967.096103] wlan: [26093:F:DP] Packets In LDPC = 0
[1446967.101152] wlan: [26093:F:DP] Packet Retries = 0
[1446967.106107] wlan: [26093:F:DP] MSDU's Part of AMSDU = 0
[1446967.110700] wlan: [26093:F:DP] Msdu's As Part of Ampdu = 0
[1446967.116244] wlan: [26093:F:DP] Msdu's As Ampdu = 0
[1446967.121984] wlan: [26093:F:DP] Last Packet RSSI = 0
[1446967.126920] wlan: [26093:F:DP] Dropped At FW: Removed Pkts = 0
[1446967.131883] wlan: [26093:F:DP] Dropped At FW: Removed transmitted = 0
[1446967.137947] wlan: [26093:F:DP] Dropped At FW: Removed Untransmitted = 0
[1446967.144296] wlan: [26093:F:DP] Dropped : Age Out = 0
[1446967.151234] wlan: [26093:F:DP] NAWDS :
[1446967.156257] wlan: [26093:F:DP] Nawds multicast Drop Tx Packet = 0
[1446967.160356] wlan: [26093:F:DP]     Nawds multicast  Tx Packet Count = 0
[1446967.166419] wlan: [26093:F:DP]     Nawds multicast  Tx Packet Bytes = 0
[1446967.173030] wlan: [26093:F:DP] Rate Info:
[1446967.179531] wlan: [26093:F:DP] MSDU Count
[1446967.183613] wlan: [26093:F:DP]     OFDM 48 Mbps = 0
[1446967.187794] wlan: [26093:F:DP]     OFDM 24 Mbps = 0
[1446967.192648] wlan: [26093:F:DP]     OFDM 12 Mbps = 0
[1446967.197404] wlan: [26093:F:DP]     OFDM 6 Mbps  = 0
[1446967.202190] wlan: [26093:F:DP]     OFDM 54 Mbps = 0
[1446967.206953] wlan: [26093:F:DP]     OFDM 36 Mbps = 0
[1446967.211761] wlan: [26093:F:DP]     OFDM 18 Mbps = 0
[1446967.216503] wlan: [26093:F:DP]     OFDM 9 Mbps  = 0
[1446967.221290] wlan: [26093:F:DP]     INVALID  = 0
[1446967.226050] wlan: [26093:F:DP]
[1446967.226050]
[1446967.230492] wlan: [26093:F:DP]     CCK 11 Mbps Long   = 0
[1446967.235600] wlan: [26093:F:DP]     CCK 5.5 Mbps Long  = 0
[1446967.240914] wlan: [26093:F:DP]     CCK 2 Mbps Long    = 0
[1446967.246192] wlan: [26093:F:DP]     CCK 1 Mbps Long    = 0
[1446967.251496] wlan: [26093:F:DP]     CCK 11 Mbps Short  = 0
[1446967.256782] wlan: [26093:F:DP]     CCK 5.5 Mbps Short = 0
[1446967.262109] wlan: [26093:F:DP]     CCK 2 Mbps Short   = 0
[1446967.267372] wlan: [26093:F:DP]     INVALID  = 0
[1446967.272675] wlan: [26093:F:DP]
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
[1446967.272675]
[1446967.276920] wlan: [26093:F:DP]     HT MCS 0 (BPSK 1/2)   = 0
[1446967.282052] wlan: [26093:F:DP]     HT MCS 1 (QPSK 1/2)   = 0
[1446967.287337] wlan: [26093:F:DP]     HT MCS 2 (QPSK 3/4)   = 0
[1446967.292925] wlan: [26093:F:DP]     HT MCS 3 (16-QAM 1/2) = 0
[1446967.298448] wlan: [26093:F:DP]     HT MCS 4 (16-QAM 3/4) = 0
[1446967.304016] wlan: [26093:F:DP]     HT MCS 5 (64-QAM 2/3) = 0
[1446967.309569] wlan: [26093:F:DP]     HT MCS 6 (64-QAM 3/4) = 0
[1446967.315136] wlan: [26093:F:DP]     HT MCS 7 (64-QAM 5/6) = 0
[1446967.320685] wlan: [26093:F:DP]     INVALID  = 0
[1446967.326222] wlan: [26093:F:DP]
[1446967.326222]
[1446967.330751] wlan: [26093:F:DP]     VHT MCS 0 (BPSK 1/2)      = 0
[1446967.335862] wlan: [26093:F:DP]     VHT MCS 1 (QPSK 1/2)      = 0
[1446967.341517] wlan: [26093:F:DP]     VHT MCS 2 (QPSK 3/4)      = 0
[1446967.347407] wlan: [26093:F:DP]     VHT MCS 3 (16-QAM 1/2)    = 0
[1446967.353321] wlan: [26093:F:DP]     VHT MCS 4 (16-QAM 3/4)    = 0
[1446967.359213] wlan: [26093:F:DP]     VHT MCS 5 (64-QAM 2/3)    = 0
[1446967.365148] wlan: [26093:F:DP]     VHT MCS 6 (64-QAM 3/4)    = 0
[1446967.371029] wlan: [26093:F:DP]     VHT MCS 7 (64-QAM 5/6)    = 0
[1446967.376921] wlan: [26093:F:DP]     VHT MCS 8 (256-QAM 3/4)   = 0
[1446967.382834] wlan: [26093:F:DP]     VHT MCS 9 (256-QAM 5/6)   = 0
[1446967.388727] wlan: [26093:F:DP]     VHT MCS 10 (1024-QAM 3/4) = 0
[1446967.394670] wlan: [26093:F:DP]     VHT MCS 11 (1024-QAM 5/6) = 0
[1446967.400544] wlan: [26093:F:DP]     INVALID  = 0
[1446967.406432] wlan: [26093:F:DP]
[1446967.406432]
[1446967.410960] wlan: [26093:F:DP]     HE MCS 0 (BPSK 1/2)      = 0
[1446967.416071] wlan: [26093:F:DP]     HE MCS 1 (QPSK 1/2)      = 0
[1446967.421744] wlan: [26093:F:DP]     HE MCS 2 (QPSK 3/4)      = 0
[1446967.427529] wlan: [26093:F:DP]     HE MCS 3 (16-QAM 1/2)    = 0
[1446967.433356] wlan: [26093:F:DP]     HE MCS 4 (16-QAM 3/4)    = 0
[1446967.439161] wlan: [26093:F:DP]     HE MCS 5 (64-QAM 2/3)    = 0
[1446967.444989] wlan: [26093:F:DP]     HE MCS 6 (64-QAM 3/4)    = 0
[1446967.450803] wlan: [26093:F:DP]     HE MCS 7 (64-QAM 5/6)    = 0
[1446967.456609] wlan: [26093:F:DP]     HE MCS 8 (256-QAM 3/4)   = 0
[1446967.462441] wlan: [26093:F:DP]     HE MCS 9 (256-QAM 5/6)   = 0
[1446967.468254] wlan: [26093:F:DP]     HE MCS 10 (1024-QAM 3/4) = 0
[1446967.474076] wlan: [26093:F:DP]     HE MCS 11 (1024-QAM 5/6) = 0
[1446967.479890] wlan: [26093:F:DP]     INVALID  = 0
[1446967.485685] wlan: [26093:F:DP]
[1446967.485685]
[1446967.490127] wlan: [26093:F:DP] SGI = 0.8us 0 0.4us 0 1.6us 0 3.2us 0
[1446967.495238] wlan: [26093:F:DP] Excess Retries per AC
[1446967.501613] wlan: [26093:F:DP]      Best effort = 0
[1446967.506694] wlan: [26093:F:DP]      Background= 0
[1446967.511568] wlan: [26093:F:DP]      Video = 0
[1446967.516327] wlan: [26093:F:DP]      Voice = 0
[1446967.520595] wlan: [26093:F:DP] BW Counts = 20MHZ 0 40MHZ 0 80MHZ 0 160MHZ 0
[1446967.520595]
[1446967.524870] wlan: [26093:F:DP] NSS(1-8) =  0 0 0 0 0 0 0 0
[1446967.533617] wlan: [26093:F:DP] Transmit Type :
[1446967.539253] wlan: [26093:F:DP] SU 0, MU_MIMO 0, MU_OFDMA 0, MU_MIMO_OFDMA 0
[1446967.543904] wlan: [26093:F:DP] User position list for GID 00->15: [ 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0]
[1446967.550957] wlan: [26093:F:DP] User position list for GID 16->31: [ 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0]
[1446967.560108] wlan: [26093:F:DP] User position list for GID 32->47: [ 0 0 0 0
```

```
0 0 0 0 0 0 0 0 0 0 0 0]
[1446967.569396] wlan: [26093:F:DP] User position list for GID 48->63: [ 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0]
[1446967.578684] wlan: [26093:F:DP] Last Packet RU index [0], Size [0]
[1446967.587958] wlan: [26093:F:DP] RU Locations RU[26 52 106 242 484 996]:
[1446967.594141] wlan: [26093:F:DP] RU_26:  0
[1446967.601062] wlan: [26093:F:DP] RU 52:  0
[1446967.604954] wlan: [26093:F:DP] RU 106: 0
[1446967.609036] wlan: [26093:F:DP] RU 242: 0
[1446967.613129] wlan: [26093:F:DP] RU 484: 0
[1446967.617195] wlan: [26093:F:DP] RU 996: 0
[1446967.621287] wlan: [26093:F:DP] Aggregation:
[1446967.625356] wlan: [26093:F:DP] Number of Msdu's Part of Amsdu = 0
[1446967.629797] wlan: [26093:F:DP] Number of Msdu's With No Msdu Level
Aggregation = 0
[1446967.635866] wlan: [26093:F:DP] Bytes and Packets transmitted  in last one
sec:
[1446967.643865] wlan: [26093:F:DP]     Bytes transmitted in last sec: 0
[1446967.651238] wlan: [26093:F:DP]     Data transmitted in last sec: 0
[1446967.657239] wlan: [26093:F:DP] Node Rx Stats:
[1446967.663390] wlan: [26093:F:DP] Packets Sent To Stack = 55
[1446967.667720] wlan: [26093:F:DP] Bytes Sent To Stack = 10630
[1446967.673285] wlan: [26093:F:DP] Ring Id = 0
[1446967.679195] wlan: [26093:F:DP]     Packets Received = 0
[1446967.683442] wlan: [26093:F:DP]     Bytes Received = 0
[1446967.688378] wlan: [26093:F:DP] Ring Id = 1
[1446967.693516] wlan: [26093:F:DP]     Packets Received = 0
[1446967.697754] wlan: [26093:F:DP]     Bytes Received = 0
[1446967.702710] wlan: [26093:F:DP] Ring Id = 2
[1446967.707821] wlan: [26093:F:DP]     Packets Received = 55
[1446967.712109] wlan: [26093:F:DP]     Bytes Received = 10630
[1446967.717026] wlan: [26093:F:DP] Ring Id = 3
[1446967.722589] wlan: [26093:F:DP]     Packets Received = 0
[1446967.726832] wlan: [26093:F:DP]     Bytes Received = 0
[1446967.731794] wlan: [26093:F:DP] Multicast Packets Received = 55
[1446967.736927] wlan: [26093:F:DP] Multicast Bytes Received = 10630
[1446967.742907] wlan: [26093:F:DP] Broadcast Packets Received = 17
[1446967.748883] wlan: [26093:F:DP] Broadcast Bytes Received = 3122
[1446967.754972] wlan: [26093:F:DP] Intra BSS Packets Received = 54
[1446967.760988] wlan: [26093:F:DP] Intra BSS Bytes Received = 10570
[1446967.766938] wlan: [26093:F:DP] Raw Packets Received = 0
[1446967.772941] wlan: [26093:F:DP] Raw Bytes Received = 0
[1446967.778307] wlan: [26093:F:DP] Errors: MIC Errors = 0
[1446967.783375] wlan: [26093:F:DP] Erros: Decryption Errors = 0
[1446967.788552] wlan: [26093:F:DP] Msdu's Received As Part of Ampdu = 56
[1446967.794470] wlan: [26093:F:DP] Msdu's Recived As Ampdu = 0
[1446967.800891] wlan: [26093:F:DP] Msdu's Received Not Part of Amsdu's = 55
[1446967.806697] wlan: [26093:F:DP] MSDUs Received As Part of Amsdu = 0
[1446967.813415] wlan: [26093:F:DP] NAWDS :
[1446967.819810] wlan: [26093:F:DP]     Nawds multicast Drop Rx Packet = 0
[1446967.823712] wlan: [26093:F:DP] SGI = 0.8us 55 0.4us 0 1.6us 0 3.2us 1
[1446967.830148] wlan: [26093:F:DP] BW Counts = 20MHZ 1 40MHZ 0 80MHZ 55 160MHZ 0
[1446967.836472] wlan: [26093:F:DP] MSDU Reception Type
[1446967.843799] wlan: [26093:F:DP] SU 56 MU_MIMO 0 MU_OFDMA 0 MU_OFDMA_MIMO 0
[1446967.848885] wlan: [26093:F:DP] PPDU Reception Type
[1446967.855580] wlan: [26093:F:DP] SU 56 MU_MIMO 0 MU_OFDMA 0 MU_OFDMA_MIMO 0
[1446967.860786] wlan: [26093:F:DP] MSDU Count
[1446967.867477] wlan: [26093:F:DP]     OFDM 48 Mbps = 0
```

```
[1446967.871637] wlan: [26093:F:DP]     OFDM 24 Mbps = 0
[1446967.876485] wlan: [26093:F:DP]     OFDM 12 Mbps = 0
[1446967.881270] wlan: [26093:F:DP]     OFDM 6 Mbps  = 0
[1446967.886036] wlan: [26093:F:DP]     OFDM 54 Mbps = 0
[1446967.890840] wlan: [26093:F:DP]     OFDM 36 Mbps = 0
[1446967.895582] wlan: [26093:F:DP]     OFDM 18 Mbps = 0
[1446967.900369] wlan: [26093:F:DP]     OFDM 9 Mbps  = 0
[1446967.905130] wlan: [26093:F:DP]     INVALID  = 0
[1446967.909917] wlan: [26093:F:DP]
[1446967.909917]
[1446967.914333] wlan: [26093:F:DP]     CCK 11 Mbps Long   = 0
[1446967.919486] wlan: [26093:F:DP]     CCK 5.5 Mbps Long  = 0
[1446967.924749] wlan: [26093:F:DP]     CCK 2 Mbps Long    = 0
[1446967.930058] wlan: [26093:F:DP]     CCK 1 Mbps Long    = 0
[1446967.935341] wlan: [26093:F:DP]     CCK 11 Mbps Short  = 0
[1446967.940646] wlan: [26093:F:DP]     CCK 5.5 Mbps Short = 0
[1446967.945951] wlan: [26093:F:DP]     CCK 2 Mbps Short   = 0
[1446967.951239] wlan: [26093:F:DP]     INVALID  = 0
[1446967.956518] wlan: [26093:F:DP]
[1446967.956518]
[1446967.960787] wlan: [26093:F:DP]     HT MCS 0 (BPSK 1/2)   = 0
[1446967.965910] wlan: [26093:F:DP]     HT MCS 1 (QPSK 1/2)   = 0
[1446967.971224] wlan: [26093:F:DP]     HT MCS 2 (QPSK 3/4)   = 0
[1446967.976746] wlan: [26093:F:DP]     HT MCS 3 (16-QAM 1/2) = 0
[1446967.982346] wlan: [26093:F:DP]     HT MCS 4 (16-QAM 3/4) = 0
[1446967.987858] wlan: [26093:F:DP]     HT MCS 5 (64-QAM 2/3) = 0
[1446967.993459] wlan: [26093:F:DP]     HT MCS 6 (64-QAM 3/4) = 0
[1446967.998970] wlan: [26093:F:DP]     HT MCS 7 (64-QAM 5/6) = 0
[1446968.004536] wlan: [26093:F:DP]     INVALID  = 0
[1446968.010088] wlan: [26093:F:DP]
[1446968.010088]
[1446968.014593] wlan: [26093:F:DP]     VHT MCS 0 (BPSK 1/2)      = 0
[1446968.019737] wlan: [26093:F:DP]     VHT MCS 1 (QPSK 1/2)      = 0
[1446968.025384] wlan: [26093:F:DP]     VHT MCS 2 (QPSK 3/4)      = 0
[1446968.031276] wlan: [26093:F:DP]     VHT MCS 3 (16-QAM 1/2)    = 0
[1446968.037165] wlan: [26093:F:DP]     VHT MCS 4 (16-QAM 3/4)    = 0
[1446968.043067] wlan: [26093:F:DP]     VHT MCS 5 (64-QAM 2/3)    = 0
[1446968.048978] wlan: [26093:F:DP]     VHT MCS 6 (64-QAM 3/4)    = 0
[1446968.054875] wlan: [26093:F:DP]     VHT MCS 7 (64-QAM 5/6)    = 0
[1446968.060775] wlan: [26093:F:DP]     VHT MCS 8 (256-QAM 3/4)   = 0
[1446968.066671] wlan: [26093:F:DP]     VHT MCS 9 (256-QAM 5/6)   = 0
[1446968.072594] wlan: [26093:F:DP]     VHT MCS 10 (1024-QAM 3/4) = 0
[1446968.078477] wlan: [26093:F:DP]     VHT MCS 11 (1024-QAM 5/6) = 0
[1446968.084388] wlan: [26093:F:DP]     INVALID  = 0
[1446968.090286] wlan: [26093:F:DP]
[1446968.090286]
[1446968.094796] wlan: [26093:F:DP]     HE MCS 0 (BPSK 1/2)      = 1
[1446968.099938] wlan: [26093:F:DP]     HE MCS 1 (QPSK 1/2)      = 0
[1446968.105561] wlan: [26093:F:DP]     HE MCS 2 (QPSK 3/4)      = 0
[1446968.111386] wlan: [26093:F:DP]     HE MCS 3 (16-QAM 1/2)    = 0
[1446968.117192] wlan: [26093:F:DP]     HE MCS 4 (16-QAM 3/4)    = 0
[1446968.123014] wlan: [26093:F:DP]     HE MCS 5 (64-QAM 2/3)    = 0
[1446968.128839] wlan: [26093:F:DP]     HE MCS 6 (64-QAM 3/4)    = 4
[1446968.134666] wlan: [26093:F:DP]     HE MCS 7 (64-QAM 5/6)    = 8
[1446968.140464] wlan: [26093:F:DP]     HE MCS 8 (256-QAM 3/4)   = 6
[1446968.146272] wlan: [26093:F:DP]     HE MCS 9 (256-QAM 5/6)   = 5
[1446968.152111] wlan: [26093:F:DP]     HE MCS 10 (1024-QAM 3/4) = 5
[1446968.157905] wlan: [26093:F:DP]     HE MCS 11 (1024-QAM 5/6) = 27
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
[1446968.163726] wlan: [26093:F:DP]     INVALID  = 0
[1446968.169540] wlan: [26093:F:DP]
[1446968.169540]
[1446968.174050] wlan: [26093:F:DP] PPDU Count
[1446968.179184] wlan: [26093:F:DP]     HE MCS 0 (BPSK 1/2)      = 1
[1446968.183087] wlan: [26093:F:DP]     HE MCS 1 (QPSK 1/2)      = 0
[1446968.188983] wlan: [26093:F:DP]     HE MCS 2 (QPSK 3/4)      = 0
[1446968.194804] wlan: [26093:F:DP]     HE MCS 3 (16-QAM 1/2)    = 0
[1446968.200632] wlan: [26093:F:DP]     HE MCS 4 (16-QAM 3/4)    = 0
[1446968.206428] wlan: [26093:F:DP]     HE MCS 5 (64-QAM 2/3)    = 0
[1446968.212251] wlan: [26093:F:DP]     HE MCS 6 (64-QAM 3/4)    = 4
[1446968.218061] wlan: [26093:F:DP]     HE MCS 7 (64-QAM 5/6)    = 8
[1446968.223883] wlan: [26093:F:DP]     HE MCS 8 (256-QAM 3/4)   = 6
[1446968.229698] wlan: [26093:F:DP]     HE MCS 9 (256-QAM 5/6)   = 5
[1446968.235521] wlan: [26093:F:DP]     HE MCS 10 (1024-QAM 3/4) = 5
[1446968.241331] wlan: [26093:F:DP]     HE MCS 11 (1024-QAM 5/6) = 27
[1446968.247140] wlan: [26093:F:DP]     INVALID  = 0
[1446968.252960] wlan: [26093:F:DP]
[1446968.252960]
[1446968.257480] wlan: [26093:F:DP] PPDU Count
[1446968.262595] wlan: [26093:F:DP]     HE MU-MIMO MCS 0 (BPSK 1/2)      = 0
[1446968.266500] wlan: [26093:F:DP]     HE MU-MIMO MCS 1 (QPSK 1/2)      = 0
[1446968.273103] wlan: [26093:F:DP]     HE MU-MIMO MCS 2 (QPSK 3/4)      = 0
[1446968.279628] wlan: [26093:F:DP]     HE MU-MIMO MCS 3 (16-QAM 1/2)    = 0
[1446968.286119] wlan: [26093:F:DP]     HE MU-MIMO MCS 4 (16-QAM 3/4)    = 0
[1446968.292635] wlan: [26093:F:DP]     HE MU-MIMO MCS 5 (64-QAM 2/3)    = 0
[1446968.299139] wlan: [26093:F:DP]     HE MU-MIMO MCS 6 (64-QAM 3/4)    = 0
[1446968.305668] wlan: [26093:F:DP]     HE MU-MIMO MCS 7 (64-QAM 5/6)    = 0
[1446968.312166] wlan: [26093:F:DP]     HE MU-MIMO MCS 8 (256-QAM 3/4)   = 0
[1446968.318670] wlan: [26093:F:DP]     HE MU-MIMO MCS 9 (256-QAM 5/6)   = 0
[1446968.325186] wlan: [26093:F:DP]     HE MU-MIMO MCS 10 (1024-QAM 3/4) = 0
[1446968.331697] wlan: [26093:F:DP]     HE MU-MIMO MCS 11 (1024-QAM 5/6) = 0
[1446968.338215] wlan: [26093:F:DP]     INVALID  = 0
[1446968.344713] wlan: [26093:F:DP]
[1446968.344713]
[1446968.349138] wlan: [26093:F:DP]     HE OFDMA MCS 0 (BPSK 1/2)      = 0
[1446968.354267] wlan: [26093:F:DP]     HE OFDMA MCS 1 (QPSK 1/2)      = 0
[1446968.360613] wlan: [26093:F:DP]     HE OFDMA MCS 2 (QPSK 3/4)      = 0
[1446968.366934] wlan: [26093:F:DP]     HE OFDMA MCS 3 (16-QAM 1/2)    = 0
[1446968.373277] wlan: [26093:F:DP]     HE OFDMA MCS 4 (16-QAM 3/4)    = 0
[1446968.379613] wlan: [26093:F:DP]     HE OFDMA MCS 5 (64-QAM 2/3)    = 0
[1446968.385944] wlan: [26093:F:DP]     HE OFDMA MCS 6 (64-QAM 3/4)    = 0
[1446968.392287] wlan: [26093:F:DP]     HE OFDMA MCS 7 (64-QAM 5/6)    = 0
[1446968.398618] wlan: [26093:F:DP]     HE OFDMA MCS 8 (256-QAM 3/4)   = 0
[1446968.404960] wlan: [26093:F:DP]     HE OFDMA MCS 9 (256-QAM 5/6)   = 0
[1446968.411310] wlan: [26093:F:DP]     HE OFDMA MCS 10 (1024-QAM 3/4) = 0
[1446968.417628] wlan: [26093:F:DP]     HE OFDMA MCS 11 (1024-QAM 5/6) = 0
[1446968.423972] wlan: [26093:F:DP]     INVALID  = 0
[1446968.430304] wlan: [26093:F:DP]
[1446968.430304]
[1446968.434554] wlan: [26093:F:DP] MSDU Count
[1446968.439689] wlan: [26093:F:DP]     NSS(1-8) =  1 0 0 55 0 0 0 0
[1446968.443581] wlan: [26093:F:DP] reception mode SU
[1446968.449491] wlan: [26093:F:DP]     PPDU Count
[1446968.454255] wlan: [26093:F:DP]     NSS(1-8) =  1 0 0 55 0 0 0 0
[1446968.458687] wlan: [26093:F:DP]     MPDU OK = 56, MPDU Fail = 0
[1446968.464338] wlan: [26093:F:DP] reception mode MU MIMO
[1446968.470153] wlan: [26093:F:DP]     PPDU Count
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
[1446968.475175] wlan: [26093:F:DP]     NSS(1-8) =  0 0 0 0 0 0 0 0
[1446968.479698] wlan: [26093:F:DP]     MPDU OK = 0, MPDU Fail = 0
[1446968.485336] wlan: [26093:F:DP] reception mode MU OFDMA
[1446968.491091] wlan: [26093:F:DP]     PPDU Count
[1446968.496357] wlan: [26093:F:DP]     NSS(1-8) =  0 0 0 0 0 0 0 0
[1446968.500619] wlan: [26093:F:DP]     MPDU OK = 0, MPDU Fail = 0
[1446968.506255] wlan: [26093:F:DP] Aggregation:
[1446968.511988] wlan: [26093:F:DP]     Msdu's Part of Ampdu = 0
[1446968.516250] wlan: [26093:F:DP]     Msdu's With No Mpdu Level Aggregation =
56
[1446968.521628] wlan: [26093:F:DP]     Msdu's Part of Amsdu = 0
[1446968.528824] wlan: [26093:F:DP]     Msdu's With No Msdu Level Aggregation =
55
[1446968.534128] wlan: [26093:F:DP] Bytes and Packets received in last one sec:
[1446968.541335] wlan: [26093:F:DP]     Bytes received in last sec: 0
[1446968.548356] wlan: [26093:F:DP]     Data received in last sec: 0
[1446968.554005] wlan: [26093:F:DP] Multipass Rx Packet Drop = 0
[1446968.559945] wlan: [0:F:DP] REO queue stats (TID: 0):
[1446968.559945] ssn: 55
[1446968.559945] curr_idx  : 55
[1446968.559945] pn_31_0   : 00000000
[1446968.559945] pn_63_32  : 00000000
[1446968.559945] pn_95_64  : 00000000
[1446968.559945] pn_127_96 : 00000000
[1446968.559945] last_rx_enq_tstamp : 469dd250
[1446968.559945] last_rx_deq_tstamp : 469dd250
[1446968.559945] rx_bitmap_31_0     : 00000000
[1446968.559945] rx_bitmap_63_32    : 00000000
[1446968.559945] rx_bitmap_95_64    : 00000000
[1446968.559945] rx_bitmap_127_96   : 00000000
[1446968.559945] rx_bitmap_159_128  : 00000000
[1446968.559945] rx_bitmap_191_160  : 00000000
[1446968.559945] rx_bitmap_223_192  : 00000000
[1446968.559945] rx_bitmap_255_224  : 00000000
[1446968.559945]
[1446968.610691] wlan: [0:F:DP] curr_mpdu_cnt      : 0
[1446968.610691] curr_msdu_cnt      : 0
[1446968.610691] fwd_timeout_cnt    : 0
[1446968.610691] fwd_bar_cnt        : 0
[1446968.610691] dup_cnt            : 0
[1446968.610691] frms_in_order_cnt  : 55
[1446968.610691] bar_rcvd_cnt       : 0
[1446968.610691] mpdu_frms_cnt      : 55
[1446968.610691] msdu_frms_cnt      : 55
[1446968.610691] total_byte_cnt     : 10630
[1446968.610691] late_recv_mpdu_cnt : 0
[1446968.610691] win_jump_2k        : 0
[1446968.610691] hole_cnt           : 0
[1446968.610691]
[1446968.660427] wlan: [0:F:DP] Addba Req          : 1
[1446968.660427] Addba Resp         : 1
[1446968.660427] Addba Resp success : 1
[1446968.660427] Addba Resp failed  : 0
[1446968.660427] Delba Req received : 0
[1446968.660427] Delba Tx success   : 0
[1446968.660427] Delba Tx Fail      : 0
[1446968.660427] BA window size     : 256
[1446968.660427] Pn size            : 0
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
[1446968.660427]
[1446968.695660] wlan: [0:F:DP] REO queue stats (TID: 16):
[1446968.695660] ssn: 0
[1446968.695660] curr_idx  : 0
[1446968.695660] pn_31_0   : 00000000
[1446968.695660] pn_63_32  : 00000000
[1446968.695660] pn_95_64  : 00000000
[1446968.695660] pn_127_96 : 00000000
[1446968.695660] last_rx_enq_tstamp : 00000000
[1446968.695660] last_rx_deq_tstamp : 00000000
[1446968.695660] rx_bitmap_31_0     : 00000000
[1446968.695660] rx_bitmap_63_32    : 00000000
[1446968.695660] rx_bitmap_95_64    : 00000000
[1446968.695660] rx_bitmap_127_96   : 00000000
[1446968.695660] rx_bitmap_159_128  : 00000000
[1446968.695660] rx_bitmap_191_160  : 00000000
[1446968.695660] rx_bitmap_223_192  : 00000000
[1446968.695660] rx_bitmap_255_224  : 00000000
[1446968.695660]
[1446968.762427] wlan: [0:F:DP] curr_mpdu_cnt     : 0
[1446968.762427] curr_msdu_cnt      : 0
[1446968.762427] fwd_timeout_cnt    : 0
[1446968.762427] fwd_bar_cnt        : 0
[1446968.762427] dup_cnt            : 0
[1446968.762427] frms_in_order_cnt  : 0
[1446968.762427] bar_rcvd_cnt       : 0
[1446968.762427] mpdu_frms_cnt      : 0
[1446968.762427] msdu_frms_cnt      : 0
[1446968.762427] total_byte_cnt     : 0
[1446968.762427] late_recv_mpdu_cnt : 0
[1446968.762427] win_jump_2k        : 0
[1446968.762427] hole_cnt           : 0
[1446968.762427]
[1446968.811710] wlan: [0:F:DP] Addba Req          : 0
[1446968.811710] Addba Resp         : 0
[1446968.811710] Addba Resp success : 0
[1446968.811710] Addba Resp failed  : 0
[1446968.811710] Delba Req received : 0
[1446968.811710] Delba Tx success   : 0
[1446968.811710] Delba Tx Fail      : 0
[1446968.811710] BA window size     : 1
[1446968.811710] Pn size            : 0
[1446968.811710]
root@OpenWrt:/#
```

NOTE To reset firmware statistics, use the following command: wifistats wifi0 0 <statnum>. For example, to reset stats 1, use the root@OpenWrt:/# wifistats wifi0 0 1 command.

3.23.1.6 Firmware radio-level/PDEV Tx statistics

```
root@OpenWrt:/# iwpriv ath0 txrx_stats 1

[  896.719402] wlan: [35: F:DP] HTT_TX_PDEV_STATS_CMN_TLV:
[  896.719430] wlan: [35: F:DP] mac_id__word = 0
[  896.723556] wlan: [35: F:DP] hw_queued = 730
```

```
[  896.727997] wlan: [35: F:DP] hw_reaped = 730
[  896.732373] wlan: [35: F:DP] underrun = 0
[  896.736540] wlan: [35: F:DP] hw_paused = 24
[  896.740461] wlan: [35: F:DP] hw_flush = 15
[  896.744451] wlan: [35: F:DP] hw_filt = 0
[  896.748619] wlan: [35: F:DP] tx_abort = 0
[  896.752694] wlan: [35: F:DP] mpdu_requeued = 566
[  896.756615] wlan: [35: F:DP] tx_xretry = 566
[  896.761298] wlan: [35: F:DP] data_rc = 231
[  896.765595] wlan: [35: F:DP] mpdu_dropped_xretry = 0
[  896.769441] wlan: [35: F:DP] illegal_rate_phy_err = 0
[  896.774613] wlan: [35: F:DP] cont_xretry = 0
[  896.779509] wlan: [35: F:DP] tx_timeout = 0
[  896.783907] wlan: [35: F:DP] pdev_resets = 0
[  896.787755] wlan: [35: F:DP] phy_underrun = 0
[  896.792294] wlan: [35: F:DP] txop_ovf = 0
[  896.796522] wlan: [35: F:DP] seq_posted = 261
[  896.800530] wlan: [35: F:DP] seq_failed_queueing = 0
[  896.804873] wlan: [35: F:DP] seq_completed = 522
[  896.810029] wlan: [35: F:DP] seq_restarted = 0
[  896.814634] wlan: [35: F:DP] mu_seq_posted = 0
[  896.818748] wlan: [35: F:DP] seq_switch_hw_paused = 0
[  896.818763] send_filled_buffers_to_user: Send Failed -3 drop_count = 1
[  896.828328] wlan: [35: F:DP] next_seq_posted_dsr = 0
[  896.834745] wlan: [35: F:DP] seq_posted_isr = 0
[  896.839854] wlan: [35: F:DP] seq_ctrl_cached = 0
[  896.844107] wlan: [35: F:DP] mpdu_count_tqm = 15
[  896.848981] wlan: [35: F:DP] msdu_count_tqm = 15
[  896.853607] wlan: [35: F:DP] mpdu_removed_tqm = 0
[  896.858182] wlan: [35: F:DP] msdu_removed_tqm = 0
[  896.862781] wlan: [35: F:DP] mpdus_sw_flush = 0
[  896.867458] wlan: [35: F:DP] mpdus_hw_filter = 0
[  896.871844] wlan: [35: F:DP] mpdus_truncated = 0
[  896.876681] wlan: [35: F:DP] mpdus_ack_failed = 566
[  896.881261] wlan: [35: F:DP] mpdus_expired = 0
[  896.885857] wlan: [35: F:DP] mpdus_seq_hw_retry = 445
[  896.890382] wlan: [35: F:DP] ack_tlv_proc = 15
[  896.895489] wlan: [35: F:DP] coex_abort_mpdu_cnt_valid = 0
[  896.899845] wlan: [35: F:DP] coex_abort_mpdu_cnt = 0
[  896.899845]
[  896.905331] wlan: [35: F:DP] HTT_TX_PDEV_STATS_URRN_TLV_V:
[  896.911881] wlan: [35: F:DP] urrn_stats =  0:0, 1:0, 2:0,
[  896.911881]
[  896.917203] wlan: [35: F:DP] HTT_TX_PDEV_STATS_SIFS_TLV_V:
[  896.924112] wlan: [35: F:DP] sifs_status =  0:0, 1:15, 2:0, 3:9, 4:0, 5:0,
6:15, 7:0, 8:0,
[  896.924112]
[  896.929525] wlan: [35: F:DP] HTT_TX_PDEV_STATS_FLUSH_TLV_V:
[  896.939212] wlan: [35: F:DP] flush_errs =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0,
7:0, 8:0, 9:0, 10:0, 11:0, 12:0, 13:0, 14:0, 15:0, 16:0, 17:0, 18:0, 19:0, 20:0,
21:0, 22:0, 23:0, 24:0, 25:0, 26:0, 27:0, 28:0, 29:0, 30:0, 31:0, 32:0, 33:0,
34:0, 35:0, 36:0, 37:0, 38:15, 39:0, 40:0, 41:0, 42:0, 43:0, 44:0, 45:0, 46:0,
47:0, 48:0, 49:0, 50:0, 51:0, 52:0, 53:0, 54:0, 55:0, 56:0, 57:0, 58:0, 59:0,
```

```
60:0, 61:0, 62:0, 63:0, 64:0, 65:0, 66:0, 67:0, 68:0, 69:0, 70:0,
[  896.939212]
[  896.964534] wlan: [35: F:DP] HTT_TX_PDEV_STATS_PHY_ERR_TLV_V:
[  896.986783] wlan: [35: F:DP] phy_errs =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0,
7:0, 8:0, 9:0, 10:0, 11:0, 12:0, 13:0, 14:0, 15:0, 16:0, 17:0,
[  896.986783]
```

It is possible to also display the firmware Tx stats information using the command: wifistats wifi0 1. Display the firmware Tx stats information using the wifistats wifi0 1 command.

```
root@OpenWrt:/# wifistats wifi0 1

HTT_TX_PDEV_STATS_CMN_TLV:
mac_id = 0
comp_delivered = 2020
self_triggers = 49374959
hw_queued = 1007325
hw_reaped = 1007325
underrun = 0
hw_paused = 161028
hw_flush = 127137
hw_filt = 127340
tx_abort = 0
ppdu_ok = 836534
mpdu_requeued = 460530
tx_xretry = 33152
data_rc = 3
mpdu_dropped_xretry = 0
illegal_rate_phy_err = 0
cont_xretry = 0
tx_timeout = 0
tx_time_dur_data = 254730773
pdev_resets = 0
phy_underrun = 0
txop_ovf = 0
seq_posted = 853101
seq_failed_queueing = 71985
seq_completed = 781116
seq_restarted = 0
seq_txop_repost_stop = 2993
next_seq_cancel = 135691
mu_mimo_seq_posted = 0
mu_ofdma_seq_posted = 0
mu_mimo_peer_blacklist_count = 0
seq_qdepth_repost_stop = 42260
seq_min_msdu_repost_stop = 19313
mu_seq_min_msdu_repost_stop = 0
seq_switch_hw_paused = 0
next_seq_posted_dsr = 2698
seq_posted_isr = 115461
seq_ctrl_cached = 182845
mpdu_count_tqm = 11541819
msdu_count_tqm = 34308957
mpdu_removed_tqm = 0
msdu_removed_tqm = 0
```

Qualcomm Confidential - May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

```
remove_mpdus_max_retries = 0
mpdus_sw_flush = 0
mpdus_hw_filter = 0
mpdus_truncated = 139
mpdus_ack_failed = 58827
mpdus_expired = 0
mpdus_seq_hw_retry = 0
ack_tlv_proc = 351301
coex_abort_mpdu_cnt_valid = 386463
coex_abort_mpdu_cnt = 12487582
num_total_ppdus_tried_ota = 869676
num_data_ppdus_tried_ota = 386463
local_ctrl_mgmt_enqued = 483416
local_ctrl_mgmt_freed = 483416
local_data_enqued = 0
local_data_freed = 0
mpdu_tried = 0
isr_wait_seq_posted = 0
tx_active_dur_us_low = 0
tx_active_dur_us_high = 0
fes_offsets_err_cnt = 0

HTT_TX_PDEV_STATS_URRN_TLV_V:
urrn_stats =  0:0, 1:0, 2:0,

HTT_TX_PDEV_STATS_SIFS_TLV_V:
sifs_status =  0:0, 1:215910, 2:774, 3:902, 4:2852, 5:0, 6:37036,
7:35168, 8:0,

HTT_TX_PDEV_STATS_FLUSH_TLV_V:
flush_errs =  0:0, 1:0, 2:0, 3:0, 4:0, 5:170, 6:0, 7:777, 8:0, 9:0, 10:0,
11:0, 12:0, 13:0, 14:0, 15:0, 16:0, 17:0, 18:0, 19:0, 20:0, 21:0, 22:0,
23:0, 24:0, 25:0, 26:0, 27:0, 28:0, 29:0, 30:0, 31:0, 32:0, 33:0, 34:0,
35:0, 36:0, 37:0, 38:126189, 39:0, 40:0, 41:0, 42:1, 43:0, 44:0, 45:0,
46:0, 47:0, 48:0, 49:0, 50:0, 51:0, 52:0, 53:0, 54:0, 55:0, 56:0, 57:0,
58:0, 59:0, 60:0, 61:0, 62:0, 63:0, 64:0, 65:0, 66:0, 67:0, 68:0, 69:0,
70:0,

HTT_STATS_PDEV_CTRL_PATH_TX_STATS_TAG:
fw_tx_mgmt_subtype =  0:0, 1:1, 2:0, 3:0, 4:0, 5:1, 6:0, 7:0, 8:483411,
9:0, 10:0, 11:1, 12:0, 13:2, 14:0, 15:0,

HTT_TX_PDEV_STATS_SIFS_HIST_TLV_V:
sifs_hist_status =  1:140934, 2:4719, 3:41461, 4:9689, 5:7174, 6:4090,
7:1239, 8:313, 9:125, 10:24,
root@OpenWrt:/#
```

3.23.1.7 Firmware radio-level/PDEV Rx statistics

```
root@OpenWrt:/# iwpriv ath0 txrx_stats 2

[  899.930335] wlan: [35: F:DP] HTT_RX_PDEV_FW_STATS_TLV:
[  899.930363] wlan: [35: F:DP] mac_id__word = 0
[  899.934429] wlan: [35: F:DP] ppdu_recvd = 5401
[  899.938809] wlan: [35: F:DP] mpdu_cnt_fcs_ok = 5401
[  899.943143] wlan: [35: F:DP] mpdu_cnt_fcs_err = 0
```

```
[  899.947964] wlan: [35: F:DP] tcp_msdu_cnt = 0
[  899.952780] wlan: [35: F:DP] tcp_ack_msdu_cnt = 0
[  899.957124] wlan: [35: F:DP] udp_msdu_cnt = 8
[  899.961809] wlan: [35: F:DP] other_msdu_cnt = 5393
[  899.966151] wlan: [35: F:DP] fw_ring_mpdu_ind = 5350
[  899.970834] wlan: [35: F:DP] fw_ring_mgmt_subtype =  0:0, 1:1, 2:0, 3:0, 4:0,
5:74, 6:0, 7:0, 8:5272, 9:0, 10:0, 11:1, 12:0, 13:2, 14:0, 15:0,
[  899.976017] wlan: [35: F:DP] fw_ring_ctrl_subtype =  0:0, 1:0, 2:0, 3:0, 4:0,
5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0, 12:0, 13:0, 14:0, 15:0,
[  899.988729] wlan: [35: F:DP] fw_ring_mcast_data_msdu = 0
[  900.001096] wlan: [35: F:DP] fw_ring_bcast_data_msdu = 0
[  900.006617] wlan: [35: F:DP] fw_ring_ucast_data_msdu = 0
[  900.011900] wlan: [35: F:DP] fw_ring_null_data_msdu = 0
[  900.017194] wlan: [35: F:DP] fw_ring_mpdu_drop = 0
[  900.022176] wlan: [35: F:DP] ofld_local_data_ind_cnt = 0
[  900.027003] wlan: [35: F:DP] ofld_local_data_buf_recycle_cnt = 0
[  900.032500] wlan: [35: F:DP] drx_local_data_ind_cnt = 0
[  900.038461] wlan: [35: F:DP] drx_local_data_buf_recycle_cnt = 0
[  900.043405] wlan: [35: F:DP] local_nondata_ind_cnt = 5350
[  900.049344] wlan: [35: F:DP] local_nondata_buf_recycle_cnt = 5349
[  900.054879] wlan: [35: F:DP] fw_status_buf_ring_refill_cnt = 5401
[  900.060932] wlan: [35: F:DP] fw_status_buf_ring_empty_cnt = 0
[  900.067046] wlan: [35: F:DP] fw_pkt_buf_ring_refill_cnt = 5362
[  900.072779] wlan: [35: F:DP] fw_pkt_buf_ring_empty_cnt = 1
[  900.078498] wlan: [35: F:DP] fw_link_buf_ring_refill_cnt = 5350
[  900.084002] wlan: [35: F:DP] fw_link_buf_ring_empty_cnt = 0
[  900.089766] wlan: [35: F:DP] host_pkt_buf_ring_refill_cnt = 2098
[  900.095591] wlan: [35: F:DP] host_pkt_buf_ring_empty_cnt = 2
[  900.101557] wlan: [35: F:DP] mon_pkt_buf_ring_refill_cnt = 1023
[  900.107226] wlan: [35: F:DP] mon_pkt_buf_ring_empty_cnt = 2
[  900.112951] wlan: [35: F:DP] mon_status_buf_ring_refill_cnt = 1023
[  900.118404] wlan: [35: F:DP] mon_status_buf_ring_empty_cnt = 156334
[  900.124699] wlan: [35: F:DP] mon_desc_buf_ring_refill_cnt = 1023
[  900.130825] wlan: [35: F:DP] mon_desc_buf_ring_empty_cnt = 2
[  900.137083] wlan: [35: F:DP] mon_dest_ring_update_cnt = 0
[  900.142741] wlan: [35: F:DP] mon_dest_ring_full_cnt = 0
[  900.148009] wlan: [35: F:DP] rx_suspend_cnt = 37
[  900.153052] wlan: [35: F:DP] rx_suspend_fail_cnt = 0
[  900.157891] wlan: [35: F:DP] rx_resume_cnt = 37
[  900.162861] wlan: [35: F:DP] rx_resume_fail_cnt = 0
[  900.167097] wlan: [35: F:DP] rx_ring_switch_cnt = 0
[  900.171997] wlan: [35: F:DP] rx_ring_restore_cnt = 0
[  900.176821] wlan: [35: F:DP] rx_flush_cnt = 0
[  900.176821]
[  900.182052] wlan: [35: F:DP] HTT_RX_PDEV_FW_RING_MPDU_ERR_TLV_V:
[  900.187786] wlan: [35: F:DP] fw_ring_mpdu_err =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0,
6:0, 7:0, 8:0, 9:0, 10:0, 11:0, 12:0, 13:0, 14:0, 15:0,
[  900.187786]
[  900.193856] wlan: [35: F:DP] HTT_RX_PDEV_FW_MPDU_DROP_TLV_V:
[  900.207143] wlan: [35: F:DP] fw_mpdu_drop =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0,
6:0, 7:0, 8:0, 9:0, 10:0, 11:0, 12:0, 13:0, 14:0, 15:0,
[  900.207143]
[  900.213056] wlan: [35: F:DP] HTT_RX_SOC_FW_STATS_TLV:
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
[  900.225995] wlan: [35: F:DP] fw_reo_ring_data_msdu = 0
[  900.231171] wlan: [35: F:DP] fw_to_host_data_msdu_bcmc = 0
[  900.236247] wlan: [35: F:DP] fw_to_host_data_msdu_uc = 0
[  900.241659] wlan: [35: F:DP] ofld_remote_data_buf_recycle_cnt = 0
[  900.247153] wlan: [35: F:DP] ofld_remote_free_buf_indication_cnt = 0
[  900.247153]
[  900.253145] wlan: [35: F:DP] HTT_RX_SOC_FW_REFILL_RING_EMPTY_TLV_V:
[  900.260954] wlan: [35: F:DP] refill_ring_empty_cnt =  0:0, 1:0,
[  900.260954]
[  900.267033] wlan: [35: F:DP] HTT_RX_SOC_FW_REFILL_RING_NUM_REFILL_TLV_V:
[  900.274687] wlan: [35: F:DP] refill_ring_num_refill =  0:0, 1:0,
[  900.274687]
```

It is possible to also display the firmware Tx stats information using the command: wifistats wifi0 1. Display the firmware Rx stats information using the wifistats wifi0 2 command.

```
root@OpenWrt:/# wifistats wifi0 2

HTT_RX_PDEV_FW_STATS_TLV:
mac_id = 0
ppdu_recvd = 227033
mpdu_cnt_fcs_ok = 1580149
mpdu_cnt_fcs_err = 6607
tcp_msdu_cnt = 0
tcp_ack_msdu_cnt = 9765399
udp_msdu_cnt = 1864
other_msdu_cnt = 101406
fw_ring_mpdu_ind = 5
fw_ring_mgmt_subtype =  0:1, 1:0, 2:0, 3:0, 4:1, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0,
11:1, 12:0, 13:2, 14:0, 15:0,
fw_ring_ctrl_subtype =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0,
11:0, 12:0, 13:0, 14:0, 15:0,
fw_ring_mcast_data_msdu = 0
fw_ring_bcast_data_msdu = 0
fw_ring_ucast_data_msdu = 0
fw_ring_null_data_msdu = 0
fw_ring_mpdu_drop = 1
ofld_local_data_ind_cnt = 0
ofld_local_data_buf_recycle_cnt = 0
drx_local_data_ind_cnt = 0
drx_local_data_buf_recycle_cnt = 0
local_nondata_ind_cnt = 5
local_nondata_buf_recycle_cnt = 4
fw_status_buf_ring_refill_cnt = 227033
fw_status_buf_ring_empty_cnt = 0
fw_pkt_buf_ring_refill_cnt = 18
fw_pkt_buf_ring_empty_cnt = 1
fw_link_buf_ring_refill_cnt = 6
fw_link_buf_ring_empty_cnt = 0
host_pkt_buf_ring_refill_cnt = 9995181
host_pkt_buf_ring_empty_cnt = 2
mon_pkt_buf_ring_refill_cnt = 128
```

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yufeng@zte.com.cn

```
mon_pkt_buf_ring_empty_cnt = 2
mon_status_buf_ring_refill_cnt = 295167
mon_status_buf_ring_empty_cnt = 2
mon_desc_buf_ring_refill_cnt = 8191
mon_desc_buf_ring_empty_cnt = 2
mon_dest_ring_update_cnt = 0
mon_dest_ring_full_cnt = 0
rx_suspend_cnt = 3
rx_suspend_fail_cnt = 0
rx_resume_cnt = 3
rx_resume_fail_cnt = 0
rx_ring_switch_cnt = 0
rx_ring_restore_cnt = 0
rx_flush_cnt = 0
rx_recovery_reset_cnt = 0

HTT_RX_PDEV_FW_RING_MPDU_ERR_TLV_V:
fw_ring_mpdu_err =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0,
12:0, 13:0, 14:0, 15:0,

HTT_RX_PDEV_FW_MPDU_DROP_TLV_V:
fw_mpdu_drop =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0,
12:0, 13:0, 14:0, 15:0,
HTT_RX_SOC_FW_STATS_TLV:
fw_reo_ring_data_msdu = 0
fw_to_host_data_msdu_bcmc = 0
fw_to_host_data_msdu_uc = 0
ofld_remote_data_buf_recycle_cnt = 0
ofld_remote_free_buf_indication_cnt = 0
ofld_buf_to_host_data_msdu_uc = 0
reo_fw_ring_to_host_data_msdu_uc = 0
wbm_sw_ring_reap = 0
wbm_forward_to_host_cnt = 0
wbm_target_recycle_cnt = 0
target_refill_ring_recycle_cnt = 0
HTT_RX_SOC_FW_REFILL_RING_EMPTY_TLV_V:
refill_ring_empty_cnt =  0:0, 1:0, 2:0,

HTT_RX_SOC_FW_REFILL_RING_NUM_REFILL_TLV_V:
refill_ring_num_refill =  0:0, 1:0, 2:0,

HTT_RX_SOC_FW_REFILL_RING_NUM_RXDMA_ERR_TLV_V:
rxdma_err =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0, 12:0,
13:0, 14:0, 15:0,

HTT_RX_SOC_FW_REFILL_RING_NUM_REO_ERR_TLV_V:
reo_err =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0, 12:0,
13:0, 14:0, 15:0,

HTT_RX_PDEV_FW_STATS_PHY_ERR_TLV:
mac_id__word = 0
tota_phy_err_nct = 0
phy_errs =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0, 12:0,
```

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yuanzhi@zte.com.cn

```
13:0, 14:0, 15:0, 16:0, 17:0, 18:0, 19:0, 20:0, 21:0, 22:0, 23:0, 24:0, 25:0,
26:0, 27:0, 28:0, 29:0, 30:0, 31:0, 32:0, 33:0, 34:0, 35:0, 36:0, 37:0, 38:0,
39:0, 40:0, 41:0, 42:0,

HTT_RX_REO_RESOURCE_STATS_TLV:
sample_id = 1
total_max = 53
total_avg = 0
total_sample = 24015529
non_zeros_avg = 4
non_zeros_sample = 1028
last_non_zeros_max = 2
last_non_zeros_min 2
last_non_zeros_avg 2
last_non_zeros_sample 1

HTT_RX_REO_RESOURCE_STATS_TLV:
sample_id = 3
total_max = 171
total_avg = 0
total_sample = 24015529
non_zeros_avg = 12
non_zeros_sample = 1027
last_non_zeros_max = 1
last_non_zeros_min 1
last_non_zeros_avg 1
last_non_zeros_sample 1

HTT_RX_REO_RESOURCE_STATS_TLV:
sample_id = 7
total_max = 1
total_avg = 0
total_sample = 24015529
non_zeros_avg = 1
non_zeros_sample = 1026
last_non_zeros_max = 1
last_non_zeros_min 1
last_non_zeros_avg 1
last_non_zeros_sample 1
root@OpenWrt:/#
```

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

3.23.1.8 Firmware hardware queue statistics

```
root@OpenWrt:/# iwpriv ath0 txrx_stats 3

[  903.568108] wlan: [35: F:DP] HTT_TX_HWQ_STATS_CMN_TLV:
[  903.568137] wlan: [35: F:DP] mac_id__hwq_id__word = 0
[  903.572190] wlan: [35: F:DP] xretry = 0
[  903.577296] wlan: [35: F:DP] underrun_cnt = 0
[  903.580965] wlan: [35: F:DP] flush_cnt = 0
[  903.585444] wlan: [35: F:DP] filt_cnt = 0
[  903.589413] wlan: [35: F:DP] null_mpdu_bmap = 0
[  903.593517] wlan: [35: F:DP] user_ack_failure = 0
[  903.597874] wlan: [35: F:DP] ack_tlv_proc = 0
```

```
[  903.602723] wlan: [35: F:DP] sched_id_proc = 0
[  903.607049] wlan: [35: F:DP] null_mpdu_tx_count = 0
[  903.611375] wlan: [35: F:DP] mpdu_bmap_not_recvd = 0
[  903.616175] wlan: [35: F:DP] num_bar = 0
[  903.621364] wlan: [35: F:DP] rts = 0
[  903.625303] wlan: [35: F:DP] cts2self = 0
[  903.628860] wlan: [35: F:DP] qos_null = 0
[  903.632757] wlan: [35: F:DP] mpdu_tried_cnt = 0
[  903.636722] wlan: [35: F:DP] mpdu_queued_cnt = 0
[  903.641078] wlan: [35: F:DP] mpdu_ack_fail_cnt = 0
[  903.645970] wlan: [35: F:DP] mpdu_filt_cnt = 0
[  903.650540] wlan: [35: F:DP] false_mpdu_ack_count = 0
[  903.650540]
[  903.654968] wlan: [35: F:DP] HTT_TX_HWQ_DIFS_LATENCY_STATS_TLV_V:
[  903.661480] wlan: [35: F:DP] hist_intvl = 20
[  903.667762] wlan: [35: F:DP] difs_latency_hist =  0:0,
[  903.667762]
[  903.671994] wlan: [35: F:DP] HTT_TX_HWQ_CMD_RESULT_STATS_TLV_V:
[  903.678337] wlan: [35: F:DP] cmd_result =  0:0,
[  903.684270] wlan: [35: F:DP] HTT_TX_HWQ_CMD_STALL_STATS_TLV_V:
[  903.689068] wlan: [35: F:DP] cmd_stall_status =  0:0,
[  903.689068]
[  903.694771] wlan: [35: F:DP] HTT_TX_HWQ_FES_RESULT_STATS_TLV_V:
[  903.701237] wlan: [35: F:DP] fes_result =  0:0,
[  903.707061] wlan: [35: F:DP] HTT_TX_HWQ_STATS_CMN_TLV:
[  903.711906] wlan: [35: F:DP] mac_id__hwq_id__word = 0
[  903.716879] wlan: [35: F:DP] xretry = 0
[  903.721987] wlan: [35: F:DP] underrun_cnt = 0
[  903.725642] wlan: [35: F:DP] flush_cnt = 0
[  903.730124] wlan: [35: F:DP] filt_cnt = 0
[  903.734141] wlan: [35: F:DP] null_mpdu_bmap = 0
[  903.738199] wlan: [35: F:DP] user_ack_failure = 0
[  903.742594] wlan: [35: F:DP] ack_tlv_proc = 0
[  903.747414] wlan: [35: F:DP] sched_id_proc = 0
[  903.751740] wlan: [35: F:DP] null_mpdu_tx_count = 0
[  903.756105] wlan: [35: F:DP] mpdu_bmap_not_recvd = 0
[  903.760854] wlan: [35: F:DP] num_bar = 0
[  903.766091] wlan: [35: F:DP] rts = 0
[  903.770027] wlan: [35: F:DP] cts2self = 0
[  903.773568] wlan: [35: F:DP] qos_null = 0
[  903.777466] wlan: [35: F:DP] mpdu_tried_cnt = 0
[  903.781469] wlan: [35: F:DP] mpdu_queued_cnt = 0
[  903.785779] wlan: [35: F:DP] mpdu_ack_fail_cnt = 0
[  903.790687] wlan: [35: F:DP] mpdu_filt_cnt = 0
[  903.795239] wlan: [35: F:DP] false_mpdu_ack_count = 0
[  903.795239]
[  903.799748] wlan: [35: F:DP] HTT_TX_HWQ_DIFS_LATENCY_STATS_TLV_V:
[  903.806193] wlan: [35: F:DP] hist_intvl = 20
[  903.812345] wlan: [35: F:DP] difs_latency_hist =  0:0,
[  903.812345]
[  903.816705] wlan: [35: F:DP] HTT_TX_HWQ_CMD_RESULT_STATS_TLV_V:
[  903.823052] wlan: [35: F:DP] cmd_result =  0:0,
[  903.828910] wlan: [35: F:DP] HTT_TX_HWQ_CMD_STALL_STATS_TLV_V:
[  903.833799] wlan: [35: F:DP] cmd_stall_status =  0:0,
[  903.833799]
[  903.839464] wlan: [35: F:DP] HTT_TX_HWQ_FES_RESULT_STATS_TLV_V:
[  903.845956] wlan: [35: F:DP] fes_result =  0:0,
[  903.851741] wlan: [35: F:DP] HTT_TX_HWQ_STATS_CMN_TLV:
[  903.856625] wlan: [35: F:DP] mac_id__hwq_id__word = 0
[  903.861556] wlan: [35: F:DP] xretry = 0
[  903.866706] wlan: [35: F:DP] underrun_cnt = 0
[  903.870344] wlan: [35: F:DP] flush_cnt = 0
[  903.874892] wlan: [35: F:DP] filt_cnt = 0
[  903.878832] wlan: [35: F:DP] null_mpdu_bmap = 0
```

```
[  903.882916] wlan: [35: F:DP] user_ack_failure = 0
[  903.887379] wlan: [35: F:DP] ack_tlv_proc = 0
[  903.892113] wlan: [35: F:DP] sched_id_proc = 0
[  903.896460] wlan: [35: F:DP] null_mpdu_tx_count = 0
[  903.900792] wlan: [35: F:DP] mpdu_bmap_not_recvd = 0
[  903.905583] wlan: [35: F:DP] num_bar = 0
[  903.910773] wlan: [35: F:DP] rts = 0
[  903.914708] wlan: [35: F:DP] cts2self = 0
[  903.918247] wlan: [35: F:DP] qos_null = 0
[  903.922161] wlan: [35: F:DP] mpdu_tried_cnt = 0
[  903.926131] wlan: [35: F:DP] mpdu_queued_cnt = 0
[  903.930487] wlan: [35: F:DP] mpdu_ack_fail_cnt = 0
[  903.935373] wlan: [35: F:DP] mpdu_filt_cnt = 0
[  903.939949] wlan: [35: F:DP] false_mpdu_ack_count = 0
[  903.939949]
[  903.944375] wlan: [35: F:DP] HTT_TX_HWQ_DIFS_LATENCY_STATS_TLV_V:
[  903.950890] wlan: [35: F:DP] hist_intvl = 20
[  903.957065] wlan: [35: F:DP] difs_latency_hist =  0:0,
[  903.957065]
[  903.961418] wlan: [35: F:DP] HTT_TX_HWQ_CMD_RESULT_STATS_TLV_V:
[  903.967741] wlan: [35: F:DP] cmd_result =  0:0,
[  903.973637] wlan: [35: F:DP] HTT_TX_HWQ_CMD_STALL_STATS_TLV_V:
[  903.978546] wlan: [35: F:DP] cmd_stall_status =  0:0,
[  903.978546]
[  903.984163] wlan: [35: F:DP] HTT_TX_HWQ_FES_RESULT_STATS_TLV_V:
[  903.990647] wlan: [35: F:DP] fes_result =  0:0,
[  903.996470] wlan: [35: F:DP] HTT_TX_HWQ_STATS_CMN_TLV:
[  904.001307] wlan: [35: F:DP] mac_id__hwq_id__word = 0
[  904.006359] wlan: [35: F:DP] xretry = 0
[  904.011401] wlan: [35: F:DP] underrun_cnt = 0
[  904.015045] wlan: [35: F:DP] flush_cnt = 0
[  904.019549] wlan: [35: F:DP] filt_cnt = 0
[  904.023535] wlan: [35: F:DP] null_mpdu_bmap = 0
[  904.027622] wlan: [35: F:DP] user_ack_failure = 0
[  904.031988] wlan: [35: F:DP] ack_tlv_proc = 0
[  904.036834] wlan: [35: F:DP] sched_id_proc = 0
[  904.041149] wlan: [35: F:DP] null_mpdu_tx_count = 0
[  904.045514] wlan: [35: F:DP] mpdu_bmap_not_recvd = 0
[  904.050265] wlan: [35: F:DP] num_bar = 0
[  904.055502] wlan: [35: F:DP] rts = 0
[  904.059389] wlan: [35: F:DP] cts2self = 0
[  904.063054] wlan: [35: F:DP] qos_null = 0
[  904.066843] wlan: [35: F:DP] mpdu_tried_cnt = 0
[  904.070853] wlan: [35: F:DP] mpdu_queued_cnt = 0
[  904.075222] wlan: [35: F:DP] mpdu_ack_fail_cnt = 0
[  904.080121] wlan: [35: F:DP] mpdu_filt_cnt = 0
[  904.084651] wlan: [35: F:DP] false_mpdu_ack_count = 0
[  904.084651]
[  904.089086] wlan: [35: F:DP] HTT_TX_HWQ_DIFS_LATENCY_STATS_TLV_V:
[  904.095602] wlan: [35: F:DP] hist_intvl = 20
[  904.101772] wlan: [35: F:DP] difs_latency_hist =  0:0,
[  904.101772]
[  904.106107] wlan: [35: F:DP] HTT_TX_HWQ_CMD_RESULT_STATS_TLV_V:
[  904.112458] wlan: [35: F:DP] cmd_result =  0:0,
[  904.118336] wlan: [35: F:DP] HTT_TX_HWQ_CMD_STALL_STATS_TLV_V:
[  904.123191] wlan: [35: F:DP] cmd_stall_status =  0:0,
[  904.123191]
[  904.128874] wlan: [35: F:DP] HTT_TX_HWQ_FES_RESULT_STATS_TLV_V:
[  904.135362] wlan: [35: F:DP] fes_result =  0:0,
[  904.141149] wlan: [35: F:DP] HTT_TX_HWQ_STATS_CMN_TLV:
[  904.146035] wlan: [35: F:DP] mac_id__hwq_id__word = 0
[  904.150980] wlan: [35: F:DP] xretry = 0
[  904.156105] wlan: [35: F:DP] underrun_cnt = 0
[  904.159747] wlan: [35: F:DP] flush_cnt = 0
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
[  904.164249] wlan: [35: F:DP] filt_cnt = 0
[  904.168246] wlan: [35: F:DP] null_mpdu_bmap = 0
[  904.172351] wlan: [35: F:DP] user_ack_failure = 0
[  904.176669] wlan: [35: F:DP] ack_tlv_proc = 0
[  904.181556] wlan: [35: F:DP] sched_id_proc = 0
[  904.185880] wlan: [35: F:DP] null_mpdu_tx_count = 0
[  904.190195] wlan: [35: F:DP] mpdu_bmap_not_recvd = 0
[  904.194992] wlan: [35: F:DP] num_bar = 0
[  904.200204] wlan: [35: F:DP] rts = 0
[  904.204115] wlan: [35: F:DP] cts2self = 0
[  904.207648] wlan: [35: F:DP] qos_null = 0
[  904.211562] wlan: [35: F:DP] mpdu_tried_cnt = 0
[  904.215552] wlan: [35: F:DP] mpdu_queued_cnt = 0
[  904.219927] wlan: [35: F:DP] mpdu_ack_fail_cnt = 0
[  904.224753] wlan: [35: F:DP] mpdu_filt_cnt = 0
[  904.229359] wlan: [35: F:DP] false_mpdu_ack_count = 0
[  904.229359]
[  904.233798] wlan: [35: F:DP] HTT_TX_HWQ_DIFS_LATENCY_STATS_TLV_V:
[  904.240284] wlan: [35: F:DP] hist_intvl = 20
[  904.246474] wlan: [35: F:DP] difs_latency_hist =  0:0,
[  904.246474]
[  904.250817] wlan: [35: F:DP] HTT_TX_HWQ_CMD_RESULT_STATS_TLV_V:
[  904.257149] wlan: [35: F:DP] cmd_result =  0:0,
[  904.263048] wlan: [35: F:DP] HTT_TX_HWQ_CMD_STALL_STATS_TLV_V:
[  904.267934] wlan: [35: F:DP] cmd_stall_status =  0:0,
[  904.267934]
[  904.273543] wlan: [35: F:DP] HTT_TX_HWQ_FES_RESULT_STATS_TLV_V:
[  904.280055] wlan: [35: F:DP] fes_result =  0:0,
[  904.285919] wlan: [35: F:DP] HTT_TX_HWQ_STATS_CMN_TLV:
[  904.290717] wlan: [35: F:DP] mac_id__hwq_id__word = 0
[  904.295710] wlan: [35: F:DP] xretry = 0
[  904.300818] wlan: [35: F:DP] underrun_cnt = 0
[  904.304441] wlan: [35: F:DP] flush_cnt = 0
[  904.308958] wlan: [35: F:DP] filt_cnt = 0
[  904.312944] wlan: [35: F:DP] null_mpdu_bmap = 0
[  904.317064] wlan: [35: F:DP] user_ack_failure = 0
[  904.321357] wlan: [35: F:DP] ack_tlv_proc = 0
[  904.326243] wlan: [35: F:DP] sched_id_proc = 0
[  904.330559] wlan: [35: F:DP] null_mpdu_tx_count = 0
[  904.334922] wlan: [35: F:DP] mpdu_bmap_not_recvd = 0
[  904.339682] wlan: [35: F:DP] num_bar = 0
[  904.344920] wlan: [35: F:DP] rts = 0
[  904.348817] wlan: [35: F:DP] cts2self = 0
[  904.352354] wlan: [35: F:DP] qos_null = 0
[  904.356253] wlan: [35: F:DP] mpdu_tried_cnt = 0
[  904.360260] wlan: [35: F:DP] mpdu_queued_cnt = 0
[  904.364633] wlan: [35: F:DP] mpdu_ack_fail_cnt = 0
[  904.369464] wlan: [35: F:DP] mpdu_filt_cnt = 0
[  904.374073] wlan: [35: F:DP] false_mpdu_ack_count = 0
[  904.374073]
[  904.378480] wlan: [35: F:DP] HTT_TX_HWQ_DIFS_LATENCY_STATS_TLV_V:
[  904.385123] wlan: [35: F:DP] hist_intvl = 20
[  904.391227] wlan: [35: F:DP] difs_latency_hist =  0:0,
[  904.391227]
[  904.395611] wlan: [35: F:DP] HTT_TX_HWQ_CMD_RESULT_STATS_TLV_V:
[  904.401869] wlan: [35: F:DP] cmd_result =  0:0,
[  904.407747] wlan: [35: F:DP] HTT_TX_HWQ_CMD_STALL_STATS_TLV_V:
[  904.412631] wlan: [35: F:DP] cmd_stall_status =  0:0,
[  904.412631]
[  904.418252] wlan: [35: F:DP] HTT_TX_HWQ_FES_RESULT_STATS_TLV_V:
[  904.424771] wlan: [35: F:DP] fes_result =  0:0,
[  904.430559] wlan: [35: F:DP] HTT_TX_HWQ_STATS_CMN_TLV:
[  904.435444] wlan: [35: F:DP] mac_id__hwq_id__word = 0
[  904.440399] wlan: [35: F:DP] xretry = 557
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
[  904.445511] wlan: [35: F:DP] underrun_cnt = 0
[  904.449507] wlan: [35: F:DP] flush_cnt = 0
[  904.453830] wlan: [35: F:DP] filt_cnt = 0
[  904.457830] wlan: [35: F:DP] null_mpdu_bmap = 0
[  904.461934] wlan: [35: F:DP] user_ack_failure = 111
[  904.466252] wlan: [35: F:DP] ack_tlv_proc = 0
[  904.471096] wlan: [35: F:DP] sched_id_proc = 0
[  904.475634] wlan: [35: F:DP] null_mpdu_tx_count = 0
[  904.479951] wlan: [35: F:DP] mpdu_bmap_not_recvd = 0
[  904.484782] wlan: [35: F:DP] num_bar = 0
[  904.489999] wlan: [35: F:DP] rts = 0
[  904.493865] wlan: [35: F:DP] cts2self = 0
[  904.497413] wlan: [35: F:DP] qos_null = 0
[  904.501305] wlan: [35: F:DP] mpdu_tried_cnt = 682
[  904.505317] wlan: [35: F:DP] mpdu_queued_cnt = 682
[  904.510023] wlan: [35: F:DP] mpdu_ack_fail_cnt = 557
[  904.514699] wlan: [35: F:DP] mpdu_filt_cnt = 0
[  904.519794] wlan: [35: F:DP] false_mpdu_ack_count = 0
[  904.519794]
[  904.524076] wlan: [35: F:DP] HTT_TX_HWQ_DIFS_LATENCY_STATS_TLV_V:
[  904.530577] wlan: [35: F:DP] hist_intvl = 20
[  904.536747] wlan: [35: F:DP] difs_latency_hist =  0:0,
[  904.536747]
[  904.541130] wlan: [35: F:DP] HTT_TX_HWQ_CMD_RESULT_STATS_TLV_V:
[  904.547426] wlan: [35: F:DP] cmd_result =  0:125,
[  904.553311] wlan: [35: F:DP] HTT_TX_HWQ_CMD_STALL_STATS_TLV_V:
[  904.558212] wlan: [35: F:DP] cmd_stall_status =  0:0,
[  904.558212]
[  904.564007] wlan: [35: F:DP] HTT_TX_HWQ_FES_RESULT_STATS_TLV_V:
[  904.570493] wlan: [35: F:DP] fes_result =  0:125,
[  904.576329] wlan: [35: F:DP] HTT_TX_HWQ_STATS_CMN_TLV:
[  904.581189] wlan: [35: F:DP] mac_id__hwq_id__word = 557
[  904.586310] wlan: [35: F:DP] xretry = 0
[  904.591475] wlan: [35: F:DP] underrun_cnt = 0
[  904.595238] wlan: [35: F:DP] flush_cnt = 0
[  904.599757] wlan: [35: F:DP] filt_cnt = 0
[  904.603777] wlan: [35: F:DP] null_mpdu_bmap = 0
[  904.607831] wlan: [35: F:DP] user_ack_failure = 0
[  904.612166] wlan: [35: F:DP] ack_tlv_proc = 0
[  904.617033] wlan: [35: F:DP] sched_id_proc = 0
[  904.621372] wlan: [35: F:DP] null_mpdu_tx_count = 0
[  904.625708] wlan: [35: F:DP] mpdu_bmap_not_recvd = 0
[  904.630496] wlan: [35: F:DP] num_bar = 0
[  904.635717] wlan: [35: F:DP] rts = 0
[  904.639607] wlan: [35: F:DP] cts2self = 0
[  904.643153] wlan: [35: F:DP] qos_null = 0
[  904.647066] wlan: [35: F:DP] mpdu_tried_cnt = 0
[  904.651044] wlan: [35: F:DP] mpdu_queued_cnt = 0
[  904.655440] wlan: [35: F:DP] mpdu_ack_fail_cnt = 0
[  904.660247] wlan: [35: F:DP] mpdu_filt_cnt = 0
[  904.664872] wlan: [35: F:DP] false_mpdu_ack_count = 0
[  904.664872]
[  904.669278] wlan: [35: F:DP] HTT_TX_HWQ_DIFS_LATENCY_STATS_TLV_V:
[  904.675817] wlan: [35: F:DP] hist_intvl = 20
[  904.681984] wlan: [35: F:DP] difs_latency_hist =  0:0,
[  904.681984]
[  904.686309] wlan: [35: F:DP] HTT_TX_HWQ_CMD_RESULT_STATS_TLV_V:
[  904.692686] wlan: [35: F:DP] cmd_result =  0:0,
[  904.698543] wlan: [35: F:DP] HTT_TX_HWQ_CMD_STALL_STATS_TLV_V:
[  904.703447] wlan: [35: F:DP] cmd_stall_status =  0:0,
[  904.703447]
[  904.709035] wlan: [35: F:DP] HTT_TX_HWQ_FES_RESULT_STATS_TLV_V:
[  904.715568] wlan: [35: F:DP] fes_result =  0:0,
[  904.721411] wlan: [35: F:DP] HTT_TX_HWQ_STATS_CMN_TLV:
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
[  904.726235] wlan: [35: F:DP] mac_id__hwq_id__word = 0
[  904.731195] wlan: [35: F:DP] xretry = 0
[  904.736328] wlan: [35: F:DP] underrun_cnt = 0
[  904.739934] wlan: [35: F:DP] flush_cnt = 0
[  904.744472] wlan: [35: F:DP] filt_cnt = 0
[  904.748472] wlan: [35: F:DP] null_mpdu_bmap = 0
[  904.752529] wlan: [35: F:DP] user_ack_failure = 0
[  904.756876] wlan: [35: F:DP] ack_tlv_proc = 0
[  904.761738] wlan: [35: F:DP] sched_id_proc = 0
[  904.766071] wlan: [35: F:DP] null_mpdu_tx_count = 0
[  904.770417] wlan: [35: F:DP] mpdu_bmap_not_recvd = 0
[  904.775192] wlan: [35: F:DP] num_bar = 0
[  904.780422] wlan: [35: F:DP] rts = 0
[  904.784298] wlan: [35: F:DP] cts2self = 0
[  904.787865] wlan: [35: F:DP] qos_null = 0
[  904.791756] wlan: [35: F:DP] mpdu_tried_cnt = 0
[  904.795838] wlan: [35: F:DP] mpdu_queued_cnt = 0
[  904.800091] wlan: [35: F:DP] mpdu_ack_fail_cnt = 0
[  904.804976] wlan: [35: F:DP] mpdu_filt_cnt = 0
[  904.809551] wlan: [35: F:DP] false_mpdu_ack_count = 0
[  904.809551]
[  904.814015] wlan: [35: F:DP] HTT_TX_HWQ_DIFS_LATENCY_STATS_TLV_V:
[  904.820515] wlan: [35: F:DP] hist_intvl = 20
[  904.826732] wlan: [35: F:DP] difs_latency_hist =  0:0,
[  904.826732]
[  904.831027] wlan: [35: F:DP] HTT_TX_HWQ_CMD_RESULT_STATS_TLV_V:
[  904.837357] wlan: [35: F:DP] cmd_result =  0:0,
[  904.843250] wlan: [35: F:DP] HTT_TX_HWQ_CMD_STALL_STATS_TLV_V:
[  904.848134] wlan: [35: F:DP] cmd_stall_status =  0:0,
[  904.848134]
[  904.853766] wlan: [35: F:DP] HTT_TX_HWQ_FES_RESULT_STATS_TLV_V:
[  904.860265] wlan: [35: F:DP] fes_result =  0:0,
[  904.866075] wlan: [35: F:DP] HTT_TX_HWQ_STATS_CMN_TLV:
[  904.870946] wlan: [35: F:DP] mac_id__hwq_id__word = 0
[  904.875900] wlan: [35: F:DP] xretry = 9
[  904.881013] wlan: [35: F:DP] underrun_cnt = 0
[  904.884661] wlan: [35: F:DP] flush_cnt = 15
[  904.889150] wlan: [35: F:DP] filt_cnt = 0
[  904.893288] wlan: [35: F:DP] null_mpdu_bmap = 0
[  904.897368] wlan: [35: F:DP] user_ack_failure = 0
[  904.901688] wlan: [35: F:DP] ack_tlv_proc = 15
[  904.906524] wlan: [35: F:DP] sched_id_proc = 989486
[  904.910870] wlan: [35: F:DP] null_mpdu_tx_count = 0
[  904.915651] wlan: [35: F:DP] mpdu_bmap_not_recvd = 15
[  904.920517] wlan: [35: F:DP] num_bar = 0
[  904.925733] wlan: [35: F:DP] rts = 0
[  904.929695] wlan: [35: F:DP] cts2self = 0
[  904.933256] wlan: [35: F:DP] qos_null = 0
[  904.937169] wlan: [35: F:DP] mpdu_tried_cnt = 24
[  904.941180] wlan: [35: F:DP] mpdu_queued_cnt = 24
[  904.945862] wlan: [35: F:DP] mpdu_ack_fail_cnt = 9
[  904.950447] wlan: [35: F:DP] mpdu_filt_cnt = 0
[  904.955138] wlan: [35: F:DP] false_mpdu_ack_count = 0
[  904.955138]
[  904.959572] wlan: [35: F:DP] HTT_TX_HWQ_DIFS_LATENCY_STATS_TLV_V:
[  904.966081] wlan: [35: F:DP] hist_intvl = 20
[  904.972273] wlan: [35: F:DP] difs_latency_hist =  0:0,
[  904.972273]
[  904.976626] wlan: [35: F:DP] HTT_TX_HWQ_CMD_RESULT_STATS_TLV_V:
[  904.982916] wlan: [35: F:DP] cmd_result =  0:15,
[  904.988852] wlan: [35: F:DP] HTT_TX_HWQ_CMD_STALL_STATS_TLV_V:
[  904.993717] wlan: [35: F:DP] cmd_stall_status =  0:0,
[  904.993717]
[  904.999442] wlan: [35: F:DP] HTT_SFM_CMN_TLV:
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
[  905.005934] wlan: [35: F:DP] mac_id__word = 15
[  905.010348] wlan: [35: F:DP] buf_total = 327686
[  905.014692] wlan: [35: F:DP] mem_empty = 9
[  905.019157] wlan: [35: F:DP] deallocate_bufs = 0
[  905.023291] wlan: [35: F:DP] num_records = 0
[  905.023291]
[  905.028060] wlan: [35: F:DP] HTT_TX_HWQ_STATS_CMN_TLV:
[  905.033709] wlan: [35: F:DP] mac_id__hwq_id__word = 9
[  905.038750] wlan: [35: F:DP] xretry = 0
[  905.043863] wlan: [35: F:DP] underrun_cnt = 0
[  905.047521] wlan: [35: F:DP] flush_cnt = 0
[  905.052031] wlan: [35: F:DP] filt_cnt = 0
[  905.055991] wlan: [35: F:DP] null_mpdu_bmap = 0
[  905.060094] wlan: [35: F:DP] user_ack_failure = 0
[  905.064435] wlan: [35: F:DP] ack_tlv_proc = 0
[  905.069304] wlan: [35: F:DP] sched_id_proc = 0
[  905.073623] wlan: [35: F:DP] null_mpdu_tx_count = 0
[  905.077971] wlan: [35: F:DP] mpdu_bmap_not_recvd = 0
[  905.082987] wlan: [35: F:DP] num_bar = 0
[  905.088004] wlan: [35: F:DP] rts = 0
[  905.091868] wlan: [35: F:DP] cts2self = 0
[  905.095401] wlan: [35: F:DP] qos_null = 0
[  905.099323] wlan: [35: F:DP] mpdu_tried_cnt = 0
[  905.103363] wlan: [35: F:DP] mpdu_queued_cnt = 0
[  905.107664] wlan: [35: F:DP] mpdu_ack_fail_cnt = 0
[  905.112528] wlan: [35: F:DP] mpdu_filt_cnt = 0
[  905.117135] wlan: [35: F:DP] false_mpdu_ack_count = 0
[  905.117135]
[  905.121535] wlan: [35: F:DP] HTT_TX_HWQ_DIFS_LATENCY_STATS_TLV_V:
[  905.128072] wlan: [35: F:DP] hist_intvl = 20
[  905.134272] wlan: [35: F:DP] difs_latency_hist =  0:0,
[  905.134272]
[  905.138550] wlan: [35: F:DP] HTT_TX_HWQ_CMD_RESULT_STATS_TLV_V:
[  905.144909] wlan: [35: F:DP] cmd_result =  0:0,
[  905.150847] wlan: [35: F:DP] HTT_TX_HWQ_CMD_STALL_STATS_TLV_V:
[  905.155656] wlan: [35: F:DP] cmd_stall_status =  0:0,
[  905.155656]
[  905.161325] wlan: [35: F:DP] HTT_TX_HWQ_FES_RESULT_STATS_TLV_V:
[  905.167840] wlan: [35: F:DP] fes_result =  0:0,
[  905.173637] wlan: [35: F:DP] HTT_TX_HWQ_STATS_CMN_TLV:
[  905.178476] wlan: [35: F:DP] mac_id__hwq_id__word = 0
[  905.183478] wlan: [35: F:DP] xretry = 0
[  905.188545] wlan: [35: F:DP] underrun_cnt = 0
[  905.192214] wlan: [35: F:DP] flush_cnt = 0
[  905.196703] wlan: [35: F:DP] filt_cnt = 0
[  905.200712] wlan: [35: F:DP] null_mpdu_bmap = 0
[  905.204836] wlan: [35: F:DP] user_ack_failure = 0
[  905.209146] wlan: [35: F:DP] ack_tlv_proc = 0
[  905.214006] wlan: [35: F:DP] sched_id_proc = 0
[  905.218344] wlan: [35: F:DP] null_mpdu_tx_count = 0
[  905.222684] wlan: [35: F:DP] mpdu_bmap_not_recvd = 0
[  905.227435] wlan: [35: F:DP] num_bar = 0
[  905.232696] wlan: [35: F:DP] rts = 0
[  905.236546] wlan: [35: F:DP] cts2self = 0
[  905.240122] wlan: [35: F:DP] qos_null = 0
[  905.244024] wlan: [35: F:DP] mpdu_tried_cnt = 0
[  905.248067] wlan: [35: F:DP] mpdu_queued_cnt = 0
[  905.252361] wlan: [35: F:DP] mpdu_ack_fail_cnt = 0
[  905.257237] wlan: [35: F:DP] mpdu_filt_cnt = 0
[  905.261834] wlan: [35: F:DP] false_mpdu_ack_count = 0
[  905.261834]
[  905.266298] wlan: [35: F:DP] HTT_TX_HWQ_DIFS_LATENCY_STATS_TLV_V:
[  905.272773] wlan: [35: F:DP] hist_intvl = 20
[  905.278941] wlan: [35: F:DP] difs_latency_hist =  0:0,
```

```
[  905.278941]
[  905.283278] wlan: [35: F:DP] HTT_TX_HWQ_CMD_RESULT_STATS_TLV_V:
[  905.289615] wlan: [35: F:DP] cmd_result =  0:0,
[  905.295501] wlan: [35: F:DP] HTT_TX_HWQ_CMD_STALL_STATS_TLV_V:
[  905.300375] wlan: [35: F:DP] cmd_stall_status =  0:0,
[  905.300375]
[  905.306070] wlan: [35: F:DP] HTT_TX_HWQ_FES_RESULT_STATS_TLV_V:
[  905.312531] wlan: [35: F:DP] fes_result =  0:0,
[  905.318334] wlan: [35: F:DP] HTT_TX_HWQ_STATS_CMN_TLV:
[  905.323205] wlan: [35: F:DP] mac_id__hwq_id__word = 0
[  905.328161] wlan: [35: F:DP] xretry = 0
[  905.333272] wlan: [35: F:DP] underrun_cnt = 0
[  905.336896] wlan: [35: F:DP] flush_cnt = 0
[  905.341423] wlan: [35: F:DP] filt_cnt = 0
[  905.345411] wlan: [35: F:DP] null_mpdu_bmap = 0
[  905.349504] wlan: [35: F:DP] user_ack_failure = 0
[  905.353846] wlan: [35: F:DP] ack_tlv_proc = 0
[  905.358713] wlan: [35: F:DP] sched_id_proc = 0
[  905.363046] wlan: [35: F:DP] null_mpdu_tx_count = 0
[  905.367365] wlan: [35: F:DP] mpdu_bmap_not_recvd = 0
[  905.372195] wlan: [35: F:DP] num_bar = 0
[  905.377346] wlan: [35: F:DP] rts = 0
[  905.381276] wlan: [35: F:DP] cts2self = 0
[  905.384824] wlan: [35: F:DP] qos_null = 0
[  905.388734] wlan: [35: F:DP] mpdu_tried_cnt = 0
[  905.392719] wlan: [35: F:DP] mpdu_queued_cnt = 0
[  905.397074] wlan: [35: F:DP] mpdu_ack_fail_cnt = 0
[  905.402037] wlan: [35: F:DP] mpdu_filt_cnt = 0
[  905.406545] wlan: [35: F:DP] false_mpdu_ack_count = 0
[  905.406545]
[  905.410998] wlan: [35: F:DP] HTT_TX_HWQ_DIFS_LATENCY_STATS_TLV_V:
[  905.417479] wlan: [35: F:DP] hist_intvl = 20
[  905.423668] wlan: [35: F:DP] difs_latency_hist =  0:0,
[  905.423668]
[  905.427960] wlan: [35: F:DP] HTT_TX_HWQ_CMD_RESULT_STATS_TLV_V:
[  905.434319] wlan: [35: F:DP] cmd_result =  0:0,
[  905.440217] wlan: [35: F:DP] HTT_TX_HWQ_CMD_STALL_STATS_TLV_V:
[  905.445083] wlan: [35: F:DP] cmd_stall_status =  0:0,
[  905.445083]
[  905.450720] wlan: [35: F:DP] HTT_TX_HWQ_FES_RESULT_STATS_TLV_V:
[  905.457249] wlan: [35: F:DP] fes_result =  0:0,
[  905.463048] wlan: [35: F:DP] HTT_TX_HWQ_STATS_CMN_TLV:
[  905.467927] wlan: [35: F:DP] mac_id__hwq_id__word = 0
[  905.472912] wlan: [35: F:DP] xretry = 0
[  905.477953] wlan: [35: F:DP] underrun_cnt = 0
[  905.481615] wlan: [35: F:DP] flush_cnt = 0
[  905.486145] wlan: [35: F:DP] filt_cnt = 0
[  905.490133] wlan: [35: F:DP] null_mpdu_bmap = 0
[  905.494203] wlan: [35: F:DP] user_ack_failure = 0
[  905.498557] wlan: [35: F:DP] ack_tlv_proc = 0
[  905.503427] wlan: [35: F:DP] sched_id_proc = 0
[  905.507729] wlan: [35: F:DP] null_mpdu_tx_count = 0
[  905.512128] wlan: [35: F:DP] mpdu_bmap_not_recvd = 0
[  905.516878] wlan: [35: F:DP] num_bar = 0
[  905.522077] wlan: [35: F:DP] rts = 0
[  905.525956] wlan: [35: F:DP] cts2self = 0
[  905.529531] wlan: [35: F:DP] qos_null = 0
[  905.533432] wlan: [35: F:DP] mpdu_tried_cnt = 0
[  905.537432] wlan: [35: F:DP] mpdu_queued_cnt = 0
[  905.541804] wlan: [35: F:DP] mpdu_ack_fail_cnt = 0
[  905.546646] wlan: [35: F:DP] mpdu_filt_cnt = 0
[  905.551234] wlan: [35: F:DP] false_mpdu_ack_count = 0
[  905.551234]
[  905.555675] wlan: [35: F:DP] HTT_TX_HWQ_DIFS_LATENCY_STATS_TLV_V:
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
[  905.562181] wlan: [35: F:DP] hist_intvl = 20
[  905.568352] wlan: [35: F:DP] difs_latency_hist =  0:0,
[  905.568352]
[  905.572687] wlan: [35: F:DP] HTT_TX_HWQ_CMD_RESULT_STATS_TLV_V:
[  905.579014] wlan: [35: F:DP] cmd_result =  0:0,
[  905.584907] wlan: [35: F:DP] HTT_TX_HWQ_CMD_STALL_STATS_TLV_V:
[  905.589785] wlan: [35: F:DP] cmd_stall_status =  0:0,
[  905.589785]
[  905.595442] wlan: [35: F:DP] HTT_TX_HWQ_FES_RESULT_STATS_TLV_V:
[  905.601953] wlan: [35: F:DP] fes_result =  0:0,
[  905.607730] wlan: [35: F:DP] HTT_TX_HWQ_STATS_CMN_TLV:
[  905.612698] wlan: [35: F:DP] mac_id__hwq_id__word = 0
[  905.617539] wlan: [35: F:DP] xretry = 0
[  905.622683] wlan: [35: F:DP] underrun_cnt = 0
[  905.626304] wlan: [35: F:DP] flush_cnt = 0
[  905.630835] wlan: [35: F:DP] filt_cnt = 0
[  905.634843] wlan: [35: F:DP] null_mpdu_bmap = 0
[  905.638899] wlan: [35: F:DP] user_ack_failure = 0
[  905.643272] wlan: [35: F:DP] ack_tlv_proc = 0
[  905.648110] wlan: [35: F:DP] sched_id_proc = 0
[  905.652458] wlan: [35: F:DP] null_mpdu_tx_count = 0
[  905.656774] wlan: [35: F:DP] mpdu_bmap_not_recvd = 0
[  905.661596] wlan: [35: F:DP] num_bar = 0
[  905.666764] wlan: [35: F:DP] rts = 0
[  905.670676] wlan: [35: F:DP] cts2self = 0
[  905.674229] wlan: [35: F:DP] qos_null = 0
[  905.678143] wlan: [35: F:DP] mpdu_tried_cnt = 0
[  905.682129] wlan: [35: F:DP] mpdu_queued_cnt = 0
[  905.686484] wlan: [35: F:DP] mpdu_ack_fail_cnt = 0
[  905.691336] wlan: [35: F:DP] mpdu_filt_cnt = 0
[  905.696051] wlan: [35: F:DP] false_mpdu_ack_count = 0
[  905.696051]
[  905.700355] wlan: [35: F:DP] HTT_TX_HWQ_DIFS_LATENCY_STATS_TLV_V:
[  905.706888] wlan: [35: F:DP] hist_intvl = 20
[  905.713078] wlan: [35: F:DP] difs_latency_hist =  0:0,
[  905.713078]
[  905.717419] wlan: [35: F:DP] HTT_TX_HWQ_CMD_RESULT_STATS_TLV_V:
[  905.723724] wlan: [35: F:DP] cmd_result =  0:0,
[  905.729620] wlan: [35: F:DP] HTT_TX_HWQ_CMD_STALL_STATS_TLV_V:
[  905.734497] wlan: [35: F:DP] cmd_stall_status =  0:0,
[  905.734497]
[  905.740133] wlan: [35: F:DP] HTT_TX_HWQ_FES_RESULT_STATS_TLV_V:
[  905.746651] wlan: [35: F:DP] fes_result =  0:0,
[  905.752461] wlan: [35: F:DP] HTT_TX_HWQ_STATS_CMN_TLV:
[  905.757327] wlan: [35: F:DP] mac_id__hwq_id__word = 0
[  905.762268] wlan: [35: F:DP] xretry = 0
[  905.767377] wlan: [35: F:DP] underrun_cnt = 0
[  905.771009] wlan: [35: F:DP] flush_cnt = 0
[  905.775546] wlan: [35: F:DP] filt_cnt = 0
[  905.779515] wlan: [35: F:DP] null_mpdu_bmap = 0
[  905.783638] wlan: [35: F:DP] user_ack_failure = 0
[  905.787946] wlan: [35: F:DP] ack_tlv_proc = 0
[  905.792828] wlan: [35: F:DP] sched_id_proc = 0
[  905.797138] wlan: [35: F:DP] null_mpdu_tx_count = 0
[  905.801494] wlan: [35: F:DP] mpdu_bmap_not_recvd = 0
[  905.806293] wlan: [35: F:DP] num_bar = 0
[  905.811475] wlan: [35: F:DP] rts = 0
[  905.815375] wlan: [35: F:DP] cts2self = 0
[  905.818996] wlan: [35: F:DP] qos_null = 0
[  905.822842] wlan: [35: F:DP] mpdu_tried_cnt = 0
[  905.826841] wlan: [35: F:DP] mpdu_queued_cnt = 0
[  905.831173] wlan: [35: F:DP] mpdu_ack_fail_cnt = 0
[  905.836072] wlan: [35: F:DP] mpdu_filt_cnt = 0
[  905.840635] wlan: [35: F:DP] false_mpdu_ack_count = 0
```

```
[  905.840635]
[  905.845089] wlan: [35: F:DP] HTT_TX_HWQ_DIFS_LATENCY_STATS_TLV_V:
[  905.851584] wlan: [35: F:DP] hist_intvl = 20
[  905.857793] wlan: [35: F:DP] difs_latency_hist =  0:0,
[  905.857793]
[  905.862082] wlan: [35: F:DP] HTT_TX_HWQ_CMD_RESULT_STATS_TLV_V:
[  905.868424] wlan: [35: F:DP] cmd_result =  0:0,
[  905.874332] wlan: [35: F:DP] HTT_TX_HWQ_CMD_STALL_STATS_TLV_V:
[  905.879180] wlan: [35: F:DP] cmd_stall_status =  0:0,
[  905.879180]
[  905.884864] wlan: [35: F:DP] HTT_TX_HWQ_FES_RESULT_STATS_TLV_V:
[  905.891340] wlan: [35: F:DP] fes_result =  0:0
```

3.23.1.9 Firmware TQM statistics

```
root@OpenWrt:/# iwpriv wlan0 txrx_stats 7 <cmdq>
[  912.741913] wlan: [35: F:DP] mac_id__word = 0
[  912.745960] wlan: [35: F:DP] max_cmdq_id = 0
[  912.750384] wlan: [35: F:DP] list_mpdu_cnt_hist_intvl = 30
[  912.754704] wlan: [35: F:DP] add_msdu = 0
[  912.759991] wlan: [35: F:DP] q_empty = 15
[  912.764108] wlan: [35: F:DP] q_not_empty = 15
[  912.768066] wlan: [35: F:DP] drop_notification = 0
[  912.772401] wlan: [35: F:DP] desc_threshold = 0
[  912.772401]
[  912.777098] wlan: [35: F:DP] HTT_TX_TQM_ERROR_STATS_TLV:
[  912.783376] wlan: [35: F:DP] q_empty_failure = 0
[  912.788546] wlan: [35: F:DP] q_not_empty_failure = 0
[  912.793180] wlan: [35: F:DP] add_msdu_failure = 0
[  912.793180]
[  912.798110] wlan: [35: F:DP] HTT_TX_TQM_GEN_MPDU_STATS_TLV_V:
[  912.804112] wlan: [35: F:DP] gen_mpdu_end_reason =  0:0, 1:0, 2:0, 3:0, 4:0,
5:0, 6:0, 7:0, 8:0, 9:72, 10:0, 11:0, 12:0, 13:0, 14:0, 15:0,
[  912.804112]
[  912.809978] wlan: [35: F:DP] HTT_TX_TQM_LIST_MPDU_STATS_TLV_V:
[  912.823676] wlan: [35: F:DP] list_mpdu_end_reason =  0:39, 1:0, 2:0, 3:0, 4:0,
5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0, 12:39, 13:0, 14:0, 15:0,
[  912.823676]
[  912.829528] wlan: [35: F:DP] HTT_TX_TQM_LIST_MPDU_CNT_TLV_V:
[  912.843729] wlan: [35: F:DP] list_mpdu_cnt_hist =  0:39, 1:0, 2:0, 3:0, 4:0,
5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0, 12:0, 13:0, 14:0, 15:0,
[  912.843729]
[  912.849427] wlan: [35: F:DP] HTT_TX_TQM_PDEV_STATS_TLV_V:
[  912.863078] wlan: [35: F:DP] msdu_count = 15
[  912.868500] wlan: [35: F:DP] mpdu_count = 15
[  912.872850] wlan: [35: F:DP] remove_msdu = 0
[  912.877078] wlan: [35: F:DP] remove_mpdu = 0
[  912.881345] wlan: [35: F:DP] remove_msdu_ttl = 0
[  912.885665] wlan: [35: F:DP] send_bar = 0
[  912.890215] wlan: [35: F:DP] bar_sync = 0
[  912.894118] wlan: [35: F:DP] notify_mpdu = 0
[  912.898099] wlan: [35: F:DP] sync_cmd = 79
[  912.902432] wlan: [35: F:DP] write_cmd = 78
[  912.906345] wlan: [35: F:DP] hwsch_trigger = 0
[  912.910443] wlan: [35: F:DP] ack_tlv_proc = 15
[  912.914948] wlan: [35: F:DP] gen_mpdu_cmd = 72
[  912.919397] wlan: [35: F:DP] gen_list_cmd = 39
```

```
[  912.923802] wlan: [35: F:DP] remove_mpdu_cmd = 1
[  912.928207] wlan: [35: F:DP] remove_mpdu_tried_cmd = 1
[  912.933013] wlan: [35: F:DP] mpdu_queue_stats_cmd = 0
[  912.937944] wlan: [35: F:DP] mpdu_head_info_cmd = 0
[  912.943091] wlan: [35: F:DP] msdu_flow_stats_cmd = 2645
[  912.947756] wlan: [35: F:DP] remove_msdu_cmd = 0
[  912.952955] wlan: [35: F:DP] remove_msdu_ttl_cmd = 0
[  912.957880] wlan: [35: F:DP] flush_cache_cmd = 0
[  912.962767] wlan: [35: F:DP] update_mpduq_cmd = 0
[  912.967371] wlan: [35: F:DP] enqueue = 15
[  912.972054] wlan: [35: F:DP] enqueue_notify = 0
[  912.975950] wlan: [35: F:DP] notify_mpdu_at_head = 0
[  912.980400] wlan: [35: F:DP] notify_mpdu_state_valid = 0
```

3.23.1.10 Firmware Tx classification (tx_de) statistics

```
root@OpenWrt:/# iwpriv ath0 txrx_stats 8

[  918.324785] wlan: [35: F:DP] HTT_TX_DE_CMN_STATS_TLV:
[  918.324814] wlan: [35: F:DP] mac_id__word = 0
[  918.328816] wlan: [35: F:DP] tcl2fw_entry_count = 15
[  918.333251] wlan: [35: F:DP] not_to_fw = 0
[  918.338206] wlan: [35: F:DP] invalid_pdev_vdev_peer = 0
[  918.342114] wlan: [35: F:DP] tcl_res_invalid_addrx = 0
[  918.347234] wlan: [35: F:DP] wbm2fw_entry_count = 0
[  918.352437] wlan: [35: F:DP] invalid_pdev = 0
[  918.352437]
[  918.357229] wlan: [35: F:DP] HTT_TX_DE_EAPOL_PACKETS_STATS_TLV:
[  918.363155] wlan: [35: F:DP] m1_packets = 0
[  918.368940] wlan: [35: F:DP] m2_packets = 0
[  918.373133] wlan: [35: F:DP] m3_packets = 0
[  918.377252] wlan: [35: F:DP] m4_packets = 0
[  918.381468] wlan: [35: F:DP] g1_packets = 0
[  918.385598] wlan: [35: F:DP] g2_packets = 0
[  918.385598]
[  918.389772] wlan: [35: F:DP] HTT_TX_DE_CLASSIFY_STATS_TLV:
[  918.395681] wlan: [35: F:DP] arp_packets = 0
[  918.400968] wlan: [35: F:DP] igmp_packets = 0
[  918.405408] wlan: [35: F:DP] dhcp_packets = 0
[  918.409645] wlan: [35: F:DP] host_inspected = 0
[  918.414135] wlan: [35: F:DP] htt_included = 0
[  918.418312] wlan: [35: F:DP] htt_valid_mcs = 0
[  918.422852] wlan: [35: F:DP] htt_valid_nss = 0
[  918.427166] wlan: [35: F:DP] htt_valid_preamble_type = 0
[  918.431643] wlan: [35: F:DP] htt_valid_chainmask = 0
[  918.437089] wlan: [35: F:DP] htt_valid_guard_interval = 0
[  918.442019] wlan: [35: F:DP] htt_valid_retries = 0
[  918.447320] wlan: [35: F:DP] htt_valid_bw_info = 0
[  918.452007] wlan: [35: F:DP] htt_valid_power = 0
[  918.456796] wlan: [35: F:DP] htt_valid_key_flags = 0
[  918.461586] wlan: [35: F:DP] htt_valid_no_encryption = 0
[  918.466501] wlan: [35: F:DP] fse_entry_count = 0
[  918.471864] wlan: [35: F:DP] fse_priority_be = 0
```

```
[  918.476419] wlan: [35: F:DP] fse_priority_high = 0
[  918.481001] wlan: [35: F:DP] fse_priority_low = 0
[  918.485621] wlan: [35: F:DP] fse_traffic_ptrn_be = 0
[  918.490378] wlan: [35: F:DP] fse_traffic_ptrn_over_sub = 0
[  918.495422] wlan: [35: F:DP] fse_traffic_ptrn_bursty = 0
[  918.500700] wlan: [35: F:DP] fse_traffic_ptrn_interactive = 0
[  918.506199] wlan: [35: F:DP] fse_traffic_ptrn_periodic = 0
[  918.511841] wlan: [35: F:DP] fse_hwqueue_alloc = 0
[  918.517237] wlan: [35: F:DP] fse_hwqueue_created = 0
[  918.522100] wlan: [35: F:DP] fse_hwqueue_send_to_host = 0
[  918.527212] wlan: [35: F:DP] mcast_entry = 15
[  918.532403] wlan: [35: F:DP] bcast_entry = 0
[  918.532403]
[  918.536734] wlan: [35: F:DP] HTT_TX_DE_CLASSIFY_FAILED_STATS_TLV:
[  918.542465] wlan: [35: F:DP] ap_bss_peer_not_found = 0
[  918.548543] wlan: [35: F:DP] ap_bcast_mcast_no_peer = 0
[  918.553594] wlan: [35: F:DP] sta_delete_in_progress = 0
[  918.558694] wlan: [35: F:DP] ibss_no_bss_peer = 0
[  918.563909] wlan: [35: F:DP] invaild_vdev_type = 0
[  918.568762] wlan: [35: F:DP] invalid_ast_peer_entry = 0
[  918.573472] wlan: [35: F:DP] peer_entry_invalid = 0
[  918.578571] wlan: [35: F:DP] ethertype_not_ip = 0
[  918.583430] wlan: [35: F:DP] eapol_lookup_failed = 0
[  918.588293] wlan: [35: F:DP] qpeer_not_allow_data = 0
[  918.593401] wlan: [35: F:DP] fse_tid_override = 0
[  918.593401]
[  918.598275] wlan: [35: F:DP] HTT_TX_DE_CLASSIFY_STATUS_STATS_TLV:
[  918.604387] wlan: [35: F:DP] eok = 0
[  918.610512] wlan: [35: F:DP] classify_done = 15
[  918.614153] wlan: [35: F:DP] lookup_failed = 0
[  918.618470] wlan: [35: F:DP] send_host_dhcp = 0
[  918.622953] wlan: [35: F:DP] send_host_mcast = 0
[  918.627357] wlan: [35: F:DP] send_host_unknown_dest = 0
[  918.632212] wlan: [35: F:DP] send_host = 0
[  918.637162] wlan: [35: F:DP] status_invalid = 0
[  918.637162]
[  918.641320] wlan: [35: F:DP] HTT_TX_DE_ENQUEUE_PACKETS_STATS_TLV:
[  918.647511] wlan: [35: F:DP] enqueued_pkts = 255
[  918.653516] wlan: [35: F:DP] to_tqm = 15
[  918.658187] wlan: [35: F:DP] to_tqm_bypass = 240
[  918.658187]
[  918.662089] wlan: [35: F:DP] HTT_TX_DE_ENQUEUE_DISCARD_STATS_TLV:
[  918.668100] wlan: [35: F:DP] discarded_pkts = 0
[  918.674138] wlan: [35: F:DP] local_frames = 0
[  918.678488] wlan: [35: F:DP] is_ext_msdu = 0
[  918.678488]
[  918.682993] wlan: [35: F:DP] HTT_TX_DE_COMPL_STATS_TLV:
[  918.688727] wlan: [35: F:DP] tcl_dummy_frame = 0
[  918.693781] wlan: [35: F:DP] tqm_dummy_frame = 0
[  918.698624] wlan: [35: F:DP] tqm_notify_frame = 0
[  918.703252] wlan: [35: F:DP] fw2wbm_enq = 15
[  918.707809] wlan: [35: F:DP] tqm_bypass_frame = 0
[  918.707809]
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

3.23.1.11 Firmware Tx Rate statistics

```
root@OpenWrt:/# iwpriv ath0 txrx_stats 9
[  921.805149] wlan: [35: F:DP] HTT_TX_PDEV_RATE_STATS_TLV:
[  921.805179] wlan: [35: F:DP] mac_id__word = 0
[  921.809566] wlan: [35: F:DP] tx_ldpc = 24
[  921.813831] wlan: [35: F:DP] rts_cnt = 0
[  921.817811] wlan: [35: F:DP] ack_rssi = 0
[  921.821820] wlan: [35: F:DP] tx_mcs =  0:0, 1:0, 2:0, 3:0, 4:0, 5:14, 6:0,
7:10, 8:0, 9:0, 10:0, 11:0,
[  921.825704] wlan: [35: F:DP] tx_su_mcs =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0,
7:0, 8:0, 9:0, 10:0, 11:0,
[  921.834917] wlan: [35: F:DP] tx_mu_mcs =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0,
7:0, 8:0, 9:0, 10:0, 11:0,
[  921.844687] wlan: [35: F:DP] tx_nss =  0:0, 1:0, 2:0, 3:0, 4:0, 5:2, 6:8,
7:14,
[  921.854094] wlan: [35: F:DP] tx_bw =  0:0, 1:0, 2:24, 3:0,
[  921.861468] wlan: [35: F:DP] tx_stbc =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0,
7:0, 8:0, 9:0, 10:0, 11:0,
[  921.866786] wlan: [35: F:DP] tx_pream =  0:682, 1:0, 2:0, 3:24, 4:0, 5:0, 6:0,

[  921.876160] wlan: [35: F:DP] tx_gi[0] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0,
7:0, 8:0, 9:0, 10:0, 11:0,
[  921.883352] wlan: [35: F:DP] tx_gi[1] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:14, 6:0,
7:10, 8:0, 9:0, 10:0, 11:0,
[  921.892761] wlan: [35: F:DP] tx_gi[2] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0,
7:0, 8:0, 9:0, 10:0, 11:0,
[  921.902589] wlan: [35: F:DP] tx_gi[3] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0,
7:0, 8:0, 9:0, 10:0, 11:0,
[  921.911645] wlan: [35: F:DP] tx_dcm =  0:0, 1:0, 2:0, 3:0, 4:0,
```

It is possible to also display the firmware Tx rate stats information using the command: wifistats wifi0 9. Display the firmware Tx rate stats information using the wifistats wifi0 9 command.

```
root@OpenWrt:/# wifistats wifi0 9

HTT_TX_PDEV_RATE_STATS_TLV:
mac_id = 0
tx_ldpc = 0
ac_mu_mimo_tx_ldpc = 0
ax_mu_mimo_tx_ldpc = 0
ofdma_tx_ldpc = 0
rts_cnt = 0
rts_success = 0
ack_rssi = 96
tx_11ax_su_ext = 0
Legacy CCK Rates: 1 Mbps: 0, 2 Mbps: 0, 5.5 Mbps: 0, 11 Mbps: 0
Legacy OFDM Rates: 6 Mbps: 237, 9 Mbps: 0, 12 Mbps: 0, 18 Mbps: 0
                   24 Mbps: 0, 36 Mbps: 0, 48 Mbps: 0, 54 Mbps: 0
tx_mcs =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0, 12:0,
13:0,
ac_mu_mimo_tx_mcs =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0,
11:0, 12:0, 13:0,
```

```
ax_mu_mimo_tx_mcs =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0,
11:0, 12:0, 13:0,
ofdma_tx_mcs =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0,
12:0, 13:0,
tx_nss =  1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0,
ac_mu_mimo_tx_nss =  1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0,
ax_mu_mimo_tx_nss =  1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0,
ofdma_tx_nss =  1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0,
tx_bw =  0:0, 1:0, 2:0, 3:0,
ac_mu_mimo_tx_bw =  0:0, 1:0, 2:0, 3:0,
ax_mu_mimo_tx_bw =  0:0, 1:0, 2:0, 3:0,
ofdma_tx_bw =  0:0, 1:0, 2:0, 3:0,
tx_stbc =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0, 12:0,
13:0,
tx_pream =  0:237, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0,
HE LTF: 1x: 0, 2x: 0, 4x: 0
tx_gi[0] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0, 12:0,
13:0,
tx_gi[1] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0, 12:0,
13:0,
tx_gi[2] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0, 12:0,
13:0,
tx_gi[3] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0, 12:0,
13:0,
ac_mu_mimo_tx_gi[0] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0,
11:0,
ac_mu_mimo_tx_gi[1] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0,
11:0,
ac_mu_mimo_tx_gi[2] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0,
11:0,
ac_mu_mimo_tx_gi[3] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0,
11:0,
ax_mu_mimo_tx_gi[0] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0,
11:0, 12:0, 13:0,
ax_mu_mimo_tx_gi[1] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0,
11:0, 12:0, 13:0,
ax_mu_mimo_tx_gi[2] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0,
11:0, 12:0, 13:0,
ax_mu_mimo_tx_gi[3] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0,
11:0, 12:0, 13:0,
ofdma_tx_gi[0] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0,
12:0, 13:0,
ofdma_tx_gi[1] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0,
12:0, 13:0,
ofdma_tx_gi[2] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0,
12:0, 13:0,
ofdma_tx_gi[3] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0,
12:0, 13:0,
tx_dcm =  0:0, 1:0, 2:0, 3:0, 4:0,
11ax_trigger_type =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0,
```

Qualcomm Confidential - May Contain Trade Secrets 2021-05-13 03:37:44 PDT pandyafen@zte.com.cn

3.23.1.12 Firmware Rx Rate statistics

```
root@OpenWrt:/# iwpriv wlan0 txrx_stats 10 <mac_id>

[  924.554582] wlan: [35: F:DP] HTT_RX_PDEV_RATE_STATS_TLV:
[  924.554611] wlan: [35: F:DP] mac_id__word = 0
[  924.559135] wlan: [35: F:DP] nsts = 0
[  924.563258] wlan: [35: F:DP] rx_ldpc = 0
[  924.566858] wlan: [35: F:DP] rts_cnt = 0
[  924.570866] wlan: [35: F:DP] rssi_mgmt = 0
[  924.574808] wlan: [35: F:DP] rssi_data = 0
[  924.578679] wlan: [35: F:DP] rssi_comb = 0
[  924.582760] wlan: [35: F:DP] rx_mcs =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
8:0, 9:0, 10:0, 11:0,
[  924.586892] wlan: [35: F:DP] rx_nss =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,

[  924.596154] wlan: [35: F:DP] rx_dcm =  0:0, 1:0, 2:0, 3:0, 4:0,
[  924.603286] wlan: [35: F:DP] rx_stbc =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0,
7:0, 8:0, 9:0, 10:0, 11:0,
[  924.609609] wlan: [35: F:DP] rx_bw =  0:0, 1:0, 2:0, 3:0,
[  924.618626] wlan: [35: F:DP] rssi_chain[0] =  0:0, 1:0, 2:0, 3:0,
[  924.624123] wlan: [35: F:DP] rssi_chain[1] =  0:0, 1:0, 2:0, 3:0,
[  924.630248] wlan: [35: F:DP] rssi_chain[2] =  0:0, 1:0, 2:0, 3:0,
[  924.636422] wlan: [35: F:DP] rssi_chain[3] =  0:0, 1:0, 2:0, 3:0,
[  924.642571] wlan: [35: F:DP] rssi_chain[4] =  0:0, 1:0, 2:0, 3:0,
[  924.648759] wlan: [35: F:DP] rssi_chain[5] =  0:0, 1:0, 2:0, 3:0,
[  924.654926] wlan: [35: F:DP] rssi_chain[6] =  0:0, 1:0, 2:0, 3:0,
[  924.661123] wlan: [35: F:DP] rssi_chain[7] =  0:0, 1:0, 2:0, 3:0,
[  924.667227] wlan: [35: F:DP] rx_gi[0] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0,
7:0, 8:0, 9:0, 10:0, 11:0,
[  924.673438] wlan: [35: F:DP] rx_gi[1] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0,
7:0, 8:0, 9:0, 10:0, 11:0,
[  924.682714] wlan: [35: F:DP] rx_gi[2] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0,
7:0, 8:0, 9:0, 10:0, 11:0,
[  924.692079] wlan: [35: F:DP] rx_gi[3] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0,
7:0, 8:0, 9:0, 10:0, 11:0,
[  924.701461] wlan: [35: F:DP] rx_pream =  0:5642, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0,
```

It is possible to also display the firmware Rx stats information using the command: wifistats wifi0 10. Display the firmware Rx rate stats information using the wifistats wifi0 10 command.
Some of the newly added Rx rate stats do no fit under wifistats 10. These Rx rate stats are made as part of Rx extended rate stats wifistats wifi0 30.
Especially MCS counters have space only until MCS 11 in wifistats 10. Some hardware support MCS up to 13.. Those counts are only available in wifistats 30.

```
root@OpenWrt:/# wifistats wifi0 10

HTT_RX_PDEV_RATE_STATS_TLV:
mac_id = 0
nsts = 0
rx_ldpc = 0
rts_cnt = 0
rssi_mgmt = 96
rssi_data = 96
rssi_comb = 0
```

```
rssi_in_dbm = 10
rx_mcs =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0,
rx_nss =  1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0,
rx_dcm =  0:0, 1:0, 2:0, 3:0, 4:0,
rx_stbc =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0,
rx_bw =  0:2813, 1:0, 2:0, 3:0,
rx_evm_nss_count = 0
rx_evm_pilot_count = 0
pilot_evm_dB[0] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0,
12:0, 13:0, 14:0, 15:0,
pilot_evm_dB[1] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0,
12:0, 13:0, 14:0, 15:0,
pilot_evm_dB[2] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0,
12:0, 13:0, 14:0, 15:0,
pilot_evm_dB[3] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0,
12:0, 13:0, 14:0, 15:0,
pilot_evm_dB[4] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0,
12:0, 13:0, 14:0, 15:0,
pilot_evm_dB[5] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0,
12:0, 13:0, 14:0, 15:0,
pilot_evm_dB[6] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0,
12:0, 13:0, 14:0, 15:0,
pilot_evm_dB[7] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0,
12:0, 13:0, 14:0, 15:0,
pilot_evm_dB_mean =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
rssi_chain[0] =  0:128, 1:128, 2:128, 3:128,
rssi_chain[1] =  0:8, 1:10, 2:10, 3:8,
rssi_chain[2] =  0:128, 1:128, 2:128, 3:128,
rssi_chain[3] =  0:128, 1:128, 2:128, 3:128,
rssi_chain[4] =  0:128, 1:128, 2:128, 3:128,
rssi_chain[5] =  0:128, 1:128, 2:128, 3:128,
rssi_chain[6] =  0:9, 1:12, 2:12, 3:10,
rssi_chain[7] =  0:3, 1:5, 2:6, 3:8,
rx_gi[0] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0,
rx_gi[1] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0,
rx_gi[2] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0,
rx_gi[3] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0,
rx_pream =  0:2813, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0,
rx_11ax_su_ext = 0
rx_11ac_mumimo = 0
rx_11ax_mumimo = 0
rx_11ax_ofdma = 0
txbf = 0
rx_su_ndpa = 0
rx_11ax_su_txbf_mcs =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0,
11:0,
rx_mu_ndpa = 42
rx_11ax_mu_txbf_mcs =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0,
11:0,
rx_br_poll = 0
rx_legacy_cck_rate =  0:0, 1:0, 2:0, 3:0,
rx_legacy_ofdm_rate =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
rx_active_dur_us_low = 0
```

```
rx_active_dur_us_high = 0
rx_11ax_ul_ofdma = 0
ul_ofdma_rx_mcs =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0,
ul_ofdma_rx_gi[0] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0,
11:0,
ul_ofdma_rx_gi[1] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0,
11:0,
ul_ofdma_rx_gi[2] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0,
11:0,
ul_ofdma_rx_gi[3] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0,
11:0,
ul_ofdma_rx_nss =  1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0,
ul_ofdma_rx_bw =  0:0, 1:0, 2:0, 3:0,
ul_ofdma_rx_stbc = 0
ul_ofdma_rx_ldpc = 0
rx_ulofdma_non_data_ppdu =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
rx_ulofdma_data_ppdu =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
rx_ulofdma_mpdu_ok =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
rx_ulofdma_mpdu_fail =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
rx_ul_fd_rssi: nss[0] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
rx_ul_fd_rssi: nss[1] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
rx_ul_fd_rssi: nss[2] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
rx_ul_fd_rssi: nss[3] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
rx_ul_fd_rssi: nss[4] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
rx_ul_fd_rssi: nss[5] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
rx_ul_fd_rssi: nss[6] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
rx_ul_fd_rssi: nss[7] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
per_chain_rssi_pkt_type = 0
rx_per_chain_rssi_in_dbm[0] =  0:128, 1:128, 2:128, 3:128,
rx_per_chain_rssi_in_dbm[1] =  0:168, 1:170, 2:170, 3:168,
rx_per_chain_rssi_in_dbm[2] =  0:128, 1:128, 2:128, 3:128,
rx_per_chain_rssi_in_dbm[3] =  0:128, 1:128, 2:128, 3:128,
rx_per_chain_rssi_in_dbm[4] =  0:128, 1:128, 2:128, 3:128,
rx_per_chain_rssi_in_dbm[5] =  0:128, 1:128, 2:128, 3:128,
rx_per_chain_rssi_in_dbm[6] =  0:169, 1:172, 2:172, 3:170,
rx_per_chain_rssi_in_dbm[7] =  0:163, 1:165, 2:166, 3:168,
rx_11ax_dl_ofdma_mcs =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0,
11:0,
rx_11ax_dl_ofdma_ru =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0,

root@OpenWrt:/# wifistats wifi0 30
rssi_mcast_in_dbm = 71
rssi_mgmt_in_dbm = -93
rssi_chain_ext[0] =  0:128, 1:128, 2:128, 3:128,
rssi_chain_ext[1] =  0:128, 1:128, 2:128, 3:128,
rssi_chain_ext[2] =  0:128, 1:128, 2:128, 3:128,
rssi_chain_ext[3] =  0:128, 1:128, 2:128, 3:128,
rssi_chain_ext[4] =  0:0, 1:0, 2:0, 3:0,
rssi_chain_ext[5] =  0:128, 1:0, 2:0, 3:0,
rssi_chain_ext[6] =  0:128, 1:0, 2:0, 3:0,
rssi_chain_ext[7] =  0:128, 1:0, 2:0, 3:0,
rx_per_chain_rssi_ext_in_dbm[0] =  0:128, 1:0, 2:0, 3:0,
rx_per_chain_rssi_ext_in_dbm[1] =  0:0, 1:0, 2:0, 3:0,
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
rx_per_chain_rssi_ext_in_dbm[2] =  0:0, 1:0, 2:0, 3:0,
rx_per_chain_rssi_ext_in_dbm[3] =  0:0, 1:0, 2:0, 3:0,
rx_per_chain_rssi_ext_in_dbm[4] =  0:224, 1:46, 2:69, 3:1,
rx_per_chain_rssi_ext_in_dbm[5] =  0:16, 1:227, 2:92, 3:1,
rx_per_chain_rssi_ext_in_dbm[6] =  0:0, 1:0, 2:0, 3:0,
rx_per_chain_rssi_ext_in_dbm[7] =  0:0, 1:0, 2:0, 3:0,
rx_mcs_ext =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0, 12:0,
13:0,
rx_stbc_ext =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0,
12:0, 13:0,
rx_gi_ext[0] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0,
12:0, 13:0,
rx_gi_ext[1] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0,
12:0, 13:0,
rx_gi_ext[2] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0,
12:0, 13:0,
rx_gi_ext[3] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0,
12:0, 13:0,
ul_ofdma_rx_mcs_ext =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0,
11:0, 12:0, 13:0,
ul_ofdma_rx_gi_ext[0] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0,
11:0, 12:0, 13:0,
ul_ofdma_rx_gi_ext[1] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0,
11:0, 12:0, 13:0,
ul_ofdma_rx_gi_ext[2] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0,
11:0, 12:0, 13:0,
ul_ofdma_rx_gi_ext[3] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0,
11:0, 12:0, 13:0,
rx_11ax_su_txbf_mcs =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0,
11:0, 12:0, 13:0,
rx_11ax_mu_txbf_mcs_ext =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0,
10:0, 11:0, 12:0, 13:0,
rx_11ax_dl_ofdma_mcs_ext =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0,
10:0, 11:0, 12:0, 13:0,
```

3.23.1.13 Firmware self-generated frame statistics

```
root@OpenWrt:/# iwpriv ath0 txrx_stats 12
[  929.197240] wlan: [35: F:DP] HTT_TX_SELFGEN_CMN_STATS_TLV:
[  929.197269] wlan: [35: F:DP] mac_id__word = 0
[  929.201619] wlan: [35: F:DP] su_bar = 0
[  929.206115] wlan: [35: F:DP] rts = 0
[  929.209707] wlan: [35: F:DP] cts2self = 0
[  929.213572] wlan: [35: F:DP] qos_null = 0
[  929.217416] wlan: [35: F:DP] delayed_bar_1 = 0
[  929.221424] wlan: [35: F:DP] delayed_bar_2 = 0
[  929.225763] wlan: [35: F:DP] delayed_bar_3 = 0
[  929.230232] wlan: [35: F:DP] delayed_bar_4 = 0
[  929.234617] wlan: [35: F:DP] delayed_bar_5 = 0
[  929.239047] wlan: [35: F:DP] delayed_bar_6 = 0
```

Qualcomm Confidential – May Contain Trade Secrets 2021-05-17 03:57:44 PDT pan.yafeng@zte.com.cn

```
[  929.243470] wlan: [35: F:DP] delayed_bar_7 = 0
[  929.243470]
[  929.247904] wlan: [35: F:DP] HTT_TX_SELFGEN_AC_STATS_TLV:
[  929.253735] wlan: [35: F:DP] ac_su_ndpa = 0
[  929.259270] wlan: [35: F:DP] ac_su_ndp = 0
[  929.263289] wlan: [35: F:DP] ac_mu_mimo_ndpa = 0
[  929.267528] wlan: [35: F:DP] ac_mu_mimo_ndp = 0
[  929.272218] wlan: [35: F:DP] ac_mu_mimo_brpoll_1 = 0
[  929.276481] wlan: [35: F:DP] ac_mu_mimo_brpoll_2 = 0
[  929.281669] wlan: [35: F:DP] ac_mu_mimo_brpoll_3 = 0
[  929.281669]
[  929.286630] wlan: [35: F:DP] HTT_TX_SELFGEN_AX_STATS_TLV:
[  929.292947] wlan: [35: F:DP] ax_su_ndpa = 0
[  929.298333] wlan: [35: F:DP] ax_su_ndp = 0
[  929.302344] wlan: [35: F:DP] ax_mu_mimo_ndpa = 0
[  929.306493] wlan: [35: F:DP] ax_mu_mimo_ndp = 0
[  929.311285] wlan: [35: F:DP] ax_mu_mimo_brpoll_1 = 0
[  929.315519] wlan: [35: F:DP] ax_mu_mimo_brpoll_2 = 0
[  929.320732] wlan: [35: F:DP] ax_mu_mimo_brpoll_3 = 0
[  929.325706] wlan: [35: F:DP] ax_mu_mimo_brpoll_4 = 0
[  929.330627] wlan: [35: F:DP] ax_mu_mimo_brpoll_5 = 0
[  929.335583] wlan: [35: F:DP] ax_mu_mimo_brpoll_6 = 0
[  929.340507] wlan: [35: F:DP] ax_mu_mimo_brpoll_7 = 0
[  929.345478] wlan: [35: F:DP] ax_basic_trigger = 0
[  929.350410] wlan: [35: F:DP] ax_bsr_trigger = 0
[  929.355041] wlan: [35: F:DP] ax_mu_bar_trigger = 0
[  929.359369] wlan: [35: F:DP] ax_mu_rts_trigger = 0
[  929.359369]
[  929.364219] wlan: [35: F:DP] HTT_TX_SELFGEN_AC_ERR_STATS_TLV:
[  929.370427] wlan: [35: F:DP] ac_su_ndp_err = 0
[  929.376329] wlan: [35: F:DP] ac_su_ndpa_err = 0
[  929.380611] wlan: [35: F:DP] ac_mu_mimo_ndpa_err = 0
[  929.385064] wlan: [35: F:DP] ac_mu_mimo_ndp_err = 0
[  929.390247] wlan: [35: F:DP] ac_mu_mimo_brp1_err = 0
[  929.394874] wlan: [35: F:DP] ac_mu_mimo_brp2_err = 0
[  929.400074] wlan: [35: F:DP] ac_mu_mimo_brp3_err = 0
[  929.400074]
[  929.405047] wlan: [35: F:DP] HTT_TX_SELFGEN_AX_ERR_STATS_TLV:
[  929.411358] wlan: [35: F:DP] ax_su_ndp_err = 0
[  929.417093] wlan: [35: F:DP] ax_su_ndpa_err = 0
[  929.421442] wlan: [35: F:DP] ax_mu_mimo_ndpa_err = 0
[  929.425861] wlan: [35: F:DP] ax_mu_mimo_ndp_err = 0
[  929.431045] wlan: [35: F:DP] ax_mu_mimo_brp1_err = 0
[  929.435670] wlan: [35: F:DP] ax_mu_mimo_brp2_err = 0
[  929.440863] wlan: [35: F:DP] ax_mu_mimo_brp3_err = 0
[  929.445835] wlan: [35: F:DP] ax_mu_mimo_brp4_err = 0
[  929.450780] wlan: [35: F:DP] ax_mu_mimo_brp5_err = 0
[  929.455727] wlan: [35: F:DP] ax_mu_mimo_brp6_err = 0
[  929.460662] wlan: [35: F:DP] ax_mu_mimo_brp7_err = 0
[  929.465605] wlan: [35: F:DP] ax_basic_trigger_err = 0
[  929.470589] wlan: [35: F:DP] ax_bsr_trigger_err = 0
[  929.475556] wlan: [35: F:DP] ax_mu_bar_trigger_err = 0
[  929.480179] wlan: [35: F:DP] ax_mu_rts_trigger_err = 0
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
[  929.480179]
```

Display the firmware selfgen stats information using the command for PDEV: wifistats wifi0 12 --vdevid 255

NOTE For stats for a specific VDEV, specify it as follows: wifistats wifi0 12 --vdevid 0

```
root@OpenWrt:/# wifistats wifi0 12 --vdevid 255
root@OpenWrt:~# wifistats wifi0 12

HTT_TX_SELFGEN_CMN_STATS_TLV:
mac_id = 0
su_bar = 0
rts = 0
cts2self = 0
qos_null = 0
delayed_bar_1 = 0
delayed_bar_2 = 0
delayed_bar_3 = 0
delayed_bar_4 = 0
delayed_bar_5 = 0
delayed_bar_6 = 0
delayed_bar_7 = 0

HTT_TX_SELFGEN_AC_STATS_TLV:
ac_su_ndpa_queued = 0
ac_su_ndpa_tried = 0
ac_su_ndp_queued = 0
ac_su_ndp_tried = 0
ac_mu_mimo_ndpa_queued = 0
ac_mu_mimo_ndpa_tried = 0
ac_mu_mimo_ndp_queued = 0
ac_mu_mimo_ndp_tried = 0
ac_mu_mimo_brpoll1_queued = 0
ac_mu_mimo_brpoll1_tried = 0
ac_mu_mimo_brpoll2_queued = 0
ac_mu_mimo_brpoll2_tried = 0
ac_mu_mimo_brpoll3_queued = 0
ac_mu_mimo_brpoll3_tried = 0

HTT_TX_SELFGEN_AX_STATS_TLV:
ax_su_ndpa_queued = 917
ax_su_ndpa_tried = 917
ax_su_ndp_queued = 917
ax_su_ndp_tried = 988
ax_mu_mimo_ndpa_queued = 0
ax_mu_mimo_ndpa_tried = 0
ax_mu_mimo_ndp_queued = 0
ax_mu_mimo_ndp_tried = 0
ax_mu_mimo_brpoll_queued =  1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
ax_mu_mimo_brpoll_tried =  1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
ax_basic_trigger = 0
ax_ulmumimo_trigger = 0
ax_bsr_trigger = 0
ax_mu_bar_trigger = 0
```

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

```
ax_mu_rts_trigger = 0

HTT_TX_SELFGEN_AC_ERR_STATS_TLV:
ac_su_ndp_err = 0
ac_su_ndp_flushed = 0
ac_su_ndpa_err = 0
ac_su_ndpa_flushed = 0
ac_mu_mimo_ndpa_err = 0
ac_mu_mimo_ndpa_flushed = 0
ac_mu_mimo_ndp_err = 0
ac_mu_mimo_ndp_flushed = 0
ac_mu_mimo_brp1_err = 0
ac_mu_mimo_brp2_err = 0
ac_mu_mimo_brp3_err = 0
ac_mu_mimo_brp1_flushed = 0
ac_mu_mimo_brp2_flushed = 0
ac_mu_mimo_brp3_flushed = 0

HTT_TX_SELFGEN_AX_ERR_STATS_TLV:
ax_su_ndp_err = 71
ax_su_ndp_flushed = 0
ax_su_ndpa_err = 0
ax_su_ndpa_flushed = 0
ax_mu_mimo_ndpa_err = 0
ax_mu_mimo_ndpa_flushed = 0
ax_mu_mimo_ndp_err = 0
ax_mu_mimo_ndp_flushed = 0
ax_mu_mimo_brp_err =  1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
ax_mu_mimo_brpoll_flushed =  1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
ax_basic_trigger_err = 0
ax_ulmumimo_trigger_err = 0
ax_bsr_trigger_err = 0
ax_mu_bar_trigger_err = 0
ax_mu_rts_trigger_err = 0

HTT_TX_SELFGEN_AC_SCHED_STATUS_STATS_TLV:
ac_su_ndpa_sch_status =  1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
ac_su_ndp_sch_status =  1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
ac_mu_mimo_ndpa_sch_status =  1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
ac_mu_mimo_ndp_sch_status =  1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
ac_mu_mimo_brp_sch_status =  1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
ac_su_ndp_sch_flag_err =  1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0,
ac_mu_mimo_ndp_sch_flag_err =  1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0,
ac_mu_mimo_brp_sch_flag_err =  1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0,

HTT_TX_SELFGEN_AX_SCHED_STATUS_STATS_TLV:
ax_su_ndpa_sch_status =  1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
ax_su_ndp_sch_status =  1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
ax_mu_mimo_ndpa_sch_status =  1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
ax_mu_mimo_ndp_sch_status =  1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
ax_mu_brp_sch_status =  1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
ax_mu_bar_sch_status =  1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
ax_basic_trig_sch_status =  1:14475, 2:79, 3:7, 4:0, 5:0, 6:0, 7:0,
ax_su_ndp_sch_flag_err =  1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0,
ax_mu_mimo_ndp_sch_flag_err =  1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0,
ax_mu_brp_sch_flag_err =  1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0,
```

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

```
ax_mu_bar_sch_flag_err =  1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0,
ax_basic_trig_sch_flag_err =  1:2, 2:5, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0,
```

Selfgen sounding user stats

PDEV-level stats: wifistats wifi0 22 -vdevid 255

VDEV-level stats: wifistats wifi0 22 -vdevid 0

```
HTT_TX_AC_SOUNDING_STATS_TLV:
ac_cbf_20 = IBF : 0, SU_SIFS : 0, SU_RBO : 0, MU_SIFS : 0, MU_RBO : 0
ac_cbf_40 = IBF : 0, SU_SIFS : 0, SU_RBO : 0, MU_SIFS : 0, MU_RBO : 0
ac_cbf_80 = IBF : 0, SU_SIFS : 0, SU_RBO : 0, MU_SIFS : 0, MU_RBO : 0
ac_cbf_160 = IBF : 0, SU_SIFS : 0, SU_RBO : 0, MU_SIFS : 0, MU_RBO : 0
Sounding User 0 = 20MHz: 0, 40MHz : 0, 80MHz: 0, 160MHz: 0
Sounding User 1 = 20MHz: 0, 40MHz : 0, 80MHz: 0, 160MHz: 0
Sounding User 2 = 20MHz: 0, 40MHz : 0, 80MHz: 0, 160MHz: 0
Sounding User 3 = 20MHz: 0, 40MHz : 0, 80MHz: 0, 160MHz: 0

HTT_TX_AX_SOUNDING_STATS_TLV:
ax_cbf_20 = IBF : 0, SU_SIFS : 0, SU_RBO : 0, MU_SIFS : 0, MU_RBO : 0
ax_cbf_40 = IBF : 0, SU_SIFS : 0, SU_RBO : 0, MU_SIFS : 0, MU_RBO : 0
ax_cbf_80 = IBF : 0, SU_SIFS : 0, SU_RBO : 917, MU_SIFS : 0, MU_RBO : 0
ax_cbf_160 = IBF : 0, SU_SIFS : 0, SU_RBO : 0, MU_SIFS : 0, MU_RBO : 0
Sounding User 0 = 20MHz: 0, 40MHz : 0, 80MHz: 917, 160MHz: 0
Sounding User 1 = 20MHz: 0, 40MHz : 0, 80MHz: 0, 160MHz: 0
Sounding User 2 = 20MHz: 0, 40MHz : 0, 80MHz: 0, 160MHz: 0
Sounding User 3 = 20MHz: 0, 40MHz : 0, 80MHz: 0, 160MHz: 0
Sounding User 4 = 20MHz: 0, 40MHz : 0, 80MHz: 0, 160MHz: 0
Sounding User 5 = 20MHz: 0, 40MHz : 0, 80MHz: 0, 160MHz: 0
Sounding User 6 = 20MHz: 0, 40MHz : 0, 80MHz: 0, 160MHz: 0
Sounding User 7 = 20MHz: 0, 40MHz : 0, 80MHz: 0, 160MHz: 0
```

3.23.1.14 PDEV Tx scheduler statistics

```
root@OpenWrt:/# iwpriv ath0 txrx_stats 4
[1669764.299286] wlan: [6802:F:DP] HTT_STATS_TX_SCHED_CMN_TLV:
```

3.23.1.15 PDEV error statistics

```
root@OpenWrt:/# iwpriv ath0 txrx_stats 5
[1669961.268412] wlan: [6802:F:DP] HTT_HW_STATS_PDEV_ERRS_TLV:
```

3.23.1.16 PDEV TQM statistics

```
root@OpenWrt:/# iwpriv ath0 txrx_stats 6
[1669988.236951] wlan: [6802:F:DP] HTT_TX_TQM_CMN_STATS_TLV:
```

3.23.1.17 Source ring (SRNG) info statistics

```
root@OpenWrt:/# iwpriv ath0 txrx_stats 15
[1742287.192379] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
```

```
[1742287.192429] wlan: [6802:F:DP] data =  0:1598833492, 1:1598378823,
2:1598311747, 3:1196312914, 4:0,
[1742287.192429]
[1742287.196463] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742287.207210] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 16842752
[1742287.212271] wlan: [6802:F:DP] base_addr_lsb = 1353990176
[1742287.218629] wlan: [6802:F:DP] base_addr_msb = 0
[1742287.223970] wlan: [6802:F:DP] ring_size = 256
[1742287.228747] wlan: [6802:F:DP] elem_size = 8
[1742287.233093] wlan: [6802:F:DP] num_avail_words__num_valid_words = 248
[1742287.237590] wlan: [6802:F:DP] head_ptr__tail_ptr = 0
[1742287.244069] wlan: [6802:F:DP] consumer_empty__producer_full = 0
[1742287.249131] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr = 0
[1742287.249131]
[1742287.255334] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742287.263485] wlan: [6802:F:DP] data =  0:1598833492, 1:1598378823,
2:1413567571, 3:1381978965, 4:4673097,
[1742287.263485]
[1742287.268544] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742287.279719] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 65792
[1742287.284828] wlan: [6802:F:DP] base_addr_lsb = 1353991264
[1742287.290728] wlan: [6802:F:DP] base_addr_msb = 0
[1742287.296219] wlan: [6802:F:DP] ring_size = 64
[1742287.300961] wlan: [6802:F:DP] elem_size = 8
[1742287.305295] wlan: [6802:F:DP] num_avail_words__num_valid_words = 56
[1742287.309721] wlan: [6802:F:DP] head_ptr__tail_ptr = 0
[1742287.316170] wlan: [6802:F:DP] consumer_empty__producer_full = 0
[1742287.321230] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr = 0
[1742287.321230]
[1742287.327357] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742287.335532] wlan: [6802:F:DP] data =  0:843858772, 1:1331648326,
2:1431327829, 3:1230135124, 4:18254,
[1742287.335532]
[1742287.340487] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742287.351442] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 66048
[1742287.356519] wlan: [6802:F:DP] base_addr_lsb = 1353991584
[1742287.362592] wlan: [6802:F:DP] base_addr_msb = 0
[1742287.368003] wlan: [6802:F:DP] ring_size = 7040
[1742287.372747] wlan: [6802:F:DP] elem_size = 110
[1742287.377431] wlan: [6802:F:DP] num_avail_words__num_valid_words =
6930
[1742287.381775] wlan: [6802:F:DP] head_ptr__tail_ptr = 0
[1742287.388287] wlan: [6802:F:DP] consumer_empty__producer_full = 0
[1742287.393536] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr = 0
[1742287.393536]
[1742287.399659] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742287.407843] wlan: [6802:F:DP] data =  0:1598902612, 1:1380993352,
2:1296260937, 3:1230135108, 4:18254,
[1742287.407843]
[1742287.412782] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742287.423758] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 16843520
[1742287.428912] wlan: [6802:F:DP] base_addr_lsb = 1354019808
[1742287.435261] wlan: [6802:F:DP] base_addr_msb = 0
[1742287.440717] wlan: [6802:F:DP] ring_size = 4096
[1742287.445434] wlan: [6802:F:DP] elem_size = 2
```

```
[1742287.450076] wlan: [6802:F:DP] num_avail_words__num_valid_words =
4094
[1742287.454364] wlan: [6802:F:DP] head_ptr__tail_ptr = 0
[1742287.460776] wlan: [6802:F:DP] consumer_empty__producer_full = 0
[1742287.465982] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr = 0
[1742287.465982]
[1742287.472184] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742287.480312] wlan: [6802:F:DP] data =  0:1598902612, 1:1347899212,
2:1130318162, 3:1381975117, 4:4673097,
[1742287.480312]
[1742287.485269] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742287.496665] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 16843776
[1742287.501572] wlan: [6802:F:DP] base_addr_lsb = 1354036256
[1742287.507909] wlan: [6802:F:DP] base_addr_msb = 0
[1742287.513291] wlan: [6802:F:DP] ring_size = 4096
[1742287.518173] wlan: [6802:F:DP] elem_size = 2
[1742287.522746] wlan: [6802:F:DP] num_avail_words__num_valid_words =
4094
[1742287.527001] wlan: [6802:F:DP] head_ptr__tail_ptr = 127403928
[1742287.533437] wlan: [6802:F:DP] consumer_empty__producer_full = 443
[1742287.539331] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr =
127401984
[1742287.539331]
[1742287.545537] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742287.554560] wlan: [6802:F:DP] data =  0:1412585286, 1:1163873617,
2:1381979214, 3:4673097,
[1742287.554560]
[1742287.559469] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742287.569422] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 16844032
[1742287.574573] wlan: [6802:F:DP] base_addr_lsb = 1354052704
[1742287.580928] wlan: [6802:F:DP] base_addr_msb = 0
[1742287.586285] wlan: [6802:F:DP] ring_size = 512
[1742287.591066] wlan: [6802:F:DP] elem_size = 8
[1742287.595512] wlan: [6802:F:DP] num_avail_words__num_valid_words = 504
[1742287.599912] wlan: [6802:F:DP] head_ptr__tail_ptr = 524296
[1742287.606357] wlan: [6802:F:DP] consumer_empty__producer_full = 1
[1742287.611896] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr =
524288
[1742287.611896]
[1742287.618099] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742287.626580] wlan: [6802:F:DP] data =  0:1598902612, 1:1380993352,
2:1414750025, 3:1398101057, 4:1313428063, 5:71,
[1742287.626580]
[1742287.631634] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742287.643640] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 67072
[1742287.648693] wlan: [6802:F:DP] base_addr_lsb = 1354153184
[1742287.654703] wlan: [6802:F:DP] base_addr_msb = 0
[1742287.660152] wlan: [6802:F:DP] ring_size = 12288
[1742287.664947] wlan: [6802:F:DP] elem_size = 12
[1742287.669643] wlan: [6802:F:DP] num_avail_words__num_valid_words =
12276
[1742287.674042] wlan: [6802:F:DP] head_ptr__tail_ptr = 0
[1742287.680839] wlan: [6802:F:DP] consumer_empty__producer_full = 0
[1742287.685770] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr = 0
[1742287.685770]
[1742287.691978] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
```

```
[1742287.700102] wlan: [6802:F:DP] data =  0:1598902612, 1:1347899212,
2:1398753618, 3:1431585108, 4:1230135123, 5:18254,
[1742287.700102]
[1742287.705066] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742287.717434] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 67328
[1742287.722404] wlan: [6802:F:DP] base_addr_lsb = 1354054816
[1742287.728476] wlan: [6802:F:DP] base_addr_msb = 0
[1742287.733868] wlan: [6802:F:DP] ring_size = 24576
[1742287.738623] wlan: [6802:F:DP] elem_size = 12
[1742287.743359] wlan: [6802:F:DP] num_avail_words__num_valid_words =
24564
[1742287.747651] wlan: [6802:F:DP] head_ptr__tail_ptr = 382211784
[1742287.754611] wlan: [6802:F:DP] consumer_empty__producer_full = 0
[1742287.760156] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr = 0
[1742287.760156]
[1742287.766349] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742287.774529] wlan: [6802:F:DP] data =  0:843858772, 1:1598902612,
2:1599360581, 3:1196312914, 4:0,
[1742287.774529]
[1742287.779442] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742287.790050] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 133120
[1742287.795180] wlan: [6802:F:DP] base_addr_lsb = 1245184
[1742287.801140] wlan: [6802:F:DP] base_addr_msb = 16
[1742287.806341] wlan: [6802:F:DP] ring_size = 1024
[1742287.811214] wlan: [6802:F:DP] elem_size = 8
[1742287.815967] wlan: [6802:F:DP] num_avail_words__num_valid_words =
1016
[1742287.820259] wlan: [6802:F:DP] head_ptr__tail_ptr = 0
[1742287.826656] wlan: [6802:F:DP] consumer_empty__producer_full = 0
[1742287.831959] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr = 0
[1742287.831959]
[1742287.838030] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742287.846316] wlan: [6802:F:DP] data =  0:843924055, 1:1598902612,
2:1129530692, 3:1313428063, 4:71,
[1742287.846316]
[1742287.851145] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742287.861763] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 133376
[1742287.866964] wlan: [6802:F:DP] base_addr_lsb = 1249312
[1742287.872926] wlan: [6802:F:DP] base_addr_msb = 16
[1742287.878128] wlan: [6802:F:DP] ring_size = 512
[1742287.882994] wlan: [6802:F:DP] elem_size = 2
[1742287.887419] wlan: [6802:F:DP] num_avail_words__num_valid_words =
33423360
[1742287.891979] wlan: [6802:F:DP] head_ptr__tail_ptr = 131072
[1742287.898699] wlan: [6802:F:DP] consumer_empty__producer_full = 65536
[1742287.904271] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr = 0
[1742287.904271]
[1742287.910957] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742287.919050] wlan: [6802:F:DP] data =  0:843927892, 1:1598898775,
2:1598834002, 3:1196312914, 4:0,
[1742287.919050]
[1742287.923976] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742287.934591] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 133632
[1742287.939674] wlan: [6802:F:DP] base_addr_lsb = 1251392
[1742287.945701] wlan: [6802:F:DP] base_addr_msb = 16
[1742287.950874] wlan: [6802:F:DP] ring_size = 2048
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
[1742287.955806] wlan: [6802:F:DP] elem_size = 8
[1742287.960511] wlan: [6802:F:DP] num_avail_words__num_valid_words =
2040
[1742287.964764] wlan: [6802:F:DP] head_ptr__tail_ptr = 0
[1742287.971233] wlan: [6802:F:DP] consumer_empty__producer_full = 0
[1742287.976383] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr = 0
[1742287.976383]
[1742287.982579] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742287.990825] wlan: [6802:F:DP] data =  0:843924055, 1:1599030610,
2:1129530692, 3:1313428063, 4:71,
[1742287.990825]
[1742287.995699] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742288.006284] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 133888
[1742288.011473] wlan: [6802:F:DP] base_addr_lsb = 1259616
[1742288.017484] wlan: [6802:F:DP] base_addr_msb = 16
[1742288.022660] wlan: [6802:F:DP] ring_size = 256
[1742288.027518] wlan: [6802:F:DP] elem_size = 2
[1742288.031961] wlan: [6802:F:DP] num_avail_words__num_valid_words =
15728654
[1742288.036461] wlan: [6802:F:DP] head_ptr__tail_ptr = 240
[1742288.043279] wlan: [6802:F:DP] consumer_empty__producer_full = 0
[1742288.048784] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr = 0
[1742288.048784]
[1742288.054863] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742288.063104] wlan: [6802:F:DP] data =  0:844055890, 1:1598898775,
2:1598834002, 3:1196312914, 4:0,
[1742288.063104]
[1742288.068016] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742288.078535] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 134144
[1742288.083606] wlan: [6802:F:DP] base_addr_lsb = 1260672
[1742288.089705] wlan: [6802:F:DP] base_addr_msb = 16
[1742288.094868] wlan: [6802:F:DP] ring_size = 1024
[1742288.099650] wlan: [6802:F:DP] elem_size = 8
[1742288.104461] wlan: [6802:F:DP] num_avail_words__num_valid_words =
1016
[1742288.108681] wlan: [6802:F:DP] head_ptr__tail_ptr = 0
[1742288.115123] wlan: [6802:F:DP] consumer_empty__producer_full = 0
[1742288.120349] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr = 0
[1742288.120349]
[1742288.126492] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742288.134672] wlan: [6802:F:DP] data =  0:843924055, 1:1381979974,
2:1113541701, 3:1381975637, 4:4673097,
[1742288.134672]
[1742288.139602] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742288.151009] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 68864
[1742288.155822] wlan: [6802:F:DP] base_addr_lsb = 1354202400
[1742288.161821] wlan: [6802:F:DP] base_addr_msb = 0
[1742288.167317] wlan: [6802:F:DP] ring_size = 512
[1742288.172051] wlan: [6802:F:DP] elem_size = 8
[1742288.176381] wlan: [6802:F:DP] num_avail_words__num_valid_words = 504
[1742288.180923] wlan: [6802:F:DP] head_ptr__tail_ptr = 0
[1742288.187418] wlan: [6802:F:DP] consumer_empty__producer_full = 0
[1742288.192494] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr = 0
[1742288.192494]
[1742288.198618] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
```

```
[1742288.206798] wlan: [6802:F:DP] data =  0:843924055, 1:1281316678,
2:1598770761, 3:1196312914, 4:0,
[1742288.206798]
[1742288.211742] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742288.222447] wlan: [6802:F:DP] data =  0:1462916934, 1:1381977410,
2:1381977157, 3:4673097,
[1742288.222447]
[1742288.227442] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742288.237361] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 16846592
[1742288.242583] wlan: [6802:F:DP] base_addr_lsb = 1354204512
[1742288.248866] wlan: [6802:F:DP] base_addr_msb = 0
[1742288.254270] wlan: [6802:F:DP] ring_size = 512
[1742288.259022] wlan: [6802:F:DP] elem_size = 8
[1742288.263377] wlan: [6802:F:DP] num_avail_words__num_valid_words = 504
[1742288.267920] wlan: [6802:F:DP] head_ptr__tail_ptr = 0
[1742288.274314] wlan: [6802:F:DP] consumer_empty__producer_full = 0
[1742288.279431] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr = 0
[1742288.279431]
[1742288.285613] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742288.293887] wlan: [6802:F:DP] data =  0:1598898775, 1:1162626121,
2:1397048415, 3:1230135107, 4:18254,
[1742288.293887]
[1742288.298790] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742288.309671] wlan: [6802:F:DP] data =  0:1412585299, 1:1597066307,
2:1431326281, 3:1230135124, 4:18254,
[1742288.309671]
[1742288.314878] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742288.325838] wlan: [6802:F:DP] data =  0:1379030854, 1:1163874117,
2:1381979214, 3:4673097,
[1742288.325838]
[1742288.331021] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742288.340915] wlan: [6802:F:DP] data =  0:844055890, 1:1147098950,
2:1599361861, 3:1196312914, 4:0,
[1742288.340915]
[1742288.346134] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742288.356712] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 4864
[1742288.361821] wlan: [6802:F:DP] base_addr_lsb = 1347406432
[1742288.367834] wlan: [6802:F:DP] base_addr_msb = 0
[1742288.373262] wlan: [6802:F:DP] ring_size = 1024
[1742288.377950] wlan: [6802:F:DP] elem_size = 16
[1742288.382643] wlan: [6802:F:DP] num_avail_words__num_valid_words =
1008
[1742288.386893] wlan: [6802:F:DP] head_ptr__tail_ptr = 0
[1742288.393451] wlan: [6802:F:DP] consumer_empty__producer_full = 0
[1742288.398611] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr = 0
[1742288.398611]
[1742288.404799] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742288.413039] wlan: [6802:F:DP] data =  0:843924055, 1:1597003603,
2:1598834002, 3:1196312914, 4:0,
[1742288.413039]
[1742288.417930] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742288.428502] wlan: [6802:F:DP] data =  0:843924055, 1:1597069139,
2:1598834002, 3:1196312914, 4:0,
[1742288.428502]
[1742288.433620] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
```

```
[1742288.444251] wlan: [6802:F:DP] data =  0:843924055, 1:1597134675,
2:1598834002, 3:1196312914, 4:0,
[1742288.444251]
[1742288.449312] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742288.460011] wlan: [6802:F:DP] data =  0:843924055, 1:1597200211,
2:1598834002, 3:1196312914, 4:0,
[1742288.460011]
[1742288.465048] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742288.475660] wlan: [6802:F:DP] data =  0:1599030610, 1:1598311747,
2:1196312914, 3:0,
[1742288.475660]
[1742288.480757] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742288.490295] wlan: [6802:F:DP] data =  0:1599030610, 1:1413567571,
2:1381978965, 3:4673097,
[1742288.490295]
[1742288.495369] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742288.505249] wlan: [6802:F:DP] data =  0:1598898775, 1:1162626121,
2:1179992671, 3:1599227206, 4:1196312914, 5:0,
[1742288.505249]
[1742288.510441] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742288.522461] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 6656
[1742288.527512] wlan: [6802:F:DP] base_addr_lsb = 1347410560
[1742288.533519] wlan: [6802:F:DP] base_addr_msb = 0
[1742288.538804] wlan: [6802:F:DP] ring_size = 2048
[1742288.543616] wlan: [6802:F:DP] elem_size = 2
[1742288.548255] wlan: [6802:F:DP] num_avail_words__num_valid_words =
2046
[1742288.552546] wlan: [6802:F:DP] head_ptr__tail_ptr = 4294967295
[1742288.558942] wlan: [6802:F:DP] consumer_empty__producer_full = 0
[1742288.564867] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr = 0
[1742288.564867]
[1742288.571144] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742288.579267] wlan: [6802:F:DP] data =  0:1279346775, 1:1162891359,
2:1279611487, 3:1163084101, 4:1313428063, 5:71,
[1742288.579267]
[1742288.584206] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742288.596205] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 16784128
[1742288.601294] wlan: [6802:F:DP] base_addr_lsb = 1347398208
[1742288.607629] wlan: [6802:F:DP] base_addr_msb = 0
[1742288.613057] wlan: [6802:F:DP] ring_size = 2048
[1742288.617797] wlan: [6802:F:DP] elem_size = 8
[1742288.622468] wlan: [6802:F:DP] num_avail_words__num_valid_words =
2040
[1742288.626718] wlan: [6802:F:DP] head_ptr__tail_ptr = 0
[1742288.633163] wlan: [6802:F:DP] consumer_empty__producer_full = 0
[1742288.638355] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr = 0
[1742288.638355]
[1742288.644595] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742288.652690] wlan: [6802:F:DP] data =  0:1296324690, 1:1113010753,
2:1163026253, 3:1230135116, 4:18254,
[1742288.652690]
[1742288.657631] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742288.668601] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 0
[1742288.673773] wlan: [6802:F:DP] base_addr_lsb = 1347389952
[1742288.679420] wlan: [6802:F:DP] base_addr_msb = 0
[1742288.684883] wlan: [6802:F:DP] ring_size = 1024
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
[1742288.689702] wlan: [6802:F:DP] elem_size = 8
[1742288.694362] wlan: [6802:F:DP] num_avail_words__num_valid_words =
1016
[1742288.698596] wlan: [6802:F:DP] head_ptr__tail_ptr = 0
[1742288.705029] wlan: [6802:F:DP] consumer_empty__producer_full = 0
[1742288.710229] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr = 0
[1742288.710229]
[1742288.716407] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742288.724580] wlan: [6802:F:DP] data =  0:1129535304, 1:1364472392,
2:1296260941, 3:1230135108, 4:18254,
[1742288.724580]
[1742288.729513] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742288.740462] wlan: [6802:F:DP] data =  0:1296324690, 1:1163014721,
2:1296260943, 3:1230135108, 4:18254,
[1742288.740462]
[1742288.745673] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742288.756646] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 131584
[1742288.761800] wlan: [6802:F:DP] base_addr_lsb = 1264800
[1742288.767855] wlan: [6802:F:DP] base_addr_msb = 16
[1742288.772989] wlan: [6802:F:DP] ring_size = 1024
[1742288.777852] wlan: [6802:F:DP] elem_size = 8
[1742288.782624] wlan: [6802:F:DP] num_avail_words__num_valid_words =
1016
[1742288.786879] wlan: [6802:F:DP] head_ptr__tail_ptr = 0
[1742288.793330] wlan: [6802:F:DP] consumer_empty__producer_full = 0
[1742288.798512] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr = 0
[1742288.798512]
[1742288.804688] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742288.812844] wlan: [6802:F:DP] data =  0:843924055, 1:1296324690,
2:1162108737, 3:1381974867, 4:4673097,
[1742288.812844]
[1742288.817812] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742288.829079] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 768
[1742288.834022] wlan: [6802:F:DP] base_addr_lsb = 1347387840
[1742288.840005] wlan: [6802:F:DP] base_addr_msb = 0
[1742288.845362] wlan: [6802:F:DP] ring_size = 256
[1742288.850064] wlan: [6802:F:DP] elem_size = 2
[1742288.854418] wlan: [6802:F:DP] num_avail_words__num_valid_words =
15728654
[1742288.858927] wlan: [6802:F:DP] head_ptr__tail_ptr = 240
[1742288.865714] wlan: [6802:F:DP] consumer_empty__producer_full = 0
[1742288.871320] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr = 0
[1742288.871320]
[1742288.877252] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742288.885413] wlan: [6802:F:DP] data =  0:1398757458, 1:1381974866,
2:1598508617, 3:1094930252, 4:1414750028, 5:1398101057, 6:0,
[1742288.885413]
[1742288.890366] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742288.903413] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 16777216
[1742288.908579] wlan: [6802:F:DP] base_addr_lsb = 1348474336
[1742288.914927] wlan: [6802:F:DP] base_addr_msb = 0
[1742288.920355] wlan: [6802:F:DP] ring_size = 64
[1742288.925076] wlan: [6802:F:DP] elem_size = 2
[1742288.929411] wlan: [6802:F:DP] num_avail_words__num_valid_words =
1966112
[1742288.933850] wlan: [6802:F:DP] head_ptr__tail_ptr = 3538994
```

```
[1742288.940621] wlan: [6802:F:DP] consumer_empty__producer_full = 0
[1742288.946446] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr =
1310750
[1742288.946446]
[1742288.952432] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742288.960937] wlan: [6802:F:DP] data =  0:1398757458, 1:1381974866,
2:1598508617, 3:1094930252, 4:1263558476, 5:84,
[1742288.960937]
[1742288.966060] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742288.978061] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 16777472
[1742288.983152] wlan: [6802:F:DP] base_addr_lsb = 1352250816
[1742288.989527] wlan: [6802:F:DP] base_addr_msb = 0
[1742288.994936] wlan: [6802:F:DP] ring_size = 128
[1742288.999633] wlan: [6802:F:DP] elem_size = 2
[1742289.003984] wlan: [6802:F:DP] num_avail_words__num_valid_words =
4718646
[1742289.008492] wlan: [6802:F:DP] head_ptr__tail_ptr = 3407886
[1742289.015277] wlan: [6802:F:DP] consumer_empty__producer_full = 0
[1742289.021104] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr =
4587538
[1742289.021104]
[1742289.027080] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742289.035596] wlan: [6802:F:DP] data =  0:1398757458, 1:1381974866,
2:1598508617, 3:1330464082, 4:1348420948, 5:21579,
[1742289.035596]
[1742289.040719] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742289.053115] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 16777728
[1742289.058112] wlan: [6802:F:DP] base_addr_lsb = 1946910720
[1742289.064413] wlan: [6802:F:DP] base_addr_msb = 0
[1742289.069793] wlan: [6802:F:DP] ring_size = 8192
[1742289.074597] wlan: [6802:F:DP] elem_size = 2
[1742289.079238] wlan: [6802:F:DP] num_avail_words__num_valid_words =
534642720
[1742289.083499] wlan: [6802:F:DP] head_ptr__tail_ptr = 8190
[1742289.090696] wlan: [6802:F:DP] consumer_empty__producer_full = 0
[1742289.095929] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr = 0
[1742289.095929]
[1742289.102059] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742289.110146] wlan: [6802:F:DP] data =  0:1398757458, 1:1381974866,
2:1598508617, 3:1094930252, 4:1162108748, 5:17235,
[1742289.110146]
[1742289.115113] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742289.127453] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 16777984
[1742289.132459] wlan: [6802:F:DP] base_addr_lsb = 1352251360
[1742289.138796] wlan: [6802:F:DP] base_addr_msb = 0
[1742289.144192] wlan: [6802:F:DP] ring_size = 128
[1742289.148937] wlan: [6802:F:DP] elem_size = 2
[1742289.153333] wlan: [6802:F:DP] num_avail_words__num_valid_words =
4325436
[1742289.157796] wlan: [6802:F:DP] head_ptr__tail_ptr = 1704052
[1742289.164583] wlan: [6802:F:DP] consumer_empty__producer_full = 0
[1742289.170427] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr =
3276824
[1742289.170427]
[1742289.176379] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
```

```
[1742289.184899] wlan: [6802:F:DP] data =  0:1398757458, 1:1381974866,
2:1598508617, 3:1229868877, 4:1599229780, 5:1413567571, 6:21333,
[1742289.184899]
[1742289.190020] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742289.203415] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 16778240
[1742289.208580] wlan: [6802:F:DP] base_addr_lsb = 1994342400
[1742289.214968] wlan: [6802:F:DP] base_addr_msb = 0
[1742289.220313] wlan: [6802:F:DP] ring_size = 4096
[1742289.225072] wlan: [6802:F:DP] elem_size = 2
[1742289.229815] wlan: [6802:F:DP] num_avail_words__num_valid_words =
266076194
[1742289.234020] wlan: [6802:F:DP] head_ptr__tail_ptr = 229510562
[1742289.241139] wlan: [6802:F:DP] consumer_empty__producer_full = 0
[1742289.246796] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr =
231079960
[1742289.246796]
[1742289.252953] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742289.261803] wlan: [6802:F:DP] data =  0:1398757458, 1:1381974866,
2:1598508617, 3:1229868877, 4:1599229780, 5:5524304,
[1742289.261803]
[1742289.266761] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742289.279104] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 16778496
[1742289.284280] wlan: [6802:F:DP] base_addr_lsb = 1947009024
[1742289.290619] wlan: [6802:F:DP] base_addr_msb = 0
[1742289.296010] wlan: [6802:F:DP] ring_size = 16384
[1742289.300769] wlan: [6802:F:DP] elem_size = 2
[1742289.305454] wlan: [6802:F:DP] num_avail_words__num_valid_words =
1071120422
[1742289.309842] wlan: [6802:F:DP] head_ptr__tail_ptr = 916076172
[1742289.316940] wlan: [6802:F:DP] consumer_empty__producer_full = 0
[1742289.322645] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr =
917766170
[1742289.322645]
[1742289.328859] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742289.337675] wlan: [6802:F:DP] data =  0:1398757458, 1:1381974866,
2:1598508617, 3:1229868877, 4:1599229780, 5:1129530692, 6:0,
[1742289.337675]
[1742289.342647] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742289.355647] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 16778752
[1742289.360846] wlan: [6802:F:DP] base_addr_lsb = 1938948096
[1742289.367196] wlan: [6802:F:DP] base_addr_msb = 0
[1742289.372556] wlan: [6802:F:DP] ring_size = 16384
[1742289.377329] wlan: [6802:F:DP] elem_size = 2
[1742289.382017] wlan: [6802:F:DP] num_avail_words__num_valid_words =
1070858282
[1742289.386412] wlan: [6802:F:DP] head_ptr__tail_ptr = 781594248
[1742289.393501] wlan: [6802:F:DP] consumer_empty__producer_full = 0
[1742289.399205] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr =
783548446
[1742289.399205]
[1742289.405385] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742289.414241] wlan: [6802:F:DP] data =  0:1147099218, 1:1599361861,
2:1196312914, 3:1599555167, 4:1414743364, 5:0,
[1742289.414241]
[1742289.419199] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742289.431170] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 0
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
[1742289.436184] wlan: [6802:F:DP] base_addr_lsb = 1357280352
[1742289.441898] wlan: [6802:F:DP] base_addr_msb = 0
[1742289.447289] wlan: [6802:F:DP] ring_size = 1024
[1742289.452074] wlan: [6802:F:DP] elem_size = 8
[1742289.456750] wlan: [6802:F:DP] num_avail_words__num_valid_words =
1016
[1742289.461018] wlan: [6802:F:DP] head_ptr__tail_ptr = 6815848
[1742289.467506] wlan: [6802:F:DP] consumer_empty__producer_full = 0
[1742289.473341] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr = 0
[1742289.473341]
[1742289.479334] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742289.487498] wlan: [6802:F:DP] data =  0:1147099218, 1:1599361861,
2:1196312914, 3:1313819999, 4:1380930633, 5:1397048415, 6:84,
[1742289.487498]
[1742289.492475] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742289.505821] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 256
[1742289.510759] wlan: [6802:F:DP] base_addr_lsb = 1939079168
[1742289.516739] wlan: [6802:F:DP] base_addr_msb = 0
[1742289.522055] wlan: [6802:F:DP] ring_size = 65528
[1742289.526802] wlan: [6802:F:DP] elem_size = 8
[1742289.531509] wlan: [6802:F:DP] num_avail_words__num_valid_words =
3211200
[1742289.535840] wlan: [6802:F:DP] head_ptr__tail_ptr = 2504037744
[1742289.542680] wlan: [6802:F:DP] consumer_empty__producer_full = 0
[1742289.548421] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr = 0
[1742289.548421]
[1742289.554694] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742289.562891] wlan: [6802:F:DP] data =  0:1147099218, 1:1599361861,
2:1196312914, 3:1397704799, 4:1162108756, 5:21587,
[1742289.562891]
[1742289.567821] wlan: [6802:F:DP] HTT_SRING_STATS_TLV:
[1742289.580121] wlan: [6802:F:DP] mac_id__ring_id__arena__ep = 512
[1742289.585147] wlan: [6802:F:DP] base_addr_lsb = 1939341312
[1742289.591140] wlan: [6802:F:DP] base_addr_msb = 0
[1742289.596444] wlan: [6802:F:DP] ring_size = 32768
[1742289.601202] wlan: [6802:F:DP] elem_size = 8
[1742289.605889] wlan: [6802:F:DP] num_avail_words__num_valid_words =
32760
[1742289.610229] wlan: [6802:F:DP] head_ptr__tail_ptr = 0
[1742289.617072] wlan: [6802:F:DP] consumer_empty__producer_full = 0
[1742289.621952] wlan: [6802:F:DP] prefetch_count__internal_tail_ptr = 0
[1742289.621952]
[1742289.628119] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742289.636288] wlan: [6802:F:DP] data =  0:1147099218, 1:1599361861,
2:1196312914, 3:1329943135, 4:1397048415, 5:84,
[1742289.636288]
[1742289.641235] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742289.653236] wlan: [6802:F:DP] data =  0:1147099218, 1:1599361861,
2:1196312914, 3:1296193375, 4:1397048415, 5:84,
[1742289.653236]
root@OpenWrt:/#
```

3.23.1.18 SFM info statistics

```
root@OpenWrt:/# iwpriv ath0 txrx_stats 16
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
[1742315.937436] wlan: [6802:F:DP] HTT_SFM_CMN_TLV:
[1742315.937466] wlan: [6802:F:DP] mac_id__word = 0
[1742315.941223] wlan: [6802:F:DP] buf_total = 1021
[1742315.945906] wlan: [6802:F:DP] mem_empty = 2047
[1742315.950346] wlan: [6802:F:DP] deallocate_bufs = 0
[1742315.955007] wlan: [6802:F:DP] num_records = 10
[1742315.955007]
[1742315.959573] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742315.965855] wlan: [6802:F:DP] data =  0:1598899795, 1:1162431555,
2:1597002830, 3:1129338962, 4:12373,
[1742315.965855]
[1742315.971075] wlan: [6802:F:DP] HTT_SFM_CLIENT_TLV:
[1742315.981986] wlan: [6802:F:DP] client_id = 0
[1742315.986904] wlan: [6802:F:DP] buf_min = 74
[1742315.991342] wlan: [6802:F:DP] buf_max = 112
[1742315.995692] wlan: [6802:F:DP] buf_busy = 0
[1742315.999915] wlan: [6802:F:DP] buf_alloc = 74
[1742316.004280] wlan: [6802:F:DP] buf_avail = 74
[1742316.008506] wlan: [6802:F:DP] num_users = 39
[1742316.008506]
[1742316.012958] wlan: [6802:F:DP] HTT_SFM_CLIENT_USER_TLV_V:
[1742316.019263] wlan: [6802:F:DP] dwords_used_by_user_n =  0:0, 1:0,
2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0, 12:0, 13:0, 14:0,
15:0, 16:0, 17:0, 18:0, 19:0, 20:0, 21:0, 22:0, 23:0, 24:0, 25:0, 26:0,
27:0, 28:0, 29:0, 30:0, 31:0, 32:0, 33:0, 34:0, 35:0, 36:0, 37:0, 38:0,
[1742316.019263]
[1742316.028577] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742316.051023] wlan: [6802:F:DP] data =  0:1598899795, 1:1162431555,
2:1597068366, 3:1129338962, 4:12629,
[1742316.051023]
[1742316.056016] wlan: [6802:F:DP] HTT_SFM_CLIENT_TLV:
[1742316.067001] wlan: [6802:F:DP] client_id = 1
[1742316.071829] wlan: [6802:F:DP] buf_min = 16
[1742316.076299] wlan: [6802:F:DP] buf_max = 16
[1742316.080665] wlan: [6802:F:DP] buf_busy = 0
[1742316.084894] wlan: [6802:F:DP] buf_alloc = 16
[1742316.089271] wlan: [6802:F:DP] buf_avail = 16
[1742316.093475] wlan: [6802:F:DP] num_users = 1
[1742316.093475]
[1742316.097926] wlan: [6802:F:DP] HTT_SFM_CLIENT_USER_TLV_V:
[1742316.104122] wlan: [6802:F:DP] dwords_used_by_user_n =  0:0,
[1742316.104122]
[1742316.109304] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742316.116904] wlan: [6802:F:DP] data =  0:1598899795, 1:1162431555,
2:1597133902, 3:1129338964, 4:12373,
[1742316.116904]
[1742316.121818] wlan: [6802:F:DP] HTT_SFM_CLIENT_TLV:
[1742316.132763] wlan: [6802:F:DP] client_id = 2
[1742316.137591] wlan: [6802:F:DP] buf_min = 64
[1742316.142150] wlan: [6802:F:DP] buf_max = 647
[1742316.146439] wlan: [6802:F:DP] buf_busy = 0
[1742316.150701] wlan: [6802:F:DP] buf_alloc = 64
[1742316.155040] wlan: [6802:F:DP] buf_avail = 64
[1742316.159288] wlan: [6802:F:DP] num_users = 37
[1742316.159288]
[1742316.163733] wlan: [6802:F:DP] HTT_SFM_CLIENT_USER_TLV_V:
```

```
[1742316.169957] wlan: [6802:F:DP] dwords_used_by_user_n =  0:0, 1:0,
2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0, 12:0, 13:0, 14:0,
15:0, 16:0, 17:0, 18:0, 19:0, 20:0, 21:0, 22:0, 23:0, 24:0, 25:0, 26:0,
27:0, 28:0, 29:0, 30:0, 31:0, 32:0, 33:0, 34:0, 35:0, 36:0,
[1742316.169957]
[1742316.178403] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742316.200630] wlan: [6802:F:DP] data =  0:1598899795, 1:1162431555,
2:1597199438, 3:1129338964, 4:12629,
[1742316.200630]
[1742316.205754] wlan: [6802:F:DP] HTT_SFM_CLIENT_TLV:
[1742316.216780] wlan: [6802:F:DP] client_id = 3
[1742316.221572] wlan: [6802:F:DP] buf_min = 8
[1742316.226036] wlan: [6802:F:DP] buf_max = 30
[1742316.230047] wlan: [6802:F:DP] buf_busy = 0
[1742316.234563] wlan: [6802:F:DP] buf_alloc = 8
[1742316.238806] wlan: [6802:F:DP] buf_avail = 8
[1742316.243096] wlan: [6802:F:DP] num_users = 3
[1742316.243096]
[1742316.247406] wlan: [6802:F:DP] HTT_SFM_CLIENT_USER_TLV_V:
[1742316.253497] wlan: [6802:F:DP] dwords_used_by_user_n =  0:0, 1:0,
2:0,
[1742316.253497]
[1742316.258865] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742316.267081] wlan: [6802:F:DP] data =  0:1598899795, 1:1162431555,
2:1597264974, 3:1129535304, 4:72,
[1742316.267081]
[1742316.272161] wlan: [6802:F:DP] HTT_SFM_CLIENT_TLV:
[1742316.283107] wlan: [6802:F:DP] client_id = 4
[1742316.287673] wlan: [6802:F:DP] buf_min = 392
[1742316.292240] wlan: [6802:F:DP] buf_max = 402
[1742316.296524] wlan: [6802:F:DP] buf_busy = 0
[1742316.300883] wlan: [6802:F:DP] buf_alloc = 392
[1742316.305217] wlan: [6802:F:DP] buf_avail = 392
[1742316.309457] wlan: [6802:F:DP] num_users = 22
[1742316.309457]
[1742316.313998] wlan: [6802:F:DP] HTT_SFM_CLIENT_USER_TLV_V:
[1742316.320284] wlan: [6802:F:DP] dwords_used_by_user_n =  0:0, 1:0,
2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0, 12:0, 13:0, 14:0,
15:0, 16:0, 17:0, 18:0, 19:0, 20:0, 21:0,
[1742316.320284]
[1742316.325646] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742316.343503] wlan: [6802:F:DP] data =  0:1598899795, 1:1162431555,
2:1597330510, 3:4670544,
[1742316.343503]
[1742316.348272] wlan: [6802:F:DP] HTT_SFM_CLIENT_TLV:
[1742316.358191] wlan: [6802:F:DP] client_id = 5
[1742316.363035] wlan: [6802:F:DP] buf_min = 10
[1742316.367584] wlan: [6802:F:DP] buf_max = 37
[1742316.371882] wlan: [6802:F:DP] buf_busy = 0
[1742316.376121] wlan: [6802:F:DP] buf_alloc = 10
[1742316.380425] wlan: [6802:F:DP] buf_avail = 10
[1742316.384634] wlan: [6802:F:DP] num_users = 37
[1742316.384634]
[1742316.389073] wlan: [6802:F:DP] HTT_SFM_CLIENT_USER_TLV_V:
[1742316.395305] wlan: [6802:F:DP] dwords_used_by_user_n =  0:0, 1:0,
2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0, 12:0, 13:0, 14:0,
```

```
15:0, 16:0, 17:0, 18:0, 19:0, 20:0, 21:0, 22:0, 23:0, 24:0, 25:0, 26:0,
27:0, 28:0, 29:0, 30:0, 31:0, 32:0, 33:0, 34:0, 35:0, 36:0,
[1742316.395305]
[1742316.403664] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742316.426042] wlan: [6802:F:DP] data =  0:1598899795, 1:1162431555,
2:1597396046, 3:1296324692, 4:65,
[1742316.426042]
[1742316.431117] wlan: [6802:F:DP] HTT_SFM_CLIENT_TLV:
[1742316.442077] wlan: [6802:F:DP] client_id = 6
[1742316.446685] wlan: [6802:F:DP] buf_min = 50
[1742316.451136] wlan: [6802:F:DP] buf_max = 111
[1742316.455474] wlan: [6802:F:DP] buf_busy = 0
[1742316.459718] wlan: [6802:F:DP] buf_alloc = 50
[1742316.464084] wlan: [6802:F:DP] buf_avail = 50
[1742316.468356] wlan: [6802:F:DP] num_users = 37
[1742316.468356]
[1742316.472756] wlan: [6802:F:DP] HTT_SFM_CLIENT_USER_TLV_V:
[1742316.478951] wlan: [6802:F:DP] dwords_used_by_user_n =  0:0, 1:0,
2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0, 12:0, 13:0, 14:0,
15:0, 16:0, 17:0, 18:0, 19:0, 20:0, 21:0, 22:0, 23:0, 24:0, 25:0, 26:0,
27:0, 28:0, 29:0, 30:0, 31:0, 32:0, 33:0, 34:0, 35:0, 36:0,
[1742316.478951]
[1742316.487341] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742316.509737] wlan: [6802:F:DP] data =  0:1598899795, 1:1162431555,
2:1597461582, 3:1296324690, 4:12353,
[1742316.509737]
[1742316.514796] wlan: [6802:F:DP] HTT_SFM_CLIENT_TLV:
[1742316.525755] wlan: [6802:F:DP] client_id = 7
[1742316.530584] wlan: [6802:F:DP] buf_min = 8
[1742316.535067] wlan: [6802:F:DP] buf_max = 50
[1742316.539068] wlan: [6802:F:DP] buf_busy = 0
[1742316.543619] wlan: [6802:F:DP] buf_alloc = 8
[1742316.547824] wlan: [6802:F:DP] buf_avail = 8
[1742316.552108] wlan: [6802:F:DP] num_users = 1
[1742316.552108]
[1742316.556426] wlan: [6802:F:DP] HTT_SFM_CLIENT_USER_TLV_V:
[1742316.562596] wlan: [6802:F:DP] dwords_used_by_user_n =  0:0,
[1742316.562596]
[1742316.567931] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742316.575366] wlan: [6802:F:DP] data =  0:1598899795, 1:1162431555,
2:1597527118, 3:1280268370, 4:69,
[1742316.575366]
[1742316.580318] wlan: [6802:F:DP] HTT_SFM_CLIENT_TLV:
[1742316.591250] wlan: [6802:F:DP] client_id = 8
[1742316.595854] wlan: [6802:F:DP] buf_min = 294
[1742316.600336] wlan: [6802:F:DP] buf_max = 304
[1742316.604689] wlan: [6802:F:DP] buf_busy = 0
[1742316.609018] wlan: [6802:F:DP] buf_alloc = 294
[1742316.613367] wlan: [6802:F:DP] buf_avail = 294
[1742316.617638] wlan: [6802:F:DP] num_users = 37
[1742316.617638]
[1742316.622135] wlan: [6802:F:DP] HTT_SFM_CLIENT_USER_TLV_V:
[1742316.628413] wlan: [6802:F:DP] dwords_used_by_user_n =  0:0, 1:0,
2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0, 12:0, 13:0, 14:0,
15:0, 16:0, 17:0, 18:0, 19:0, 20:0, 21:0, 22:0, 23:0, 24:0, 25:0, 26:0,
27:0, 28:0, 29:0, 30:0, 31:0, 32:0, 33:0, 34:0, 35:0, 36:0,
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
[1742316.628413]
[1742316.636810] wlan: [6802:F:DP] HTT_STATS_STRING_TLV:
[1742316.659157] wlan: [6802:F:DP] data =  0:1598899795, 1:1162431555,
2:1597592654, 3:1296324690, 4:12609,
[1742316.659157]
[1742316.664258] wlan: [6802:F:DP] HTT_SFM_CLIENT_TLV:
[1742316.675260] wlan: [6802:F:DP] client_id = 9
[1742316.680028] wlan: [6802:F:DP] buf_min = 8
[1742316.684554] wlan: [6802:F:DP] buf_max = 50
[1742316.688531] wlan: [6802:F:DP] buf_busy = 0
[1742316.693080] wlan: [6802:F:DP] buf_alloc = 8
[1742316.697299] wlan: [6802:F:DP] buf_avail = 8
[1742316.701557] wlan: [6802:F:DP] num_users = 1
[1742316.701557]
[1742316.705902] wlan: [6802:F:DP] HTT_SFM_CLIENT_USER_TLV_V:
[1742316.711987] wlan: [6802:F:DP] dwords_used_by_user_n =  0:0,
[1742316.711987]
root@OpenWrt:/#
root@OpenWrt:/#
```

3.23.1.19 Transmit MU statistics

```
root@OpenWrt:/# iwpriv ath0 txrx_stats 17
[1742330.808909] wlan: [6802:F:DP] HTT_TX_PDEV_MU_MIMO_SCH_STATS_TLV:
[1742330.808940] wlan: [6802:F:DP] mu_mimo_sch_posted = 0
[1742330.814180] wlan: [6802:F:DP] mu_mimo_sch_failed = 0
[1742330.819176] wlan: [6802:F:DP] mu_mimo_ppdu_posted = 0
[1742330.819176]
[1742330.824285] wlan: [6802:F:DP] 11ac MU_MIMO SCH STATS:
[1742330.831165] wlan: [6802:F:DP] ac_mu_mimo_sch_nusers_0 = 0
[1742330.836241] wlan: [6802:F:DP] ac_mu_mimo_sch_nusers_1 = 0
[1742330.841812] wlan: [6802:F:DP] ac_mu_mimo_sch_nusers_2 = 0
[1742330.847385] wlan: [6802:F:DP] ac_mu_mimo_sch_nusers_3 = 0
[1742330.852924] wlan: [6802:F:DP]
[1742330.852924] 11ax MU_MIMO SCH STATS:
[1742330.858462] wlan: [6802:F:DP] ax_mu_mimo_sch_nusers_0 = 0
[1742330.865526] wlan: [6802:F:DP] ax_mu_mimo_sch_nusers_1 = 0
[1742330.870917] wlan: [6802:F:DP] ax_mu_mimo_sch_nusers_2 = 0
[1742330.876425] wlan: [6802:F:DP] ax_mu_mimo_sch_nusers_3 = 0
[1742330.882008] wlan: [6802:F:DP] ax_mu_mimo_sch_nusers_4 = 0
[1742330.887539] wlan: [6802:F:DP] ax_mu_mimo_sch_nusers_5 = 0
[1742330.893240] wlan: [6802:F:DP] ax_mu_mimo_sch_nusers_6 = 0
[1742330.898645] wlan: [6802:F:DP] ax_mu_mimo_sch_nusers_7 = 0
[1742330.904223] wlan: [6802:F:DP]
[1742330.904223] 11ax OFDMA SCH STATS:
[1742330.909768] wlan: [6802:F:DP] ax_ofdma_sch_nusers_0 = 0
[1742330.916722] wlan: [6802:F:DP] ax_ofdma_sch_nusers_1 = 0
[1742330.922042] wlan: [6802:F:DP] ax_ofdma_sch_nusers_2 = 0
[1742330.927385] wlan: [6802:F:DP] ax_ofdma_sch_nusers_3 = 0
[1742330.932776] wlan: [6802:F:DP] ax_ofdma_sch_nusers_4 = 0
[1742330.938145] wlan: [6802:F:DP] ax_ofdma_sch_nusers_5 = 0
[1742330.943574] wlan: [6802:F:DP] ax_ofdma_sch_nusers_6 = 0
[1742330.948902] wlan: [6802:F:DP] ax_ofdma_sch_nusers_7 = 0
[1742330.954307] wlan: [6802:F:DP] HTT_TX_PDEV_MU_MIMO_AC_MPDU_STATS:
[1742330.954307]
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
[1742330.959676] wlan: [6802:F:DP] ac_mu_mimo_mpdus_queued_usr_0 = 0
[1742330.967623] wlan: [6802:F:DP] ac_mu_mimo_mpdus_tried_usr_0 = 0
[1742330.973649] wlan: [6802:F:DP] ac_mu_mimo_mpdus_failed_usr_0 = 0
[1742330.979648] wlan: [6802:F:DP] ac_mu_mimo_mpdus_requeued_usr_0 = 0
[1742330.985640] wlan: [6802:F:DP] ac_mu_mimo_err_no_ba_usr_0 = 0
[1742330.991721] wlan: [6802:F:DP] ac_mu_mimo_mpdu_underrun_usr_0 = 0
[1742330.997640] wlan: [6802:F:DP] ac_mu_mimo_ampdu_underrun_usr_0 = 0
[1742330.997640]
[1742331.003799] wlan: [6802:F:DP] ac_mu_mimo_mpdus_queued_usr_1 = 0
[1742331.011689] wlan: [6802:F:DP] ac_mu_mimo_mpdus_tried_usr_1 = 0
[1742331.017868] wlan: [6802:F:DP] ac_mu_mimo_mpdus_failed_usr_1 = 0
[1742331.023917] wlan: [6802:F:DP] ac_mu_mimo_mpdus_requeued_usr_1 = 0
[1742331.029903] wlan: [6802:F:DP] ac_mu_mimo_err_no_ba_usr_1 = 0
[1742331.036002] wlan: [6802:F:DP] ac_mu_mimo_mpdu_underrun_usr_1 = 0
[1742331.041929] wlan: [6802:F:DP] ac_mu_mimo_ampdu_underrun_usr_1 = 0
[1742331.041929]
[1742331.048121] wlan: [6802:F:DP] ac_mu_mimo_mpdus_queued_usr_2 = 0
[1742331.056029] wlan: [6802:F:DP] ac_mu_mimo_mpdus_tried_usr_2 = 0
[1742331.062185] wlan: [6802:F:DP] ac_mu_mimo_mpdus_failed_usr_2 = 0
[1742331.068225] wlan: [6802:F:DP] ac_mu_mimo_mpdus_requeued_usr_2 = 0
[1742331.074179] wlan: [6802:F:DP] ac_mu_mimo_err_no_ba_usr_2 = 0
[1742331.080255] wlan: [6802:F:DP] ac_mu_mimo_mpdu_underrun_usr_2 = 0
[1742331.086164] wlan: [6802:F:DP] ac_mu_mimo_ampdu_underrun_usr_2 = 0
[1742331.086164]
[1742331.092492] wlan: [6802:F:DP] ac_mu_mimo_mpdus_queued_usr_3 = 0
[1742331.100214] wlan: [6802:F:DP] ac_mu_mimo_mpdus_tried_usr_3 = 0
[1742331.106400] wlan: [6802:F:DP] ac_mu_mimo_mpdus_failed_usr_3 = 0
[1742331.112458] wlan: [6802:F:DP] ac_mu_mimo_mpdus_requeued_usr_3 = 0
[1742331.118464] wlan: [6802:F:DP] ac_mu_mimo_err_no_ba_usr_3 = 0
[1742331.124580] wlan: [6802:F:DP] ac_mu_mimo_mpdu_underrun_usr_3 = 0
[1742331.130434] wlan: [6802:F:DP] ac_mu_mimo_ampdu_underrun_usr_3 = 0
[1742331.130434]
[1742331.136580] wlan: [6802:F:DP] HTT_TX_PDEV_MU_MIMO_AX_MPDU_STATS:
[1742331.136580]
[1742331.144520] wlan: [6802:F:DP] ax_mu_mimo_mpdus_queued_usr_0 = 0
[1742331.152400] wlan: [6802:F:DP] ax_mu_mimo_mpdus_tried_usr_0 = 0
[1742331.158451] wlan: [6802:F:DP] ax_mu_mimo_mpdus_failed_usr_0 = 0
[1742331.164545] wlan: [6802:F:DP] ax_mu_mimo_mpdus_requeued_usr_0 = 0
[1742331.170558] wlan: [6802:F:DP] ax_mu_mimo_err_no_ba_usr_0 = 0
[1742331.176590] wlan: [6802:F:DP] ax_mu_mimo_mpdu_underrun_usr_0 = 0
[1742331.182513] wlan: [6802:F:DP] ax_mu_mimo_ampdu_underrun_usr_0 = 0
[1742331.182513]
[1742331.188667] wlan: [6802:F:DP] ax_mu_mimo_mpdus_queued_usr_1 = 0
[1742331.196603] wlan: [6802:F:DP] ax_mu_mimo_mpdus_tried_usr_1 = 0
[1742331.202871] wlan: [6802:F:DP] ax_mu_mimo_mpdus_failed_usr_1 = 0
[1742331.208798] wlan: [6802:F:DP] ax_mu_mimo_mpdus_requeued_usr_1 = 0
[1742331.214803] wlan: [6802:F:DP] ax_mu_mimo_err_no_ba_usr_1 = 0
[1742331.220901] wlan: [6802:F:DP] ax_mu_mimo_mpdu_underrun_usr_1 = 0
[1742331.226776] wlan: [6802:F:DP] ax_mu_mimo_ampdu_underrun_usr_1 = 0
[1742331.226776]
[1742331.232948] wlan: [6802:F:DP] ax_mu_mimo_mpdus_queued_usr_2 = 0
[1742331.240846] wlan: [6802:F:DP] ax_mu_mimo_mpdus_tried_usr_2 = 0
[1742331.247018] wlan: [6802:F:DP] ax_mu_mimo_mpdus_failed_usr_2 = 0
[1742331.253083] wlan: [6802:F:DP] ax_mu_mimo_mpdus_requeued_usr_2 = 0
[1742331.259058] wlan: [6802:F:DP] ax_mu_mimo_err_no_ba_usr_2 = 0
[1742331.265153] wlan: [6802:F:DP] ax_mu_mimo_mpdu_underrun_usr_2 = 0
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
[1742331.271072] wlan: [6802:F:DP] ax_mu_mimo_ampdu_underrun_usr_2 = 0
[1742331.271072]
[1742331.277208] wlan: [6802:F:DP] ax_mu_mimo_mpdus_queued_usr_3 = 0
[1742331.285173] wlan: [6802:F:DP] ax_mu_mimo_mpdus_tried_usr_3 = 0
[1742331.291280] wlan: [6802:F:DP] ax_mu_mimo_mpdus_failed_usr_3 = 0
[1742331.297367] wlan: [6802:F:DP] ax_mu_mimo_mpdus_requeued_usr_3 = 0
[1742331.303343] wlan: [6802:F:DP] ax_mu_mimo_err_no_ba_usr_3 = 0
[1742331.309414] wlan: [6802:F:DP] ax_mu_mimo_mpdu_underrun_usr_3 = 0
[1742331.315322] wlan: [6802:F:DP] ax_mu_mimo_ampdu_underrun_usr_3 = 0
[1742331.315322]
[1742331.321578] wlan: [6802:F:DP] ax_mu_mimo_mpdus_queued_usr_4 = 0
[1742331.329378] wlan: [6802:F:DP] ax_mu_mimo_mpdus_tried_usr_4 = 0
[1742331.335558] wlan: [6802:F:DP] ax_mu_mimo_mpdus_failed_usr_4 = 0
[1742331.341625] wlan: [6802:F:DP] ax_mu_mimo_mpdus_requeued_usr_4 = 0
[1742331.347622] wlan: [6802:F:DP] ax_mu_mimo_err_no_ba_usr_4 = 0
[1742331.353690] wlan: [6802:F:DP] ax_mu_mimo_mpdu_underrun_usr_4 = 0
[1742331.359638] wlan: [6802:F:DP] ax_mu_mimo_ampdu_underrun_usr_4 = 0
[1742331.359638]
[1742331.365768] wlan: [6802:F:DP] ax_mu_mimo_mpdus_queued_usr_5 = 0
[1742331.373675] wlan: [6802:F:DP] ax_mu_mimo_mpdus_tried_usr_5 = 0
[1742331.379810] wlan: [6802:F:DP] ax_mu_mimo_mpdus_failed_usr_5 = 0
[1742331.385898] wlan: [6802:F:DP] ax_mu_mimo_mpdus_requeued_usr_5 = 0
[1742331.391907] wlan: [6802:F:DP] ax_mu_mimo_err_no_ba_usr_5 = 0
[1742331.397948] wlan: [6802:F:DP] ax_mu_mimo_mpdu_underrun_usr_5 = 0
[1742331.403864] wlan: [6802:F:DP] ax_mu_mimo_ampdu_underrun_usr_5 = 0
[1742331.403864]
[1742331.410018] wlan: [6802:F:DP] ax_mu_mimo_mpdus_queued_usr_6 = 0
[1742331.417959] wlan: [6802:F:DP] ax_mu_mimo_mpdus_tried_usr_6 = 0
[1742331.424080] wlan: [6802:F:DP] ax_mu_mimo_mpdus_failed_usr_6 = 0
[1742331.430159] wlan: [6802:F:DP] ax_mu_mimo_mpdus_requeued_usr_6 = 0
[1742331.436226] wlan: [6802:F:DP] ax_mu_mimo_err_no_ba_usr_6 = 0
[1742331.442255] wlan: [6802:F:DP] ax_mu_mimo_mpdu_underrun_usr_6 = 0
[1742331.448116] wlan: [6802:F:DP] ax_mu_mimo_ampdu_underrun_usr_6 = 0
[1742331.448116]
[1742331.454302] wlan: [6802:F:DP] ax_mu_mimo_mpdus_queued_usr_7 = 0
[1742331.462199] wlan: [6802:F:DP] ax_mu_mimo_mpdus_tried_usr_7 = 0
[1742331.468383] wlan: [6802:F:DP] ax_mu_mimo_mpdus_failed_usr_7 = 0
[1742331.474429] wlan: [6802:F:DP] ax_mu_mimo_mpdus_requeued_usr_7 = 0
[1742331.480430] wlan: [6802:F:DP] ax_mu_mimo_err_no_ba_usr_7 = 0
[1742331.486491] wlan: [6802:F:DP] ax_mu_mimo_mpdu_underrun_usr_7 = 0
[1742331.492429] wlan: [6802:F:DP] ax_mu_mimo_ampdu_underrun_usr_7 = 0
[1742331.492429]
[1742331.498567] wlan: [6802:F:DP] HTT_TX_PDEV_AX_MU_OFDMA_MPDU_STATS:
[1742331.498567]
[1742331.506478] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_0 = 0
[1742331.514432] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_0 = 0
[1742331.520554] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_0 = 0
[1742331.526676] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_0 = 0
[1742331.532779] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_0 = 0
[1742331.539270] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_0 = 0
[1742331.544945] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_0 = 0
[1742331.544945]
[1742331.551242] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_1 = 0
[1742331.559153] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_1 = 0
[1742331.565415] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_1 = 0
[1742331.571592] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_1 = 0
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
[1742331.577641] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_1 = 0
[1742331.584159] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_1 = 0
[1742331.589795] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_1 = 0
[1742331.589795]
[1742331.596140] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_2 = 0
[1742331.604042] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_2 = 0
[1742331.610274] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_2 = 0
[1742331.616449] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_2 = 0
[1742331.622556] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_2 = 0
[1742331.629028] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_2 = 0
[1742331.634678] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_2 = 0
[1742331.634678]
[1742331.640933] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_3 = 0
[1742331.648929] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_3 = 0
[1742331.655166] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_3 = 0
[1742331.661400] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_3 = 0
[1742331.667499] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_3 = 0
[1742331.673908] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_3 = 0
[1742331.679557] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_3 = 0
[1742331.679557]
[1742331.685813] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_4 = 0
[1742331.693821] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_4 = 0
[1742331.700031] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_4 = 0
[1742331.706210] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_4 = 0
[1742331.712295] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_4 = 0
[1742331.718809] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_4 = 0
[1742331.724428] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_4 = 0
[1742331.724428]
[1742331.730697] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_5 = 0
[1742331.738665] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_5 = 0
[1742331.745006] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_5 = 0
[1742331.751077] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_5 = 0
[1742331.757155] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_5 = 0
[1742331.763671] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_5 = 0
[1742331.769337] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_5 = 0
[1742331.769337]
[1742331.775621] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_6 = 0
[1742331.783558] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_6 = 0
[1742331.789791] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_6 = 0
[1742331.795988] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_6 = 0
[1742331.802045] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_6 = 0
[1742331.808538] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_6 = 0
[1742331.814191] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_6 = 0
[1742331.814191]
[1742331.820468] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_7 = 0
[1742331.828484] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_7 = 0
[1742331.834679] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_7 = 0
[1742331.840853] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_7 = 0
[1742331.846930] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_7 = 0
[1742331.853431] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_7 = 0
[1742331.859057] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_7 = 0
[1742331.859057]
[1742331.865334] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_8 = 0
[1742331.873336] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_8 = 0
[1742331.879558] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_8 = 0
[1742331.885717] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_8 = 0
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
[1742331.891887] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_8 = 0
[1742331.898293] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_8 = 0
[1742331.904013] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_8 = 0
[1742331.904013]
[1742331.910189] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_9 = 0
[1742331.918217] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_9 = 0
[1742331.924430] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_9 = 0
[1742331.930607] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_9 = 0
[1742331.936666] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_9 = 0
[1742331.943210] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_9 = 0
[1742331.948811] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_9 = 0
[1742331.948811]
[1742331.955083] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_10 = 0
[1742331.963072] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_10 = 0
[1742331.969332] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_10 = 0
[1742331.975557] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_10 = 0
[1742331.981795] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_10 = 0
[1742331.988318] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_10 = 0
[1742331.994429] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_10 = 0
[1742331.994429]
[1742332.000739] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_11 = 0
[1742332.008871] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_11 = 0
[1742332.014801] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_11 = 0
[1742332.021070] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_11 = 0
[1742332.027206] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_11 = 0
[1742332.033818] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_11 = 0
[1742332.039880] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_11 = 0
[1742332.039880]
[1742332.046257] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_12 = 0
[1742332.054361] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_12 = 0
[1742332.060426] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_12 = 0
[1742332.066533] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_12 = 0
[1742332.072755] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_12 = 0
[1742332.079290] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_12 = 0
[1742332.085373] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_12 = 0
[1742332.085373]
[1742332.091761] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_13 = 0
[1742332.099791] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_13 = 0
[1742332.105768] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_13 = 0
[1742332.112011] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_13 = 0
[1742332.118241] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_13 = 0
[1742332.124769] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_13 = 0
[1742332.130854] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_13 = 0
[1742332.130854]
[1742332.137240] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_14 = 0
[1742332.145278] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_14 = 0
[1742332.151242] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_14 = 0
[1742332.157493] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_14 = 0
[1742332.163661] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_14 = 0
[1742332.170274] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_14 = 0
[1742332.176330] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_14 = 0
[1742332.176330]
[1742332.182677] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_15 = 0
[1742332.190753] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_15 = 0
[1742332.196744] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_15 = 0
[1742332.202988] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_15 = 0
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
[1742332.209138] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_15 = 0
[1742332.215792] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_15 = 0
[1742332.221834] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_15 = 0
[1742332.221834]
[1742332.228155] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_16 = 0
[1742332.236280] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_16 = 0
[1742332.242237] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_16 = 0
[1742332.248459] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_16 = 0
[1742332.254633] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_16 = 0
[1742332.261230] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_16 = 0
[1742332.267324] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_16 = 0
[1742332.267324]
[1742332.273642] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_17 = 0
[1742332.281719] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_17 = 0
[1742332.287699] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_17 = 0
[1742332.294020] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_17 = 0
[1742332.300109] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_17 = 0
[1742332.306718] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_17 = 0
[1742332.312791] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_17 = 0
[1742332.312791]
[1742332.319145] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_18 = 0
[1742332.327207] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_18 = 0
[1742332.333190] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_18 = 0
[1742332.339436] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_18 = 0
[1742332.345620] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_18 = 0
[1742332.352250] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_18 = 0
[1742332.358271] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_18 = 0
[1742332.358271]
[1742332.364626] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_19 = 0
[1742332.372760] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_19 = 0
[1742332.378672] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_19 = 0
[1742332.384925] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_19 = 0
[1742332.391092] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_19 = 0
[1742332.397699] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_19 = 0
[1742332.403770] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_19 = 0
[1742332.403770]
[1742332.410096] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_20 = 0
[1742332.418201] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_20 = 0
[1742332.424157] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_20 = 0
[1742332.430424] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_20 = 0
[1742332.436570] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_20 = 0
[1742332.443193] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_20 = 0
[1742332.449289] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_20 = 0
[1742332.449289]
[1742332.455594] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_21 = 0
[1742332.463705] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_21 = 0
[1742332.469663] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_21 = 0
[1742332.475896] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_21 = 0
[1742332.482065] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_21 = 0
[1742332.488653] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_21 = 0
[1742332.494758] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_21 = 0
[1742332.494758]
[1742332.501075] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_22 = 0
[1742332.509146] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_22 = 0
[1742332.515136] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_22 = 0
[1742332.521403] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_22 = 0
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
[1742332.527594] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_22 = 0
[1742332.534146] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_22 = 0
[1742332.540221] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_22 = 0
[1742332.540221]
[1742332.546581] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_23 = 0
[1742332.554642] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_23 = 0
[1742332.560620] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_23 = 0
[1742332.566889] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_23 = 0
[1742332.573032] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_23 = 0
[1742332.579674] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_23 = 0
[1742332.585707] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_23 = 0
[1742332.585707]
[1742332.592070] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_24 = 0
[1742332.600114] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_24 = 0
[1742332.606153] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_24 = 0
[1742332.612356] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_24 = 0
[1742332.618533] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_24 = 0
[1742332.625116] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_24 = 0
[1742332.631200] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_24 = 0
[1742332.631200]
[1742332.637530] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_25 = 0
[1742332.645625] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_25 = 0
[1742332.651588] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_25 = 0
[1742332.657841] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_25 = 0
[1742332.664009] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_25 = 0
[1742332.670624] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_25 = 0
[1742332.676671] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_25 = 0
[1742332.676671]
[1742332.683069] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_26 = 0
[1742332.691139] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_26 = 0
[1742332.697092] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_26 = 0
[1742332.703332] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_26 = 0
[1742332.709488] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_26 = 0
[1742332.716090] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_26 = 0
[1742332.722184] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_26 = 0
[1742332.722184]
[1742332.728502] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_27 = 0
[1742332.736583] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_27 = 0
[1742332.742585] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_27 = 0
[1742332.748805] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_27 = 0
[1742332.754980] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_27 = 0
[1742332.761620] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_27 = 0
[1742332.767667] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_27 = 0
[1742332.767667]
[1742332.773988] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_28 = 0
[1742332.782064] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_28 = 0
[1742332.788045] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_28 = 0
[1742332.794320] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_28 = 0
[1742332.800463] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_28 = 0
[1742332.807100] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_28 = 0
[1742332.813137] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_28 = 0
[1742332.813137]
[1742332.819492] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_29 = 0
[1742332.827552] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_29 = 0
[1742332.833540] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_29 = 0
[1742332.839830] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_29 = 0
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
[1742332.845966] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_29 = 0
[1742332.852550] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_29 = 0
[1742332.858617] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_29 = 0
[1742332.858617]
[1742332.864973] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_30 = 0
[1742332.873052] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_30 = 0
[1742332.879018] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_30 = 0
[1742332.885275] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_30 = 0
[1742332.891439] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_30 = 0
[1742332.898047] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_30 = 0
[1742332.904113] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_30 = 0
[1742332.904113]
[1742332.910450] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_31 = 0
[1742332.918637] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_31 = 0
[1742332.924506] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_31 = 0
[1742332.930764] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_31 = 0
[1742332.936917] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_31 = 0
[1742332.943540] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_31 = 0
[1742332.949586] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_31 = 0
[1742332.949586]
[1742332.955941] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_32 = 0
[1742332.964011] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_32 = 0
[1742332.970013] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_32 = 0
[1742332.976244] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_32 = 0
[1742332.982413] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_32 = 0
[1742332.989003] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_32 = 0
[1742332.995145] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_32 = 0
[1742332.995145]
[1742333.001422] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_33 = 0
[1742333.009492] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_33 = 0
[1742333.015486] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_33 = 0
[1742333.021747] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_33 = 0
[1742333.027891] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_33 = 0
[1742333.034544] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_33 = 0
[1742333.040572] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_33 = 0
[1742333.040572]
[1742333.046921] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_34 = 0
[1742333.055036] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_34 = 0
[1742333.061024] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_34 = 0
[1742333.067289] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_34 = 0
[1742333.073379] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_34 = 0
[1742333.079974] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_34 = 0
[1742333.086057] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_34 = 0
[1742333.086057]
[1742333.092470] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_35 = 0
[1742333.100470] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_35 = 0
[1742333.106452] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_35 = 0
[1742333.112705] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_35 = 0
[1742333.118884] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_35 = 0
[1742333.125460] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_35 = 0
[1742333.131549] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_35 = 0
[1742333.131549]
[1742333.137875] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_queued_usr_36 = 0
[1742333.145978] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_tried_usr_36 = 0
[1742333.152036] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_failed_usr_36 = 0
[1742333.158181] wlan: [6802:F:DP] ax_mu_ofdma_mpdus_requeued_usr_36 = 0
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
[1742333.164360] wlan: [6802:F:DP] ax_mu_ofdma_err_no_ba_usr_36 = 0
[1742333.170966] wlan: [6802:F:DP] ax_mu_ofdma_mpdu_underrun_usr_36 = 0
[1742333.177022] wlan: [6802:F:DP] ax_mu_ofdma_ampdu_underrun_usr_36 = 0
[1742333.177022]
root@OpenWrt:/#
```

3.23.1.20 Host AST entries for peer

Check AST entries for Peer view. This will list out all the currently associated STA

```
root@OpenWrt:/# iwpriv ath0 txrx_stats 262

[ 4524.205681] wlan: [7291:F:DP] AST Stats:

[ 4524.205930] wlan: [7291:F:DP]        Entries Added   = 0

[ 4524.209001] wlan: [7291:F:DP]        Entries Deleted = 0
```

3.23.1.21 Host ring pointers

Check ring pointers. This command provides a view of all the DMA rings. It can indicate any discrepancies in the refill path. (REO, WBM , TCL TQM , TxTx DMA)

```
root@OpenWrt:/# iwpriv ath0 txrx_stats 263
[434273.120177] wlan: [21268:F:DP] Reo_exception:SW:Head pointer = 0 Tail
Pointer = 0
[434273.120177]
[434273.120221] wlan: [21268:F:DP] Reo_exception:HW:Head pointer = 0 Tail
Pointer = 0
[434273.120221]
[434273.128453] wlan: [21268:F:DP] Reo_reinject:SW:Head pointer = 0 Tail Pointer
= 0
[434273.128453]
[434273.137541] wlan: [21268:F:DP] Reo_reinject:HW:Head pointer = 0 Tail Pointer
= 0
[434273.137541]
[434273.146295] wlan: [21268:F:DP] Reo_cmd:SW:Head pointer = 440 Tail Pointer =
440
[434273.146295]
[434273.155300] wlan: [21268:F:DP] Reo_cmd:HW:Head pointer = 440 Tail Pointer =
440
[434273.155300]
[434273.164345] wlan: [21268:F:DP] Reo_status:SW:Head pointer = 676 Tail Pointer
= 676
[434273.164345]
[434273.173283] wlan: [21268:F:DP] Reo_status:HW:Head pointer = 676 Tail Pointer
= 676
```

```
[434273.173283]
[434273.182575] wlan: [21268:F:DP] wbm2sw_release:SW:Head pointer = 1720 Tail
Pointer
……..(snippet cut)…….
```

In monitor mode, this command checks monitor ring pointers. If monitor status or buffer ring pointers are not changing, it indicates error condition.

```
root@OpenWrt:/# iwpriv ath0 txrx_stats 263
       Rx Refill Buf Ring : Head pointer = 4095  Tail Pointer = 0
        Rxdma Mon Buf Ring : Head pointer = 856  Tail Pointer = 858
       Rxdma Mon Dst Ring : Head pointer = 1625  Tail Pointer = 1624
       Rxdma Mon Status Ring : Head pointer = 228  Tail Pointer = 229
       Rxdma mon desc Ring : Head pointer = 269  Tail Pointer = 271
root@OpenWrt:/# iwpriv ath0 txrx_stats 263
       Rx Refill Buf Ring : Head pointer = 4095  Tail Pointer = 0
       Rxdma Mon Buf Ring : Head pointer = 3153  Tail Pointer = 3155
       Rxdma Mon Dst Ring : Head pointer = 546  Tail Pointer = 545
       Rxdma Mon Status Ring : Head pointer = 785  Tail Pointer = 786
       Rxdma mon desc Ring : Head pointer = 1278  Tail Pointer = 1280
```

3.23.1.22 Host monitor Rx statistics

Check monitor RX counters. This command indicates PDEV level monitor stats

```
iwpriv athN txrx_stats 264
root@OpenWrt:/# iwpriv ath0 txrx_stats 264
PDEV Rx Monitor Stats:
   status_ppdu_compl_cnt = 96288200
    status_ppdu_start_cnt = 192995354
   status_ppdu_end_cnt = 965867296
   status_ppdu_start_mis_cnt = 869579096
    status_ppdu_end_mis_cnt = 96707154
    status_ppdu_done_cnt = 96288200
……..(snippet cut)…….
```

3.23.1.23 Firmware MU/OFDMA statistics

Display the firmware multiuser/orthogonal frequency division multiple access (MU/OFDMA) stats information using the command for PDEV:
wifistats wifi0 17 --vdevid 255

NOTE For stats for a specific VDEV, specify it as follows: wifistats wifi0 17 --vdevid 0

```
root@OpenWrt:/# wifistats wifi0 17 --vdevid 255

HTT_TX_PDEV_MU_MIMO_SCH_STATS_TLV:
mu_mimo_sch_posted = 0
mu_mimo_sch_failed = 0
```

```
mu_mimo_ppdu_posted = 0

11ac MU_MIMO SCH STATS:
ac_mu_mimo_sch_nusers_0 = 0
ac_mu_mimo_sch_nusers_1 = 0
ac_mu_mimo_sch_nusers_2 = 0
ac_mu_mimo_sch_nusers_3 = 0

11ax MU_MIMO SCH STATS:
ax_mu_mimo_sch_nusers_0 = 0
ax_mu_mimo_sch_nusers_1 = 0
ax_mu_mimo_sch_nusers_2 = 0
ax_mu_mimo_sch_nusers_3 = 0
ax_mu_mimo_sch_nusers_4 = 0
ax_mu_mimo_sch_nusers_5 = 0
ax_mu_mimo_sch_nusers_6 = 0
ax_mu_mimo_sch_nusers_7 = 0

11ax OFDMA SCH STATS:
ax_ofdma_sch_nusers_0 = 0
ax_ofdma_sch_nusers_1 = 0
ax_ofdma_sch_nusers_2 = 0
ax_ofdma_sch_nusers_3 = 0
ax_ofdma_sch_nusers_4 = 0
ax_ofdma_sch_nusers_5 = 0
ax_ofdma_sch_nusers_6 = 0
ax_ofdma_sch_nusers_7 = 0
..
ax_ofdma_sch_nusers_73 = 0

HTT_TX_PDEV_MU_MIMO_AC_MPDU_STATS:
ac_mu_mimo_mpdus_queued_usr_0 = 0
ac_mu_mimo_mpdus_tried_usr_0 = 0
ac_mu_mimo_mpdus_failed_usr_0 = 0
ac_mu_mimo_mpdus_requeued_usr_0 = 0
ac_mu_mimo_err_no_ba_usr_0 = 0
ac_mu_mimo_mpdu_underrun_usr_0 = 0
ac_mu_mimo_ampdu_underrun_usr_0 = 0

ac_mu_mimo_mpdus_queued_usr_1 = 0
ac_mu_mimo_mpdus_tried_usr_1 = 0
ac_mu_mimo_mpdus_failed_usr_1 = 0
ac_mu_mimo_mpdus_requeued_usr_1 = 0
ac_mu_mimo_err_no_ba_usr_1 = 0
ac_mu_mimo_mpdu_underrun_usr_1 = 0
ac_mu_mimo_ampdu_underrun_usr_1 = 0

ac_mu_mimo_mpdus_queued_usr_2 = 0
ac_mu_mimo_mpdus_tried_usr_2 = 0
ac_mu_mimo_mpdus_failed_usr_2 = 0
ac_mu_mimo_mpdus_requeued_usr_2 = 0
ac_mu_mimo_err_no_ba_usr_2 = 0
ac_mu_mimo_mpdu_underrun_usr_2 = 0
ac_mu_mimo_ampdu_underrun_usr_2 = 0

ac_mu_mimo_mpdus_queued_usr_3 = 0
```

```
ac_mu_mimo_mpdus_tried_usr_3 = 0
ac_mu_mimo_mpdus_failed_usr_3 = 0
ac_mu_mimo_mpdus_requeued_usr_3 = 0
ac_mu_mimo_err_no_ba_usr_3 = 0
ac_mu_mimo_mpdu_underrun_usr_3 = 0
ac_mu_mimo_ampdu_underrun_usr_3 = 0

HTT_TX_PDEV_MU_MIMO_AX_MPDU_STATS:
ax_mu_mimo_mpdus_queued_usr_0 = 0
ax_mu_mimo_mpdus_tried_usr_0 = 0
ax_mu_mimo_mpdus_failed_usr_0 = 0
ax_mu_mimo_mpdus_requeued_usr_0 = 0
ax_mu_mimo_err_no_ba_usr_0 = 0
ax_mu_mimo_mpdu_underrun_usr_0 = 0
ax_mu_mimo_ampdu_underrun_usr_0 = 0

ax_mu_mimo_mpdus_queued_usr_1 = 0
ax_mu_mimo_mpdus_tried_usr_1 = 0
ax_mu_mimo_mpdus_failed_usr_1 = 0
ax_mu_mimo_mpdus_requeued_usr_1 = 0
ax_mu_mimo_err_no_ba_usr_1 = 0
ax_mu_mimo_mpdu_underrun_usr_1 = 0
ax_mu_mimo_ampdu_underrun_usr_1 = 0

ax_mu_mimo_mpdus_queued_usr_2 = 0
ax_mu_mimo_mpdus_tried_usr_2 = 0
ax_mu_mimo_mpdus_failed_usr_2 = 0
ax_mu_mimo_mpdus_requeued_usr_2 = 0
ax_mu_mimo_err_no_ba_usr_2 = 0
ax_mu_mimo_mpdu_underrun_usr_2 = 0
ax_mu_mimo_ampdu_underrun_usr_2 = 0

ax_mu_mimo_mpdus_queued_usr_3 = 0
ax_mu_mimo_mpdus_tried_usr_3 = 0
ax_mu_mimo_mpdus_failed_usr_3 = 0
ax_mu_mimo_mpdus_requeued_usr_3 = 0
ax_mu_mimo_err_no_ba_usr_3 = 0
ax_mu_mimo_mpdu_underrun_usr_3 = 0
ax_mu_mimo_ampdu_underrun_usr_3 = 0

ax_mu_mimo_mpdus_queued_usr_4 = 0
ax_mu_mimo_mpdus_tried_usr_4 = 0
ax_mu_mimo_mpdus_failed_usr_4 = 0
ax_mu_mimo_mpdus_requeued_usr_4 = 0
ax_mu_mimo_err_no_ba_usr_4 = 0
ax_mu_mimo_mpdu_underrun_usr_4 = 0
ax_mu_mimo_ampdu_underrun_usr_4 = 0

ax_mu_mimo_mpdus_queued_usr_5 = 0
ax_mu_mimo_mpdus_tried_usr_5 = 0
ax_mu_mimo_mpdus_failed_usr_5 = 0
ax_mu_mimo_mpdus_requeued_usr_5 = 0
ax_mu_mimo_err_no_ba_usr_5 = 0
ax_mu_mimo_mpdu_underrun_usr_5 = 0
ax_mu_mimo_ampdu_underrun_usr_5 = 0
```

Qualcomm Confidential – May Contain Trade Secrets 2021-07-26 03:57:44 PDT hanyafan@zte.com.cn

```
ax_mu_mimo_mpdus_queued_usr_6 = 0
ax_mu_mimo_mpdus_tried_usr_6 = 0
ax_mu_mimo_mpdus_failed_usr_6 = 0
ax_mu_mimo_mpdus_requeued_usr_6 = 0
ax_mu_mimo_err_no_ba_usr_6 = 0
ax_mu_mimo_mpdu_underrun_usr_6 = 0
ax_mu_mimo_ampdu_underrun_usr_6 = 0

ax_mu_mimo_mpdus_queued_usr_7 = 0
ax_mu_mimo_mpdus_tried_usr_7 = 0
ax_mu_mimo_mpdus_failed_usr_7 = 0
ax_mu_mimo_mpdus_requeued_usr_7 = 0
ax_mu_mimo_err_no_ba_usr_7 = 0
ax_mu_mimo_mpdu_underrun_usr_7 = 0
ax_mu_mimo_ampdu_underrun_usr_7 = 0

HTT_TX_PDEV_AX_MU_OFDMA_MPDU_STATS:
ax_mu_ofdma_mpdus_queued_usr_0 = 0
ax_mu_ofdma_mpdus_tried_usr_0 = 0
ax_mu_ofdma_mpdus_failed_usr_0 = 0
ax_mu_ofdma_mpdus_requeued_usr_0 = 0
ax_mu_ofdma_err_no_ba_usr_0 = 0
ax_mu_ofdma_mpdu_underrun_usr_0 = 0
ax_mu_ofdma_ampdu_underrun_usr_0 = 0

ax_mu_ofdma_mpdus_queued_usr_1 = 0
ax_mu_ofdma_mpdus_tried_usr_1 = 0
ax_mu_ofdma_mpdus_failed_usr_1 = 0
ax_mu_ofdma_mpdus_requeued_usr_1 = 0
ax_mu_ofdma_err_no_ba_usr_1 = 0
ax_mu_ofdma_mpdu_underrun_usr_1 = 0
ax_mu_ofdma_ampdu_underrun_usr_1 = 0

ax_mu_ofdma_mpdus_queued_usr_2 = 0
ax_mu_ofdma_mpdus_tried_usr_2 = 0
ax_mu_ofdma_mpdus_failed_usr_2 = 0
ax_mu_ofdma_mpdus_requeued_usr_2 = 0
ax_mu_ofdma_err_no_ba_usr_2 = 0
ax_mu_ofdma_mpdu_underrun_usr_2 = 0
ax_mu_ofdma_ampdu_underrun_usr_2 = 0

ax_mu_ofdma_mpdus_queued_usr_3 = 0
ax_mu_ofdma_mpdus_tried_usr_3 = 0
ax_mu_ofdma_mpdus_failed_usr_3 = 0
ax_mu_ofdma_mpdus_requeued_usr_3 = 0
ax_mu_ofdma_err_no_ba_usr_3 = 0
ax_mu_ofdma_mpdu_underrun_usr_3 = 0
ax_mu_ofdma_ampdu_underrun_usr_3 = 0

ax_mu_ofdma_mpdus_queued_usr_4 = 0
ax_mu_ofdma_mpdus_tried_usr_4 = 0
ax_mu_ofdma_mpdus_failed_usr_4 = 0
ax_mu_ofdma_mpdus_requeued_usr_4 = 0
ax_mu_ofdma_err_no_ba_usr_4 = 0
ax_mu_ofdma_mpdu_underrun_usr_4 = 0
```

```
ax_mu_ofdma_ampdu_underrun_usr_4 = 0

ax_mu_ofdma_mpdus_queued_usr_5 = 0
ax_mu_ofdma_mpdus_tried_usr_5 = 0
ax_mu_ofdma_mpdus_failed_usr_5 = 0
ax_mu_ofdma_mpdus_requeued_usr_5 = 0
ax_mu_ofdma_err_no_ba_usr_5 = 0
ax_mu_ofdma_mpdu_underrun_usr_5 = 0
ax_mu_ofdma_ampdu_underrun_usr_5 = 0

ax_mu_ofdma_mpdus_queued_usr_6 = 0
ax_mu_ofdma_mpdus_tried_usr_6 = 0
ax_mu_ofdma_mpdus_failed_usr_6 = 0
ax_mu_ofdma_mpdus_requeued_usr_6 = 0
ax_mu_ofdma_err_no_ba_usr_6 = 0
ax_mu_ofdma_mpdu_underrun_usr_6 = 0
ax_mu_ofdma_ampdu_underrun_usr_6 = 0

ax_mu_ofdma_mpdus_queued_usr_7 = 0
ax_mu_ofdma_mpdus_tried_usr_7 = 0
ax_mu_ofdma_mpdus_failed_usr_7 = 0
ax_mu_ofdma_mpdus_requeued_usr_7 = 0
ax_mu_ofdma_err_no_ba_usr_7 = 0
ax_mu_ofdma_mpdu_underrun_usr_7 = 0
ax_mu_ofdma_ampdu_underrun_usr_7 = 0
..
ax_mu_ofdma_mpdus_queued_usr_36 = 0
ax_mu_ofdma_mpdus_tried_usr_36 = 0
ax_mu_ofdma_mpdus_failed_usr_36 = 0
ax_mu_ofdma_mpdus_requeued_usr_36 = 0
ax_mu_ofdma_err_no_ba_usr_36 = 0
ax_mu_ofdma_mpdu_underrun_usr_36 = 0
ax_mu_ofdma_ampdu_underrun_usr_36 = 0
```

Only selfgen and MU stats have support to log per VDEV level.

3.23.1.24 SU Txbf/OL/IBF statistics

This stats give information on the number of MPDUs transmitted per Txbf/OL/IBF mode per mcs/bw/nss.

```
root@OpenWrt:/# wifistats wifi0 31
```

Sample:

```
HTT_STATS_PDEV_TX_RATE_TXBF_STATS:
tx_ol_mcs =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0,
12:0, 13:0,
tx_ibf_mcs =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0,
11:0, 12:0, 13:0,
tx_txbf_mcs =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:17, 7:30, 8:32, 9:71,
10:55, 11:1853, 12:0, 13:0,
tx_ol_nss =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
tx_ibf_nss =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
tx_txbf_nss =  0:2058, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
tx_ol_bw =  0:0, 1:0, 2:0, 3:0, 4:0,
tx_ibf_bw =  0:0, 1:0, 2:0, 3:0, 4:0,
```

```
    tx_txbf_bw =  0:5, 1:31, 2:2022, 3:0, 4:0,
```

3.23.1.25 Firmware peer/node statistics

```
root@OpenWrt:/# iwpriv wifi0 dp_peer_stats 00:03:7f:12:1b:1b:0x3
[  256.235442] wlan: [35: F:DP] HTT_PEER_DETAILS_TLV:
[  256.235469] wlan: [35: F:DP] peer_type = 0
[  256.239147] wlan: [35: F:DP] sw_peer_id = 2
[  256.243232] wlan: [35: F:DP] vdev_pdev_ast_idx = 9895936
[  256.247304] wlan: [35: F:DP] mac_addr(upper 4 bytes) = 310313728
[  256.252871] wlan: [35: F:DP] mac_addr(lower 2 bytes) = 6939
[  256.258873] wlan: [35: F:DP] peer_flags = 24704
[  256.264151] wlan: [35: F:DP] qpeer_flags = 8
[  256.264151]
[  256.268669] wlan: [35: F:DP] HTT_PEER_STATS_CMN_TLV:
[  256.274553] wlan: [35: F:DP] ppdu_cnt = 0
[  256.279599] wlan: [35: F:DP] mpdu_cnt = 25
[  256.283509] wlan: [35: F:DP] msdu_cnt = 0
[  256.287501] wlan: [35: F:DP] pause_bitmap = 0
[  256.291555] wlan: [35: F:DP] block_bitmap = 0
[  256.295922] wlan: [35: F:DP] current_timestamp = 0
[  256.295922]
[  256.300260] wlan: [35: F:DP] HTT_TX_PEER_RATE_STATS_TLV:
[  256.306375] wlan: [35: F:DP] tx_ldpc = 7
[  256.311893] wlan: [35: F:DP] rts_cnt = 0
[  256.315780] wlan: [35: F:DP] ack_rssi = 0
[  256.319702] wlan: [35: F:DP] tx_mcs =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:7,
8:0, 9:0, 10:0, 11:0,
[  256.323613] wlan: [35: F:DP] tx_su_mcs =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0,
7:0, 8:0, 9:0, 10:0, 11:0,
[  256.332849] wlan: [35: F:DP] tx_mu_mcs =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0,
7:0, 8:0, 9:0, 10:0, 11:0,
[  256.342386] wlan: [35: F:DP] tx_nss =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:7,

[  256.351865] wlan: [35: F:DP] tx_bw =  0:0, 1:0, 2:7, 3:0,
[  256.358844] wlan: [35: F:DP] tx_stbc =
[  256.364414] wlan: [35: F:DP] tx_pream =  0:5, 1:0, 2:0, 3:7, 4:7, 5:0, 6:0,
[  256.368492] wlan: [35: F:DP] tx_gi[0] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0,
7:0, 8:0, 9:0, 10:0, 11:0,
[  256.375542] wlan: [35: F:DP] tx_gi[1] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0,
7:7, 8:0, 9:0, 10:0, 11:0,
[  256.384672] wlan: [35: F:DP] tx_gi[2] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0,
7:0, 8:0, 9:0, 10:0, 11:0,
[  256.394027] wlan: [35: F:DP] tx_gi[3] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0,
7:0, 8:0, 9:0, 10:0, 11:0,
[  256.403429] wlan: [35: F:DP] tx_dcm =  0:0, 1:0, 2:0, 3:0, 4:0,
[  256.403429]
[  256.412779] wlan: [35: F:DP] HTT_RX_PEER_RATE_STATS_TLV:
[  256.420395] wlan: [35: F:DP] nsts = 0
[  256.425719] wlan: [35: F:DP] rx_ldpc = 0
[  256.429230] wlan: [35: F:DP] rts_cnt = 0
[  256.433262] wlan: [35: F:DP] rssi_mgmt = 0
[  256.437127] wlan: [35: F:DP] rssi_data = 0
[  256.441034] wlan: [35: F:DP] rssi_comb = 0
[  256.445143] wlan: [35: F:DP] rx_mcs =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
8:0, 9:0, 10:0, 11:0,
[  256.449241] wlan: [35: F:DP] rx_nss =  0:0, 1:0, 2:0, 3:0, 4:0, 5:3, 6:0,
```

```
7:14,
[  256.458523] wlan: [35: F:DP] rx_dcm =  0:0, 1:0, 2:0, 3:0, 4:0,
[  256.465966] wlan: [35: F:DP] rx_stbc =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0,
7:0, 8:0, 9:0, 10:0, 11:0,
[  256.471979] wlan: [35: F:DP] rx_bw =  0:0, 1:0, 2:0, 3:0,
[  256.481093] wlan: [35: F:DP] rssi_chain[0] =  0:0, 1:0, 2:0, 3:0,
[  256.486551] wlan: [35: F:DP] rssi_chain[1] =  0:0, 1:0, 2:0, 3:0,
[  256.492716] wlan: [35: F:DP] rssi_chain[2] =  0:0, 1:0, 2:0, 3:0,
[  256.498971] wlan: [35: F:DP] rssi_chain[3] =  0:0, 1:0, 2:0, 3:0,
[  256.505042] wlan: [35: F:DP] rssi_chain[4] =  0:0, 1:0, 2:0, 3:0,
[  256.511203] wlan: [35: F:DP] rssi_chain[5] =  0:0, 1:0, 2:0, 3:0,
[  256.517367] wlan: [35: F:DP] rssi_chain[6] =  0:0, 1:0, 2:0, 3:0,
[  256.523531] wlan: [35: F:DP] rssi_chain[7] =  0:0, 1:0, 2:0, 3:0,
[  256.529699] wlan: [35: F:DP] rx_gi[0] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0,
7:0, 8:0, 9:0, 10:0, 11:0,
[  256.535875] wlan: [35: F:DP] rx_gi[1] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:3, 6:0,
7:14, 8:0, 9:0, 10:0, 11:0,
[  256.545190] wlan: [35: F:DP] rx_gi[2] =  0:0, 1:0, 2:0, 3:0, 4:0, 5:0, 6:0,
7:0, 8:0, 9:0, 10:0, 11:0,
[  256.554882] wlan: [35: F:DP] rx_gi[3] =  0:0, 1:8, 2:0, 3:0, 4:17, 5:0, 6:0,
7:0, 8:0, 9:0, 10:0, 11:0,
[  256.563990] wlan: [35: F:DP] rx_pream =  0:8, 1:0, 2:0, 3:17, 4:0, 5:0, 6:0,
[  256.563990]
```

3.23.1.26 Display firmware per-peer statistics

The design of wifistats application is used to get the DP statistics from FW. Wifistats application is nl80211 application, which sends the HTT statistics request to the FW and prints the statistics on receiving the response. Application sends a nl80211 request for statistics and sends the cookie as PID of the process and waits for the asynchronous response from the FW

Driver, up on receiving the HTT FW statistics response, sends a nl80211 broadcast message for the application to print the statistics. The application listening for the nl messages looks filter outs the message based on message type and the cookie value. If cookie value corresponds to current application, it prints the statistics and exits.

This is an alternative for existing DP statistics commands, “iwpriv athX txrx_stats <1-19>” and “iwpriv wifiX dp_peer_stats”, with the difference being that the statistics are displayed in user space instead of being logged in the kernel.

Generic usage (iwpriv athX txrx_stats)

```
wifistats
         - necessary args
                <radio_name>    - ex: wifiX
                <cmd_id>        - 1 - 19
         - optional args
                 <config_param0>         default 0 if not mentioned
                 <config_param1>         default 0 if not mentioned
                 <config_param2>         default 0 if not mentioned
                 <config_param3>         default 0 if not mentioned
Example:
        wifistats wifiX <cmd_id> <optional config_params>
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Usage for peer stats (iwpriv wifiX dp_peer_stats)

```
wifistats
         - necessary args
                <radio_name>    - ex: wifiX
                 <cmid_id> is 11
                 One of either two below: --peerid or--mac
                        --mac            followed by MAC in
aa:bb:cc:dd:ee:ff format
                                -OR-
                        --peerid         followed by sw_peer_id
         - optional args
                 --mode  followed by peer_stats_req_mode
                 --mask  followed by req_type_bitmask
Example:
        wifistats wifiX 11 --mac aa:bb:cc:dd:ee:ff <optional args>
                -OR-
        wifistats wifiX 11 --peerid N <optional args>
```

3.23.2 Support for control frame Tx and Rx statistics

Control frame statistics for Tx and Rx packets are displayed in the outputs of the iwpriv athX txrx_stats and iwpriv wifi0 fc_peer_stats commands.

Control frame information in Tx path statistics

- BAR frames—In the output of the `iwpriv ath0 txrx_stats 12` command, the `su_bar` and `ax_mu_bar_trigger` fields display the Block Acknowledgement Request (BAR) frame information.

```
[  4]  0.0-10.0 sec   159 MBytes   133 Mbits/sec
root@OpenWrt:/# iwpriv ath0 txrx_stats 12
[  436.728276] wlan: [35: F:DP] HTT_TX_SELFGEN_CMN_STATS_TLV:
[  436.728304] wlan: [35: F:DP] mac_id__word = 0
[  436.732729] wlan: [35: F:DP] su_bar = 8
[  436.737102] wlan: [35: F:DP] rts = 0
```

```
[172694.997686] wlan: [2041: F:DP] ax_mu_mimo_brp6_err = 0
[172695.002914] wlan: [2041: F:DP] ax_mu_mimo_brp7_err = 0
[172695.008111] wlan: [2041: F:DP] ax_basic_trigger_err = 0
[172695.013344] wlan: [2041: F:DP] ax_bsr_trigger_err = 0
[172695.018866] wlan: [2041: F:DP] ax_mu_bar_trigger_err = 0
[172695.023827] wlan: [2041: F:DP] ax_mu_rts_trigger_err = 0
```

```
root@OpenWrt:/# iwpriv ath0 txrx_stats 12
[172694.725066] wlan: [2041: F:DP] HTT_TX_SELFGEN_CMN_STATS_TLV:
[172694.725094] wlan: [2041: F:DP] mac_id__word = 0
[172694.729817] wlan: [2041: F:DP] su_bar = 0
[172694.734517] wlan: [2041: F:DP] rts = 0
[172694.738414] wlan: [2041: F:DP] cts2self = 0
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

- RTS frames—In the output of the `iwpriv ath0 txrx_stats 9` command, the `rts_cnt` field displays the receive to send (RTS) frame information.

```
root@OpenWrt:/# iwpriv ath0 txrx_stats 9
[ 6660.287115] wlan: [35: F:DP] HTT_TX_PDEV_RATE_STATS_TLV:
[ 6660.287219] wlan: [35: F:DP] mac_id__word = 0
[ 6660.291545] wlan: [35: F:DP] tx_ldpc = 15371
[ 6660.295855] wlan: [35: F:DP] rts_cnt = 162
[ 6660.300169] wlan: [35: F:DP] ack_rssi = 0
```

- RTS success—In the output of the `iwpriv ath0 txrx_stats 9` command, the `rts_success` field displays the RTS success information.

```
root@OpenWrt:/# iwpriv ath0 txrx_stats 9
[  378.847402] wlan: [129: F:DP] HTT_TX_PDEV_RATE_STATS_TLV:
[  378.847429] wlan: [129: F:DP] mac_id__word = 0
[  378.851839] wlan: [129: F:DP] tx_ldpc = 528
[  378.856151] wlan: [129: F:DP] rts_cnt = 1456
[  378.860318] wlan: [129: F:DP] rts_success = 833
[  378.864758] wlan: [129: F:DP] ack_rssi = 0
```

- CTS frames—In the output of the `iwpriv ath0 txrx_stats 12` command, the `cts2self` field displays the clear to send (CTS) frame information.

```
root@OpenWrt:/# iwpriv ath0 txrx_stats 12
[ 6845.212241] wlan: [33: F:DP] HTT_TX_SELFGEN_CMN_STATS_TLV:
[ 6845.212270] wlan: [33: F:DP] mac_id__word = 0
[ 6845.216802] wlan: [33: F:DP] su_bar = 0
[ 6845.221101] wlan: [33: F:DP] rts = 0
[ 6845.224735] wlan: [33: F:DP] cts2self = 0
[ 6845.228519] wlan: [33: F:DP] qos_null = 0
[ 6845.232437] wlan: [33: F:DP] delayed_bar_1 = 0
```

Control frame information in Tx path statistics

- BA and ACK frames—In the output of the `iwpriv ath0 txrx_stats 1` command, the HTT_TX_PDEV_STATS_CMN_TLV: ack_tlv_proc field displays the block acknowledgment (BA) and acknowledgment (ACK) frame information.

```
root@OpenWrt:/# iwpriv ath0 txrx_stats 1
[ 6717.951308] wlan: [35: F:DP] HTT_TX_PDEV_STATS_CMN_TLV:
[ 6717.951351] wlan: [35: F:DP] mac_id__word = 0
[ 6717.955387] wlan: [35: F:DP] hw_queued = 100221
[ 6717.959900] wlan: [35: F:DP] hw_reaped = 100221
```

```
[ 6718.099715] wlan: [35: F:DP] mpdus_truncated = 1
[ 6718.104613] wlan: [35: F:DP] mpdus_ack_failed = 10613
[ 6718.109171] wlan: [35: F:DP] mpdus_expired = 0
[ 6718.114169] wlan: [35: F:DP] mpdus_seq_hw_retry = 2707
[ 6718.118462] wlan: [35: F:DP] ack_tlv_proc = 13071
```

- BAR frames—In the output of the `iwpriv wifi0 fc_peer_stats xx:xx:xx:xx:xx:xx` command, BAR statistics are provided per peer, per TID in the REO queue stats (TID: 0): bar_rcvd_cnt field.

```
^Croot@OpenWrt:/# iwpriv wifi0 fc_peer_stats 8C:FD:F0:07:34:0B
[ 6529.729660] wlan: [1730: F:DP] Node Tx Stats:
[ 6529.729660]
[ 6529.729690] wlan: [1730: F:DP] Total Packet Completions = 13384004
[ 6529.734400] wlan: [1730: F:DP] Total Bytes Completions = -1131434808
[ 6529.740545] wlan: [1730: F:DP] Success Packets = 13384004
[ 6529.747062] wlan: [1730: F:DP] Success Bytes = -1131434808
[ 6530.201090] wlan: [0: F:DP] REO queue stats (TID: 0):
[ 6530.201090] ssn: 1485
[ 6530.201090] curr_idx  : 0
[ 6530.201090] pn_31_0   : 00000000
[ 6530.201090] pn_63_32  : 00000000
[ 6530.201090] pn_95_64  : 00000000
[ 6530.201090] pn_127_96 : 00000000
[ 6530.201090] last_rx_enq_tstamp : 7f52f8e6
[ 6530.201090] last_rx_deq_tstamp : 7f52f8e6
[ 6530.201090] rx_bitmap_31_0     : 00000000
[ 6530.201090] rx_bitmap_63_32    : 00000000
[ 6530.201090] rx_bitmap_95_64    : 00000000
[ 6530.201090] rx_bitmap_127_96   : 00000000
[ 6530.201090] rx_bitmap_159_128  : 00000000
[ 6530.201090] rx_bitmap_191_160  : 00000000
[ 6530.201090] rx_bitmap_223_192  : 00000000
[ 6530.201090] rx_bitmap_255_224  : 00000000
[ 6530.201090]
[ 6530.250973] wlan: [0: F:DP] curr_mpdu_cnt      : 0
[ 6530.250973] curr_msdu_cnt      : 0
[ 6530.250973] fwd_timeout_cnt    : 0
[ 6530.250973] fwd_bar_cnt        : 0
[ 6530.250973] dup_cnt            : 10031
[ 6530.250973] frms_in_order_cnt  : 4761041
[ 6530.250973] bar_rcvd_cnt       : 0
[ 6530.250973] mpdu_frms_cnt      : 4771072
[ 6530.250973] msdu_frms_cnt      : 4771072
```

3.23.3 Supported Tx and Rx per-PPDU statistical details

For Tx, the following per-PPDU statistics are supported:

status	Completion status - OK/Filter/Abort/Timeout
ppdu_id	PPDU Id
vdev_id	VAP Id
num_users	Number of Users in transmission (Valid for MU/OFDMA)
num_mpdu	Number of MPDUs in PPDU
num_msdu	Number of MSDUs in PPDU
channel	Channel information
ack_rssi	RSSI value of last ack packet (units=dB above noise floor)
ppdu_start_timestamp	TSF at PPDU start
Per User	
tid	TID
peer_id	Datapath Peer ID
peer mac address	Peer MAC Address
frame_ctrl	802.11 header - Frame Control field
qos_ctrl	802.11 header - QoS Control field

mpdu_tried	Number of MPDUs enqueued for transmission
mpdu_success	Number of MPDUs successfully transmitted
long_retries	Number of MPDUs retried (long retries)
short_retries	Number of MPDUs retried (short retries)
is_ampdu	AMPDU or non-AMPDU
success_bytes	Bytes transmitted successfully
retry_bytes	Bytes for MPDUs retried
failed_mpdus	Number of MPDUs failed transmission
ltf_size	PLCP preamble size
stbc	STBC transmission or not
bw	Transmission bandwidth
	<enum 0 transmit_bw_20_MHz>
	<enum 1 transmit_bw_40_MHz>
	<enum 2 transmit_bw_80_MHz>
	<enum 3 transmit_bw_160_MHz>
nss	NSS 1,2, ...8
mcs	MCS index
preamble	Legacy, HT, VHT, HE
gi	guard interval 800/400/1600/3200 ns
dcm	dcm
ldpc	ldpc
ppdu_type	SU/MU_MIMO/MU_OFDMA/MU_MIMO_OFDMA
ba_seq_no	Block Ack sequence number
ba_bitmap	Block Ack bitmap
start_seq	Sequence number of first MPDU
enq_bitmap	Enqueue MPDU bitmap

For Rx, the following per PPDU statistics are supported:

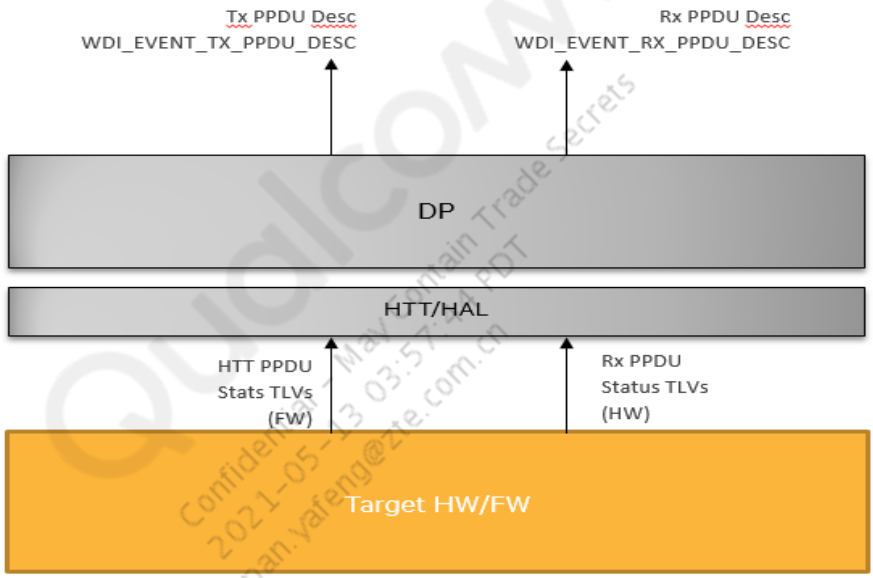
ppdu_id	PPDU Id
peer_id	Peer ID
peer_mac_addr	Peer Mac Address
first_data_seq_ctrl	Sequence control field of the first data frame
bw	Bandwidth
	<enum 0 bw_20_MHz>
	<enum 1 bw_40_MHz>
	<enum 2 bw_80_MHz>
	<enum 3 bw_160_MHz>

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

nss	NSS 1,2, ...8
mcs	MCS index
preamble	preamble
rssi	RSSI value (units = dB above noise floor)
channel	Channel information

To enable the Tx and Rx per-PPDU statistics, enter the iwpriv ath0 enable_ol_stats 1 command.

The following flow diagram illustrates the collection of Tx and Rx per-PPDU statistics:



3.23.3.1 Examples of console output for Tx and Rx per-PPDU statistics

Tx statistics example

```
[  912.901171] ****** Tx ppdu stats ******
[  912.903348] PPDU Id      : 983136
[  912.906890] MPDU count   : 0
[  912.910362] MSDU count   : 0
[  912.913318] Channel      : 5180
[  912.916178] Start TSF    : 891699199 End TSF     : 0
[  912.919046] Ack TSF      : 0
[  912.924256] Ack RSSI     : 0
[  912.927115]  Frame_type      :0
[  912.929984]  User     : 0
[  912.933196]     mac_addr        : 00:03:7f:12:ba:ba
[  912.935713]     peer_id         : 1
[  912.940745]     tid             : 18
[  912.944308]     is_ampdu        : 0
[  912.947864]     mpdu_tried_mcast: 0
```

```
[  912.951509]     mpdu_tried_ucast: 1
[  912.955073]     mpdu_success    : 1
[  912.958627]     long_retries    : 0
[  912.962187]     short_retries   : 0
[  912.965750]     success_msdus   : 0
[  912.969304]     retry_msdus     : 0
[  912.972867]     failed_msdus    : 0
[  912.976422]     success_bytes   : 0
[  912.979981]     retry_bytes     : 0
[  912.983546]     bw              : 2
[  912.987100]     nss             : 0
[  912.990658]     mcs             : 3
[  912.994222]     preamble        : 0
[  912.997777]     gi              : 0
[  913.001336]     dcm             : 0
[  913.004900]     ldpc            : 0
[  913.008454]     ppdu_type       : 0
[  913.012012]     ltf_size        : 1
[  913.015578]     stbc            : 0
[  913.019130]     he_re           : 0
[  913.022695]     txbf            : 0
```

Rx statistics example

```
[  913.026494] ****** Rx ppdu stats ******
[  913.029807] mac_addr              :8c:fd:f0:02:6e:4f
[  913.033375] rssi :                :16
[  913.038574] ppdu_id               :323
[  913.042134] duration              :0
[  913.045784] peer_id               :3
[  913.049513] tid                   :0
[  913.053076] first_data_seq_ctrl   :0
[  913.056631] bw                    :0
[  913.060190] mcs                   :3
[  913.063752] nss                   :0
[  913.067308] preamble              :1
[  913.070867] channel               :0
[  913.074429] timestamp             :0
[  913.077985] lsig_a                :0
[  913.081543] is_ampdu              :0
[  913.085108] num_mpdu              :0
[  913.088662] num_msdu              :0
[  913.092221] rate_info             :77824
[  913.095784] ltf_size              :0
[  913.099686] stbc                  :0
[  913.103249] he_re                 :0
[  913.106804] gi                    :0
[  913.110363] dcm                   :0
[  913.113927] ldpc                  :0
[  913.117481] ppdu_type             :0
```

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:37:44 PDT pan.yafeng@zte.com.cn

3.23.3.2 Support for Rx statistics to contain MU-MIMO and OFDMA parameters

In the WLAN driver, the following Rx information is already supported:

Rx info	Comment
Per PPDU	
rs_rssi	Per PPDU RX – rssi
rs_ratephy1	Already available in per PPDU
rs_ratephy2	Already available in per PPDU
rs_ratephy3	Already available in per PPDU
rs_noisefloor	
Per MSDU	
rs flags	

For MU-MIMO, the following is the definition in struct rx_msdu_start:

```
reception_type

            Indicates what type of reception this is.

            <enum 0     reception_type_SU >

            <enum 1     reception_type_MU_MIMO >

            <enum 2     reception_type_MU_OFDMA >

            <enum 3     reception_type_MU_OFDMA_MIMO >

      <legal all>
```

OFDMA is also supported on IPQ807x/IPQ807xA/IPQ817x/IPQ60xx/IPQ50xx chipsets. The new reception type is added in the current Rx info framework. The new field is called ppdu_type. The values of the ppdu_type field must be interpreted as follows:

- 0—reception_type_SU
- 1—reception_type_MU_MIMO
- 2—reception_type_MU_OFDMA
- 3—reception_type_MU_OFDMA_MIMO

Consider a sample configuration in which an AP is set up and a ping test to the uplink connection. Enable enhanced statistics using the following command, and view the value of the ppdu_type field.

```
root@OpenWrt:/# iwpriv wifi0 enable_ol_stats 1
root@OpenWrt:/# iwpriv ath1 txrx_fw_stats 1
[  299.237654] PPDU Id      : 983277
[  299.237670] MPDU count   : 0
[  299.239944] MSDU count   : 0
[  299.242889] Channel      : 5500
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
[  299.245747] Start TSF    : 276479999 End TSF     : 0
[  299.248615] Ack TSF      : 0
[  299.253826] Ack RSSI     : 0
[  299.256684]  User     : 0
[  299.259549]     mac_addr        : 00:03:7f:12:3c:3c
[  299.262070]     peer_id         : 1
[  299.267107]     tid             : 18
[  299.270660]     is_ampdu        : 0
[  299.274225]     mpdu_tried_mcast: 0
[  299.277866]     mpdu_tried_ucast: 1
[  299.281424]     mpdu_success    : 1
[  299.284989]     long_retries    : 0
[  299.288543]     short_retries   : 0
[  299.292102]     success_msdus   : 0
[  299.295666]     retry_msdus     : 0
[  299.299219]     failed_msdus    : 0
[  299.302778]     success_bytes   : 0
[  299.306342]     retry_bytes     : 0
[  299.309899]     bw              : 2
[  299.313461]     nss             : 0
[  299.317015]     mcs             : 3
[  299.320573]     preamble        : 0
[  299.324138]     gi              : 0
[  299.327692]     dcm             : 0
[  299.331251]     ldpc            : 0
[  299.334819]    ppdu_type    : 0
[  299.338369]     ltf_size        : 1
[  299.341928]     stbc            : 0
[  299.345491]     he_re           : 0
[ 299.349046]    txbf      : 0
```

3.23.3.3 Support for Tx statistics to contain MU-MIMO and OFDMA parameters

On packet transmission, the vendor driver returns parameters about MU-MIMO and OFDMA with which the packet was transmitted.

```
root@OpenWrt:/# iwpriv wifi0 enable_ol_stats 1[  252.579515] ****** Tx
ppdu stats ******
[  252.582261] PPDU Id      : 985476
[  252.585813] MPDU count   : 0
[  252.589283] MSDU count   : 0
[  252.592239] Channel      : 5180
[  252.595097] Start TSF    : 41122944 End TSF     : 0
[  252.597964] Ack TSF      : 0
[  252.602828] Ack RSSI     : 0
[  252.605947]  Frame_type      :0
[  252.608827] ****** Rx ppdu stats ******
[  252.612032] mac_addr             :00:03:7f:12:81:81
[  252.615588] rssi :               :32
[  252.620793] ppdu_id              :542
[  252.624356] duration             :0
[  252.627997] peer_id              :2
[  252.631730] tid                  :0
[  252.635294] first_data_seq_ctrl  :1
[  252.638847] bw                   :0
[  252.642412] mcs                  :7
```

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT panxiafeng@zte.com.cn

```
[  252.645966] nss                  :3
[  252.649525] preamble             :4
[  252.653089] channel              :0
[  252.656643] timestamp            :0
[  252.660202] lsig_a               :0
[  252.663766] is_ampdu             :0
[  252.667320] num_mpdu             :0
[  252.670879] num_msdu             :0
[  252.674442] rate_info            :291584
[  252.677998] ltf_size             :0
[  252.681908] stbc                 :0
[  252.685550] he_re                :0
[  252.689109] gi                   :0
[  252.692672] dcm                  :0
[  252.696227] ldpc                 :0
[  252.699786] ppdu_type            :0
[  252.703415] ****** Tx ppdu stats ******
[  252.706908] PPDU Id      : 989573
[  252.710463] MPDU count   : 0
[  252.713939] MSDU count   : 0
[  252.716885] Channel      : 5180
[  252.719751] Start TSF    : 41123447 End TSF     : 0
[  252.722623] Ack TSF      : 0
[  252.727475] Ack RSSI     : 0
[  252.730601]  Frame_type      :0
[  253.549014] ****** Tx ppdu stats ******
[  253.549036] PPDU Id      : 985480
[  253.551651] MPDU count   : 0
[  253.555114] MSDU count   : 0
[  253.558064] Channel      : 5180
[  253.560929] Start TSF    : 42122803 End TSF     : 0
[  253.563796] Ack TSF      : 0
[  253.568655] Ack RSSI     : 24
[  253.571779]  Frame_type      :0
[  253.574829] ****** Tx ppdu stats ******
[  253.577946] PPDU Id      : 985481
[  253.581499] MPDU count   : 0
[  253.584984] MSDU count   : 0
[  253.587922] Channel      : 5180
[  253.590787] Start TSF    : 42122954 End TSF     : 0
[  253.593659] Ack TSF      : 0
[  253.598515] Ack RSSI     : 0
[  253.601637]  Frame_type      :0
[  253.604672] ****** Tx ppdu stats ******
[  253.607724] PPDU Id      : 989578
[  253.611274] MPDU count   : 0
[  253.614751] MSDU count   : 0
[  253.617697] Channel      : 5180
[  253.620563] Start TSF    : 42123491 End TSF     : 0
[  253.623434] Ack TSF      : 0
[  253.628286] Ack RSSI     : 0
[  253.631412]  Frame_type      :0
[  253.634387] ****** Rx ppdu stats ******
[  253.637493] mac_addr             :00:03:7f:12:81:81
[  253.641051] rssi :               :32
[  253.646262] ppdu_id              :603
```

```
[  253.649816] duration            :0
[  253.653466] peer_id             :2
[  253.657195] tid                 :0
[  253.660753] first_data_seq_ctrl  :1
[  253.664316] bw                  :0
[  253.667872] mcs                 :7
[  253.671430] nss                 :3
[  253.674994] preamble            :1
[  253.678549] channel             :0
[  253.682112] timestamp           :0
[  253.685666] lsig_a              :0
[  253.689226] is_ampdu            :0
[  253.692790] num_mpdu            :0
[  253.696343] num_msdu            :0
[  253.699902] rate_info           :94976
[  253.703468] ltf_size            :0
[  253.707369] stbc                :0
[  253.710929] he_re               :0
[  253.714490] gi                  :0
[  253.718045] dcm                 :0
[  253.721604] ldpc                :0
[  253.725167] ppdu_type           :0

root@OpenWrt:/# [  256.549220] ****** Tx ppdu stats ******
[  256.549241] PPDU Id        : 985495
[  256.551857] MPDU count     : 0
[  256.555320] MSDU count     : 0
[  256.558269] Channel        : 5180
[  256.561134] Start TSF      : 45122970 End TSF     : 0
[  256.564002] Ack TSF        : 0
[  256.568860] Ack RSSI       : 0
[  256.571986]  Frame_type       :0
[  256.574850]  User        : 0
[  256.578066]     mac_addr        : 00:03:7f:12:81:81
[  256.580582]     peer_id         : 2
[  256.585615]     tid             : 0
[  256.589173]     is_ampdu        : 0
[  256.592733]     mpdu_tried_mcast: 0
[  256.596292]     mpdu_tried_ucast: 1
[  256.599852]     mpdu_success    : 0
[  256.603411]     long_retries    : 1
[  256.606969]     short_retries   : 0
[  256.610528]     success_msdus   : 0
[  256.614087]     retry_msdus     : 0
[  256.617646]     failed_msdus    : 0
[  256.621204]     success_bytes   : 0
[  256.624763]     retry_bytes     : 0
[  256.628323]     bw              : 4
[  256.631882]     nss             : 3
[  256.635440]     mcs             : 11
[  256.639000]     preamble        : 3
[  256.642560]     gi              : 1
[  256.646204]     dcm             : 0
[  256.649763]     ldpc            : 1
[  256.653323]     ppdu_type       : 0
[  256.656882]     ltf_size        : 1
```

```
[  256.660442]     stbc            : 0
[  256.663999]     he_re           : 0
[  256.667559]     txbf            : 0
[  256.671142] ****** Rx ppdu stats ******
[  256.674684] mac_addr             :00:03:7f:12:81:81
[  256.678236] rssi :               :32
[  256.683446] ppdu_id              :736
[  256.687000] duration             :0
[  256.690647] peer_id              :2
[  256.694384] tid                  :0
[  256.697938] first_data_seq_ctrl  :1
[  256.701498] bw                   :0
[  256.705061] mcs                  :7
[  256.708615] nss                  :3
[  256.712178] preamble             :3
[  256.715733] channel              :0
[  256.719292] timestamp            :0
[  256.722856] lsig_a               :0
[  256.726410] is_ampdu             :0
[  256.729969] num_mpdu             :0
[  256.733533] num_msdu             :0
[  256.737087] rate_info            :226048
[  256.740647] ltf_size             :0
[  256.744556] stbc                 :0
[  256.748198] he_re                :0
[  256.751757] gi                   :0
[  256.755321] dcm                  :0
[  256.758875] ldpc                 :0
[  256.762440] ppdu_type            :0
[  256.766094] ****** Tx ppdu stats ******
[  256.769557] PPDU Id      : 989593
[  256.773118] MPDU count   : 0
[  256.776583] MSDU count   : 0
[  256.779534] Channel      : 5180
[  256.782407] Start TSF    : 45123185 End TSF     : 0
[  256.785266] Ack TSF      : 0
[  256.790125] Ack RSSI     : 0
[  256.793253]  Frame_type      :0
[  256.796115]  User     : 0
[  256.799325]     mac_addr        : 00:03:7f:12:81:81
[  256.801846]     peer_id         : 2
[  256.806884]     tid             : 0
[  256.810438]     is_ampdu        : 0
[  256.814001]     mpdu_tried_mcast: 0
[  256.817557]     mpdu_tried_ucast: 1
[  256.821115]     mpdu_success    : 0
[  256.824678]     long_retries    : 1
[  256.828233]     short_retries   : 0
[  256.831792]     success_msdus   : 0
[  256.835355]     retry_msdus     : 0
[  256.838911]     failed_msdus    : 0
[  256.842473]     success_bytes   : 0
[  256.846029]     retry_bytes     : 0
[  256.849587]     bw              : 4
[  256.853150]     nss             : 3
[  256.856706]     mcs             : 11
```

```
[  256.860264]     preamble        : 3
[  256.863827]     gi              : 1
[  256.867469]     dcm             : 0
[  256.871028]     ldpc            : 1
[  256.874593]     ppdu_type       : 0
[  256.878147]     ltf_size        : 1
[  256.881705]     stbc            : 0
[  256.885270]     he_re           : 0
[  256.888823]     txbf            : 0
root@OpenWrt:/# ifconfig br-lan 192.168.10.9[  256.896264] ****** Tx ppdu
stats ******
[  256.899853] PPDU Id      : 985499
[  256.903673] MPDU count   : 0
[  256.907139] MSDU count   : 0
[  256.910091] Channel      : 5180
[  256.912960] Start TSF    : 45123431 End TSF     : 0
[  256.915822] Ack TSF      : 0
[  256.920680] Ack RSSI     : 24
[  256.923810]  Frame_type      :0
[  256.926944] ****** Tx ppdu stats ******
[  256.929974] PPDU Id      : 985500
[  256.933535] MPDU count   : 0
[  256.937001] MSDU count   : 0
[  256.939951] Channel      : 5180
[  256.942820] Start TSF    : 45123582 End TSF     : 0
[  256.945683] Ack TSF      : 0
[  256.950542] Ack RSSI     : 0
[  256.953670]  Frame_type      :0
[  256.956799] ****** Tx ppdu stats ******
[  256.959749] PPDU Id      : 989597
[  256.963308] MPDU count   : 0
[  256.966774] MSDU count   : 0
[  256.969724] Channel      : 5180
[  256.972595] Start TSF    : 45124110 End TSF     : 0
[  256.975457] Ack TSF      : 0
[  256.980316] Ack RSSI     : 0
[  256.983444]  Frame_type      :0
[  257.549283] ****** Tx ppdu stats ******
[  257.549303] PPDU Id      : 985504
[  257.551918] MPDU count   : 0
[  257.555382] MSDU count   : 0
[  257.558333] Channel      : 5180
[  257.561197] Start TSF    : 46123017 End TSF     : 0
[  257.564063] Ack TSF      : 0
[  257.568921] Ack RSSI     : 0
[  257.572046]  Frame_type      :0
[  257.574912]  User     : 0
[  257.578123]     mac_addr        : 00:03:7f:12:81:81
[  257.580645]     peer_id         : 2
[  257.585677]     tid             : 0
[  257.589236]     is_ampdu        : 0
[  257.592794]     mpdu_tried_mcast: 0
[  257.596353]     mpdu_tried_ucast: 1
[  257.599913]     mpdu_success    : 0
[  257.603476]     long_retries    : 1
[  257.607031]     short_retries   : 0
```

```
[  257.610590]     success_msdus   : 0
[  257.614149]     retry_msdus     : 0
[  257.617710]     failed_msdus    : 0
[  257.621267]     success_bytes   : 0
[  257.624825]     retry_bytes     : 0
[  257.628385]     bw              : 4
[  257.631944]     nss             : 3
[  257.635503]     mcs             : 10
[  257.639061]     preamble        : 3
[  257.642621]     gi              : 1
[  257.646267]     dcm             : 0
[  257.649826]     ldpc            : 1
[  257.653385]     ppdu_type       : 0
[  257.656943]     ltf_size        : 1
[  257.660503]     stbc            : 0
[  257.664062]     he_re           : 0
[  257.667622]     txbf            : 0
[  257.671200] ****** Rx ppdu stats ******
[  257.674746] mac_addr             :00:03:7f:12:81:81
[  257.678298] rssi :               :32
[  257.683509] ppdu_id              :791
[  257.687063] duration             :0
[  257.690709] peer_id              :2
[  257.694447] tid                  :0
[  257.698001] first_data_seq_ctrl  :1
[  257.701559] bw                   :0
[  257.705123] mcs                  :8
[  257.708678] nss                  :3
[  257.712241] preamble             :0
[  257.715795] channel              :0
[  257.719354] timestamp            :0
[  257.722919] lsig_a               :0
[  257.726472] is_ampdu             :0
[  257.730031] num_mpdu             :0
[  257.733595] num_msdu             :0
[  257.737149] rate_info            :33536
[  257.740711] ltf_size             :0
[  257.744618] stbc                 :0
[  257.748174] he_re                :0
[  257.751733] gi                   :0
[  257.755296] dcm                  :0
[  257.758851] ldpc                 :0
[  257.762414] ppdu_type            :0
```

3.23.4 Display statistics and packet logs for LTE-U

3.23.4.1 Tx and Rx firmware statistics

The txrx_fw_stats command needs one input argument. The most commonly used argument options available and their use are listed as follows:

```
iwpriv ath<if> txrx_fw_stats <argument>
```

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:41 PDT pan.yafeng@zte.com.cn

Argument Option	Output
1	Physical device statistics
3	Rx rate statistics

Physical device statistics

The physical device target data stats show the number of times various expected and unexpected transmit and receive events have happened. In addition to that it also displays remaining IRAM and DRAM memory in bytes.

The following is a sample output of the iwpriv ath0 txrx_fw_stats 1 command:

```
  iwpriv ath0 txrx_fw_stats 1
WAL Pdev stats:

### Tx ###
comp_queued       :     7
comp_delivered    :     7
msdu_enqued       :     196
wmm_drop          :     0
local_enqued      :     196
local_freed       :     196
hw_queued         :     196
hw_reaped         :     196
mac underrun      :     0
phy underrun      :     0
tx_abort          :     0
mpdus_requed      :     0
excess retries    :     1
last rc           :     3
sched self trig   :     0
ampdu retry failed:     0
illegal rate errs :     0
pdev cont xretry  :     0
pdev tx timeout   :     0
pdev resets       :     1
ppdu txop ovf     :     0
mcast Drop        :     0

### Rx ###
ppdu_route_change :     0
```

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

```
status_rcvd       :     21
r0_frags          :     0
r1_frags          :     0
r2_frags          :     0
r3_frags          :     0
htt_msdus         :     21
htt_mpdus         :     21
loc_msdus         :     17
loc_mpdus         :     17
oversize_amsdu    :     0
phy_errs          :     0
phy_errs dropped  :     0
mpdu_errs         :     0
 ############## Free memory###########
IRAM Remaining:         6308
DRAM Remaining:         3768
```

The following list describes the fields displayed in the Rx section of the output:

- ppdu_route_change: # of times for a received PPDU, part of MPDUs are data frames and part of the MPDUs are non-data frames
- status_rcvd: # of Rx status is used. One Rx status usually represents one MSDU
- r0_frags: # of buffer fragmentation happened in Ring 0. The buffer fragmentation means that a MSDU occupies more than one Rx buffer
- r1_frags: # of buffer fragmentation happened in Ring 1
- r2_frags: # of buffer fragmentation happened in Ring 2
- r3_frags: # of buffer fragmentation happened in Ring 3
- htt_msdus: # of data MSDUs received
- htt_mpdus: # of data MPDUs received
- loc_msdus: # of non-data MSDUs received
- loc_mpdus: # of non-data MPDUs received
- oversize_amsdu: # of the times that receiving an A-MSDU which has SDUs more than the size of Rx status ring
- phy_errs : # of received PPDUs as Phy errors
- phy_errs dropped : # of received PPDUs as Phy errors dropped
- mpdu_errs :
- # of MPDU received with error.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Receive rate control statistics

The Rx rate statistical details display fields, such as the number of times Rx frames were received using different rates modulation, coding, short vs. long GI, and RSSI.

The following is a sample output of the iwpriv ath0 txrx_fw_stats 3 command:

```
iwpriv ath0 txrx_fw_stats 3
RX Rate Info:
MCS counts (0..9): 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
SGI counts (0..9): 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
NSS  counts: 1x1 0, 2x2 0, 3x3 0, 4x4 0
NSTS count: 0
BW counts: 20MHz 0, 40MHz 0, 80MHz 0
Preamble counts: 0, 0, 0, 0, 0, 0
STBC rate counts (0..9): 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
LDPC TXBF Counts: 0, 0
RSSI (data, mgmt): 0, 13
RSSI Chain 0 (0x00 0x00 0x00 0x00)
RSSI Chain 1 (0x00 0x00 0x00 0x00)
RSSI Chain 2 (0x00 0x00 0x00 0x00)
```

The following list describes the output fields:

- MCS counts: Signifies the counters for each MCSs 0..9 in case of VHT, and MCS0..7 in the case of the HT association. For 802.11n MCS8..23 combine this field with NSS field, for example, MCS8 is NSS 2 MCS0. Note that this parameter does not include legacy OFDM/CCK rates.
- SGI counts: Displays the counters for each SGI enabled MCS
- NSS counts: Captures the number of spatial streams. This field indicates whether 1x1, 2x2, or 3x3 rate is being used. Combined with MCS, this field provides the actual (802.11n) MCS for HT.
- NSTS count: Indicates whether the frames are being sent with STBC enabled and the transmission is at a 1x1 rate. The NSTS count can be seen to be equal to the sum of STBC counts.
- BW counts: Indicates the number of received frames on 20, 40 and 80MHz. This parameter is useful to debug all the bandwidths (BWs) that the transmitter STA is currently using.
- Preamble counts: index 0 counts legacy (CCK/OFDM) PPDUs, 1 HT, 2 HT with BF (on QCA9880 always 0), 3 VHT and 4 VHT with BF (on QCA9880 always zero), 5 all other, for example, PHY error.
- STBC rate counts: Similar to MCS counts, this field displays the MCSs for which STBC is enabled.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

- LDPC TXBF counts: The first counter increments for each received LDPC PPDU, whereas the second counter increments for each received TxBF frames (which is not supported on QCA9880).
- RSSI (data, mgmt.): absolute RSSI value as seen in received MAC descriptor for data and management frame respectively.
- RSSI chain 0/1/2: (sec80, sec40, sec20, pri20) displays the RSSI present in MAC descriptor for given chain across primary/secondary channels. This might be useful to ensure that all chains are balanced.

NOTE When rate is fixed, only one of MCS counts is incremented with autorate. Most of the MCSs are used depending on the environment.

3.23.4.2 Packet log analysis for 11ac chipsets

For details regarding packet log analysis on IPQ807x chipsets, see the *Packet log analysis for IPQ807x/IPQ807xA/IPQ817x devices* section *Using logs and diagnostic tools for debugging* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6.

Packet log (pktlog) tool captures the TX, RX descriptors in real time and provides rates statistics, RSSI statistics, aggregate information, data type description, RA/TA/ BSSID addresses, MPDU sequence numbers, BA, TID, MSDU length, and timestamp.

Pktlog tool essentially consists of ath_pktlog.ko (driver), pktlogconf (user space utility) and the pktlogdecoder_11ac.pl Perl script.

Assuming that the pktlog is compiled into the codebase, perform the following procedures to successfully set up the pktlog for QCA9880:

Collect pktlog and analyze pktlog data

To collect pktlog, enter the following pktlogconf command:

```
root@OpenWrt:/# pktlogconf -h
```

Usage of the packet log configuration command is as follows:

Usage: pktlogconf [-a adapter] [-e[event-list]] [-d adapter] [-s log-size] [-t -k -l]

[-b -p -i]

-h show this usage

-a configures packet logging for specific 'adapter';

configures system-wide logging if this option is not specified

-d disable packet logging

-e enable logging events listed in the 'event-list'; event-list is an optional comma separated list

of one or more of the following: rx tx rcf rcu ani (for example, pktlogconf -erx,rcu,tx)

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

-s change the size of log-buffer to "log-size" bytes

-t enable logging of TCP headers

-k enable triggered stop by a threshold number of TCP SACK packets

-l change the number of packets to log after triggered stop

-b enable triggered stop by a throughput threshold

-p enable triggered stop by a PER threshold

-i change the time period of counting throughput/PER

To analyze pktlog, use the pktlogdecoder_11ac.pl command as follows:

```
pktlogdecoder_11ac.pl
```

Usage: ./pktlogdecoder_11ac.pl

[-agmnprtuvBDERGKJISVP] [-d <delta>] [-D <start>,<end>]

[-T <seqStart>,<ackStart>] [-I <offset>,<slot>]

(-x | -X ap_addr | pktLog.dat)

The following are the basic options:

- -h Print this message

The following are the options for the record-by-record log display (default):

- -a show all
- -v show detailed EVM info in log

The following are the options for displaying info other than the record-by-record log (each of the following is mutually-exclusive):

- -D show full Tx desc info from <start index> to <end index>
- -G print A-MPDU stats
- -R print rate statistics
- -K print PHY error stats
- -S print RSSI statistics
- -V print EVM statistics

Enable packet log and view statistics

To enable pktlogconf on wifi1, enter the following command:

```
pktlogconf -a wifi1 -e
```

To stop pktlog, enter the following command:

```
pktlogconf -d wifi1
```

After pktlog is stopped, the pktlog can be copied from the following sysfile:

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
/proc/ath_pktlog/wifi1
```

Example: cp /proc/ath_pktlog/wifi1 /tmp/pktlog.dat

To perform a file transfer using TFTP, use following commands:

- For AP135, “tftp <ip>”
- For IPQ4019, “tftp-hpa –m binary <ip>”

IPQ4019 tftp-hpa tool, by default, uses ASCII mode to transfer file. If –m binary is not used, the transferred pktlog file is corrupt if transferred in ASCII mode.

To display rate statistics:

```
pktlogdecoder_11ac.pl -R pktlog-wifi1-AP135-10.2.2.18_default.dat
File endianness:   Big (AP)
File version:      10010
File size:         1048544 bytes
MAC Ver/Rev:       0x0/0x0 (Owl 1.0 - sw hardcoded version)
Number of records: 4920
Finished loading file

                        -- Tx Rate Statistics --

         Num    Avg #           Subframes Failures   PER     Excess Failures
Excess Failures %
Rate    Frames  Subfr       %      Good   Bad                 RTS    Data
RTS    Data
-------------------------------------------------------------------------------
----------------
 vM09x3h    537   10.0 100.00%     5358    0        0.00%         0      0
0.00%     0.00%
-------------------------------------------------------------------------------
---------------
Total Data frames    : 537
Total MSDUs          : 16069
Total RTS failures   : 0 (PER = 0.00%)
Total Data failures  : 0 (PER = 0.00%)
Note: Excessive Retries (ERs) are accounted for by the last rate of the series.


                       -- Rx Rate Statistics --

         Num Avg #           Subframes         Weighted         ACK/BA
Rate  Frames Subfr       %   Good  Bad     PER      PER Retries  Fails <---  %
-----------------------------------------------------------------------------
 vM09x3h     40  18.9 100.00%    757    0   0.00%    0.00%       0      0   0.00%
-----------------------------------------------------------------------------
Total Data frames         : 40
Total MSDUs               : 2226
Total A-MPDU Subframe PER : 0.00% (Good=757, Bad=0)
Total retries             : 0 (0.00% of good frames)
ACK/BA (dup seq) failures : 0.00%
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

3.23.5 Enhanced client statistics for acknowledgment failures, modulation rates, authentication details

This section describes the display of enhanced statistics of acknowledgment failures, modulation rates, authentication details. Use the `apstats -s -m xx:xx:xx:xx:xx:xx` (MAC address of station) command, the `athstats` command, the `wlanconfig ath1 list` command, and the `iwpriv ath1 txrx_fw_stats 1` command for 802.11ac chipsets or the `iwpriv ath1 txrx_stats 1` command for802.11ax chipsets, appropriately, to view these enhanced client or STA statistical details. The following additional statistics are displayed:

1. **tx unaggregated unacked frames, tx aggr unacked subframes**, and **MPDUs_ack_failed**—Number of unacknowledged data MPDU transmissions. Firmware transmits this statistical counter by enabling the packetlog event, WMI_PKTLOG_EVENT_TX, for QCA9880 and as a part of host target transport (HTT) statistics for QCA9980. This statistic was previously available for display for QCA9980 family of chipsets, and this implementation is extended for displaying values for QCA9880 chipsets.

   NOTE These values are counts based for offload and DA.

   For direct attach (DA) models, the *tx unaggregated unacked frames, tx aggr unacked subframes* fields are displayed in the output of the `athstats -i wifiX` command. For offload (OL) models, the *MPDUs_ack_failed* field is displayed in the output of the `iwpriv athX txrx_fw_stats 1` command.

   NOTE For offload models, enter the `iwpriv wifix enable_ol_stats 1` command to enable the collection of offload statistics before viewing this parameter in the output of the `iwpriv athX txrx_fw_stats 1` command.

2. **Average Tx Rate (kbps)**—Weighted average of the Tx modulation rate for data MPDU. Firmware transmits this information when WMI_PKTLOG_EVENT_RCU is enabled for QCA9880 and as a part of enhanced statistics for QCA9980. The calculated average Tx modulation rate value is updated to display in apstats tool.

   The *Average Tx Rate (kbps)* field is displayed in the output of the `apstats -s -m <mac_addr>` command for DA radios. The *Avg ppdu Tx Rate (kbps)* field is displayed in the output of the `apstats -s -m <mac_addr>` command for OL radios.

   NOTE For offload models, enter the `iwpriv wifix enable_ol_stats 1` command to enable the collection of offload statistics before viewing this parameter in the output of the `apstats -s -m <mac_addr>` command.

3. **Average Rx Rate (kbps)**—Weighted average of the Rx modulation rate for data MPDU. This information is retrieved from Rx descriptors and handled in rx_indication_msg. No separate path is implemented for QCA9880 or QCA9980 per design. The calculated average Rx modulation rate value is updated to display in apstats tool.

   The *Average Rx Rate (kbps)* field is displayed in the output of the `apstats -s -m <mac_addr>` command for DA radios. The *Avg ppdu Rx Rate (kbps)* field is displayed in the output of the `apstats -s -m <mac_addr>` command for OL radios.

NOTE For offload models, enter the `iwpriv wifix enable_ol_stats 1` command to enable the collection of offload statistics before viewing this parameter in the output of the `apstats -s -m <mac_addr>` command.

4. **Channel Busy**—Percentage of each second for which the channel is busy (TX/RX/noise). This value is already displayed in the output of the apstats command in the Channel Utilization field. The channel utilization parameter is disabled by default; it can be enabled by the `iwpriv athx chutil_enab 1` command.

   The *value* displayed in the output of the `iwpriv athx get_chutil` command represents the channel utilization value in percentage.

5. **802.11 Auth Attempts**—Number of user equipment/UE authentication attempts. This parameter displays the values that the WLAN driver calculates per VAP and per radio in the output of the apstats command.

   The *802.11 Auth Attempts* field is displayed in the output of the `apstats -r -i wifiX` command.

6. **MLME Authorize Success**—Number of UE authentications for MAC layer management entity (MLME). This parameter considers the number of UEs that are in MLME authorized state. This value is calculated as the number of authorized successes and displayed in the output of the apstats command for VAP-level and radio-level statistics.

   The *MLME Authorize Success* field is displayed in the output of the `apstats -r -i wifiX` command.

7. **Self BSS chan util, OBSS chan util** (Busy by Other Radio, Busy by non-WiFi)—Percentage of each second for which the transmitter is busy by another radio and the percentage of each second for which the transmitter is busy by non-Wi-Fi. Both these types of percentages are not separated and calculated by hardware counters. As a result, the combined value of (Busy by other radio + Busy by non-Wi-Fi) and Self-BSS are displayed in the output of the `apstats -r -i wifiX` command as per-radio statistics.

   These *Self BSS chan util, OBSS chan util* parameters apply per-radio and not per-VAP, and are displayed in the output of the `apstats -r -i wifiX` command.

8. **ASSOCTIME**—Total amount of time for which the clients are connected. This parameter is per radio, per AP, and per SSID.

   The *ASSOCTIME* parameter is also displayed in the output of the `wlanconfig athx list` command.

9. **802.11 Auth Attempts, 802.11 Auth Success**—Number of user equipment/UE (client) authentication failures. This value is calculated as follows:

   Authentication failures = Authentication Attempts - Authentication Successes **(1)**

   This value is displayed in apstats tool in both radio-level and VAP-level statistics.

   The *802.11 Auth Attempts, 802.11 Auth Success* fields are displayed in the output of the `apstats -r -i wifiX` command.

10. **Connections refuse Radio limit**—Number of connection that are rejected because the connected users limit is exceeded. Number of clients that are rejected after the radio limit (num_vaps) and vap limit (max_aid) are exceeded is updated from driver when node join occurs. This value is displayed in apstats tool in both radio and VAP-level statistics. Also, a

iwpriv command has been added to configure the radio limit dynamically that was earlier hardcoded to a predefined value during attach. Enter the `iwpriv wifix max_sta <val>` command to configure the maximum number of client connections, where `<val>` is max number of connections allowed per radio (limit) +1.

The *Connection refuse Radio limit* field is displayed in the output of the `apstats -r -i wifix` command.

11. **AID**—A unique session ID that might be common for more than one record (long). The unique session ID is considered as the association ID (AID).

    This *Session ID* parameter is displayed in the output of the `wlanconfig athx list` command for both DA and OL models.

NOTE The acknowledgment failures, Tx average modulation rate, and Rx average modulation rate parameters have different path implementation for direct attach and offload architectures. For offload models only, enter the `iwpriv wifix enable_ol_stats 1` command to enable the collection of offload statistics before viewing these parameters in the output of the apstats command. This behavior is because these parameters are retrieved from firmware through resource-intensive data path, and also a slight performance impact is observed when these parameters are collected and displayed.

The following sample outputs illustrate the newly added display fields:

Output of apstats for DA radios:

```
root@OpenWrt:/# apstats -r -i wifi0
Radio Level Stats: wifi0
Tx Data Packets                 = 0
Tx Data Bytes                   = 0
Rx Data Packets                 = 376
Rx Data Bytes                   = 59264
Tx Unicast Data Packets         = 0
Tx Multi/Broadcast Data Packets = 0
Tx Data Packets per AC:
 Best effort                    = 0
 Background                     = 0
 Video                          = 0
 Voice                          = 0
Rx Data Packets per AC:
 Best effort                    = 0
 Background                     = 0
 Video                          = 0
 Voice                          = 0
Channel Utilization (0-255)     = <DISABLED>
Tx Beacon Frames                = 1266
Tx Mgmt Frames                  = 1619
Rx Mgmt Frames                  = 14004
Tx Ctl Frames                   = 0
Rx Ctl Frames                   = 0
Rx RSSI                         = 41
Rx PHY errors                   = 710
Rx CRC errors                   = 583
Rx MIC errors                   = 0
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
Rx Decryption errors            = 0
Rx errors                       = 0
Tx failures                     = 0
Tx Dropped                      = 2
Connections refuse Radio limit  = 0
Connections refuse Vap limit    = 0
802.11 Auth Attempts            = 1
802.11 Auth Success             = 1
MLME Authorize Attempts         = 1
MLME Authorize Success          = 1
Self BSS chan util              = 16
OBSS chan util                  = 80
Throughput (kbps)               = <DISABLED>
PER over configured period (%)  = <DISABLED>
Total PER (%)                   = 0
```

Output of apstats for OL radios:

```
root@OpenWrt:/# apstats -s -m 04:1b:6d:ba:40:29
Node Level Stats: 04:1b:6d:ba:40:29 (under VAP ath1)
Tx Data Packets                 = 0
Tx Data Bytes                   = 0
Tx Success Data Packets         = 0
Tx Success Data Bytes           = 0
Tx Data Packets per AC:
 Best effort                    = 0
 Background                     = 0
 Video                          = 0
 Voice                          = 0
Rx Data Packets per AC:
 Best effort                    = 0
 Background                     = 0
 Video                          = 0
 Voice                          = 0
Tx Success Unicast Data Packets = 0
Tx Success Unicast Data Bytes   = 0
Tx Success Multicast Data Packets = 0
Tx Success Multicast Data Bytes = 0
Last Packet Error Rate (PER)    = 13
Rx Data Packets                 = 0
Rx Data Bytes                   = 0
Rx Unicast   Data Packets       = 0
Rx Unicast   Data Bytes         = 0
Rx Multicast Data Packets       = 0
Rx Multicast Data Bytes         = 0
Avg ppdu Tx Rate (kbps)          = 0
Avg ppdu Rx Rate (kbps)          = 0
Last tx rate                    = 175500
Last rx rate                    = 325000
Last mgmt rx rate               = 6000
Rx MIC Errors                   = 0
Rx Decryption errors            = 0
Rx errors                       = 0
Packets Queued                  = 0
Host Discard                    = 0
Tx Mgmt Packets                 = 0
```

Qualcomm Confidential - May Contain Trade Secrets 2021-05-13 03:59:44 PDT pan.yafeng@zte.com.cn

```
Rx Mgmt Packets                 = 0
Rx mpdu count                   = 0
Rx ppdu count                   = 0
Rx retry count                  = 0
Tx failures                     = 0
Rx RSSI                         = 24
Rx MGMT RSSI                    = 18
Excessive retries per AC:
 Best effort                    = 1170
 Background                     = 0
 Video                          = 0
 Voice                          = 11
Ack RSSI chain 1                = 0
Ack RSSI chain 2                = 0
Ack RSSI chain 3                = 0
Ack RSSI chain 4                = 0
Band Width                      = 80
stbc                            tx(0) rx(0)
chainmask (NSS)                 tx(1) rx(1)
Tx packets for last one second  = 0
Tx bytes for last one second    = 0
Rx packets for last one second  = 0
Rx bytes for last one second    = 0


root@OpenWrt:/# athstats -i wifi0 | grep unacked
       277 tx unaggregated unacked frames
         0 tx aggr unacked subframes
         0 tx rifs: unacked subframes

root@OpenWrt:/# wlanconfig ath1 list
ADDR              AID CHAN TXRATE RXRATE RSSI MINRSSI MAXRSSI IDLE  TXSEQ
RXSEQ  CAPS       ACAPS    ERP    STATE MAXRATE(DOT11) HTCAPS ASSOCTIME
IEs   MODE                    PSMODE RXNSS TXNSS
04:1b:6d:ba:40:29    1  149   0M      0M   27       0      27    0      0
65535     E         0          b              0            AWQS 00:02:50
WME IEEE80211_MODE_11AC_VHT80   0        1     1


root@OpenWrt:/# apstats -s -m 04:1b:6d:ba:40:29
Node Level Stats: 04:1b:6d:ba:40:29 (under VAP ath0)
Tx Data Packets                 = 37554
Tx Data Bytes                   = 49797250
Tx Success Data Packets         = 0
Tx Success Data Bytes           = 0
Tx Data Packets per AC:
 Best effort                    = 0
 Background                     = 0
 Video                          = 0
 Voice                          = 0
Rx Data Packets per AC:
 Best effort                    = 0
 Background                     = 0
 Video                          = 0
 Voice                          = 0
Tx Success Unicast Data Packets = 0
```

```
Tx Success Unicast Data Bytes   = 0
Tx Success Multicast Data Packets = 0
Tx Success Multicast Data Bytes = 0
Last Packet Error Rate (PER)    = 0
Rx Data Packets                 = 42722
Rx Data Bytes                   = 39231993
Rx Unicast   Data Packets       = 42682
Rx Unicast   Data Bytes         = 39227006
Rx Multicast Data Packets       = 40
Rx Multicast Data Bytes         = 4987
Average Tx Rate (kbps)          = 71247
Average Rx Rate (kbps)          = 71686
Last tx rate                    = 0
Last rx rate                    = 0
Last mgmt rx rate               = 0
Rx MIC Errors                   = 0
Rx Decryption errors            = 0
Rx errors                       = 0
Packets Queued                  = 37554
Host Discard                    = 0
Tx Mgmt Packets                 = 0
Rx Mgmt Packets                 = 0
Rx mpdu count                   = 0
Rx ppdu count                   = 0
Rx retry count                  = 0
Tx failures                     = 0
Rx RSSI                         = 64
Rx MGMT RSSI                    = 0
Excessive retries per AC:
 Best effort                    = 0
 Background                     = 0
 Video                          = 0
 Voice                          = 0
Ack RSSI chain 1                = 0
Ack RSSI chain 2                = 0
Ack RSSI chain 3                = 0
Ack RSSI chain 4                = 0
Band Width                      = 20
stbc                            tx(0) rx(0)
chainmask (NSS)                 tx(1) rx(1)
Tx packets for last one second  = 0
Tx bytes for last one second    = 0
Rx packets for last one second  = 0
Rx bytes for last one second    = 0

root@OpenWrt:/# iwpriv ath1 txrx_fw_stats 1
[ 1811.510000] ### HOST MSDU TTL Stats ###
[ 1811.510000] Host_msdu_ttl     :0
[ 1811.520000] WAL Pdev stats:
[ 1811.530000]
[ 1811.530000] ### MEM ###
[ 1811.530000]  ############## Free memory###########
[ 1811.540000] IRAM Remaining: 1836
[ 1811.540000] DRAM Remaining: 540
[ 1811.550000] SRAM Remaining: 0
[ 1811.550000]
```

Qualcomm Confidential - May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

```
[ 1811.550000] ### Tx ###
[ 1811.560000] comp_queued       :1995897
[ 1811.560000] comp_delivered    :1995897
[ 1811.560000] msdu_enqued       :2004696
[ 1811.570000] wmm_drop          :0
[ 1811.570000] local_enqued      :8716
[ 1811.580000] local_freed       :8716
[ 1811.580000] hw_queued         :200322
[ 1811.590000] hw_reaped         :200320
[ 1811.590000] mac underrun      :0
[ 1811.590000] phy underrun      :0
[ 1811.600000] hw_paused         :0
[ 1811.600000] seq_posted        :0
[ 1811.610000] mu_seq_posted     :0
[ 1811.610000] seq_failed        :0
[ 1811.620000] seq_restarted     :0
[ 1811.620000] tx_abort          :0
[ 1811.620000] mpdus_requed      :129736
[ 1811.630000] mpdus_sw_flush    :0
[ 1811.630000] mpdus_hw_filter   :0
[ 1811.640000] mpdus_truncated   :0
[ 1811.640000] mpdus_ack_failed  :0
[ 1811.640000] mpdus_expired     :0
[ 1811.650000] excess retries    :6399
[ 1811.650000] last rc           :196
[ 1811.660000] sched self trig   :3636
[ 1811.660000] ampdu retry failed:10
[ 1811.660000] illegal rate errs :0
[ 1811.670000] pdev cont xretry  :0
[ 1811.670000] pdev tx timeout   :0
[ 1811.680000] pdev resets       :1
[ 1811.680000] ppdu txop ovf     :0
[ 1811.680000] mcast Drop        :0
[ 1811.690000]
[ 1811.690000] ### Rx ###
[ 1811.690000] ppdu_route_change :123
[ 1811.700000] status_rcvd       :1562499
[ 1811.700000] r0_frags          :0
[ 1811.700000] r1_frags          :45
[ 1811.710000] r2_frags          :55
[ 1811.710000] htt_msdus         :1562499
[ 1811.720000] htt_mpdus         :1562497
[ 1811.720000] loc_msdus         :17697
[ 1811.730000] loc_mpdus         :17697
[ 1811.730000] oversize_amsdu    :0
[ 1811.740000] phy_errs          :0
[ 1811.740000] phy_errs dropped  :18
[ 1811.740000] mpdu_errs         :11376
[ 1811.750000] pdev_rx_timeout   :0
[ 1811.750000] ovfl_mpdu_errs    :0
```

3.23.6 Extended VoW statistics

Support for VoW statistics is extended. Debug counters are added. The host-change involves the TID value that was being overwritten in Tx path of host driver. It was caused as part of Mesh Mode changes, which unconditionally modified TID using SKB priority field. The fix is to retain TID as sent by transmitter.

Debug counters are added at following points in Wi-Fi host:

- Counters to keep track of number of video packets
- Counters to calculate interframe delay between video packets.

3.23.6.1 Host debug changes

- The debug counters added only for video packets are as follows:
  - Tx : No. of pkts from Linux subsystem to WIFI Driver Entry
  - Tx : No. of pkts from Wifi Driver Entry to OL Layer inside driver
  - Tx : No. of Tx Completion pkts
  - Rx : No. of pkts from received by Host from FW
  - Rx : No., of pkts dropped/discarded
  - Rx : No. of pkts delivered to stack
- The delay counters added for video packets are as follows:
  - Linux subsystem to Wi-Fi interframe delay –
    - interframe delay both tx (linux to wifi) and rx (wifi to linux)
  - Delay between Wi-Fi host to CE submission
  - Delay between Wi-Fi host to TX completion
- All these debug counters are added to following iwpriv commands :
  - iwpriv wifiX fc_stats_global
  - iwpriv athX txrx_fw_stats 8

3.23.6.2 Wi-Fi host

In the `iwpriv wifiX fc_stats_global 1` command, replace X with index number currently in use

The following are the counters of interest:

- Peer_q_full—This counter gets incremented whenever it fails to enqueue the packet to Host Queue. Any non-zero value here denotes packet loss.

  Enqueue_fail_cnt—This counter gets incremented if it fails to find WDS entries for oncoming packet. These are not genuine packet losses because a certain amount of time is consumed to populate bridge/WDS entries after association is complete. Therefore, bidirectional traffic is recommended to resolve ARP/WDS entries before start of test.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

3.23.6.3 Usage of VoW packet counters

This section explains each packet counters among different software components:

GMAC

Assuming that a user is configuring eth0 on AP and STA, and running downlink traffic.

Video packet statistics:

Command: 'ethtool -S eth0'

Useful stats: 'rx_video_frames': total video packets received by AP

'tx_video_frames': packets video transmitted by STA

Few more useful statistics:

Command: 'ethtool -S eth0'

Useful statistics:

AP side:

'rx_overflow_errors': Error due to RX Fifo Overflow

'rx_descriptor_errors': Error due to invalid RX descriptor

'rx_crc_errors': Error due to Checksum error

'rx_length_errors': Error due to invalid packet length

'rx_ip_header_errors': Error due to invalid IP Header

'rx_ip_payload_errors': Error due to Invalid IP Payload

'rx_no_buffer_errors': Error as we ran out of RX descriptors

'rx_receive_errors': Generic Receive error

STA side:

'tx_underflow_errors': Error due to MAC Aborting frame as underflow

'tx_ip_header_errors': Error due to invalid IP Header

'tx_ip_payload_errors': Error detected in TCP/UDP/ICMP datagram

Command: ifconfig eth0 | grep dropped

Useful statistics: Sample output

RX packets:15275 errors:0 dropped:0 overruns:0 frame:0

TX packets:435980 errors:0 dropped:0 overruns:0 carrier:0

Qualcomm Confidential – May Contain Trade Secrets 2022-05-13 09:57:14 PDT pan.yang@zte.com.cn

dropped count in TX indicates invalid skb received from the stack

dropped count in RX indicates Erroneous packet received from NSS-FW

3.24 Enhanced NSS offloading statistics

Starting with QCA_Networking_2020.SPF.11.2 release, this feature is introduced on IPQ40xx, IPQ806x, IPQ807x/IPQ807xA/IPQ817x, IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx, and QCN90xx or QCN602x platforms. NSS Wi-Fi offload supports the following statistics as part of this enhancement. Types of supported statistics are as follows:

- Per client (MAC address)
- Per direction (uplink/downlink)
- Per access category

1. Number of packets (MSDUs) and bytes, successful and with errors – Tx: tx_success, tx_failed and Rx: to_stack, rx_failed
2. Average physical-layer data rate
3. Average latency
4. RSSI and SNR
5. RSSI and chan_nf
6. Number of retries per MSDU – Tx: tx_retry, Rx: rx_retry.
7. Number of bytes, tried, successful, with errors – Tx: ucast, bcast, tx_failed and Rx: unicast, bcast, rx_failed
8. Number of PPDUs, successful, with errors and number of retries
9. Histogram of MCS, spatial streams

For offload models, enter the iwpriv wifix enable_ol_stats 1 command to enable the collection of offload statistics before viewing the different statistical details.

The following are the commands to display the enhanced NSS-offloaded statistics. They are categorized based on the commands that are displayed. Some of the statistics above are displayed both on per client/per VAP or per radio level. The level is indicated against each of the commands. Any notes if to be taken care of are also mentioned against each statistic.

3.24.1 athstats -i wifiX: This is per-radio level statistic

tx_success:

Successful Tx bytes : 1160

Successful Tx Packets : 11

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT

to_stack:

Total packets sent up the stack : 160

Total bytes sent up the stack : 37120

Tx/Rx msdu count on each NSS.

Rx :

NSS(1-8) = 0 0 0 0 0 0 0 0 0

Tx:

NSS(1-8) = 0 0 0 0 0 0 0 0

Tx/Rx MCS histogram

MCS histogram is displayed per radio level for legacy, AC, and AX

Command: athstats -i wifiX

```
Tx Rate Info:
HT MCS 0 (BPSK 1/2)    = 0
HT MCS 1 (QPSK 1/2)    = 0
HT MCS 2 (QPSK 3/4)    = 0
HT MCS 3 (16-QAM 1/2) = 0
HT MCS 4 (16-QAM 3/4) = 0
HT MCS 5 (64-QAM 2/3) = 0
HT MCS 6 (64-QAM 3/4) = 0
HT MCS 7 (64-QAM 5/6) = 0
INVALID = 0

VHT MCS 0 (BPSK 1/2)      = 0
VHT MCS 1 (QPSK 1/2)      = 0
VHT MCS 2 (QPSK 3/4)      = 0
VHT MCS 3 (16-QAM 1/2)    = 0
VHT MCS 4 (16-QAM 3/4)    = 0
VHT MCS 5 (64-QAM 2/3)    = 0
VHT MCS 6 (64-QAM 3/4)    = 0
VHT MCS 7 (64-QAM 5/6)    = 0
VHT MCS 8 (256-QAM 3/4)   = 0
VHT MCS 9 (256-QAM 5/6)   = 0
VHT MCS 10 (1024-QAM 3/4) = 0
VHT MCS 11 (1024-QAM 5/6) = 0
INVALID = 0

HE MCS 0 (BPSK 1/2)      = 0
HE MCS 1 (QPSK 1/2)      = 0
HE MCS 2 (QPSK 3/4)      = 0
HE MCS 3 (16-QAM 1/2)    = 0
HE MCS 4 (16-QAM 3/4)    = 0
```

Qualcomm Confidential - May Contain Trade Secrets 2022-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

```
HE MCS 5 (64-QAM 2/3)     = 0
HE MCS 6 (64-QAM 3/4)     = 0
HE MCS 7 (64-QAM 5/6)     = 0
HE MCS 8 (256-QAM 3/4)    = 0
HE MCS 9 (256-QAM 5/6)    = 0
HE MCS 10 (1024-QAM 3/4) = 0
HE MCS 11 (1024-QAM 5/6) = 0
INVALID = 0

Rx Rate Info:

HT MCS 0 (BPSK 1/2)   = 0
HT MCS 1 (QPSK 1/2)   = 0
HT MCS 2 (QPSK 3/4)   = 0
HT MCS 3 (16-QAM 1/2) = 0
HT MCS 4 (16-QAM 3/4) = 0
HT MCS 5 (64-QAM 2/3) = 0
HT MCS 6 (64-QAM 3/4) = 0
HT MCS 7 (64-QAM 5/6) = 0
INVALID  = 0

VHT MCS 0 (BPSK 1/2)      = 0
VHT MCS 1 (QPSK 1/2)      = 0
VHT MCS 2 (QPSK 3/4)      = 0
VHT MCS 3 (16-QAM 1/2)    = 0
VHT MCS 4 (16-QAM 3/4)    = 0
VHT MCS 5 (64-QAM 2/3)    = 6
VHT MCS 6 (64-QAM 3/4)    = 5
VHT MCS 7 (64-QAM 5/6)    = 0
VHT MCS 8 (256-QAM 3/4)   = 0
VHT MCS 9 (256-QAM 5/6)   = 0
VHT MCS 10 (1024-QAM 3/4) = 0
VHT MCS 11 (1024-QAM 5/6) = 0
INVALID  = 0

HE MCS 0 (BPSK 1/2)      = 0
HE MCS 1 (QPSK 1/2)      = 0
HE MCS 2 (QPSK 3/4)      = 0
HE MCS 3 (16-QAM 1/2)    = 0
HE MCS 4 (16-QAM 3/4)    = 0
HE MCS 5 (64-QAM 2/3)    = 0
HE MCS 6 (64-QAM 3/4)    = 0
HE MCS 7 (64-QAM 5/6)    = 0
HE MCS 8 (256-QAM 3/4)   = 0
HE MCS 9 (256-QAM 5/6)   = 0
HE MCS 10 (1024-QAM 3/4) = 0
HE MCS 11 (1024-QAM 5/6) = 0
INVALID  = 0
```

Spatial streams

Rx :

NSS(1-8) = 0 11 0 0 0 0 0 0

Qualcomm Confidential – May Contain Trade Secrets 2021-03-13 03:57:44 PDT pan yafeng@zte.com.cn

Tx:

nss = 30 0 0 0 0 0 0 0

Number of PPDUs, successful, with errors and number of retries [***]

PPDU stats are mainly collected as part of the wifistats (or FW stats). Host collects per PPDU level stats but not aggregate stats. Hence, the below stats will only be displayed as part of the host stats.

/* Number of total PPDUs(DATA, MGMT, excludes selfgen) tried over the air (OTA) *

num_total_ppdus_tried_ota : 211415

/* Number of data PPDUs tried over the air (OTA) */

num_data_ppdus_tried_ota : 96

/* Num PPDU status processed from HW */

ppdu_recvd = 23769533;

/* Num MPDU across PPDUs with FCS ok */

mpdu_cnt_fcs_ok :

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT fan.yafeng@zte.com.cn

3.24.2 apstats -v -i athx: This is per-VAP level statistic

Tx failed/Rx failed: Only counts are currently collected.

Rx errors = 0

Rx Dropped = 0

Tx failures = 0

Tx Dropped = 0

Average physical-layer data rate

Command: apstats -v -i <vap-name>

Average Tx Rate (kbps) = 130000

Average Rx Rate (kbps) = 866700

3.24.3 wlanconfig ath0 list: This is per client-level statistic

RSSI : Receive signal strength indicator

SNR : Sound to noise ratio

Rssi = -41

SNR 52

3.24.4 iwconfig ath0: This is per-VAP level statistic

chan_nf

Noise level=-93 dBm

3.24.5 apstats -s -m <mac_addr>: This is per-client level statistic

Command: apstats -s -m <mac_addr>

Tx Data Packets : Number of Tx data packets

Tx Data Bytes : Number of Tx data bytes

Tx Success Unicast Data Packets : Number of Tx unicast data packets

Tx Success Unicast Data Bytes : Number of Tx unicast data bytes

Tx Success Multicast Data Packets : Number of Tx multicast data packets

Tx Success Multicast Data Bytes : Number of Tx multicast data bytes

Tx Success Broadcast Data Packets : Number of Tx broadcast data packets

Tx Success Broadcast Data Bytes : Number of Tx broadcast data bytes

Tx retry count : Number of Tx data retried msdu count

Tx failed : Number of Tx data msdu transmission failure count

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:41 PDT pan.yifeng@zte.com.cn

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Rx Data Packets : Number of Rx data packets

Rx Data Bytes : Number of Rx data bytes

Rx Unicast Data Packets : Number of Rx unicast data packets

Rx Unicast Data Bytes : Number of Rx unicast data bytes

Rx Multicast Data Packets : Number of Rx multicast data packets

Rx Multicast Data Bytes :Number of Rx multicast data packets

Rx retry count : Number of Rx retries msdu count

Rx error : Number Rx data msdu errors

Tx and Rx retries

Rx retry count = 0

Tx retry count = 0

tx_success, ucast, bcast, tx_failed

Tx Data Packets = 0

Tx Data Bytes = 0

Tx Success Unicast Data Packets = 0

Tx Success Unicast Data Bytes = 0

Tx Success Multicast Data Packets = 0

Tx Success Multicast Data Bytes = 0

Tx Success Broadcast Data Packets = 0

Tx Success Broadcast Data Bytes = 0

Tx failed = 0

to_stack, unicast, bcast, rx_failed, rx_retries

Rx Data Packets = 8

Rx Data Bytes = 1313

Rx Unicast Data Packets = 0

Rx Unicast Data Bytes = 0

Qualcomm Confidential – May Contain Trade Secrets 2025-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Rx Multicast Data Packets = 8

Rx Multicast Data Bytes = 1313

Rx retry count = 0

Rx errors = 0

Tx/Rx per access category : Per AC Tx and Rx stats

Tx Data Packets per AC:

Best effort = 0

Background = 0

Video = 0

Voice = 0

Rx Data Packets per AC:

Best effort = 17

Background = 0

Video = 0

Voice = 0

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

3.24.6 iwpriv ath0 fc_peer_stats <mac_addr>: This is per-client DP level statistic.

Average Jitter : Exponential Rolling Average Jitter between Tx HW enqueue to Completion

Average Delay : Exponential Rolling Average delay between Tx HW enqueue to Completion

Total error count : Tx average computation error count

Total Success Count : Tx success count

Total Drop : Tx FW/HW drop count.

Command:

```
iwpriv wifiX enable_statsv3 2
iwpriv athX fc_peer_stats <mac_addr>
```

This statistic display average delay and average jitter per client and per TID. The average compute unit here is in us second unit. All the average computed using exponential rolling average method. The following is the formula used to compute the values:

avg_delay = avg_delay + ((curr_delay - avg_delay) >> AVG_DELAY_WEIGHT_DENOM)

The average weight factor is chosen to be 8 for average delay and 16 for average jitter.

The memory will increase on per client level approximately ~192 bytes

Performance drop can be observed ~10 - 15%. These heavy computation sensitive statistics and hence performance drop is expected. This statistic is mainly used for troubleshooting by customer.

There is a separate command to enable this statistic: iwpriv wifiX enable_statsv3 2

The sample output of iwpriv ath0 fc_peer_stats <mac_addr> command is as follows:

```
Node Tx Average Jitter and Average Delay :
[ 8109.551143] wlan: [15881:F:DP] Node tid = 0
[ 8109.551143] Average Jiiter            : 0 (us)
[ 8109.551143] Average Delay             : 0 (us)
[ 8109.551143] Total Average error count : 0
[ 8109.551143] Total Success Count       : 0
[ 8109.551143] Total Drop                : 0
[ 8109.551143]
[ 8109.563737] wlan: [15881:F:DP] Node tid = 1
[ 8109.563737] Average Jiiter            : 0 (us)
[ 8109.563737] Average Delay             : 0 (us)
[ 8109.563737] Total Average error count : 0
[ 8109.563737] Total Success Count       : 0
[ 8109.563737] Total Drop                : 0
[ 8109.563737]
[ 8109.590211] wlan: [15881:F:DP] Node tid = 2
[ 8109.590211] Average Jiiter            : 0 (us)
[ 8109.590211] Average Delay             : 0 (us)
[ 8109.590211] Total Average error count : 0
[ 8109.590211] Total Success Count       : 0
[ 8109.590211] Total Drop                : 0
[ 8109.590211]
[ 8109.616691] wlan: [15881:F:DP] Node tid = 3
[ 8109.616691] Average Jiiter            : 0 (us)
[ 8109.616691] Average Delay             : 0 (us)
[ 8109.616691] Total Average error count : 0
[ 8109.616691] Total Success Count       : 0
[ 8109.616691] Total Drop                : 0
[ 8109.616691]
[ 8109.643155] wlan: [15881:F:DP] Node tid = 4
[ 8109.643155] Average Jiiter            : 0 (us)
[ 8109.643155] Average Delay             : 0 (us)
[ 8109.643155] Total Average error count : 0
[ 8109.643155] Total Success Count       : 0
[ 8109.643155] Total Drop                : 0
[ 8109.643155]
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
[ 8109.669624] wlan: [15881:F:DP] Node tid = 5
[ 8109.669624] Average Jiiter            : 0 (us)
[ 8109.669624] Average Delay             : 0 (us)
[ 8109.669624] Total Average error count : 0
[ 8109.669624] Total Success Count       : 0
[ 8109.669624] Total Drop                : 0
[ 8109.669624]
[ 8109.696104] wlan: [15881:F:DP] Node tid = 6
[ 8109.696104] Average Jiiter            : 0 (us)
[ 8109.696104] Average Delay             : 0 (us)
[ 8109.696104] Total Average error count : 0
[ 8109.696104] Total Success Count       : 0
[ 8109.696104] Total Drop                : 0
[ 8109.696104]
[ 8109.722583] wlan: [15881:F:DP] Node tid = 7
[ 8109.722583] Average Jiiter            : 0 (us)
[ 8109.722583] Average Delay             : 0 (us)
[ 8109.722583] Total Average error count : 0
[ 8109.722583] Total Success Count       : 0
[ 8109.722583] Total Drop                : 0
```

3.25 Unassociated power consumption improvement parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Regulatory compliance of WLAN APs* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-26 Unassociated power consumption improvement parameters

Parameter	Command	Description	

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

		0	Process 11d beacon
		1	Ignore 11d beacon

3.26 Smart antenna parameters

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 3-27 Smart antenna parameters

Parameter	Command	DA	OL	Description
set_sa_param **get_sa_param**	iwpriv wifidev [*wifi0*|*wifi1*] set_sa_param *dword1 dword2 dword3 dword4* iwpriv wifidev [*wifi0*|*wifi1*] get_sa_param *dword1 dword2 dword3*	Y	Y	Sets and gets Smart Antenna parameters. Each dword attribute is defined as given in this table. At any time, iwpriv wifi0 set_sa_param 0 0 0 0 can be used to list all the paramID that are used as a third argument in set_sa_param. **NOTE** dword4 is not required on the get command.
				dword1: 0xAABBCCDD
				0xAA: param type: 0 = radio param, 1 = node param For radio param, MAC is 00:00:00:00:00:00. For node param, proper MAC address must be specified.
				0xBB: Reserved (should be 00)
				0xCCDD: bytes 5 and 6 of MAC
				dword2: 0xEEFFGGHH
				0xEEFFGGHH: bytes 4, 3, 2, 1 of MAC. For example, if the MAC address is 00:03:7f:48:d8:73 then: CC = 00, DD = 03, EE = 7f, FF = 48, GG = d8, HH = 73.
				dword3: paramID. See definitions and descriptions in Table 3-28.
				dword4: paramValue (required only for set_sa_param)

Table 3-28 dword3 parameters

Parameter name	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.

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

Table 3-28 dword3 parameters (cont.)

Parameter name	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	Good put averaging interval. By default it is 2 seconds.
SMART_ANT_PARAM_ DEFAULT_ANTENNA	13	Radio	Default antenna for Rx, Tx multicast and Tx broadcast. By default it is antenna 0.
SMART_ANT_PARAM_ DEFAULT_TX_ANTENNA	14	Radio	Default Tx antenna for Tx. By default it is antenna 0. Once a new node connects, by default this antenna is used as unicast Tx antenna.
SMART_ANT_PARAM_ TX_ANTENNA	15	Node	Once this command is set, no training will be done for this node and this antenna will be used for all unicast Tx.
SMART_ANT_PARAM_ DBG_LEVEL	16	Radio	It's a 4 bit value used for controlling the prints. By default it is log level 1. Bit 1 controls log level 1, bit 2 controls log level 2, bit 3 controls log level 3 and bit 4 controls log level 4.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-28 dword3 parameters (cont.)

Parameter name	Param ID	Node or Radio param	Description
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 bandwidth packets to detect bandwidth change. By default it is 5.
SMART_ANT_PARAM_ GOODPUT_IGNORE_ INTERVAL	19	Radio	By default good put ignoring interval is 1 second.
SMART_ANT_PARAM_ MIN_PKT_TH_BW20	20	Radio	Minimum number of packets in 20 MHz BW to indicate active BW. By default it is 20.
SMART_ANT_PARAM_ MIN_PKT_TH_BW40	21	Radio	Minimum number of packets in 40 MHz BW to indicate active BW. By default it is 10.
SMART_ANT_PARAM_ MIN_PKT_TH_BW80	22	Radio	Minimum number of packets in 80 MHz BW to indicate active BW. By default it is 5.
SMART_ANT_PARAM_ DEBUG_INFO	23	Node	Displays Last training time, Periodic triggers and performance triggers for specific node.
SMART_ANT_PARAM_ MAX_TRAIN_PPDU	24	Radio	Maximum number of train PPDUs in train command. By default it is 50
SMART_ANT_PARAM_ PERF_HYSTERESIS	25	Radio	Hysteresis for performance based trigger By default it is 3.
SMART_ANT_PARAM_ BCN_ANTENNA	27	Radio	To Configure Beacon Antenna
SMART_ANT_PARAM_ TRAIN_RATE_TESTMODE	28	Node	Train rate code in test mode
SMART_ANT_PARAM_ TRAIN_ANTENNA_ TESTMODE	29	Node	Train antenna in test mode
SMART_ANT_PARAM_ TRAIN_PACKETS_ TETSMODE	30	Node	Number of train packets in test mode
SMART_ANT_PARAM_ TRAIN_START_ TETSMODE	31	Node	Start Training in test mode

3.27 WDS parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *WDS functionality* section of the *Repeater access point functions* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-29 WDS parameters

Parameter	Command	DA	OL	Description

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

athrewind
get_athrewind
	iwpriv ath*N* athrewind *{0-3}*	Y	N	Enables (1) or disables (0) enhanced independent repeater mode.
				If this option is enabled, the STA VAP will scan for the Root AP in all the available channels and connect to it. The AP VAP will start and continue to transmit beacons independently of the STA VAP connection status.
				The default value is 0. The get parameter returns the current value.
				`#iwpriv ath0 athnewind 1`
				`#iwpriv ath0 get_athnewind`
				`ath0 get_athnewind:1`

wds
get_wds
	iwpriv ath*N* wds *{1|0}*	Y	Y	Enables (1) or disables (0) 4-address frame format for this VAP. Used for WDS configurations (see “Wi-Fi Distribution System (WDS)” in the *AP Driver User’s Guide* for details). The default value is 0. The get parameter returns the current value.
				`#iwpriv ath0 wds 1`
				`#iwpriv ath0 get_wds`
				`ath0 get_wds:1`

Qualcomm Confidential - May Contain Trade Secrets 2021-05-13 09:53:00 pan.yafeng@qualcomm.com

3.28 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).

For more information on usage guidelines and the functionality regarding these parameters, see the *WMM flow control and buffer management* section of the *Voice and multimedia features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-30 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		

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-30 Access categories and modes (cont.)

Value	Symbol	Description
0	AP	AP mode: Update the AP WMM table
1	STA	STA mode: Update the STA WMM tables

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

Table 3-31 WMM parameters

Parameter	Command	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 3-30. *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.

Qualcomm Confidential – Qualcomm Technologies, Inc. and/or its affiliated companies – May Contain Trade Secrets 2021-05-13 03:27:19 PDT pan.yafeng@zte.com.cn

Table 3-31 WMM parameters

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

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

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 15:44 pan.yafeng@qualcomm.com

Table 3-31 WMM parameters

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

wmm
get_wmm
	iwpriv ath*N* wmm *{1|0}*
	iwpriv ath*N* get_wmm
		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`
				
				WMM cannot be disabled for HT/VHT/HE modes.

3.29 256QAM rate support parameters

To achieve higher throughput in 2.4 GHz operating modes, the 256 QAM rate-support feature enables 256-QAM rates. According to the 802.11n protocol, devices operating in 2.4 GHz band uses 64-QAM modulation to support maximum rates. With this feature, 256-QAM modulation technique would be enabled to support higher maximum rates.

NOTE If a NAWDS link breaks, it is necessary to add the peer again, although the peer entry is already available in the NAWDS table. For example, if the iwpriv ath0 vht_11ng 1 command is issued to enable 256QAM, the NAWDS link breaks. In this scenario, the entry must be added again to the table for NAWDS to work.

For more information on usage guidelines and the functionality regarding these parameters, see the *256 QAM rate support in 2.4 GHz operation* section of the *Rate control features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-32 256QAM parameters

Parameter	Command	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 1. 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`

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

3.30 Hy-Fi – WMM DSCP prioritization parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Hy-Fi WLAN* section of the *WLAN AP modes* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-33 Hy-Fi parameters

Parameter	Command	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`
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`

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 multicast 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 multicast packets. iwpriv option sHmmcDscpOvrid should be set to 1 for this command to work
				`#iwpriv wifix sHmmcDscpTidMap <tid>`
				`#iwpriv wifix gHmmcDscpTidMap`
				`gHmmcDscpTidMap:<tid>`

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

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

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-33 Hy-Fi parameters

Parameter	Command	DA	OL	Description

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

set_dscp_ovride
get_dscp_ovride
	iwpriv athX set_dscp_ovride *<mapping table ID between 1 to 64>*	Y	Y	To enable/disable dscp override feature. Packets with specific dscp value set can be mapped to a specific TID through this feature on per vap basis. For each VAP with a different DSCP-TID mapping, a new map id must be provided. Therefore, specify a unique tableID for each VAP.
				<map_id>:0,1 are mapped to HW tables
				<map_id>:2-16 are mapped to SW mapping tables
				`#iwpriv ath0 set_dscp_ovride 1`
				`#iwpriv ath0 get_dscp_ovride get_dscp_ovride:1`

s_dscp_tid_map g_dscp_tid_map	iwpriv athX s_dscp_tid_map <dscp> (hex_val)>
	<tid (0-8)>
		Y	Y	To configure a specific tid mapping table for a VAP. iwpriv option vap based set_dscp_ovride should be set to 1.
				Enter the iwpriv athX s_dscp_tid_map <map_id> <dscp> <tid> command to configure a specific dscp-tid map value for a specific table_id.
				`#iwpriv ath0 s_dscp_tid_map 0xe0 1`
				`#iwpriv ath0 g_dscp_tid_map 0xe0 g_dscp_tid_map:1`

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

3.31 Channel loading/Channel hopping parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *UMAC auto channel selection (scanning)* section of the *Advanced channel selection capabilities* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-34 Channel loading/Channel hopping parameters

Parameter	Command	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`
				This command is applicable only for ACS scan reporting, and not for ACS scan and channel selection. To invoke ACS report, use the command iwpriv athX acsreport 1.

acsmaxdwell
get_acsmaxdwell
	iwpriv ath*N* acsmaxdwell *value_in_ms*	N	N	Maximum time in milliseconds than can be spent on a channel. Default value is 300 msec.
				The value to be set should be greater than or equal to acsmindwell. So check the value of acsmindwell and choose a value accordingly, else the command returns error.
				`#iwpriv ath0 acsmaxdwell 100`
				`#iwpriv ath0 get_acsmaxwell`
				`ath0 get_acsmindwell:100`
				This command is applicable only for ACS scan reporting, and not for ACS scan and channel selection. To invoke ACS report, use the command iwpriv athX acsreport 1.

acsreport	iwpriv ath*N* acsreport *value*	Y	Y	Enable (1) or disable (0) channel loading. If this command is issued during DFS CAC, a fresh scan is not triggered and the previously generated report is flushed.
				`#iwpriv ath0 acsreport 1|0`
				The following is the format in which ACS report will be displayed:
				`# iwpriv ath2 acsreport 1`
				`[ 1686.266895] wlan: [8645:I:ANY] ieee80211_autoselect_infra_bss_channel: 3920: ACS started: vap:0xffffffc02fac0000 root@OpenWrt:/# [ 1693.127287] wlan: [6278:I:ANY] ieee80211_acs_scan_ evhandler: 3657: lock held duration: 0(ms)`

get_acsreport	iwpriv ath*N* get_ acsreport	Y	Y	This command is used to check whether ACS scan report is active or not.

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`

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-34 Channel loading/Channel hopping parameters (cont.)

Parameter	Command	DA	OL	Description

ch_cntwn_dur
get_
ch_cntwn_dur
	iwpriv ath*N* ch_cntwn_
	dur *{value in seconds}*
		Y	N	Set/get counter window duration for channel hopping
				`#iwpriv ath0 ch_cntwn_dur 60`
				`#iwpriv ath0 g_ch_cntwn_dur`
				`g_ch_cntwn_dur:60`

ch_noise_th
get_ch_noise_th
	iwpriv athN ch_noise_th
	{value}
		Y	N	Set/get noise threshold in dB
				`iwpriv ath0 ch_noise_th -90`
				`#iwpriv ath0 get_ch_noise_th`
				`get_ch_noise_th:-90`

ch_cnt_th
get_ch_cnt_th
	iwpriv athN ch_cnt_th
	{value}
		Y	N	Set/get counter threshold
				`iwpriv ath0 ch_cnt_th 60`
				`iwpriv ath0 get_ ch_cnt_th`
				`get_ ch_cnt_th:60`

use_custom_
chan
	iwpriv athx use_
	custom_chan 1 or 0
		Y	N	Set/get custom channel list usage
				`iwpriv athx use_custom_chan 1 or 0`
				`iwpriv athx get_custom_chan 1 or 0`

3.32 802.11k parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Voice Enterprise* section in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-35 802.11k Parameters

Parameter	Command	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`

3.33 Block channel list parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Channel blocking* section of the *Core channel selection capabilities* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:27:44 PDT Pan.xufeng@zte.com.cn

Table 3-36 Block Channel List Parameters

Parameter	Com	DA	OL	Descriptio

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

3.34 Aggregate size scaling parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Video over wireless (offload)* section of the *Video and multimedia features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-37 Aggregate Size Parameters

Parameter	Command	DA	OL	Description

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

3.35 Target recovery parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Restore WLAN configuration after a firmware assert for IPQ807x/IPQ807xA/IPQ817x* section of the *WLAN AP driver operations* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-38 Target recovery parameters

Parameter	Command	DA	OL	Description

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

Table 3-38 Target recovery parameters

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.
set_fw_ hang <value>	**iwpriv wifiX set_fw_ hang <value>**	Y	Y	This parameter is used to trigger a target assertion after a crash of the FW is as follows Value - Time in msecs after which the Target Assert is triggered. For IPQ807x/IPQ60xx platforms, because the FW is the same for both 5G and 2.4G radio, FW assert will recovery both the radios.

3.36 Uncategorized protocol layer parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Crypto service module and data structures* and *Retrieve Wi-Fi medium utilization statistics* sections in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-39 Uncategorized protocol layer parameters

Parameter	Command	Description	

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

		0	Disable
		1	Enable
setparam getparam	iwpriv athN setparam	For sub-ioctl handlers, and usually not used directly. For example, `iwpriv ath0 ampdu 0` should be equivalent to `iwpriv ath0 setparam 73 0`.	
chutil_enab val	iwpriv athX chutil_ enab val	Activate/Deactivate Channel Utilization measurement. If val is 1, then it is activated. If val is 0, then it is deactivated.	
get_chutil_enab	iwpriv athX get_ chutil_enab	Gets status on whether Channel Utilization measurement is activated or not. If 1 is returned, then it is activated, else if 0 is returned, it is deactivated.	
get_chutil	iwpriv athX get_chutil	Not required if using apstats. This command retrieves the channel utilization value in percentage (1-100).	

3.37 2.4 GHz VHT 256-QAM Broadcom interoperability parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *256 QAM rate support in 2.4 GHz operation* section of the *Rate control features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-40 lists the parameters for 2.4 GHz VHT 256-QAM Broadcom interoperability support.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-40 2.4 GHz VHT 256-QAM Broadcom interoperability support

Parameter	Command	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`

3.38 QWRAP debug parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *ProxySTA functionality* and *QWRAP support for tri-radio standalone repeater* sections of the *Repeater access point functions* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-41 lists the QWRAP debug parameters.

Table 3-41 QWRAP debug

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

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

3.39 ATF parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Airtime fairness* section of the *Memory and bandwidth optimization features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-42 lists the Airtime Fairness (ATF) parameters.

Table 3-42 ATF parameters

Parameter	Command	DA	OL	Description

commitatf	*iwpriv athN* commitatf {1/0}	Y	Y	This is commit command, it must be issued once user finishes any setting for ssid/sta percentage configuration by wlanconfig tool.
				The example is the following.
				`#iwpriv ath0 commitatf 1`
				`/*setting effective*/`
				`#iwpriv ath0 commitatf 0`
				`/*setting ineffective*/`
				Changes to any configuration for ssid/sta reflects after 5 secs of the issuance of commit command.
				This behavior indicates that a user must wait for 5 secs after `commitatf` command to issue `showatftable` command by wlanconfig tool. Otherwise it shows previous/default configured data.

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

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

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

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

Table 3-42 ATF parameters (cont.)

Parameter	Command	DA	OL	Description

gatfgrouppolicy	iwpriv wifiN gatfgrouppolicy	Y	Y	The command is used to checking the inter group scheduling (Across groups)
				Example command:
				`# iwpriv wifi0 gatfgrouppolicy`
				`wifi0gatfgrouppolicy:1`
				NOTE 1 denotes strict scheduling across groups and 0 denotes fair scheduling across groups. This is the same for both direct attach and partial offload architecture.

atf_sched_dur	iwpriv wifi*N* atf_ sched_dur ac {0-3} dur {ac is the access category and dur is the token}	N	Y	This command is used to set the number of tokens to be allocated for a particular access category Example Command: `iwpriv wifi1 atf_sched_dur 2 5` The first parameter is the Access Category which should be between 0 and 3 0 – Best Effort 1 – Background – Video – Voice The second parameter should be the number of tokens which should be a positive value.

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

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

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

g_atf_shr_buf	iwpriv ath*N* g_atf_ shr_ buf	Y	N	This command is used to query whether Tx Buffer sharing between clients is enabled or not.
				Example Command:
				`#iwpriv ath0 g_atf_shr_buf ath0 g_atf_shr_`
				`buf:1`

atfmaxclient	iwpriv ath*N* atfmaxclient	Y	N	This command is used to enable maxclient support on direct attach architecture. This feature is disabled by default `#iwpriv ath0 atfmaxclient`

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-42 ATF parameters (cont.)

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

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

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

atf_log	iwpriv wifiX atf_log <0,1>	Y	Y	Enable or disable logging of airtime used as a percentage of total airtime allotted across all clients, at every ATF interval.
				Example Command:
				`#iwpriv wifi0 atf_log 1`
				`/* enable ATF logging */`

get_atf_log	iwpriv wifi*N get_ atf_log*	Y	Y	This command is used to query whether ATF logging is enabled or not. Example Command: `#iwpriv wifi0 get_atf_log`
enable_atf_stats 1 atf_stats_ timeout <10 - 50>	cfg80211tool athX enable_atf_stats 1 cfg80211tool athX atf_stats_timeout <10 - 50>			To enable ATF stats feature. Before enabling stats, cfg80211tool athX commitatf 1 command must be entered to enable ATF. Get the status of ATF stats feature.
atf_stats_ timeout <10 - 50> g_atf_stats_ timeout	cfg80211tool athX atf_stats_timeout <10 - 50> cfg80211tool athX g_atf_stats_ timeout			Set ATF stats timeout period. The default value is 30 seconds. The minimum value is 10 seconds and maximum value is 50 seconds. Get ATF stats timeout period. If enable_atf_stats is not issued at least once default it will show 0. After the stats feature is enabled, it will show default value if user has not modified to any other valid value.

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

A combination of Strict-queue scheduling within group and Fair-queue scheduling across groups is invalid. For example:

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

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

The above configuration is invalid.

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

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

3.40 Wake on wireless – AP assist parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Wake on Wireless AP assist* section of the *Beacons and frames transmission* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-43 lists the wake-on-wireless AP assist parameters.

Table 3-43 Wake-on-wireless AP assist parameters

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

3.41 Dynamic Frequency Selection (DFS) parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Dynamic frequency selection (DFS)* and *Zero CAC wait for DFS channels* sections of the *DFS parameters* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-44 DFS parameters

Parameter	Command	DA	OL	Description

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

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

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

Table 3-45 Repeater DFS channel switch options

Parameter	Command	DA	OL	Description

CSwOpts	iwpriv wifiN CSwOpts {1/2/4/8/10/20/F/0}	Y	Y	This command is used to set the channel switch options in Repeater AP. Each bit in the parameter value represents one option. Below are the options:
				■ IEEE80211_CSH_OPT_NONDFS_RANDOM 0x00000001. Enable Random selection of channel from Non-DFS channel list when local radar detected. If a Non-DFS channel is not found in the list it chooses a DFS channel.
				■ IEEE80211_CSH_OPT_IGNORE_CSA_DFS 0x00000002. When a CSA comes from Root AP and the channel in the CSA is DFS, choose a new random Non-DFS channel and use this new channel in the CSA announcement in Repeater AP's BSS.
				■ IEEE80211_CSH_OPT_CAC_APUP_BYSTA 0x00000004. Do a CAC before joining Root AP if the Root AP is in DFS channel.
				■ IEEE80211_CSH_OPT_CSA_APUP_BYSTA 0x00000008. Announce a CSA in Repeater BSS when the Repeater connects to the Root AP and the Root AP's channel is not same as Repeater AP's channel.
				■ IEEE80211_CSH_OPT_RCSA_TO_UPLINK 0x00000010. Enable sending RCSA to uplink/parent/Root AP when a repeater AP detects RADAR.
				■ IEEE80211_CSH_OPT_PROCESS_RCSA 0x00000020. Enable processing of RCSA from any downlink/client/RE/Repeater AP.
				Both IEEE80211_CSH_OPT_RCSA_TO_UPLINK and IEEE80211_CSH_OPT_PROCESS_RCSA bits should be set for the Repeater and Root to work together and to change the channel after RADAR detection. The bit IEEE80211_CSH_OPT_PROCESS_RCSA can be disabled to simulate the situation that Root/uplink does not respond to the RCSA.
				Usage:
				Enable option 1
				`#iwpriv wifi0 CSwOpts 0x1`
				Enable option 2
				`#iwpriv wifi0 CSwOpts 0x2`
				Enable option 5 and 6
				`#iwpriv wifi0 CSwOpts 0x30`
				■ IEEE80211_CSH_OPT_APRIORI_NEXT_CHANNEL 0x00000040.
				Enable apriori random next channel information propagation in the Root and its entire BSS (which includes cascaded Repeaters)
				To enable Apriori random channel information propagation, below flags need to be set -
				`"IEEE80211_CSH_OPT_RCSA_TO_UPLINK 0x00000010\n"`
				`"IEEE80211_CSH_OPT_PROCESS_RCSA 0x00000020\n"`
				`"IEEE80211_CSH_OPT_APRIORI_NEXT_CHANNEL 0x00000040\n"`
				Enable option 5,6 and 7
				`#iwpriv wifi0 CSwOpts 0x70`

Table 3-45 Repeater DFS channel switch options

Parameter	Command	DA	OL	Description
get_CSwOpts	iwpriv wifiN get_ CSwOpts	Y	Y	This command is used to query the channel switch options enabled in Repeater AP. `#iwpriv wifi0 get_CSwOpts`
get_ dfsdomain	iwpriv athiN get_ dfsdomain	Y	Y	This command is used to get the DFS domain.

Table 3-46 Zero CAC DFS options

Parameter	Command	D	O	Description
preCACEn	iwpriv wifiN preCACEn {1/0}	N	Y	This command enables/disables the pre-CAC mode (zero-CAC-DFS). The zero-CAC-DFS in a RADIO that has VHT80_80 mode support. Right now it is enabled only for QCA9984 where the primary and secondary VHT80s can be programmed independently with two different HT80 channels. Usage: `#iwpriv wifiX preCACEn 1`
get_preCACEn	iwpriv wifiN get_ preCACEn	N	Y	This command is used to query the pre-cac option enabled in Root AP. Usage: `#iwpriv wifiX get_preCACEn`
pCACTimeout	iwpriv wifiN pCACTimeout <timeout value>	N	Y	Override the pre-cac timeout value. Usage: `#iwpriv wifiX pCACTimeout 60`
get_ pCACTimeout	iwpriv wifiN get_ pCACTimeout	N	Y	Get the pre-cac timeout value. Usage: `#iwpriv wifiX get_pCACTimeout`

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-46 Zero CAC DFS options

interCACChan <non_dfs_chan_num> **getInterCACChan)**	iwpriv wifiX interCACChan <non_dfs_chan_num> iwpriv wifiX getInterCACChan)	N	Y	Perform preCAC on configured channel and switch to the channel if free; otherwise, select a new random channel. Perform preCAC on remaining predefined channels and move them to clear list. This iwpriv command accepts an argument as user-selected channel number/intermediate channel number. The optional argument also ensures backward compatibility for existing preCAC users, if any. To set and get intermediate CAC channels, enter the following commands: iwpriv wifiX interCACChan <non_dfs_chan_num> iwpriv wifiX getInterCACChan)
acs_pcaconly <1 | 0>	cfg80211tool wifiX acs_pcaconly <1 | 0>	N	Y	Set the ACS Pre-CAC support: cfg80211tool wifiX acs_pcaconly 1 Set the country code, such as cfg80211tool wifiX setCountry DK, and set the ETSI Pre-CAC feature using cfg80211tool wifiX preCACEn 1 before setting ACS Pre-CAC support. These configurations can be added as radio-level UCI commands as well during configuration. A user can also set the pre-CAC timeout value for each channel in seconds as follows: cfg80211tool wifiX pCACTimeout <seconds>

3.42 NSS offload parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *NSS Wi-Fi offload on IPQ806x* and *NSS Wi-Fi offload on IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx, IPQ807x/IPQ807xA/IPQ817x, and QCN90xx/QCN602x + IPQ807xA/IPQ60xx/IPQ50xx* sections of the *Networking offload and acceleration of link and multilink features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-47 NSS Offload - Tx Flow Control buffer pool size

Parameter	Command	DA	OL	Description

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

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-47 NSS Offload - Tx Flow Control buffer pool size

Parameter	Command	DA	OL	Description
fc_peer_ stats <mac_ address>	iwpriv wifi1 fc_peer_ stats <mac_ address>	N	Y	Displays statistical information associated with flow control for a peer with the specified MAC address. These counters include the following attributes: ■ Association ID ■ byte_cnt as a decimal value ■ high_watermark for each queue (the largest queue length a queue has seen) ■ Per-client Tx_Desc_fail

3.43 Disable selected MCS for a given SSID

For more information on usage guidelines and the functionality regarding these parameters, see the *Disable selected MCS for a given SSID* section of the *Rate control features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Parameter	Command	DA	OL	Description
disable11nmcs	iwpriv athX disable11nmcs 0xFFFFFFF0	Y	Y	Every bit in this command represents a MCS rate. LSB bit 0 represents and MSB bit 31 represents MCS 31. If a particular bit is set, then that MCS rate will be disabled for that vap.
g_disable11nmcs	iwpriv athX g_ disable11nmcs	Y	Y	iwpirv athX g_disable11nmcs will display the configured value on the VAP.
conf_11acmcs	iwpriv athX conf_ 11acmcs 0xFFFFFFF5	N	Y	The LSB 16 bits passed to conf_11acmcs represents the 16 bits in Rx MCS Map and Tx MCS Map field of VHT Capabilities IE. The MSB 16 bits should be 0xFFFF always. See the figure that follows the table on how to interpret the LSB bits of Rx MCS Map and Tx MCS Map fields of VHT Capabilities IE. Only those bits are that valid with respect to the number of Tx and Rx spatial streams will be considered. For platform that support MCS 10 and 11 in VHT mode, it is necessary to enter additional command of 'cfg80211tool athX vht_mcs_10_11_supp 0' to disable MCS 10 and 11, apart from restricting the MCS/NSS mapping commands using conf_11acmcs.
g_conf_11acmcs	iwpriv athX g_conf_ 11acmcs	N	Y	iwpirv athX g_conf_11acmcs will display the configured value on the VAP.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

	B0 B1	B2 B3	B4 B5	B6 B7	B8 B9	B10 B11	B12 B13	B14 B15
	Max VHT-MCS For 1 SS	Max VHT-MCS For 2 SS	Max VHT-MCS For 3 SS	Max VHT-MCS For 4 SS	Max VHT-MCS For 5 SS	Max VHT-MCS For 6 SS	Max VHT-MCS For 7 SS	Max VHT-MCS For 8 SS
Bits:	2	2	2	2	2	2	2	2

iwpriv athX conf_11acmcs and g_conf_11acmcs configuration examples

```
iwpriv athX conf_11acmcs 0xFFFFFFC0
```

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

```
iwpriv athX conf_11acmcs 0xFFFFFFF5
```

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

```
iwpriv athX conf_11acmcs 0xFFFFFFFA
```

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

```
iwpriv athX conf_11acmcs 0xFFFFAAAA
```

This command enables VHTMCS 0 - 9 in Spatial Streams 1-8

```
iwpriv athX   g_conf_11acmcs
```

This command displays the configured value on the VAP.

iwpirv athX disable11nmcs and g_disable11nmcs configuration examples

```
iwpriv athX disable11nmcs 0xFFFFFFF0
```

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

```
iwpriv athX disable11nmcs 0xFFFF0FFF
```

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

```
iwpirv athX   g_disable11nmcs
```

This command displays the configured value on the VAP.

3.44 Disable selected legacy rates for an SSID

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

For more information on usage guidelines and the functionality regarding these parameters, see the *Disable selected legacy rates for an SSID* section of the *Rate control features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

For 802.11 B

B3	B2	B1	B0
11	5.5	2	1

For 802.11G / 802.11NG

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

For 802.11A/802.11NA/802.11AC

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

List of legacy rates per desired modes

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

iwpriv command

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

iwpriv athX g_dis_legacy : Displays the configured value on the VAP

Examples

- iwpriv athX dis_legacy 0x9

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

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

- iwpriv athX dis_legacy 0x1f

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

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

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

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

- iwpriv athx dis_legacy 0x0

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

NOTE Do not disable all the basic supported rates.

Example:

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

3.45 Revised signaling for 160/80+80 MHz parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *160/80+80 MHz operation* section of the *IEEE 802.11 features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Parameter	Command		Description

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

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

3.46 AP diagnostics for carrier parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *AP diagnostics for carrier* section of the *Using statistics for debugging* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

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

3.47 Reject clients with low SNR parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Reject clients with low SNR* section of the *QoS operations* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Parameter	Command	DA	OL	Description
set_min_snr_en	iwpriv wifiN set_min_snr_en {1/0}	Y	Y	Reject clients with low SNR facility can be enabled/disabled by using this command. Usage: `#iwpriv wifi0 set_min_snr_en 1`
get_min_snr_en	iwpriv wifiN get_min_snr_en	Y	Y	This command is used to query whether reject clients with low SNR facility is enabled. Usage: `#iwpriv wifi0 get_min_snr_en`

Parameter	Command	DA	OL	Description
set_min_snr	iwpriv wifiN `set_min_snr` {SNR value in dB}	Y	Y	This command is used to set the client reject snr threshold. Usage: `#iwpriv wifi0 set_min_snr 15`
get_min_snr	iwpriv wifiN get_min_snr	Y	Y	This command is used to get the configured client reject snr threshold Usage: `#iwpriv wifi0 get_min_snr`

3.48 VLAN tagging parameters

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

For more information on usage guidelines and the functionality regarding these parameters, see the *VLAN support* section of the *WLAN AP modes* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Parameter	Command	DA	OL	Description

vlan_tag	iwpriv athN vlan_tag {1/0}	Y	Y	This command is used to enable/disable the VLAN Tag to be sent out over Wi-Fi
				Usage:
				Following enables VLAN Tag to be sent out over WIFI for the VLAN interface created over WIFI (i.e. for ath0.1, ath0.2 etc.)
				`#iwpriv athN vlan_tag 1`
				Following disables VLAN Tag to be sent out over WIFI even for the VLAN interface created over Wi-Fi.
				`#iwpriv athN vlan_tag 0`
				To enable VLAN tagging:
				`#iwpriv athN vlan_tag 1`
				To disable VLAN tagging:
				`#iwpriv athN vlan_tag 0`

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

3.49 Mesh support parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *NoDS frame support for Mesh feature* section of the *Coexistence of WLAN for mesh* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

The following commands provide control to configure mesh:

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Parameter	Command	D	O	Description
addlocalpeer	iwpriv athN addlocalpeer 0x<peer flags><byte5><byte4> 0x<byte3><byte2><byte 1><byte0>	N	Y	To add peer: `#iwpriv athN` *0x12040034 0x56788797* *mac addr 00:34:56:78:87:97 and peer flags 0x1204* Each bit in peer flags indicate below capability. Dual Stream support BIT(0) i.e 0x1 Three Stream support BIT(1) i.e 0x02 Four Stream support BIT(2) i.e 0x04 Reserved bits BIT(3) to BIT(7) 0xF8 Peer support HT20 BIT(8) i.e 0x0100 Peer support HT20 & HT40 BIT(9) i.e 0x0200 Peer support 11ACVHT20 BIT(10) i.e 0x0400 Peer support 11ACVHT40 BIT(11) i.e 0x0800 Peer support 11ACVHT80 BIT(12) i.e 0x1000 Peer support 11ACVHT80_80 BIT(13) i.e 0x2000 Peer support 11ACVHT160 BIT(14) i.e 0x4000 Reserved bit BIT(15) ie. 0x8000 Support will be added in a subsequent release for 5-8 streams (5 GHz <= 80 MHz), and 3-4 streams (5 GHz 160 MHz).
allowdata	iwpriv athN allowdata 0x<byte5><byte4> 0x<byte3><byte2><byte 1><byte0>	N	Y	Authorize key for the local peer. Usage: `# iwpriv athN allow data 0x8cfd 0xf0021f9c`
rmode_pktsim	iwpriv athN rmode_ pktsim	N	Y	Allow packets in raw mode. Usage `# iwpriv athN rmode_pktsim 1`
meshdbg	iwpriv athN meshdbg	N	Y	BIT usage: To add the meta header in Ethernet format BIT(0) i.e 0x1 Rx debug info BIT(1) i.e 0x2 To set the meta header BIT(16-31) Usage `iwpriv athN meshdbg 1`
conf_meshtx	iwpriv athN conf_meshtx	N	Y	Prevent the mesh vap from sending disassociation frame Usage: `# iwpriv conf_meshtx 1`

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Parameter	Command	D	O	Description
mesh_rxfilter	iwpriv athN mesh_rxfilter	N	Y	To apply Rx filters for mesh Below is the bitwise info: Drop DS Frames BIT(0) i.e 0x1 Drop ToDS Frames BIT(1) i.e0x2 Drop NoDS Frames BIT(2) i.e 0x4 Drop RA Frames BIT(3) i.e0x8 Drop TA Frames BIT(4) i.e 0x10

data_txstats
g_data_txstats
	iwpriv wifiN data_txstats
	iwpriv wifiN g_data_ txstats
		N	Y	To enable the Tx completion handler statistics
				The get command displays whether the Tx completion handler statistics are enabled.
				OI stats have to be enabled before enabling these stats
				Usage:
				`iwpriv wifiN enable_ol_stats 1`
				`iwpriv wifiN data_txstats 1`

3.50 DBDC repeater support parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *DBDC repeater functionality* section of the *Repeater access point functions* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

The following commands provide control to configure DBDC repeater:

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

Parameter	Command	DA	OL	Description

alwaysprimary
galwaysprimary
	iwpriv wifiN alwaysprimary {1|0}	Y	Y	Enable (1) or disable (0) alwaysprimary option on DBDC repeater mode.
				If this option is enabled, DBDC Repeater sends all of its client traffic to DBDC RootAP through primary radio STA Vap.
				The default value is 0. The get parameter returns the current value.
				alwaysprimary flag is a system flag,
				we can set this flag through iwpriv command on any radio.
				
				`#iwpriv wifi0 alwaysprimary 1`
				`#iwpriv wifi0 galwaysprimary`
				`wifi0 galwaysprimary:1`

primaryradio
getprimaryradio
	iwpriv wifiN primaryradio {1}	Y	Y	Set primary radio as 1 for anyone radio on DBDC repeater mode.
				If alwaysprimary option is unset(0) and if primaryradio option is set(1) for anyone radio,
				DBDC Repeater sends wireless client traffic through corresponding radio's STA VAP and Ethernet client traffic through primary radio's STA VAP.
				If alwaysprimary option is set(1) and if primaryradio option is set(1) for anyone radio,
				DBDC Repeater sends all clients traffic through primary radio's STA VAP.
				By default, primaryradio option is set as 1 for wifi0
				The get parameter returns the current value.
				
				`#iwpriv wifi0 primaryradio 1`
				`#iwpriv wifi0 getprimaryradio`
				`wifi0 getprimaryradio:1`

dbdc_enable
get_dbdc_
enable
	iwpriv wifiN dbdc_enable {1|0}	Y	Y	Enable (1) or disable (0) dbdc_enable on DBDC repeater mode.
				If this option is enabled, it enables DBDC repeater processing on driver.
				dbdc_enable flag is a system flag,
				we can set this flag through iwpriv command on any radio.
				The default value is 1. The get parameter returns the current value.
				
				`#iwpriv wifi0 dbdc_enable 1`
				`#iwpriv wifi0 get_dbdc_enable wifi0`
				`get_dbdc_enable:1`

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Parameter	Command	DA	OL	Description

delay_stavapup
g_
delay_stavapup
	iwpriv wifiN
	delay_
	stavapup
	{1|0}
		Y	Y	Enable (1) or disable (0) delay_stavapup on DBDC repeater mode. If this option is enabled, it delays secondary radio's STA VAP connection with RootAP during first scan. This delay avoids looping of multicast packets when Hy-Fi bridge takes time to attach and dbdc_enable flag is disabled.
				delay_stavapup flag is a system flag,
				we can set this flag through iwpriv command on any radio.
				The default value is 0. The get parameter returns the current value.
				
				`#iwpriv wifi0 delay_stavapup 1`
				`#iwpriv wifi0 g_delay_stavapup`
				`wifi0 g_delay_stavapup:1`

client_mcast
getclient_mcast
	iwpriv wifiN
	client_mcast
	{1|0}
		Y	Y	Enable (1) or disable (0) client_mcast on DBDC repeater mode. If this option is enabled, DBDC repeater sends its connected clients MCAST traffic on corresponding STA VAP, even when DBDC Repeater is connected with two different root APs.
				client_mcast flag is a system flag,
				we can set this flag through iwpriv command on any radio.
				
				The default value is 0. The get parameter returns the current value.
				
				`#iwpriv wifi0 client_mcast 1`
				`#iwpriv wifi0 getclient_mcast wifi0`
				`getclient_mcast:1`

- If the platform is tri-radio and has NSS Wi-Fi offload support, then for TBTC repeater feature to work, NSS Wi-Fi offload has to be disabled through UCI command
- User need to provide the below user level configuration to disable NSS wifi offload mode.

```
uci set wireless.qcawifi=qcawifi
uci set wireless.qcawifi.nss_wifi_olcfg=0
uci commit
```

- By default, wifi0 will be set as primary radio. On tri-radio board, if wifi0 is disabled and if wif1 and wifi2 are configured in DBDC repeater mode, then user needs to explicitly set wifi1 or wifi2 as primary radio.
- Dynamically changing `alwaysprimary` configuration during runtime is not supported. The `alwaysprimary` configuration has to be set only through UCI.

3.51 TR-069 parameters

Table 3-48 shows the list of TR-069 parameters.

Table 3-48 TR-069 commands

Parameter	Command	DA	OL	Description
g_phyofdmerr	iwpriv wifiN g_ phyofdmerr	Y	Y	Command to retrieve OFDM Phy error count from the radio. Usage: `#iwpriv wifi0 g_phyofdmerr`
g_fcserr	iwpriv wifiN g_fcserr	Y	Y	Command to retrieve FCS error count from the radio. Usage: `#iwpriv wifi1 g_fcserr`
g_chanutil	iwpriv wifiN g_chanutil	Y	Y	Command to retrieve channel utilization from the radio. Usage: `#iwpriv wifi1 g_chanutil`

3.52 Rx monitor filter parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Special VAP mode for third radio* section of the *WLAN AP modes* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-49 List of monitor filters

Parameter	Com-mand	DA	OL	Description
set_monrxfilter get_monrxfilter	iwpriv athX set_ monrxfilter iwpriv athX get_ monrxfilter	N	Y	Used to set various filters in monitor/ap_monitor mode BIT(0) Enable/disable all filters (1/0), that is, 0x1 BIT(1) Enable OSIF MAC address-based filtering, that is, e0x2 MAC address needs to be added to the monitor mode before entering the preceding iwpriv command BIT(2) Drop HTT unicast data frames, that is, 0x04 BIT(3) Drop HTT multicast data frames, that is, 0x08 BIT(4) Drop HTT control/management frames, that is, 0x10 0x1f-last filter; value should not be greater than 0x1f. The get command retrieves the configured value.

3.53 Bluetooth coexistence parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Bluetooth and WLAN coexistence* section of the *Coexistence of WLAN for mesh* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-50 Bluetooth coexistence parameters

Parameter	Command	DA	OL	Description

becoex_enable	iwpriv wifiX btcoex_enable {1|0}	N	Y	Enable/Disable BT Coex support.
				`# iwpriv wifi0 btcoex_enable 1`
				`# iwpriv wifi0 btcoex_enable 0`

g_btcoex_enable	iwpriv wifiX g_ btcoex_enable	N	Y	Get current status of BT Coex support `# iwpriv wifi0 g_btcoex_enable`
btcoex_wl_pri	iwpriv wifiX btcoex_wl_pri {bitmap}	N	Y	Set priority frame types of WLAN traffic using the bit fields below: BIT0: BE BIT1: BK BIT2: VI BIT3: VO BIT4: Beacon BIT5: Management Frames `# iwpriv wifi0 btcoex_wl_pri 0x34`
g_btcoex_wl_pri	iwpriv wifiX g_ btcoex_wl_pri	N	Y	Get current priority frame types of WLAN traffic `# iwpriv wifi0 g_btcoex_wl_pri`

btcoex_dc <period> <wlan_ duration>	iwpriv wifi0 btcoex_ dc <period> <wlan_duration>	N	Y	Set duty cycle parameters.
				`iwpriv wifi0 btcoex_dc <period> <wlan_ duration>`
				WLAN TX Duty cycle = Time WLAN TX is active / Period **(2)**
				For example:
				`iwpriv wifi0 btcoex_dc 100 80`

g_btcoex_dc	iwpriv wifi0 g_ btcoex_dc	N	Y	Display current duty cycle parameters.

3.54 Interface Manager parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Interface manager (iface_mgr) application* section of the *Repeater access point functions* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-51 List of Interface Manager parameters

Parameter	Command	DA	OL	Description

discon_time
gdiscon_time
	iwpriv wifi*N* discon_time
	{secs}
		Y	Y	Time that can be used by STA VAP to establish reconnection with RootAP.
				The default value is 10 secs. The get parameter returns the current value.
				`#iwpriv wifi0 discon_time 20`
				`#iwpriv wifi0 gdiscon_time`
				`wifi0 gdiscon_time:20`

reconfig_time
greconfig_time
	iwpriv wifiN reconfig_time
	{secs}
		Y	Y	Time that can be used to reconfigure RootAP.
				The default value is 60 secs. The get parameter returns the current value.
				`#iwpriv wifi0 reconfig_time 20`
				`#iwpriv wifi0 greconfig_time wifi0`
				`greconfig_time:20`

3.55 Override MU-MIMO capability parameter

For more information on usage guidelines and the functionality regarding these parameters, see the *Override MU-MIMO capability based on vendor OUI* section of the *Performance enhancement capabilities* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Qualcomm Confidential - May Contain Trade Secrets 2021-05-13 03:37 PDT pan.yafeng@zte.com.cn

Parameter	Command	DA	OL	Description
mucapwar <1|0> get_mucapwar	iwpriv athx mucapwar <1|0> iwpriv athX get_ mucapwar	Y	Y	Enable (1) or disable (0) the capability to override MU-MIMO based on the vendor organizational unit identifier (OUI). Phones become 2x2 spatial streams-capable only when working with APs that are not MU-Capable. The workaround (WAR) implements a hack where we ensure that S7 phones are in SU-2x2 when it is the only MU-Capable client in the BSS. Also, all phones are ensured to be in MU-1x1 mode when other MU-Capable clients are present in the BSS (including presence of more than 1 MU-1x1 phone). When the MU-CAP-WAR is enabled, the Probe-Response is modified before it is sent to dedicated-client so that it joins as SU-2x2 The MU-Disabled Probe-Response (MU-Disabled in VHT CAP) is sent to the clients having Broadcom Vendor-IE. This hacked probe response is sent only when the following conditions are satisfied: ■ The number of MU-Capable clients in the system is 0. ■ The override variable is not set (this override variable prevents some race conditions). 0 Enables the feature 1 Disables the feature Enter the get command to retrieve the configured value.

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:07:44 PDT pan.yafeng@zte.com.cn

3.56 OCE parameters

The Technical Specification for WFA Optimized Connectivity Experience (OCE) defines architecture, protocols, and functionality for interoperability of Wi-Fi OCE devices. The OCE program aims to deliver a better overall connectivity experience by taking advantage of systemic information available within managed networks (for example, hotspot, workplace, and operator-deployed networks). The program is intended to address issues identified by operators, including prolonged connection setup times, poor wireless local area network (WLAN) connectivity, and airtime consumed by management frames.

For more information on usage guidelines and the functionality regarding these parameters, see the *OCE overview* section of the *Beacons and frames transmission* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-52 OCE parameters

Parameter	Command	Description
oce <0 or 1>	iwpriv athN oce <0 or 1>	Enable or disable OCE. An OCE AP transmits an OCE Capability Indication Attribute in the MBO-OCE IE in its Beacon, Probe Response and (Re-)Association Response frames. An OCE STA transmits an OCE Capability Indication Attribute in the MBO-OCE IE in its Probe Request and (Re-)Association Request frames. All OCE devices implementing this version of the OCE set the "OCE Release" sub-field bits of the OCE Control field of the OCE Capability Indication Attribute to 001.

g_oce	iwpriv athN g_oce	Retrieve the status of OCE.
		`#iwpriv ath0 oce 0`
		`#iwpriv ath0 g_oce`
		`ath0 g_oce:0`

oce_asoc_rej <0 or 1>	iwpriv athN oce_ asoc_rej <0 or 1>	Enable or disable OCE RSSI-based association rejection. When an OCE STA receives a (Re-)Association Response frame from an AP containing the RSSI-based Association Rejection attribute in the MBO-OCE IE, that STA shall not send a (Re-)Association Request frame or a unicast Probe Request frame to that AP until both of the following criteria are met: ■ "The time period indicated in the Retry Delay field of the most recently received RSSI-based Association Rejection attribute has expired, AND ■ "Based on the value indicated in the Delta RSSI field of the most recently received RSSI-based Association Rejection attribute, the STA estimates that the AP receives its (Re-)Association Request frame or unicast Probe Request frame with at least the minimum sufficient RSSI.
g_oce_asoc_rej	iwpriv athN g_oce_ asoc_rej	Retrieve the status of OCE RSSI-based association rejection.
oce_asoc_rssi <value>	iwpriv athN oce_ asoc_rssi <value>	Set the RSSI threshold value for association. An OCE AP must, when configured to do so, reject a (Re-)Association request if the RSSI of the (Re-)Association request received from a STA is below a minimum threshold value. This minimum threshold value is within the range of -60 to -90 dBm. How an OCE AP selects the minimum threshold value and field deployment configurability of an OCE AP with the minimum threshold value are out of scope of this specification.
g_oce_asoc_rssi	iwpriv athN g_oce_ asoc_rssi	Retrieve the specified RSSI threshold value for association.
oce_asoc_dly <value>	iwpriv athN oce_ asoc_dly <value>	Set the retry delay for reassociation. Retry delay set to a non-negative value in seconds for which the AP does not accept any subsequent (re-)association requests from the requesting STA, regardless of the RSSI at which they may be received. The value of the Retry Delay field is an 8-bit integer between 0 and 255, equal to the time period in seconds for which the AP will not accept any subsequent (re-)association requests from the requesting STA.
g_oce_asoc_dly	iwpriv athN g_oce_ asoc_dly	Retrieve the specified retry delay value for reassociation.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-52 OCE parameters (cont.)

Parameter	Command	Description
set_bpr_enable <0 or 1>	iwpriv athN set_bpr_enable <0 or 1>	To enable or disable the Broadcast Probe Response (BPR) feature for OCE, enter the iwpriv athN set_bpr_enable <0 or 1> command. An OCE STA shall support receiving broadcast Probe Response frames and using them in AP selection. It is expected that, if an OCE STA performs active scan, it will typically send a broadcast Probe Request frame. In such case, if an OCE AP responds with a Probe Response frame, it will send that frame to the broadcast address so that it can also be received by other STAs that are performing active scan. However, in certain cases where an OCE STA wishes to solicit an OCE AP's retry mechanism for more reliable delivery of the Probe Response frame (e.g. if the OCE STA is at the edge of coverage of an AP it is trying to discover, whose BSSID it knows a-priori), an OCE STA may decide to send a unicast Probe Request frame. In such case, if an OCE AP responds with a Probe Response frame, it will send that frame unicast to the STA's MAC address and the STA will send an acknowledgment if it receives the frame.
get_bpr_enable	iwpriv athN get_bpr_enable	To retrieve the configured setting for the BPR feature of OCE, enter the iwpriv athN get_bpr_enable command.
oce_hlp <0 or 1>	iwpriv athN oce_hlp <0 or 1>	In the MBO_OCE IE configuration, to enable/disable OCE HLP support, enter the iwpriv athN oce_hlp <0 or 1> command. The Vendor Specific information element format shall be used to define the MBO-OCE Information Element (MBO-OCE IE) in this specification. Little endian encoding is used for multi-byte fields and subfields. The "HLP Enabled" sub-field indicates whether or not an OCE AP has FILS Higher Layer Setup with Higher Layer Protocol Encapsulation enabled. An OCE STA and OCE STA-CFON set this sub-field to zero.
oce_hlp <0 or 1>	iwpriv athN oce_hlp <0 or 1>	To retrieve the configured setting for OCE HLP, enter the iwpriv athN g_oce_hlp command.
oce_wan_mtr <ul metric> <dl metric>	iwpriv athN oce_wan_mtr <ul metric> <dl metric>	To set WAN metrics in the uplink and downlink direction for OCE AP, enter the iwpriv athN oce_wan_mtr <ul metric> <dl metric> command. An OCE AP that has an estimate of the available WAN backhaul link capacity shall transmit the Reduced WAN Metrics attribute of the MBO-OCE IE in its Beacon Frames and Probe Response Frames. If estimates of the available WAN backhaul link capacity are unavailable, the OCE AP is not required to transmit this attribute. The current available (unused) WAN backhaul capacity is the product of the WAN backhaul link current speed and one minus the current fractional loading of the WAN backhaul link. The algorithm to calculate currently available (unused) WAN backhaul link capacity is implementation dependent.
g_oce_wan_mtr	iwpriv athN g_oce_wan_mtr	To retrieve the configured setting for WAN metrics of OCE AP, enter the iwpriv athN g_oce_wan_mtr command.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-52 OCE parameters (cont.)

Parameter	Command	Description
rnr <0 or 1>	iwpriv athN rnr <0 or 1>	To enable or disable the reduced neighbor report (RNR)/AP channel report functionality of OCE, enter the iwpriv athN rnr <0 or 1> command. An OCE AP shall, when configured to do so, transmit a Reduced Neighbor Report element in Beacon frames and Probe Response frames, and may transmit a Reduced Neighbor Report element in FILS Discovery frames. An OCE AP shall not transmit a Reduced Neighbor Report element in a unicast Probe Response frame to a non-OCE STA unless that STA has FILS Capability indication bit set to one, in which case transmission of a Reduced Neighbor Report element is at the discretion of the AP. If an OCE AP includes information in a Reduced Neighbor Report element about all neighboring BSS to its best knowledge, regardless of operating channel or band, that are operating with a given SSID within coverage of the AP, it shall include the corresponding Short SSID in an RNR Completeness attribute carried in the MBO-OCE IE in the same Beacon or Probe Response frame. If the information in the Reduced Neighbor Report element is not known to be complete for any SSID, the RNR Completeness attribute shall not be included. If the information in the Reduced Neighbor Report element is complete for all SSIDs (that is, information is included for every neighboring BSS), the RNR Completeness attribute shall be included with Length field set to zero (that is, wildcard), and the Short SSID List field absent.
g_rnr	iwpriv athN g_rnr	To retrieve the configured RNR/AP channel report functionality of OCE, enter the iwpriv athN g_rnr command.
rnr_fd <0 or 1>	iwpriv athN rnr_fd <0 or 1>	To enable/disable inclusion of RNR IE in FILS Discovery, enter the iwpriv athN rnr_fd <0 or 1> command. An OCE AP shall, apart from the exception below, attempt to transmit either a FILS Discovery frame or Beacon frame every 20 TUs. The Beacon Interval shall be a minimum of 100 TUs. In every beacon interval, the OCE AP shall attempt to transmit the first FILS Discovery frame 20 TUs after the TBTT. The exception to the above is if the AP has a neighboring OCE AP in the same ESS and on the same channel, and that neighboring AP is transmitting FILS Discovery frames and including the AP's BSS in the Reduced Neighbor Reports that it transmits, and that neighboring AP is sufficiently proximate such that STAs in the AP's coverage area are able to receive FILS Discovery frames and Beacon frames from that neighboring AP, in which case the AP is not required to transmit FILS Discovery frames.
g_rnr_fd	iwpriv athN g_rnr_fd	To retrieve the RNR IE included in FILS Discovery, enter the iwpriv athN g_rnr_fd command.
rnr_tbtt <0 or 1>	iwpriv athN rnr_tbtt <0 or 1>	To enable/disable calculation of target beacon transmission time (TBTT) offset in the RNR IE, enter the iwpriv athN rnr_tbtt <0 or 1> command. An OCE STA-CFON shall attempt to transmit either a FILS Discovery frame or Beacon frame every 20 TUs from the time it starts transmitting Beacon frames for a period of at least three minutes. In every beacon interval during that period, the OCE STA-CFON shall attempt to transmit the first FILS Discovery frame 20 TUs after the TBTT. A FILS Discovery frame transmitted by an OCE AP and STA-CFON shall carry information to indicate the next TBTT on the BSS of the transmitting AP or STA-CFON. The Neighbor AP TBTT Offset field indicated for those BSS shall be a valid (non-255) value. If an OCE AP includes other BSS in the Reduced Neighbor Report element and the TBTT of those BSS is unknown, it shall set the Neighbor AP TBTT Offset field to 255.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-52 OCE parameters (cont.)

Parameter	Command	Description
g_rnr_tbtt	iwpriv athN g_rnr_tbtt	To indicate whether the calculation of TBTT offset in the RNR IE is enabled or not, enter the iwpriv athN g_rnr_tbtt command.
apchanrpt <0 or 1>	iwpriv athN apchanrpt <0 or 1>	To enable/disable AP channel reporting (make sure to enable background scanning in all radios), enter the iwpriv athN apchanrpt <0 or 1> command. An OCE STA shall support receiving an AP Channel Report element in Beacon frames and Probe Response frames, and may use the received neighbor information in its scanning algorithms to prioritize scanning of channels. An OCE AP shall not include a channel in an AP Channel Report element if it has not identified an OBSS to be operating on that channel.
g_apchanrpt	iwpriv athN g_ apchanrpt	To retrieve the configured AP channel reporting functionality, enter the iwpriv athN g_apchanrpt command.
set_bcn_rate <1000 or 2000 or 5500 or 11000>	iwpriv athN set_bcn_ rate <1000 or 2000 or 5500 or 11000>	To set beacon rate, enter the iwpriv athN set_bcn_rate <1000 or 2000 or 5500 or 11000> command. An OCE AP shall transmit Beacon frames at a minimum rate of 5.5 Mbps on the 2.4 GHz band, OOB. An OCE STA-CFON shall transmit Beacon frames at a minimum rate of 5.5 Mbps on the 2.4 GHz band, OOB.
get_bcn_rate	iwpriv athN get_bcn_ rate	To retrieve the configured beacon rate, enter the iwpriv athN get_bcn_ rate command.
prb_rate <1000 or 2000 or 5500 or 11000>	iwpriv athN prb_rate <1000 or 2000 or 5500 or 11000>	To set the Probe-Response rate, enter the iwpriv athN prb_rate <1000 or 2000 or 5500 or 11000> command. An OCE AP shall transmit Probe Response frames at a minimum rate of 5.5 Mbps on the 2.4 GHz band OOB, except for: a. Unicast responses to Probe Request frames from non-OCE STAs which can be transmitted at any rate allowed, and b. Unicast responses to Probe Request frames from OCE STAs received at a rate below 5.5 Mbps which can be transmitted at the same rate as the Probe Request frame. An OCE STA-CFON shall transmit Probe Response frames at a minimum rate of 5.5 Mbps on the 2.4 GHz band OOB, except for unicast responses to Probe Request frames from non-OCE STAs which can be transmitted at any rate allowed.
g_prb_rate	iwpriv athN g_prb_ rate	To retrieve the configured Probe-Response rate, enter the iwpriv athN g_prb_rate command.

3.57 ACS/DCS/OBSS parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *EACS PLUS: ACS/OBSS* and *Channel-ranking information in ACS reports* sections of the *Advanced channel selection capabilities* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-53 ACS/DCS/OBSS iwpriv parameters

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

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-53 ACS/DCS/OBSS iwpriv parameters (cont.)

Command	Command	Description
acs_dbgtrace	iwpriv wifi1 acs_dbgtrace 0xxx	Set ACS run time debug option The hexadecimal values signify one of the following: ACS_DBG_EXT = 0x0000, /* Always enabled for all external APIs */ ACS_DBG_BASE = 0x0001, /* Base prints */ ACS_DBG_RSSI = 0x0002, /* RSSI stats */ ACS_DBG_ADJCHAN = 0x0004, /* Adjacent channel stats collection */ ACS_DBG_NF = 0x0008, /* Noise floor stats */ ACS_DBG_CHLOAD = 0x0010, /* Channel load stats */ ACS_DBG_REGPOWER = 0x0020, /* Regulatory tx power stats */ ACS_DBG_OBSS = 0x0040, /* OBSS/Coex checking */ ACS_DBG_SCAN = 0x0080, /* Scan handling */ ACS_DBG_BLOCK = 0x0100, /* Blocking logic */ ACS_DBG_FILTER = 0x0200, /* EACS-plus filtering logic */ ACS_DBG_CHLST = 0x0400, /* Channel list population */ ACS_DBG_MAX = 0xFFFF, /* All prints */
g_ acs_dbgtrace	iwpriv wifi1 g_acs_dbgtrace	Display the debug option specified
obss_rssi_th	iwpriv wifiX obss_rssi_th <value>	Configure OBSS RSSI threshold. If OBSSI RSSI is greater than configured value then only move to 20 MHz. Value range: 0 – 127.
gobss_rssi_th	iwpriv wifiX gobss_rssi_th	Retrieve OBSS RSSI threshold
obss_rx_rssi_th	iwpriv wifiX obss_rx_rssi_th <value>	Configure RSSI threshold for received frame with 40 MHz intolerance bit. If RSSI of received is greater than configured value then only move to 20 MHz. Value range: 0–127.
acs_txpwr_opt	iwpriv wifiX acs_txpwr_opt <value>	Configures ACS Tx Power parameter option. Values range: 1-2 1. Tx Power for good throughput 2. Tx Power for maximum range
g_acs_txpwr_opt	iwpriv wifiX g_acs_txpwr_opt	Retrieves the Tx power type values
antenna_plzn	iwpriv wifiX antenna_plzn <value>	Configures antenna polarization. Value – 32 Bit value Bits [24-31] – specifies to enable antenna polarization (0xFF – enable, 0 - disable). Bits [0-23] – Antenna value. Default value – 0xFF00000A (VHVH antenna polarization) V – Vertical, H – Horizontal.

Table 3-53 ACS/DCS/OBSS iwpriv parameters (cont.)

Command	Command	Description
acs_2g_allch	iwpriv wifi*N acs_2g_allch* {1|0}	Enable (1) or disable (0) overlap channel selection in 2.4GHz band with ACS. If this option is enabled, AP can select one of the overlapping channels with ACS. The default value is 0. `#iwpriv wifi0 acs_2g_allchan 1`

g_acs_2g_allchan	iwpriv wifi*N g_acs_2g_ allchan*	The get parameter returns the current value of overlap channel selection.
		`#iwpriv wifi0 g_acs_2g_allchan`
		`wifi0 g_acs_2g_allchan:1`

acs_rank_en <1 | 0>	iwpriv wifiX acs_rank_en <1 | 0>	Enable (1) or disable (0) the ACS channel-ranking functionality.
g_acs_rank_en	iwpriv wifiX g_acs_rank_en	Displays the status of the ACS channel-ranking functionality.
channel 0	iwpriv athX channel 0	Trigger the ACS scan. User must set correct mode and trigger ACS before generating the ACS channel-ranking report. On IPQ807x platforms and later, use the `iwpriv athN acsreport` value to enable channel loading. In this tri-radio solution, if the new phymode is same as the current phymode, then channel can be changed simply by using the cfg80211tool athx channel <new channel number> <band> command. If the new phymode is different from current phymode (this could happen during channel switch from 2.4 GHz to 5 GHz, 5 GHz to 2.4 GHz, 11ac to 11ng etc,) then just changing the channel may put driver in auto mode / default mode if the new channel doesn't support current phymode. This is a known behavior with the current driver. In this case, the solution is to change both channel and phymode.
acs_srloadvar	iwpriv wifi1 acs_srloadvar <value>	Set the SR load variance. If two channels differ with SR load value less then SR load variance, they are treated as having same SR load for next level evaluation Default Value: 3
g_acssrloadvar	iwpriv wifi1 g_acssrloadvar	Displays spatial reuse (SR) variance value

3.58 Dynamic encapsulation and decapsulation configuration

For more information on usage guidelines and the functionality regarding these parameters, see the *Raw mode Tx/Rx* section of the *Beacons and frames transmission* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

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

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-54 Dynamic encapsulation and decapsulation configuration

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

Qualcomm Confidential 2021-05-13 13:37:54 PDT pan.yaren@2[illegible]

3.59 Raw mode simulation parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Raw mode Tx/Rx* section of the *Beacons and frames transmission* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

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

Starting with the QCA_Networking_2020.SPF.11.3 release, the raw mode packet simulation code is compiled as a separate ko named "rawmode_sim.ko" and it is loaded only if the following user configuration is set.

```
uci set wireless.qcawifi=qcawifi
uci set wireless.qcawifi.load_rawsimulation_mod=1
uci commit wireless
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-55 Raw mode simulation parameters

Command	DA	OL	Description	
iwpriv athN rawsim_ txagr <*value*>	N	Y	Enable/Disable use of multiple fragments during Tx by Raw 802.11 mode simulation. This will result in creation of A-MSDUs, with one MSDU per fragment.	
			1	Enable
			0	Disable
iwpriv athN g_rawsim_ txagr	N	Y	Return whether use of multiple fragments during Tx by raw mode simulation is enabled or not.	
			1	Enabled
			0	Disabled
iwpriv athN rawsim_stats	N	Y	Print raw mode simulation module internal statistics. Note: These are not exhaustive and do not cover events outside the simulation module, such as higher layer failure to process successfully decapsulated MPDUs. These are meant for QCA internal debug purposes only.	
iwpriv athN clr_rawsim_ stats 0	N	Y	Clear raw mode simulation module internal statistics.	
iwpriv athN rawsim_ debug <*value*>	N	Y	Enable/disable dumping of additional debug data by raw mode simulation. Currently, this consists of hex dumps of frames before/after encapsulation/decapsulation.	
			0	Disable
			1	Enable and dump headers only
			2	Enable and dump entire frame
iwpriv athN get_rawsim_ debug	N	Y	Return whether dumping of additional debug data by raw mode simulation is enabled or not.	
			1	Enabled
			0	Disabled
cfg80211tool athN rsim_ en_frmcnt <*value*>	N	Y	Set number of frames to dump before and after encapsulation.	
			0	Print all frames
			0x1 to 0x7FFFFFFF	Print specified number of frames
cfg80211tool athN rsim_ de_frmcnt <*value*>	N	Y	Set number of frames to dump before and after decapsulation.	
			0	Print all frames
			0x1 to 0x7FFFFFFF	Print specified number of frames

3.60 RDK-B HAL API parameters

The Reference Design Kit for Broadband (RDK-B) is an open source initiative standardising software functionalities for broadband devices enabling MSOs to efficiently deploy services to a large customer base. The primary purpose of RDK-B is to create a standardized software stack for various OEM's providing them with specific features to manage complex broadband functions

such as Wide Area Networking (WAN), Local Area Networking (LAN), data reporting and management, home-networking technologies such as Wi-Fi and Multimedia over Coax Alliance (MoCA) and Internet of Things (IoT) controllers such as ZigBee, Bluetooth LE.

For more information on usage guidelines and the functionality regarding these parameters, see the *HAL API* section of the *WLAN AP driver architecture* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-56 RDK-B HAL API parameters

Parameter	Command	D	O	Description
set_waterm_th	iwpriv athX set_waterm_th <threshold value >	Y	Y	Assign the high watermark threshold value TAD (Test And Diagnostic) monitors the amount of free memory available in the system at run time. It triggers the memory/Resource Reclamation (RR) process in TDM, when the amount of free memory drops below a configurable threshold. RR process can also be triggered by memory allocation failures which results in a notification being sent to TDM to try to reclaim memory. If "Average Memory Used" reaches threshold value, necessary action will be executed. Example: `#iwpriv ath0 set_waterm_th 40`
get_waterm_th	iwpriv athX get_waterm_th	Y	Y	Get the high watermark threshold value Example: `#iwpriv ath0 get_waterm_th`
get_th_reach	iwpriv athX get_th_reach	Y	Y	Get the number of times the threshold has been breached Example: `#iwpriv ath0 get_th_reach`
get_assoc_reach	iwpriv athX get_assoc_reach	Y	Y	Get the maximum number of associated devices post the threshold value Example: `#iwpriv ath0 get_assoc_reach`
get_assoc_dev_watermark_time	Wifitool athX get_assoc_dev_watermark_time	Y	Y	Displays the time at which the maximum number of devices has been associated post the threshold Example: `#wifitool ath0 get_assoc_dev_watermark_time`
set_traf_rate	iwpriv athX set_traf_rate	Y	Y	Set the traffic rate/time at which the received signal level is to be measured `#iwpriv ath0 set_traf_rate 300`
set_traf_int	iwpriv athX set_traf_int	Y	Y	Set the traffic interval until which the received signal level has to be measured. Example `#iwpriv ath0 set_Traf_int 1800`

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-56 RDK-B HAL API parameters (cont.)

set_traf_stat	iwpriv athX set_ traf_stat	Y	Y	Enable/disable the Traffic measurement statistics for the specific rate and interval. Example: `#iwpriv ath0 set_traf_stat 1/0`
display_traffic_ statistics	Wifitool athX display_traffic_ statistics	Y	Y	Display the statistics. i.e the Received signal level maximum,minum,and median at each rate until the interval time is elapsed. Example: # `wifitool ath0 display_traffic_`

3.61 Offload statistics parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Using statistics for debugging* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6). High CPU usage is expected if offload statistics is enabled.

Table 3-57 Offload statistics parameters

Parameter	Command	Description
`enable_ol_ stats`	`iwpriv wifiX enable_ol_stats`	Enter the `iwpriv wifiX enable_ol_stats 1` command to enable the collection of offload-mode statistics. Only after the iwpriv wifi0 enable_ol_ stats 1 command is issued to enable station statistics, the output of the `apstats -s -m` command displays information pertaining to AP statistical details. For NSS offload-specific Wi-Fi statistics for Lithium, to enable enhanced Tx/Rx statistics, use the `iwpriv wifiX enable_ol_stats 1` command. The `iwpriv wifiX enable_ol_stats 1` command enables the statistics collection from NSS FW to host for the NSS offloaded radio (wifiX). Similarly, `iwpriv wifiX enable_ol_stats 0` disables the statistics collection from NSS FW to host for the NSS offloaded radio (wifiX).

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-57 Offload statistics parameters (cont.)

Parameter	Command	Description
`enable_statsv2 <Value>`	`iwpriv wifiX enable_statsv2 <Value>`	After enabling the offload-mode statistics, enter this command to selectively enable different statistics or use 0xF to enable all statistics. The following values are possible: ■ Bit0: WIFI_RX_STATA_REGULAR_PEER - Default non zero value, Rx Extended stats for regular peer updated ■ Bit1: WIFI_RX_STATS_STATUS_NOT_OK - Rx extended statistics for Errored/FCS packets are updated ■ Bit2: WIFI_RX_STATS_INTRABSS - Rx statistics for intra-BSS frames are updated ■ Bit3: WIFI_TX_STATS_TIMESTAMP - Tx Sojourn statistics are updated To enable all statistics, enter the following command: `iwpriv wifi0 enable_statsv2 0xF`
`enable_statsv3 <value>`	`iwpriv wifiX enable_statsv3 <2 | 0>` `iwpriv wifi0 fc_peer_stats <peer_mac>`	The existing enhanced v3 stats command is extended to support the jitter. The command to enable or disable the jitter stats is as follows: Enable: iwpriv wifiX enable_statsv3 2 Disable: iwpriv wifi0 enable_statsv3 0 The jitter stats are displayed per client. The command to display stats is iwpriv wifi0 fc_peer_stats <peer_mac>

3.62 Tx Ack timeout parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Configure Tx Ack timeout value* section of the *Beacons and frames transmission* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-58 Tx Ack timeout parameters

Parameter	Command	Description
acktimeout yy get_acktimeout	iwpriv wifiX acktimeout yy iwpriv wifix get_acktimeout	Specify the Tx acknowledgment packet timeout value. This command is based on the OL_ATH_PARAM_TXACKTIMEOUT to create. The OL_ATH_PARAM_TXACKTIMEOUT parameter is added for physical device (PDEV). Here, `wifiX` is wifi interface `wifi0/wifi1/wifi2`, and *`yy`* is the Tx ACK timeout value where *`yy`* ranges from 0x40–0xff. If the value less than 0x40 or big than 0xff, it prints out the assert command. This value will save to tx_ack_timeout parameter. This `iwpriv wifiX acktimeout yy` command calls the WMI_PDEV_PARAM_TX_ACK_TIMEOUT WMI command to send the yy value to FW. This capability is supported on the QCA9980, QCA98886, IPQ4019, and IPQ807x/IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx/QCN90xx or QCN602x + IPQ807xA/IPQ60xx/IPQ50xx platforms. When AP is rebooted or powered on again, reset this value again manually. The Tx Ack value ranges from 0x40 to 0xFF. When FW recovers, it is reset to 0x40 and needs to be modified accordingly manually. The Tx packet retry latency increases as timeout value changes to a value higher than 0x40 and causes performance degree in some environments. Enter the `iwpriv wifix get_acktimeout` command to retrieve the configured Tx Ack timeout value. TX ACK Time-out value must be between 0x40 and 0xFF This command is not supported for IPQ4019 and QCA9984 platforms.

3.63 11.0 WLAN driver parameters

For more information on usage guidelines and the functionality regarding the cfg80211 utility, see the *cfg80211 WLAN driver support* section of the *WLAN AP driver build and configuration methods* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-59 11.0 WLAN driver parameters

Parameter	Command	Description
qdf_cv_lvl value	**cfg80211tool athX qdf_cv_lvl value**	Sets the trace level for a particular category or module ID. Value holds 32-bits where the MSB 16-bits indicate the category and LSB 16-bits indicate the trace level. The available verbose levels are: ■ 0 - None ■ 1 - Fatal ■ 2 - Error ■ 3 - Warn ■ 4 - Info ■ 5 - Info High ■ 6 - Info Medium ■ 7 - Info Low ■ 8 - Debug ■ 9 - All An example is `iwpriv ath0 qdf_cv_lvl 0x00040009`, which enables all trace levels for category represented by 0x0004.
g_qdf_cv_lvl	**cfg80211tool athX g_ qdf_cv_lvl value**	Retrieves the configured trace level for a particular category or module ID.
he_mcs	**cfg80211tool athX he_mcs <value>** **iwpriv athX he_mcs <value>**	Sets the 11ax fixed MCS rates from MCS0 to MCS11. The value can be in the range of 0-11.
get_he_mcs	**cfg80211tool athX get_he_mcs** **iwpriv athX get_he_ mcs**	Gets the 11ax fixed MCS rates.
he_extrange	**cfg80211tool athX he_extrange <value>** **iwpriv athX he_ extrange <value>**	Sets the 11ax extended range to enable/disable state support. Enter 0 to disable, or 1 to enable.
get_he_extrange	**cfg80211tool athX get_he_extrange** **iwpriv athX get_he_ extrange**	Gets the configured 11ax extended range support.
he_dcm	**cfg80211tool athX he_dcm <value>** **iwpriv athX he_dcm <value>**	Sets the 11ax dual -carrier modulation to enabled/disabled state. Enter 0 to disable, or 1 to enable. he_dcm is applicable for non-AP modes such as STA to enable or disable the HE PHY DCM max constellation TX support. However, to enable or disable HE PHY DCM max constellation RX support across all operating modes, use the cfg80211tool ath0 he_dcm_max_const_rx 0/1 command.
he_dcm_max_ const_rx	**cfg80211tool athX he_dcm_max_const_ rx <value>** **iwpriv athX he_dcm_ max_const_rx <value>**	Enable or disable HE PHY DCM max constellation RX support across all operating modes.

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
get_he_dcm	**cfg80211tool athX get_he_dcm <value>** **iwpriv athX get_he_dcm <value>**	Retrieves the configured 11ax dual-carrier modulation setting (enabled or disabled).
he_bsscolor	**cfg80211tool wifiN he_bsscolor {0...63} {0/1}** **cfg80211tool wifiN get_he_bsscolor**	Sets fixed BSS color in 11ax mode. Enter a value in the range of 0-63. The second argument is the override{0|1} flag. Setting the override argument 1 along with the BSS Color setting will force the user set BSS color. BSS color can be disabled when the BSS color argument is passed 0 by the user. This command needs to be issued before VAP up. It is recommended to be set in wireless configuration files. This parameter is used to program the HW register field on the device. This parameter is used to program the "Spatial Reuse Parameter Set element" in the beacon.
get_he_bsscolor	**cfg80211tool athX get_he_bsscolor** **iwpriv athX get_he_bsscolor**	Gets the current BSS color in 802.11ax mode. This parameter is used to program the HW register field on the device. This parameter is used to program the "Spatial Reuse Parameter Set element" in the beacon.
set_mesh_mcast	**cfg80211tool athX set_mesh_mcast <value>**	Setting a value of 1 enables receipt of all broadcast packets. Setting a value of 0 disables receipt of all broadcast packets.
get_mesh_mcast	**cfg80211tool athX get_mesh_mcast**	Displays the current value of mesh multicast setting.
dump_chmsk_tbl	**cfg80211tool wifiX dump_chmsk_tbl**	Dump available chainmasks for a given radio with supported flags.
set_reo_dest	**cfg80211tool wifiX set_reo_dest**	Select reorder destination ring for a particular pdev; the default value is 1. For Rx Reorder (REO) ring processing, the regular MSDU Rx path includes processing REO descriptors, updating REO ring read index, and indicating packets to upper layers for further inspection/delivery. The host driver manages these per-TID REO queues, including adding/deleting TID queues based on ADDBA/DELBA requests and initializing/updating the PN and starting sequence numbers for each TID.
get_reo_dest	**cfg80211tool wifiX get_reo_dest**	Check whether the reorder destination ring is being used for that pdev.
acs_srloadvar	**cfg80211tool wifiX acs_srloadvar <value>**	Sets the spatial reuse load variance. If two channels differ with spatial reuse, load value is less than source load variance.They are treated as having same spatial load for next level evaluation. The default value is 3.
g_acssrloadvar	**cfg80211tool wifiX g_acssrloadvar**	Retrieves the value of spatial reuse load variance.
dump_chmsk_tbl	**cfg80211tool wifiX dump_chmsk_tbl**	Dumps available chainmasks for a particular radio with supported flags.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
print_obj_refs <value>	**cfg80211tool wifiX print_obj_refs <value>**	Prints unfreed objects with their reference counts. This command is an object manager debug command. Now, the value can be any integer; this value is reserved for future. For example, the `iwpriv wifi0 print_obj_refs 1` command prints all objects with their reference counts.
txrx_stats	**cfg80211tool wifiX txrx_stats <value>**	Displays host and firmware radio-level statistics for debugging purposes. Enter the *<value>* in the range of 1-16 for firmware radio-level statistics and in the range of 257-262 for host radio-level statistics.
dp_peer_stats	**cfg80211tool wifiX dp_peer_stats <mac_addr>:0x3**	Displays firmware per-peer statistics.
fc_peer_stats	**cfg80211tool wifiX fc_peer_stats <mac_addr>**	Displays host per-peer statistics.
set_qosacn_conf <value>	**cfg80211tool athX set_qosacn_conf <value>**	Setting qos_acnconf to 1 forwards the QoS action frames to upper stack after handling the frames in the host driver. Setting qos_acnconf to 3 forwards the QoS action frames to upper stack without handling the frames in the host driver. If bit 0 of value, QoS action frames are forwarded to the upper stack. If bit 1 of value is set, QoS action frames are not handled in host. This setting must be configured in conjunction with bit 0; otherwise, it is invalid.
get_qosacn_conf	**cfg80211tool athX get_qosacn_conf**	Retrieves and displays the current configuration of QoS action frames. By default, it displays a value of zero.
ext_nss 0 **g_ext_nss.**	**cfg80211tool athX ext_nss 0** **cfg80211tool athX g_ext_nss.**	Disables extended NSS signaling on the APs or STAs for the designated frequency used in the network topology. When extended NSS signaling is disabled, the "VHT Extended NSS BW Capable" Bit in the "Supported VHT-MCS and NSS Set" field of the VHT Capabilities element in the Beacon and Probe Response frame are equal to zero. The get command retrieves the configured value. EXT NSS signaling support becomes disabled if driver is not extended NSS-signaling capable.
ext_nss 1 **g_ext_nss.**	**cfg80211tool athX ext_nss 1** **cfg80211tool athX g_ext_nss.**	Enables extended NSS signaling on the APs or STAs for the designated frequency used in the network topology. When extended NSS signaling is enabled, the "VHT Extended NSS BW Capable" Bit in the "Supported VHT-MCS and NSS Set" field of the VHT Capabilities element in the Beacon and Probe Response frame are equal to one. The get command retrieves the configured value. EXT NSS signaling support becomes disabled if driver is not extended NSS-signaling capable.
ext_nss_sup <1/0> **g_ext_nss_sup**	**cfg80211tool athX ext_nss_sup <1/0>** **cfg80211tool athX g_ext_nss_sup**	Enables/disables extended NSS-signaling support. The iwpriv athX g_ext_nss_sup command retrieves the extended NSS-signaling support; 1 if enabled, 0 if disabled. If fixed MCS is set, the entire rate is fixed (MCS, NSS, BW). For a station that is connected in 160 MHz mode, it is necessary to disable extended number of spatial streams (NSS) signaling support. Else, the station cannot receive 160 MHz packets.

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
acktimeout yy **get_acktimeout**	**cfg80211tool wifiX acktimeout yy** **cfg80211tool wifix get_acktimeout**	Specify the Tx acknowledgment packet timeout value. This command is based on the OL_ATH_PARAM_TXACKTIMEOUT to create. The OL_ATH_PARAM_TXACKTIMEOUT parameter is added for physical device (PDEV). Here, `wifiX` is wifi interface `wifi0/wifi1/wifi2`, and `yy` is the Tx ACK timeout value where `yy` ranges from 0x40–0xff. If the value less than 0x40 or big than 0xff, it prints out the assert command. This value will save to tx_ack_timeout parameter. This `iwpriv wifiX acktimeout yy` command calls the WMI_PDEV_PARAM_TX_ACK_TIMEOUT WMI command to send the yy value to FW. This capability is supported on the QCA9984, QCA9980, QCA98886, IPQ4019, and IPQ807x/IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx/QCN90xx or QCN602x + IPQ807xA/IPQ60xx/IPQ50xx platforms; it is not supported on the QCA9880 chipsets. When AP is rebooted or powered on again, reset this value again manually. The Tx Ack value ranges from 0x40 to 0xFF. When FW recovers, it is reset to 0x40 and needs to be modified accordingly manually. The Tx packet retry latency increases as timeout value changes to a value higher than 0x40 and causes performance degree in some environments. Enter the `iwpriv wifix get_acktimeout` command to retrieve the configured Tx Ack timeout value. Tx ACK Time-out value must be between 0x40 and 0xFF
ampdu <0-255> **get_ampdu**	**cfg80211tool ath0 ampdu <0-255>** **cfg80211tool ath0 get_ampdu**	Set the maximum number of frames or MAC PDUs (MPDUs) that are aggregated for a custom AMPDU. This maximum number is per VAP interface per access category. The maximum value for custom AMPDU size is 255. cfg80211tool athX get_ampdu—Retrieve the configured custom aggregation size for AMPDUs per VAP interface.
amsdu <0-7> **get_amsdu**	**cfg80211tool ath0 amsdu <0-7>** **cfg80211tool ath0 get_amsdu**	Set the maximum number of frames for a custom AMSDU size. This maximum number is per VAP interface. The maximum value for custom AMPDU size is 7. cfg80211tool athX get_amsdu—Retrieve the configured custom aggregation size for AMSDUs per VAP interface per access category.
ratedrop <value> **get_ratedrop**	**cfg80211tool ath0 ratedrop <value>** **cfg80211tool ath0 get_ratedrop**	Enable or disable rate-control logic for management, single-user data, and multi-user data packets per VAP interface. The maximum value for rate drop-down logic is 7. cfg80211tool athX get_ratedrop—Retrieves the rate-drop setting configured per VAP.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
he_rtsthrshld <value> **get_he_rtsthrshld**	**cfg80211tool ath0 he_rtsthrshld <value>** **cfg80211tool ath0 get_he_rtsthrshld**	Enable or disable the TxOP duration-based high efficiency rate threshold setting. Use the get command to retrieve the rate-threshold setting configured per VAP. The TxOP Duration RTS Threshold subfield enables an HE AP to manage RTS/CTS usage by non-AP HE STAs that are associated with it. The TxOP Duration RTS Threshold subfield contains the TxOP duration RTS threshold in units of 32 us, which enables the use of RTS/CTS except for the value 1023. On the command line, the following values can be used: SPF releases earlier than SPF 11.2: ■ 0-1023: Value given is directly inserted into the beacon frames SPF 11.2 and later releases: ■ 0: HE-RTS threshold fully disabled – beacons will advertise the value 1023, and stations are not required to use RTS ■ 1-1022: supplied HE-RTS threshold value is advertised directly in the beacon ■ 1023: FW dynamically controls the HE-RTS threshold field in the beacon Currently, this command does not have support over WEXT interface Therefore, this command cannot be issued with 'iwpriv'. Because the command has support over NL interface only, it must be issued with 'cfg80211tool'.
he_ltf X **get_he_ltf**	**cfg80211tool ath0 he_ltf X** **cfg80211tool ath0 get_he_ltf** **iwpriv ath0 he_ltf X** **iwpriv ath0 get_he_ltf**	Enable or disable fixed-rate high efficiency long training field (HE-LTF). It is important to set fixed rate values using the `iwpriv athX he_mcs` command before issuing this command. Use the get command to retrieve the configured HE-LTF value. Where X is: ■ 0 - "Auto" – Basically, the HE-LTF is automatically set to the mandatory HE-LTF, based on the GI setting (see shortgi command) ■ 1 - 1x LTF (1x LTF = 3.2 us LTF) ■ 2 - 2x LTF (2x LTF = 6.4 us LTF) ■ 3 - 4x LTF (3x LTF = 12.8 us LTF)

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
he_ar_gi_ltf **get_he_ar_gi_ltf**	**cfg80211tool athN he_ar_gi_ltf X** **cfg80211tool athN get_he_ar_gi_ltf**	Sets or unsets possible set of LTF and GI combinations for the autorate case. The target is restricted to choose a proper value of ltf and gi from the possible sets of LTF and GI set using this command. Where X is a 16 bit value with following bit significance: bits 7-0 [for LTF setting]: ■ BIT0 = 1 [HE_LTF_1X] ■ BIT1 = 1 [HE_LTF_2X] ■ BIT2 = 1 [HE_LTF_4X] ■ BIT3-7 = Reserved bits bits 15-8 [for GI setting]: ■ BIT8 = 1 [400 NS] ■ BIT9 = 1 [800 NS] ■ BIT10 = 1 [1600 NS] ■ BIT11 = 1 [3200 NS] ■ BIT12-15 = Reserved bits This command modifies the possible set of values of LTF/GI (in autorate case) for Tx (self) only and the values are not reflected in any IE advertised to peer. Example input: cfg80211tool athN he_ar_gi_ltf 0x0303 sets the autorate GI set as [400ns (BIT8), 800ns (Bit9)] and LTF = [1X (Bit0), 2X (Bit1)] Passing the argument value 0xFFFF(for auto rate) will allow the FW to choose the best GI+LTF combination from the possible set of combinations. The max PHY rate displayed by default will be with 800ns SGI. For both auto-rate and fixed rate the FW will use 800ns SGI by default. If the user changes the auto-rate (he_ar_gi_ltf)/fixed rate (shortgi) SGI, the max PHY rate will reflect bit rate corresponding to the SGI setting. Max PHY rate is displayed according to 800ns SGI(4.8039 Gb/s | 1.1471 Gb/s). Because 400ns SGI is not used by default for HE., it will only be used when the user explicitly sets SGI to 400ns (in which case max PHY rate is displayed based on SGI 400ns. The following list describes the differences in the max PHY rate values between SPF 10.0 and SPF 11.0. Until SPF 10.0, the values corresponded to QCA proprietary HE 400 ns GI. Starting with SPF 11.0, the values correspond to 802.11ax standard based 800 ns GI. On IPQ807x: ■ For 5G 11axa HT80 ath0, 4.949 Gb/s in SPF 10.0 CS and 4.8039 Gb/s in SPF 11 CS ■ For 2.4G 11axg HT40 ath1, 1.1818 Gb/s in SPF 10.0 CS 1.1471 Gb/s in SPF 11.0 CS
dfs_notify **get_dfs_notify**	**cfg80211tool ath0 dfs_notify** **cfg80211tool ath0 get_dfs_notify**	Enable or disable dynamic frequency selection (DFS) notification. Use the get command to retrieve the configured DFS notification setting. The get command is used to check if RADAR detection information must be shared with user or not.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
dfs_notify **g_dfs_notify**	**iwpriv ath0 dfs_notify** **iwpriv ath0 g_dfs_notify**	Enable or disable dynamic frequency selection (DFS) notification. Use the get command to retrieve the configured DFS notification setting. The get command is used to check if RADAR detection information must be shared with user or not.
get_11hcap <mac_address>	**cfg80211tool athX get_11hcap <mac_address>**	The cfg80211tool command is introduced for checking if a connected station has 802.11h spectrum management capability with the help of the station's MAC address as the input value. The output of this cfg80211tool athX get_11hcap <mac_address> command displays the following: Output: 1: Station is 802.11h capable | 0: Station is not 802.11h capable
he_dlofdma	**cfg80211 athX he_dl_ofdma** **iwpriv athX he_dlofdma**	Enable (1) or disable (0) high efficiency downlink orthogonal frequency division multiple access (OFDMA) functionality. Ucode does not support partial BW RU allocation in 160 MHz DL OFDMA.
get_he_dlofdma	**cfg80211tool athX get_he_dlofdma** **iwpriv athX get_he_dlofdma**	Retrieve the configured setting for high efficiency downlink OFDMA functionality.
he_ulmumimo **he_ulmimo**	**cfg80211tool athX he_ulmumimo** **iwpriv athX he_ulmimo**	Enable (1) or disable (0) high efficiency uplink MU-MIMO functionality. On the AP side, this parameter controls the full bandwidth UL-MIMO bit in HE PHY-CAP IE. On the STA side, HE control frames can be sent to AP to inform the AP side not to do any trigger for that client.
get_he_ulmumimo **get_he_ulmimo**	**cfg80211tool athX get_he_ulmumimo** **iwpriv athX get_he_ulmimo**	Retrieve the configured setting for high efficiency uplink MU-MIMO functionality.
he_ulofdma	**cfg80211tool athX he_ulofdma** **iwpriv athX he_ulofdma**	Enable (1) or disable (0) high efficiency uplink orthogonal frequency division multiple access (OFDMA) functionality. With 8 STAs and UL-OFDMA traffic, when OBSS is brought up in channel 100, HE20 mode, RU26 intolerance is disabled but basic trigger packet in UL-OFDMA is not displaying the information of 4 users. However, details pertaining to 6 or 7 users are displayed in basic trigger. This is expected behavior when UL MU-MIMO is enabled. When UL MU-MIMO is disabled, information of 4 users is correctly displayed.
get_he_ulofdma	**cfg80211tool athX get_he_ulofdma** **iwpriv athX get_he_ulofdma**	Retrieve the configured setting for high efficiency uplink OFDMA functionality.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
set_drop_threshold <mac_address> <tid_num> <update_mask> <threshold_value>	**cfg80211tool athX set_drop_threshold <mac_address> <tid_num> <update_mask> <threshold_value>**	Set MAC service data unit (MSDUQ) depth parameters. The command is used to provide the host support to enable setting MSDUQ depth parameters per TID per peer on a per-VAP basis. Parameters that are added through the command include MAC address, TID number, update mask and MSDU queue depth threshold value. The update mask is as follows: ■ Bit 0 - High Priority MSDU flow queue depth threshold needs to update ■ Bit 1 - Low Priority MSDU flow queue depth threshold needs to update ■ Bit 2 - UDP MSDU flow queue depth threshold needs to update ■ Bit 3 - Non-UDP MSDU flow qdepth threshold needs to update Rest of the bits are reserved.
en_txbf_snd <peer_mac_address> <enable/disable>	**cfg80211tool athX en_txbf_snd <peer_mac_address> <enable/disable>**	Enable or disable the setting of Tx beamforming sounding intervals.
set_su_sound <peer_mac_address> <sounding_interval in ms> **set_mu_sound <peer_mac_address> <sounding_interval in ms>**	**cfg80211tool athX set_su_sound <peer_mac_address> <sounding_interval in ms>** **cfg80211tool athX set_mu_sound <peer_mac_address> <sounding_interval in ms>**	Configure the two commands for setting the single-user Tx beamforming sounding interval and multi-user Tx beamforming sounding interval after enabling the Tx beamforming sounding interval capability (these commands will not work if en_txbf_snd is 0).
sched_mu_enable <peer_mac_address> <enable/disable> **sched_ofdma_enable <peer_mac_address> <enable/disable>**	**cfg80211tool athX sched_mu_enable <peer_mac_address> <enable/disable>** **cfg80211tool athX sched_ofdma_enable <peer_mac_address> <enable/disable>**	Enable/disable the FW scheduler API and statistics on a per peer basis. This operation is performed on two modes – OFDMA and Multiuser mode. Commands: ■ To enable or disable FW scheduler API in OFDMA mode—cfg80211tool athX sched_mu_enable <peer_mac_address> <enable/disable> ■ To enable or disable FW scheduler API in MU-MIMO mode—cfg80211tool athX sched_ofdma_enable <peer_mac_address> <enable/disable>
enable_11v_dms **g_enable_11v_dms**	**cfg80211tool athX enable_11v_dms** **cfg80211tool athX g_enable_11v_dms**	Enable/disable the directed multicast service (DMS) to enable a client to request the AP to convert multicast frames that match a certain traffic classifier into unicast frames for the client. This DMS configuration is a runtime setting. The get command retrieves the configured 802.11v DMS setting.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
he_txmcsmap map **get_he_ txmcsmap**	**iwpriv athN he_ txmcsmap map** **iwpriv athN get_he_ txmcsmap**	Sets the Tx modulation coding scheme number of spatial streams (mcsnss) map in AP beacon/probe resp/assoc resp. map is a 32 bit value. This command will also ensure that self-Tx mcsnss capability is intersected with the remote mcsnss capability and Tx does not exceed intersected mcs values for the given SS. The lower 16 bit values indicate the mcsnssmap for less than and equal to 80MHz and the higher 16 bit values indicate the same for greater than 80MHz (that is, 160 MHz and 80/80 MHz. Mcsnss map for 160MHz and 80/80 MHz are the same for lithium target). Each 16 bit value is set as per 11ax D2.0 spec Figure 9-589cn. For example: iwpriv athN he_txmcsmap 0x24922492 sets the tx mcsnss map for both <= 80MHz and > 80MHz as follows: 1 SS - 11, 2 SS - 7, 3 SS - 9, 4 SS - 11, 5 SS - 7, 6 SS - 9, 7 SS - 11, 8 SS - 7 For capping MCS to 7 for all SSs, use iwpriv athN he_txmcsmap 0x0. For capping MCS to 9 for all SSs, use iwpriv athN he_txmcsmap 0x55555555,
he_rxmcsmap map **get_he_ rxmcsmap**	**iwpriv athN he_ rxmcsmap map** **iwpriv athN get_he_ rxmcsmap**	Sets the Rx mcsnss map in AP beacon/probe resp/assoc resp. map is a 32 bit value. This command will also ensure that self rx mcsnss capability is intersected with the remote mcsnss capability. The lower 16 bit values indicate the mcsnssmap for less than and equal to 80MHz and the higher 16 bit values indicate the same for greater than 80MHz (i.e., 160 MHz and 80/80 MHz. Mcsnss map for 160MHz and 80/80 MHz are the same for lithium target). Each 16 bit value is set as per 11ax D2.0 spec Figure 9-589cn. For example: iwpriv athN he_rxmcsmap 0x24922492 sets the rx mcsnss map for both <= 80MHz and > 80MHz as follows: 1 SS - 11, 2 SS - 7, 3 SS - 9, 4 SS - 11, 5 SS - 7, 6 SS - 9, 7 SS - 11, 8 SS - 7 For capping MCS to 7 for all SSs, use iwpriv athN he_rxmcsmap 0x0. For capping MCS to 9 for all SSs, use iwpriv athN he_rxmcsmap 0x55555555.
ba_bufsize {0, 1} **get_ba_bufsize**	**cfg80211tool athN ba_bufsize {0, 1}** **cfg80211tool athN get_ba_bufsize**	Set BA buffer size in ADD BA Req, where: 0 - 64 1 - 255

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
he_frag level **get_he_frag**	**iwpriv athN he_frag level** **iwpriv athN get_he_frag**	Sets the level of dynamic fragmentation that is supported by a STA as a recipient, where level is: ■ 0 - for no support for dynamic fragmentation ■ 1 - for support of dynamic fragments that are contained within an MPDU or S-MPDU, no support for dynamic fragments within an A-MPDU that is not an S-MPDU. ■ 2 - for support of dynamic fragments that are contained within an MPDU or S-MPDU and support for up to one dynamic fragment for each MSDU, each A-MSDU if supported by the recipient, and each MMPDU within an A-MPDU or multi-TID A-MPDU that is not an MPDU or S-MPDU. ■ 3 - for support of dynamic fragments that are contained within an MPDU or S-MPDU and support for multiple dynamic fragments for each MSDU and for each A-MSDU if supported by the recipient within an A-MPDU or multi-TID AMPDU and up to one dynamic fragment for each MMPDU in a multi-TID A-MPDU that is not an S-MPDU. Note: Currently, Lithium targets support only 0 and 1 values for this command.
he_subfee {0,1} **g_he_subfee**	**iwpriv athN he_subfee {0,1}** **iwpriv athN g_he_subfee**	Enable (1) or disable (0) support for operation as an SU beamformee.
he_subfer {0,1} **g_he_subfer**	**iwpriv athN he_subfer {0,1}** **iwpriv athN g_he_subfer**	Enable (1) or disable (0) support for operation as an SU beamformer.
he_mubfee {0,1} **g_he_mubfee**	**iwpriv athN he_mubfee {0,1}** **iwpriv athN g_he_mubfee**	Enable (1) or disable (0) support for operation as an MU beamformee. Enable or disable MUBFEE support from STA side. In HE mode from STA side, it is not possible to disable through HE cap IE and instead it is tied up in Q-Q proprietary IE to inform AP side to disable MU MIMO DL to that Sta. For AP side, this command is not applicable.
he_mubfer {0,1} **g_he_mubfer**	**iwpriv athN he_mubfer {0,1}** **iwpriv athN g_he_mubfer**	Enable (1) or disable (0) support for operation as an MU beamformer.
set_ba_timeout	**cfg80211 wifiX set_ba_timeout**	Set Block Ack aging timeout values per access category.
get_ba_timeout	**cfg80211 wifiX get_ba_timeout**	Get Block Ack aging timeout values per access category.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
m_copy	**iwpriv athX m_copy <0 | 1 | 2>**	Enable (1) or disable (0) mirror copy mode for Tx packet log-capture mechanism. This command cannot be enabled when tx_capture mode or data_txstats has been enabled. m_copy and lite_monitor cannot be enabled simultaneously on the same or different VAP of the same radio. Starting with QCA_Networking_2020.SPF.11.3 release, enter the cfg80211tool athX m_copy 2 command to enable the additional mode for m_copy to send the first MSDU for each MPDU, which is also called extended m_copy mode. This mode retrieves the first MSDU for each MPDU in a PPDU for both Rx and enhances the current implementation of Mcopy mode.
get_m_copy	**iwpriv athX get_m_copy**	Displays whether mirror copy mode is enabled or disabled or Tx packet log-capture mechanism.
get_nominal_noisefloor	**cfg80211tool wifiX get_nominal_noisefloor**	Get indicative nominal noise floor value in dBm. This parameter is supported for both DA and OL chipsets. Note: This is not available with iwpriv command. This is currently intended for some specific channel selection related purposes. (Channel selection applications can access this information programmatically).
widebw_scan <0 | 1>	**iwpriv wifiX widebw_scan <0 | 1>**	Enables (1) or disables (0) the 40/80 MHz scan feature. This functionality is introduced in the QCA_Networking_2019.SPF.10.0 release on IPQ4019, IPQ8064, IPQ807x/IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx/QCN90xx or QCN602x + IPQ807xA/IPQ60xx/IPQ50xx, IPQ807xA, and IPQ817x platforms.
g_widebw_scan	**iwpriv wifiX g_widebw_scan**	Displays whether the 40/80 MHz scan feature is enabled.
set_mgmt_rssi_thr <0 | 1>	**iwpriv athX set_mgmt_rssi_thr <0 | 1>**	Enables (1) or disables (0) the rule to discard management frames below a configured RSSI threshold.
g_mgmt_rssi_thr	**cfg80211tool athX g_mgmt_rssi_thr**	Displays whether the RSSI threshold for management frames is enabled or not.
nss_rdprehdr	**iwpriv athX nss_rdprehdr**	Enables (1) or disables (0) the reading of Rx packet TLV preheader received from NSS to identify AMSDU boundaries
g_nss_rdprehdr	**iwpriv athX g_nss_rdprehdr**	Displays whether the reading of Rx packet TLV preheader is enabled or disabled.
he_sounding_mode	**cfg80211tool athX he_sounding_mode**	Enables (1) or disables (0) HE-sounding mode.
get_he_sounding_mode	**cfg80211tool athX get_he_sounding_mode**	Retrieves the HE-sounding mode.
he_su_sounding	**cfg80211tool athX he_su_sounding**	Switches between HE/VHT single-user sounding.
get_he_su_sounding	**cfg80211tool athX get_he_su_sounding**	Retrieves the single-user sounding mode.
he_su_trig_sounding	**cfg80211tool athX he_su_trig_sounding**	Switches between triggered and non-triggered single-user sounding when HE single-user sounding is selected.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
get_he_su_trig_ sounding	**cfg80211tool athX get_he_su_trig_ sounding**	Retrieves the sounding mechanism (triggered or not) when HE single-user sounding is selected.
he_mu_ sounding	**cfg80211tool athX he_mu_sounding**	Switches between HE/VHT multi-user sounding.
get_he_mu_ sounding	**cfg80211tool athX get_he_mu_sounding**	Retrieves the multi-user sounding mode.
he_su_trig_ sounding	**cfg80211tool athX he_su_trig_sounding**	Switches between triggered and non-triggered multi-user sounding when HE multi-user -sounding is selected.
get_he_su_trig_ sounding	**cfg80211tool athX get_he_su_trig_ sounding**	Retrieves the sounding mechanism (triggered or not) when HE multi-user sounding is selected.
g_curr_caprssi	**iwpriv athX g_curr_ caprssi**	Retrieves the RSSI of the CAP. This command is run on a range extender and applies for Wi-Fi SON functionality. Based on this CAP RSSI, a decision is made on whether the RE connects to CAP or not.
caprssi **g_caprssi**	**cfg80211tool athX athX caprssi** **cfg80211tool athX athX g_caprssi**	Configures and retrieves the RSSI of the CAP. This command is run on a range extender and applies for Wi-Fi SON functionality. Based on this CAP RSSI, a decision is made on whether the RE connects to CAP or not.
he_ht_ctrl {0, 1} **get_he_ht_ctrl**	**iwpriv athN he_ht_ ctrl {0, 1}** **iwpriv athN get_he_ ht_ctrl**	Enables(1) or disables(0) HE HT control support. The get command returns the current value.
he_mu_edca {0/1} **get_he_edca**	**cfg80211tool wifiN he_mu_edca {0/1}** **cfg80211tool wifiN get_he_edca**	Enables(1) or disables(0) MU-EDCA support on the AP.

Confidential – May Contain Trade Secrets 2021-06-13 03:[illegible]4 PDT pan.yafeng@zte.com.cn

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
set_ muedcaparams {paramID} {AC} {value} **get_ muedcaparams {paramID} {AC}**	**cfg80211tool athN set_muedcaparams {paramID} {AC} {value}** **cfg80211tool athN get_muedcaparams {paramID} {AC}**	The set_muedcaparams cmd sets the MUEDCA parameters. It takes the paramID, AC and value as the inputs. The valid range for the muedcaparams is listed below along with the paramID and AC values. The get parameter returns the current settings. paramID ■ 1: ECWmin [Valid Range 0 - 15] ■ 2: ECWmax [Valid Range 0 - 15] ■ 3: AIFSN [Valid Range 0 - 15] ■ 4: ACM [Valid values 0/1] ■ 5: MUEDCA timer [Valid Range 0 - 255] Access Category ■ 0: Best Effort ■ 1: Background ■ 2: Video ■ 3: Voice For example: Issue the command to set the ECWmin value for Best Effort to 13 as follows - #cfg80211tool ath0 set_muedcaparams 1 0 13 [paramID - 1(ECWmin); AC - 0(Best Effort); value: 13] #cfg80211tool ath0 get_muedcaparms 1 0 ath0 get_muedcaparams: 13

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:41 PM pan.yafeng@zte.com.cn

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
set_muedca_ecwmin {paramID} {AC} **get_muedca_ecwmin {AC}** **set_muedca_ecwmax {paramID} {AC}** **get_muedca_ecwmin {AC}** **set_muedca_aifsn {paramID} {AC}** **get_muedca_aifsn {AC}** **set_muedca_acm {paramID} {AC}** **get_muedca_acm {AC}** **set_muedca_timer {paramID} {AC}** **get_muedca_timer {AC}**	**cfg80211tool athN set_muedca_ecwmin {paramID} {AC}** **cfg80211tool athN get_muedca_ecwmin {AC}** **cfg80211tool athN set_muedca_ecwmax {paramID} {AC}** **cfg80211tool athN get_muedca_ecwmin {AC}** **cfg80211tool athN set_muedca_aifsn {paramID} {AC}** **cfg80211tool athN get_muedca_aifsn {AC}** **cfg80211tool athN set_muedca_acm {paramID} {AC}** **cfg80211tool athN get_muedca_acm {AC}** **cfg80211tool athN set_muedca_timer {paramID} {AC}** **cfg80211tool athN get_muedca_timer {AC}**	Each MUEDCA param can be executed individually, without using the set/get_muedcaparams as shown in the following example. The Access Category and value remain the same. Each set parameter has a corresponding get parameter that returns the current value. Valid ranges: ■ ECWmin 0-15 ■ ECWmax 0-15 ■ AIFSN 0-15 ■ ACM 0/1 ■ MUEDCA Timer 0-255 For example: The ecwmin parameter might be given as follows: #cfg80211tool ath0 muedca_ecwmin 0 14 #cfg80211tool ath0 get_muedca_ecwmin 0 ath0 get_muedca_ecwmin: 14

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
setratemask {preamble} {ratemasklower32} {ratemaskhigher32} {ratemasklower32_2}	**cfg80211tool athN setratemask {preamble} {ratemasklower32} {ratemaskhigher32} {ratemasklower32_2}**	Set MCS NSS rate mask. The rate mask variables are used to configure the maximum NSS MCS support on the AP. The mapping of the variables to the NSS MCS rate setting is as follows: Preamble → 0: CCK/OFDM, 1: HT, 2: VHT, 3: HE ratemasklower32 ==> b0 - b31 ratemaskhigher32 ==> b32 - b63 ratemasklower32_2 ==> b64 - b95 Each bit in the map corresponds to an MCS value and depending on the max MCS supported by the target family, x bits will correspond to one NSS setting as listed and illustrated with examples below: ■ HT: □ 8 bits per NSS. ■ VHT: □ 12 bits per NSS if proprietary 1024 QAM for VHT is enabled. □ 10 bits if proprietary 1024 QAM for VHT is disabled. ■ HE: □ 14 bits per NSS if 4K (4096) QAM feature is enabled. □ 12 bits if 4K (4096) QAM feature is disabled. 14 bits will correspond to one NSS setting for HE targets when 4K (4096) QAM is enabled. For example: b0-b13 will correspond to NSS1 and MCS0-13, b14-b27 will correspond to NSS2 and MCS 0-13 and so on. 12 bits will correspond to one NSS setting for VHT targets when proprietary 1024 QAM for VHT is enabled. For example: b0-b11 will correspond to NSS1 and MCS0-11, b12-b23 will correspond to NSS2 and MCS0-11 and so on. 10 bits will correspond to one NSS setting for legacy targets. For example: b0-b9 will correspond to NSS1 and MCS0-9, b10-b19 will correspond to NSS2 and MCS0-9 and so on. Certain targets support 4k QAM rates (MCS 12 and 13). For those targets, there are 14 bits per NSS, so b0-b13 will correspond to NSS 1, MCS 0-13.
advertise_sta_maxcap <0/1> **g_advertise_sta_maxcap**	**cfg80211tool athX advertise_sta_maxcap <0/1>** **cfg80211tool athX g_advertise_sta_maxcap**	■ cfg80211tool athX advertise_sta_maxcap <0/1>—Enable(1) or disable(0) the advertisement of the STA's maximum chwidth capabilities ■ cfg80211tool athX g_advertise_sta_maxcap—Current advertisement mode (0 – Disable/1 – Enable) This command is for the STA mode only and is enabled by default for the STA mode. On disabling this feature, the STA reverts to sending its chwidth capabilities, which are "dumbed-down" based on the AP from its beacon/probe responses. At the time of association with the STA, the AP receives capability information of the STA through the association request and stores it locally in the driver's node-specific structure.
he_ul_shortgi X **get_he_ul_shortgi**	**cfg80211tool athN he_ul_shortgi X** **cfg80211tool athN get_he_ul_shortgi**	Set the uplink short guard interval (GI) setting. It also configures the short GI parameter in the trigger frame. X: 0 - 800ns, 1 - 400ns, 2 - 1600ns, 3 - 3200ns This command is supported on IPQ807xA platforms only.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
he_ul_ltf X **get_he_ul_ltf**	**cfg80211tool athN he_ul_ltf X** **cfg80211tool athN get_he_ul_ltf**	Set the uplink long training field (LTF).setting. It also configures the LTF parameter in the trigger frame. Where X is: ■ 1 - 1x LTF (1x LTF = 3.2 us LTF) ■ 2 - 2x LTF (2x LTF = 6.4 us LTF) ■ 3 - 4x LTF (3x LTF = 12.8 us LTF) This command is supported on IPQ807xA platforms only.
he_ul_nss X **get_he_ul_nss**	**cfg80211tool athN he_ul_nss X** **cfg80211tool athN get_he_ul_nss**	Set the high-efficiency (HE) uplink NSS in the trigger frame. Valid range: 0 - 8 This command is supported on IPQ807xA platforms only.
he_ul_ppdu_bw X **get_he_ul_ppdu_bw**	**cfg80211tool athN he_ul_ppdu_bw X** **cfg80211tool athN get_he_ul_ppdu_bw**	Set the HE uplink PPDU bandwidth. Where X is: ■ 1 - 20MHz ■ 2 - 40MHz ■ 3 - 80MHz ■ 4 - 160MHz This command is supported on IPQ807xA platforms only.
he_ul_ldpc {0/1} **get_he_ul_ldpc**	**cfg80211tool athN he_ul_ldpc {0/1}** **cfg80211tool athN get_he_ul_ldpc**	Enable (1) or disable (0) the UL LDPC coding. It configures the trigger frame coding parameter. This command is supported on IPQ807xA platforms only.
he_ul_stbc {0/1} **get_he_ul_stbc**	**cfg80211tool athN he_ul_stbc {0/1}** **cfg80211tool athN get_he_ul_stbc**	Enable (1) or disable (0) uplink STBC. Use the get command to display the status of uplink STBC setting. This command is supported on IPQ807xA platforms only.
he_ul_mcs X **get_he_ul_mcs**	**cfg80211tool athN he_ul_mcs X** **cfg80211tool athN get_he_ul_mcs**	Sets the UL MCS setting in the trigger frame. Valid range: 0 - 11 This command is supported on IPQ807xA platforms only.
he_ul_trig_int X **get_he_ul_trig_int**	**cfg80211tool wifiN he_ul_trig_int X** **cfg80211tool wifiN get_he_ul_trig_int**	Sets the UL trigger interval in ms. Valid range: 0 - 65535 This command is supported on IPQ807xA platforms only.
csa2g **get_csa2g**	**cfg80211tool athX csa2g** **cfg80211tool athX get_csa2g**	Enable (1) or disable (0) CSA for 2.4 GHz devices. Use the get command retrieve the configured CSA value for 2.4 GHz devices.
g_conf_meshtx	**cfg80211tool athX g_conf_meshtx**	This command is used to check whether mesh mode is enabled (1) or disabled (0) for probe and disassoc Tx management frames.
qwrap_hk_war	**cfg80211tool athN qwrap_hk_war**	This command is used to enable (1) or disable (0) QWRAP workaround (WAR) value for IPQ807x/IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx/QCN90xx or QCN602x + IPQ807xA/IPQ60xx/IPQ50xx platforms.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
g_csl	**cfg802111tool ath0 csl <1 |0>**	Displays whether the connection state logging mechanism is enabled or not. Enabling CSL should enable IEEE80211_AUTH_MSG and IEEE80211_ASSOC_MSG debug flags.
nssol_inspect	**cfg802111tool ath0 nssol_inspect**	Enables (1) or disables (0) VAP inspection mode.
get_bandinfo	**cfg802111tool {ath0 | wifi0} get_bandinfo**	Retrieves the band information whether radio supports low, or high or full band frequency ranges.
s_txpow **g_Txpow**	**cfg802111tool athX s_ txpow** **cfg802111tool athX g_txpow**	Sets Tx power for control frames; enter a value between 0 and 255. The get command retrieves the configured Tx power for control frames.
rsn_override **g_rsn_override**	**cfg802111tool athX rsn_override** **cfg802111tool athX g_rsn_override**	Overrides the RSN IE value when this command is enabled; uses the RSN IE value sent by hostapd when this command is disabled. The get command retrieves the configured RSN IE override setting.
acs_debug_ enable	**cfg80211tool athX acs_debug_enable**	Enables (1) or disables (0) ACS stub for noise floor, tx_power, clr_cnt, and cycle_count.
g_dbr_ring_ status	**cfg80211tool {athX | wifiX} g_dbr_ring_ status**	Direct Buffer Rx uses software rings that is maintained between host and firmware to DMA the data from uCode. Use API to print the ring status per module per pdev and display it.
max_radio_sta **g_max_radio_sta**	**cfg80211tool wifiX max_radio_sta <1-51>** **cfg80211tool wifiX g_ max_radio_sta**	Sets the number of stations that can be connected to the AP; enter a value in the range of 1-51. The get command retrieves the configured setting.
isolation **get_isolation**	**cfg80211tool {athX | wifiX} isolation <0 | 1>** **cfg80211tool {athX | wifiX} get_isolation**	Enables (1) or disables (0) isolation mode for QWRAP. The get command retrieves the configured setting.
esp_period **g_esp_period**	**cfg80211tool {athX | wifiX} esp_period <0-5000>** **cfg80211tool {athX | wifiX} g_esp_period**	Sets the ESP periodicity value; enter a value in the range of 0-5000. The get command retrieves the configured setting.
esp_airtime **g_esp_airtime**	**cfg80211tool {athX | wifiX} esp_airtime <0-255>** **cfg80211tool {athX | wifiX} g_esp_airtime**	Sets the ESP airtime value; enter a value in the range of 0-255. The get command retrieves the configured setting.
esp_ppdu_dur **g_esp_ppdu_dur**	**cfg80211tool {athX | wifiX} esp_ppdu_dur <0-255>** **cfg80211tool {athX | wifiX} g_esp_ppdu_ dur**	Sets the ESP periodicity value; enter a value in the range of 0-255. The get command retrieves the configured setting.

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
esp_ba_window **g_esp_ba_window**	**cfg80211tool {athX | wifiX} esp_ba_window <1-7>** **cfg80211tool {athX | wifiX} g_esp_ba_window**	Sets the ESP BA window value; enter a value in the range of 1-7. The get command retrieves the configured setting.
cbs_csa **g_cbs_csa**	**cfg80211tool wifiX cbs_csa <0 | 1>** **cfg80211tool wifiX g_cbs_csa**	Enables (1) or disables (0) CBS channel switch announcement capability. The get command retrieves the configured setting.
set_retry_stats **get_retry_stats**	**cfg80211tool {athX | wifiX} set_retry_stats <0 | 1>** **cfg80211tool {athX | wifiX} get_retry_stats**	Enables (1) or disables (0) the retry statistics capability. The get command retrieves the configured setting.
dcs_sim <1-7>	**cfg80211tool {athX | wifiX} dcs_sim <1-7>**	Enables CW, WLAN, and AWGN interference. 1:CW interference 2: WLAN interference 3: Combination of CW and WLAN interference 4: AWGN interference (supported from QCA_Networking_2021.SPF.11.4 release onwards) 5: Combination of CW and AWGN interference (supported from QCA_Networking_2021.SPF.11.4 release onwards) 6: Combination of WLAN and AWGN interference (supported from QCA_Networking_2021.SPF.11.4 release onwards) 7: Combination of CW, WLAN, and AWGN interference (supported from QCA_Networking_2021.SPF.11.4 release onwards) The dcs_sim parameter accepts individual values of 0x1, 0x2 or 0x4, provided that they are enabled in the dcs_enable flag. The values for dcs_sim parameter can be entered either in decimal format or hexadecimal format (by prefixing the 0x notation before the number).
g_ch_util_nwifi	**cfg80211tool wifiX g_ch_util_nwifi**	Retrieves the channel utilization statistics for non-Wi-Fi traffic.
fc_video_stats	**cfg80211tool wifiX fc_video_statsi**	Retrieves the video/all TID counters framework for both Tx/Rx to trace packet loss.
set_block_mgmt	**cfg80211tool athX set_block_mgmt**	Enables block management frames for ACL entries.
clr_block_mgmt	**cfg80211tool athX clr_block_mgmt**	Disables block management frames for ACL entries.
get_block_mgmt	**cfg80211tool athX get_block_mgmt**	Checks whether block mgmt frames for ACL entries is set or not.
sifs_tr_rate **g_sifs_tr_rate**	**cfg80211tool athX sifs_tr_rate** **cfg80211tool athX g_sifs_tr_rate**	Sets the SIFS rate value The get command retrieves the SIFS rate value.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
ft **g_ft**	**cfg80211tool athX ft** **cfg80211tool athX g_ft**	Enables (1) or disables (0) STA mode VAP to perform Fast BSS Transition. The get command checks whether Fast BSS transition is enabled or disabled.
g_da_wds_war **da_wds_war**	**cfg80211tool athX g_da_wds_war** **cfg80211tool athX da_wds_war**	Enables (1) or disables (0) the direct-attach WDS workaround. The get command retrieves the configured direct-attach WDS workaround value.
display_me_info	**cfg80211tool athX display_me_info**	Displays a table showing the connected peer MAC addresses (destination MAC addresses) and their corresponding next-hop peer and a duplicate bit (set if the next-hop peer is already present in a previous entry of the table). This table is useful for debugging and analysis for the multicast enhancement mode 6 feature which utilizes the next-hop peer rather than the actual destination MAC address for transmission.
rawmode_open_war **g_rawmode_open_war**	**cfg80211tool athX rawmode_open_war** **cfg80211tool athX g_rawmode_open_war**	Enables (1) or disables (0) the raw mode open workaround. The get command retrieves the configured setting.
bcn_stats_clr	**cfg80211tool athX bcn_stats_clr**	Resets the beacon statistical counters.
set_peer_next_hop	**cfg80211tool athX set_peer_next_hop**	Sets the per-peer next hop in NSS offload mode.
nbr_scan_prd **g_nbr_scan_prd**	**cfg80211tool athX nbr_scan_prd** **cfg80211tool athX g_nbr_scan_prd**	Sets the neighbor AP scan period in seconds. The get command retrieves the specified neighbor AP scan period.
bsteerrssi_log	**cfg80211tool athX bsteerrssi_log**	Enables (1) or disables (0) band steering RSSI logs.
log_flush_timer_period	**cfg80211tool athX log_flush_timer_period**	Sets flush timer period for log; value must not be greater than 3600000. This command is under compilation flag WLAN_LOGGING_SOCK_SVC_ENABLE, which is disabled by default.
log_flush_one_time	**cfg80211tool athX log_flush_one_time**	Flushes out host logs for one time.
log_dump_at_kernel_enable	**cfg80211tool athX log_dump_at_kernel_enable**	Enables (1) logs at kernel level.
set_obss_ru_tolerance_time <value>	**cfg80211tool ath0 set_obss_ru_tolerance_time <value>**	Set the OBSS RU tolerance time in the range of 5-60 mins. An AP does not allocate RU26 (or equivalently other-sized Rus) in an uplink OFDMA PPDU if a RU26- intolerant overlapping basic service set (OBSS) is detected. dot11ObssNbRuToleranceTime: This attribute indicates the time in seconds when the last beacon frame carrying a Narrow Band Tolerance indication from an OBSS AP is received. The default value of this variable is 30 mins and the maximum is 1 hour. Disable RU26 allocation only for dot11ObssNbRuToleranceTime from the last beacon, which has RU26 intolerance. This command is supported on IPQ807xA and IPQ817x platforms only.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
get_obss_ru_ tolerance_time	**cfg80211tool ath0 get_obss_ru_ tolerance_time**	Retrieve the OBSS RU tolerance time. This command is supported on IPQ807xA and IPQ817x platforms only.
get_ru26_ tolerance	**cfg80211tool ath0 get_ru26_tolerance**	Retrieve the RU-26 intolerant OBBSS. This command is supported on IPQ807xA and IPQ817x platforms only.
ampduden_ovrd	**cfg80211tool athX ampduden_ovrd**	Configures the AMPDU density override. Enter a value in the range of 0-7.
g_ampduden_ ovrd	**cfg80211tool athX g_ ampduden_ovrd**	Displays the AMPDU density override.
set_ rxProtocolTag 0 <protocol_type_ enum> <16-bit tag>	**cfg80211tool <radio_ id> set_ rxProtocolTag 0 <protocol_type_ enum> <16-bit tag>**	Enable tagging packets based on protocol type. The capability to identify Rx Flows matching a specific criterion as programmed by the application is implemented. Support is implemented for tagging of Rx packets based on following protocol types: ■ Ethernet types: ARP (NS not required), EAP ■ IP protocol (v4 and v6): TCP, UDP, ICMP ■ UDP: DHCP, DNS When a packet matches the preceding criteria, the application is to be notified with an appropriate meta header/meta data. Tagging of packets should occur on both reorder (REO) ring and monitor rings. For example: cfg80211_tool wifi0 set_rxProtocolTag 0 0 XXXX / enables addition of Rx protocol type 0 (ARP).
set_ rxProtocolTag 0 <protocol_type_ enum> <16-bit tag>	**cfg80211tool <radio_ id> set_ rxProtocolTag 1 <protocol_type_ enum> <16-bit tag>**	Disable or delete tagging of packets based on protocol type. For example, cfg80211_tool wifi0 set_rxProtocolTag 1 1 deletes Rx protocol type 1. Tagging of protocol must be ordered in such a way that application layer protocol(s) tag are programmed first, followed by transport layer, network layer, and lower layer protocols. The protocols are enumerated in RX_PKT_PROTOCOL_TAG_TYPE data structure in the same fashion. The reason for this is because that multiple application layer protocols can be based off the same lower layer protocol (transport/network) and hardware applies the tag rules in the order of programming. Therefore, the correct programming order should be maintained to obtain the expected tag results.
dump_ rxProtocolTagStats <protocol_ type>	**cfg80211tool <radio_ id> dump_ rxProtocolTagStats <protocol_type>**	Obtain statistics for protocol type tags. If protocol_type is 255, the counters for all protocol types will be provided. If a specific protocol_type is provided, then only its counter will be provided. Per-protocol counters are not maintained for monitor ring. Counters are tracked only for REO ring. Example: cfg80211_tool wifi0 dump_rxProtocolTagStats 0 //Dump counters for protocol_type = 0 (ARP). cfg80211_tool wifi0 dump_rxProtocolTagStats 255 //Dump counters for all protocol type tags

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
he_amsdu_in_ampdu_supp <0-1> **get_get_he_amsdu_in_ampdu_supp**	**cfg80211tool athN he_amsdu_in_ampdu_supp <0-1>** **cfg80211tool athN get_get_he_amsdu_in_ampdu_supp**	Configure HE AMSDU in AMPDU MCS cap.
he_bfee_sts_supp <0-7> <0-7> **get_he_bfee_sts_supp**	**cfg80211tool athN he_bfee_sts_supp <0-7> <0-7>** **cfg80211tool athN get_he_bfee_sts_supp**	Configure beamformee STS <=80 and > 80 HE PHY CAP fields. Argument 1 is beamformee STS <=80 and argument 2 is beamformee STS > 80. The get command provides hexadecimal values for both fields.
he_4xltf_800nsgi_rx <0-1> **get_he_4xltf_800nsgi_rx**	**cfg80211tool athN he_4xltf_800nsgi_rx <0-1>** **cfg80211tool athN get_he_4xltf_800nsgi_rx**	Configure HE ER SU PPDU with 4x HE-LTF and 0.8 µs GI phy cap.
he_1xltf_800nsgi_rx <0-1> **get_he_1xltf_800nsgi_rx**	**cfg80211tool athN he_1xltf_800nsgi_rx <0-1>** **cfg80211tool athN get_he_1xltf_800nsgi_rx**	Configure HE SU PPDU with 1x HE LTF and 0.8 µs GI.
he_max_nc <0-7> **get_he_bfee_sts_supp**	**cfg80211tool athN he_max_nc <0-7>** **cfg80211tool athN get_he_bfee_sts_supp**	Configure HE maximum NC field. The get command retrieves the configured parameter.
twt_responder <0-1> **get_twt_responder**	**cfg80211tool athN twt_responder <0-1>** **cfg80211tool athN get_twt_responder**	Enable or disable twt_responder cap in HE mac cap and EXT cap.
rx_lite_monitor <1 | 2>	**cfg80211tool wifi0 rx_lite_monitor <1 | 2>**	Enable Rx packet capture enhancements on IPQ807xA and IPQ817x platforms. Customers require received (Rx) packets with 128 wireless header MPDU/MSDU, together with Rx information. Every user's MPDU is delivered to the user by registered callback. The callback function is passed with required Rx info and skb for 128 bytes MPDU header. The decoded 128 bytes of MSDUs skb for the MPDU is chained after 128 bytes MPDU. The ppdu is delimited by the ppdu id in the Rx info. Run "cfg80211tool wifi0 rx_lite_monitor 1" to enable enhanced Rx capture mode with 128 bytes MPDU header TLV. Run "cfg80211tool wifi0 rx_lite_monitor 2" to enable enhanced Rx capture mode with 128 bytes MPDU and MSDU header TLV. m_copy and lite_monitor cannot be enabled simultaneously on the same or different VAP of the same radio.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
tx_monitor <val>	**cfg80211tool wifiN tx_monitor <val>**	Enable Tx packet capture enhancements on IPQ807xA and IPQ817x, IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx/QCN90xx or QCN602x + IPQ807xA/IPQ60xx/IPQ50xx platforms. Customers require transmitted (Tx) packets with 128 wireless header MPDU/MSDU, together with Tx information. Block Ack frame contains acknowledgment for MPDU of up to 64/256(firmware support either 64 or 256). Block Ack Bitmap comes to host in bitmap TLV through HTT message. which needs to be processed to find failed MPDU. This feature available in sniffer mode. Enable/disable this Tx packet capture mode.This feature is not supported in NSS offload mode, and radiotap headers are not accurately captured. ■ 0 – Disable enhanced Tx monitor mode ■ 1– Enable enhanced Tx monitor mode
tx_lat_capture <0/1>	**cfg80211 ath0 tx_lat_capture <0/1>**	Enable (10) or disable (0) per-packet hardware timestamp after Tx completion feature. This functionality is supported on IPQ4019, IPQ8064, IPQ807x, IPQ807xA, IPQ817x, IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx/QCN90xx or QCN602x + IPQ807xA/IPQ60xx/IPQ50xx platforms.
vht_mcs_10_11_supp {0/1}	**cfg80211tool athN vht_mcs_10_11_supp {0/1}**	This command will enable/disable overall support for VHT MCS 10/11. It will also handle the advertisement of the IE. The maxphyrate displayed on the AP will be as per 1024 QAM rates. AP will advertise support for MCS 10/11 rates in the Q2Q IE and non Q clients will not see this information and assumes it supports only upto MCS9. By default, for the non Q clients, which do not include this Q2Q IE, AP will send MCS10/11 disabled in the peer assoc params and use 256 QAM for that client. However, a setting is available to use to enable probing with MCS10/11 for the non Q clients and based on response we will either continue with 1024 QAM rates or drop down to 256 QAM. The customer can issue the following two commands for AP to try 1024 QAM for non Q clients: ■ cfg80211tool athN vht_mcs_10_11_supp 1 ■ cfg80211tool athN vht_mcs_10_11_nq2q_peer_supp 1—This command ensures we try 1024QAM rates for clients who do not advertise the support in Q2Q IE
get_vht_mcs_10_11_supp	**cfg80211tool athN get_vht_mcs_10_11_supp**	Display whether support for VHT MCS 10/11 is enabled or not.
vht_mcs_10_11_nq2q_peer_supp {0/1}	**cfg80211tool athN vht_mcs_10_11_nq2q_peer_supp {0/1}**	If the cfg80211tool athN vht_mcs_10_11_supp {0/1} command is disabled, setting this command has no effect for any peer. If cfg80211tool athN vht_mcs_10_11_supp {0/1} command is enabled and peer advertises Q-Q IE, setting this has no effect for that peer and peer does not advertise Q-Q IE to us or if Q-Q peer advertise no support for VHT MCS10/11, then: If cfg80211tool athN vht_mcs_10_11_nq2q_peer_supp {0/1} command is enabled, then probe peer for MCS 10/11. If cfg80211tool athN vht_mcs_10_11_nq2q_peer_supp {0/1} command is disabled, then do not probe the peer. Use normal VHT.

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
get_vht_mcs_10_11_nq2q_peer_supp	**cfg80211tool athN get_vht_mcs_10_11_nq2q_peer_supp**	Retrieve the current configuration of VHT MCS 10/11 for peer advertising Q-Q IE.
cfg80211tool wifiN self_obss_pd_tx_threshold_db <non- srg | srg> <128 to 127> **cfg80211tool wifiN get_self_obss_pd_tx_threshold_db**	**cfg80211tool wifiN self_obss_pd_tx_threshold_db <non- srg | srg> <-128 to 127>** **cfg80211tool wifiN get_self_obss_pd_tx_threshold_db**	Set the radio-level OBSS-PD threshold for self spatial reuse in dB format. It takes two arguments. Arguments are as follows. Type of the OBSS PD Spatial Reuse i.e, "srg" or "non-srg". OBSS PD threshold value in dB for the Spatial Reuse type specified in the first argument. This value should be in the range of [-128 to 127]. This value will be added to noise floor to compute the OBSS PD level. **Note**: Starting from QCN90xx/QCN602x, HW reports RSSI in dBm format for Spatial Reuse whereas older chipsets use dB format. So, this command is applicable only on chipsets older than QCN90xx/QCN602x.
cfg80211tool wifiN self_obss_pd_tx_threshold_dbm <non- srg | srg> <128 to 127> **cfg80211tool wifiN get_self_obss_pd_tx_threshold_dbm**	**cfg80211tool wifiN self_obss_pd_tx_threshold_dbm <non- srg | srg> <-128 to 127>** **cfg80211tool wifiN get_self_ obss_pd_tx_threshold_dbm**	Set the radio-level OBSS-PD threshold for self spatial reuse in dBm format. It takes two arguments. Arguments are as follows. Type of the OBSS PD Spatial Reuse i.e, "srg" or "non-srg". OBSS PD threshold value in dBm for the Spatial Reuse type specified in the first argument. This value should be in the range of [-128 to 127]. **Note:** Starting from QCN90xx/QCN602x, HW reports RSSI in dBm format for Spatial Reuse whereas older chipsets use dB format. So, this command is applicable only on chipsets starting from QCN90xx/QCN602x.
cfg80211tool wifiN self_obss_pd_tx_enable <non- srg | srg> <0|1> **cfg80211tool wifiN get_self_obss_pd_tx_enable**	**cfg80211tool wifiN self_obss_pd_tx_enable <non- srg | srg> <0|1>** **cfg80211tool wifiN get_self_ obss_pd_tx_enable**	Enable (1) or disable (0) self OBSS PD transmissions at the radio level. It takes 2 arguments. Arguments are as follows. ■ Type of the OBSS PD Spatial Reuse i.e, "srg" or "non-srg". ■ Enable(1) or disable(0) self OBSS PD transmissions for the Spatial Reuse type specified in the first argument.
cfg80211tool wifiN self_srg_bss_color_bitmap <32 bits> <32 bits> **cfg80211tool wifiN get_self_srg_bss_color_bitmap**	**cfg80211tool wifiN self_srg_bss_color_bitmap <32 bits> <32 bits>** **cfg80211tool wifiN get_self_srg_bss_color_bitmap**	Set the BSS color bitmap to be used for the self SRG operation. Each bit represents a BSS color. BSS Colors corresponding to each set bit are present in the Spatial Reuse Group in which this AP is part of. It takes 2 arguments. Arguments are as follows. 1. Upper 32 bits of the bitmap 2. Lower 32 bits of the bitmap **Note:** If SRG based Spatial Reuse on a particular BSS color is disabled via 'self_srg_obss_color_enable_bitmap' command, then even though that BSS color is present in the Spatial Reuse Group, SRG based Spatial Reuse will not be performed on that particular BSS color.

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
cfg80211tool wifiN self_srg_partial_bssid_bitmap <32bits> <32 bits> **cfg80211tool wifiN get_self_srg_partial_bssid_bitmap**	**cfg80211tool wifiN self_srg_partial_bssid_bitmap <32bits> <32 bits>** **cfg80211tool wifiN get_self_srg_partial_bssid_bitmap**	Set the partial BSSID bitmap to be used for the SRG operation. Each bit represents a BSSID. BSSIDs corresponding to each set bit are present in the Spatial Reuse Group in which this AP is part of. It takes 2 arguments. Arguments are as follows. 1. Upper 32 bits of the bitmap 2. Lower 32 bits of the bitmap **Note**: If SRG based Spatial Reuse on a particular BSSID is disabled via 'self_srg_obss_bssid_enable_bitmap' command, then even if that BSSID is present in the Spatial Reuse Group, SRG based Spatial Resue will not be performed on that particular BSSID.
cfg80211tool wifiN self_hesiga_sr15_enable <0/1> **cfg80211tool wifiN get_self_hesiga_sr15_enable**	**cfg80211tool wifiN self_hesiga_sr15_enable <0/1>** **cfg80211tool wifiN get_self_hesiga_sr15_enable**	Configure AP to use PSR_AND_NON_SRG_OBSS_PD_PROHIBITED in the 'Spatial Reuse' field of HE-SIGA of HE SU PPDU, HE ER SU PPDU, and HE MU PPDU. When this value is set, other BSSes can't treat our transmissions as Spatial Reuse opportunities.
cfg80211tool wifiN self_srg_obss_color_enable_bitmap <32 bits> <32 bits> **cfg80211tool wifiN get_self_srg_obss_color_enable_bitmap**	**cfg80211tool wifiN self_srg_obss_color_enable_bitmap <32 bits> <32 bits>** **cfg80211tool wifiN get_self_srg_obss_color_enable_bitmap**	Set the BSS color enable bitmap to be used for the self SRG based Spatial Reuse operation. It takes 2 arguments. Arguments are as follows. 1. Upper 32 bits of the bitmap 2. Lower 32 bits of the bitmap If the bit corresponding to a BSS color is cleared, then SRG based Spatial Reuse will be disabled for that BSS color; otherwise SRG based Spatial Reuse will be enabled only if that BSS color is part of the Spatial Reuse Group in which this AP is part of.
cfg80211tool wifiN self_srg_obss_bssid_enable_bitmap <32 bits> <32 bits> **cfg80211tool wifiN get_self_srg_obss_bssid_enable_bitmap**	**cfg80211tool wifiN self_srg_obss_bssid_enable_bitmap <32 bits> <32 bits>** **cfg80211tool wifiN get_self_srg_obss_bssid_enable_bitmap**	Set the OBSS BSSID enable bitmap to be used for the self SRG based Spatial Reuse operation. It takes 2 arguments. Arguments are as follows. 1. Upper 32 bits of the bitmap 2. Lower 32 bits of the bitmap If the bit corresponding to a BSSID is cleared, then SRG based Spatial Reuse is disabled for that BSSID; otherwise SRG based Spatial Reuse is enabled only if that BSSID is part of the Spatial Reuse Group in which this AP is part of.

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
cfg80211tool wifiN self_non_srg_obss_color_enable_bitmap <32 bits> <32 bits> **cfg80211tool wifiN get_self_non_srg_obss_color_enable_bitmap**	**cfg80211tool wifiN self_non_srg_obss_color_enable_bitmap <32 bits> <32 bits>** **cfg80211tool wifiN get_self_non_srg_obss_color_enable_bitmap**	Set the BSS color enable bitmap to be used for the self Non-SRG based Spatial Reuse operation. It takes 2 arguments. Arguments are as follows. 1. lower 32 bits of the bitmap 2. upper 32 bits of the bitmap If the bit corresponding to a BSS color is cleared, then Non-SRG based Spatial Reuse is disabled for that BSS color; otherwise Non-SRG based Spatial Reuse is enabled for that BSS color.
cfg80211tool wifiN self_non_srg_obss_bssid_enable_bitmap <32 bits> <32 bits> **cfg80211tool wifiN get_self_non_srg_obss_bssid_enable_bitmap**	**cfg80211tool wifiN self_non_srg_obss_bssid_enable_bitmap <32 bits> <32 bits>** **cfg80211tool wifiN get_self_non_srg_obss_bssid_enable_bitmap**	Set the OBSS BSSID enable bitmap to be used for the self Non-SRG based Spatial Reuse operation. It takes 2 arguments. Arguments are as follows. 1. lower 32 bits of the bitmap 2. upper 32 bits of the bitmap If the bit corresponding to a BSSID is cleared, then Non-SRG based Spatial Reuse is disabled for that BSSID; otherwise Non-SRG based Spatial Reuse is enabled for that BSSID.
cfg80211tool wifiN he_srp_srg_obsspd_enable <0-1> <0-20> <0-20>	**cfg80211tool wifiN he_srp_srg_obsspd_enable <0-1> <0-20> <0-20>**	This command configures information related to SRG based OBSS PD in the IE. It takes 3 arguments. Arguments are as follows. 1. Set/clear the 'SRG Information Present' bit of the SR Control Field 2. Value for the 'SRG OBSS-PD Min Offset' field. Enter a value in the range of 0 through 20. 3. Value for the 'SRG OBSS-PD Max Offset' field. Enter a value in the range of 0 through 20. Per the 802.11ax spec, the SRG OBSSPD level is obtained by adding the offset value to -82 dBm
cfg80211tool wifiN get_he_srp_srg_obsspd_enable	**cfg80211tool wifiN get_he_srp_srg_obsspd_enable**	Get the current values of the SRG OBSS-PD SR parameters set in the IE.
cfg80211tool wifiN he_srp_srg_bss_color_bitmap	**cfg80211tool wifiN he_srp_srg_bss_color_bitmap**	Set the 'SRG BSS Color Bitmap' filed in the IE based on two 32 bit arguments. The first argument represents the upper 32bits of the bitmap and the second argument corresponds to the lower 32 bits of the bitmap.
cfg80211tool wifiN get_he_srp_srg_bss_color_bitmap	**cfg80211tool wifiN get_he_srp_srg_bss_color_bitmap**	Get the current value of SRG BSS Color Bitmap field set in the Spatial Reuse IE. Output will be two 32-bit values represented in Hex format where the first value represents the upper 32bits of the bitmap and the second value corresponds to the lower 32 bits of the bitmap.
cfg80211tool wifiN he_srp_srg_partial_bssid_bitmap <32 bits> <32 bits>	**cfg80211tool wifiN he_srp_srg_partial_bssid_bitmap <32 bits> <32 bits>**	Set the 'SRG Partial BSSID Bitmap' field in the Spatial Reuse IE based on two 32 bit arguments. The first argument represents the upper 32bits of the bitmap and the second argument corresponds to the lower 32 bits of the bitmap.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
cfg80211tool wifiN get_he_srp_srg_partial_bssid_bitmap	**cfg80211tool wifiN get_he_srp_srg_partial_bssid_bitmap**	Get the current value of SRG Partial BSSID Bitmap field set in the Spatial Reuse IE. Output will be two 32-bit values represented in Hex format where the first value represents the upper 32bits of the bitmap and the second value corresponds to the lower 32 bits of the bitmap.
cfg80211tool wifiN he_srp_psr_enable <0/1>	**cfg80211tool wifiN he_srp_psr_enable <0/1>**	Configures "PSR Disallowed" bit in the SR control field of the Spatial Reuse IE. ■ Enabling this parameter (he_srp_psr_enable 1) causes the "PSR Disallowed" bit to be set to 0 in the SR control field. ■ Disabling this setting (he_srp_psr_enable 0) causes the "PSR Disallowed" bit to be set to 1 in the SR control field.
cfg80211tool wifiN get_he_srp_psr_enable	**cfg80211tool wifiN get_he_srp_psr_enable**	Get the current value of he_srp_psr_enable parameter.
cfg80211tool wifi*N* he_srp_sr15_enable <0|1>	**cfg80211tool wifi*N* he_srp_sr15_enable <0|1>**	Enable (1) or disable (0) the "HESIGA_Spatial_Reuse_value15_allowed" bit in SR Control Field of the IE. Enabling this allows connected non-AP STA to use SRP_AND_NON_SRG_OBSS_PD_PROHIBITED in the Spatial Reuse field of HE-SIGA of HE SU PPDU, HE ER SU PPDU, and HE MU PPDU that they transmit.
cfg80211tool wifi*N* get_he_srp_sr15_enable	**cfg80211tool wifi*N* get_he_srp_sr15_enable**	Get the current state he_srp_sr15_enable parameter.
cfg80211tool wifi*N* he_srp_non_srg_obsspd_enable <0-1><0-20>	**cfg80211tool wifi*N* he_srp_non_srg_obsspd_enable <0-1><0-20>**	This command configures information related to Non-SRG based OBSS PD in the IE. It takes 2 arguments. Arguments are as follows. 1. Enable (1) or disable (0) use of Non-SRG OBSS-PD Spatial Reuse. Enabling this sets 'Non-SRG OBSS PD SR Disallowed' bit to 0 & 'Non SRG Offset Present' bit to 1. Disabling this sets 'Non-SRG OBSS PD SR Disallowed' bit to 1 & 'Non SRG Offset Present' bit to 0. 2. Value for the Non-SRG OBSS-PD Max Offset field. Enter a value in the range of 0 through 20. Per the 802.11ax spec, the SRG OBSSPD level is obtained by adding the offset value to -82 dBm
cfg80211tool wifi*N* get_he_srp_non_srg_obsspd_enable	**cfg80211tool wifi*N* get_he_srp_non_srg_obsspd_enable**	Get current state and value of the Non-SRG OBSS-PD SR parameters set in the IE.
mcast_rc_stale_period <val> **g_mcast_rc_stale_period**	**cfg80211tool athX mcast_rc_stale_period <val>** **cfg80211tool athX g_mcast_rc_stale_period**	Sets the multicast rate-control stale period in seconds in the target to support the multicast dynamic rate adjustment (DRA) algorithm across the highest common supported rates. A value greater than or equal to 0 is accepted and valid. Note: The FW design does not mandate the max-limit of the accepted values, but all values are checked to be greater than zero from the host. FW will add an upper-limit check themselves.
dec_bcn_loss <0 | 1>	**cg80211tool athX dec_bcn_loss <0 | 1>**	Enable (1) or disable (0) this setting to cause the beacon loss to be less if the user is ready to compromise with the scan results. Therefore, the passive scan dwell time is reduced.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
cck_tx_enable	**cfg80211tool wifiX cck_tx_enable**	Enables (1) or disables (0) Complementary Code Keying (CCK) modulation scheme for transmission of frames. The host sends the CCK enable/disable configuration to the firmware.
g_cck_tx_enable	**cfg80211tool wifiX g_cck_tx_enable**	Displays whether the Complementary Code Keying (CCK) modulation scheme for transmission of frames is enabled or not.
en_sae_pwid	**cfg80211tool athX en_sae_pwid**	Configures Simultaneous Authentication of Equals (SAE) password identifier.
g_en_sae_pwid	**cfg80211tool athX g_en_sae_pwid**	Displays the configured SAE password identifier.
get_rx_amsdu	**cfg80211tool athX get_rx_amsdu**	Retrieves the AMSDUs that are received.
rx_amsdu_ptid	**cfg80211tool athX rx_amsdu_ptid**	Configures the per-TID for received AMSDUs. Use the rx_amsdu_ptid parameter for disabling AMSDU in STA for uplink traffic on IPQ807x chipsets. Disabling AMSDU in STA for uplink traffic is only applicable to QCA980 and QCA9984 chipsets can be performed using the sst_ht_noamsdu parameter.
tx_lat_capture	**cfg80211tool athX tx_lat_capture**	Configures the latency capture for Tx packets.
map	**cfg80211tool athX map**	Enables the multiple AP (multi-AP) functionality in Wi-Fi SON networks. Multi-AP functionality is also called Easy Mesh.
get_map	**cfg80211tool athX get_map**	Displays whether multi-AP functionality is configured or not.
MapBSSType	**cfg80211tool athX MapBSSType**	Configures the type of BSS for multi-AP.
get_MapBSSType	**cfg80211tool athX get_MapBSSType**	Displays the type of BSS configured for multi-AP.
set_dpp_mode	**cfg80211tool athX set_dpp_mode**	Configures the Device Provisioning Protocol (DPP) capability mode for an AP. When encryption is set as DPP, the AP comes up in DPP mode.
get_dpp_mode	**cfg80211tool athX get_dpp_mode**	Retrieves the configured DPP mode for an AP.
get_whc_ul_snr	**cfg80211tool athX get_whc_ul_snr**	Retrieves the uplink signal-to-noise ratio for whole home coverage (Wi-Fi SON).
scsainteropauth	**cfg80211tool athX scsainteropauth**	Specifies the interoperable authentication for CSA.
gscsainteropauth	**cfg80211tool athX gscsainteropauth**	Retrieves the interoperable authentication for CSA.
scsainteropbss	**cfg80211tool athX scsainteropbss**	Specifies the interoperable BSS information for CSA.
gscsainteropbss	**cfg80211tool athX gscsainteropbss**	Retrieves the interoperable BSS information for CSA.
suniformrssi	**cfg80211tool athX suniformrssi**	Specifies a uniform RSSI.
guniformrssi	**cfg80211tool athX guniformrssi**	Retrieves the configured uniform RSSI.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
mapset_vapup	**cfg80211tool athX mapset_vapup**	Bring up or bring down the VAP for multi-AP or Easy Mesh environment in Wi-Fi SON networks.
mapget_vapup	**cfg80211tool athX mapget_vapup**	Displays whether the VAP is up or down for multi-AP or Easy Mesh environment in Wi-Fi SON networks.
mapget_espinfo	**cfg80211tool athX mapget_espinfo**	Retrieves the Estimated Service Parameters (ESP) information for multi-AP or Easy Mesh environment in Wi-Fi SON networks.
mapget_opchan	**cfg80211tool athX mapget_opchan**	Retrieves the operating channel details for multi-AP or Easy Mesh environment in Wi-Fi SON networks.
mapset_timeracl	**cfg80211tool athX mapset_timeracl**	Specifies the ACL timer for multi-AP or Easy Mesh environment in Wi-Fi SON networks.
mapset_rssi	**cfg80211tool athX mapset_rssi**	Specifies the RSSI for multi-AP or Easy Mesh environment in Wi-Fi SON networks.
mapget_ radiohwcap	**cfg80211tool athX mapget_radiohwcap**	Displays the radio hardware CAP information for multi-AP or Easy Mesh environment in Wi-Fi SON networks.
mapget_ clientcap	**cfg80211tool athX mapget_clientcap**	Displays the client CAP information for multi-AP or Easy Mesh environment in Wi-Fi SON networks.
caprssi	**cfg80211tool athX caprssi**	Configures the RSSI of the CAP.
g_caprssi	**cfg80211tool athX g_ caprssi**	Retrieves the RSSI of the CAP.
whc_mixedbh_ bh_type	**cfg80211tool athX whc_mixedbh_bh_ type**	Specifies the backhaul interface type for mixed backhaul environment in Wi-Fi SON networks.
whc_mixedbh_ul	**cfg80211tool athX whc_mixedbh_ul**	Specifies the uplink for mixed backhaul environment in Wi-Fi SON networks.
g_whc_ mixedbh_ul	**cfg80211tool athX g_ whc_mixedbh_ul**	Retrieves the configured uplink for mixed backhaul environment in Wi-Fi SON networks.
set_tidmap_tbl_ id	**set_tidmap_tbl_id cfg80211tool athX**	Specifies the TID map table ID.
get_tidmap_tbl_ id	**cfg80211tool athX get_tidmap_tbl_id**	Retrieves the TID map table ID.
set_vlan_type	**set_vlan_type**	Specifies the type of VLAN.
he_bsr_supp	**cfg80211tool athX he_bsr_supp**	Enables(1) or disables(0) HE buffer status report (BSR) support.
get_he_bsr_ supp	**cfg80211tool athX get_he_bsr_supp**	Displays the current status of HE buffer status report (BSR) support.
set_skip_hyst	**cfg80211tool athX set_skip_hyst**	Configures the best uplink hysteresis to be skipped.
get_skip_hyst	**cfg80211tool athX get_skip_hyst**	Displays whether the best uplink hysteresis is set to be skipped.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
hw_mode <mode> **get_hw_mode**	**cfg80211tool wifiX hw_mode <mode>** **cfg80211tool wifiX get_hw_mode**	Request driver to switch HW modes. Until the QCA_Networking_2019.SPF.10.0 release, the driver was initialized in either DBS or DBS_SBS mode, depending on the static configuration *hw_mode_id* in INI file. Starting with the QCA_Networking_2019.SPF.11.0 release, a new INI configuration *dynamic_hw_mode* differentiates the behavior from only using a static configuration. A value 0 of *dynamic_hw_mode* in INI file will ensure seamless execution of existing static way of HW mode selection. A value 1, on the other hand, will enable switching HW mode dynamically. After mode change command is processed in driver, an event is sent to user space. A function is invoked in 'wifi' script to notify the completion of mode change. The function brings up VAPs, and applies VAP parameters again from the wireless config file. Use the cfg80211tool wifiX get_hw_mode command to view the configured HW mode.
mcast_rc_stale_period <val>	**cfg80211tool athX mcast_rc_stale_period <val>**	Sets the multicast rate-control stale period in seconds in the target to support the multicast dynamic rate adjustment (DRA) algorithm across the highest common supported rates. A value greater than or equal to 0 is accepted and valid. Starting with QCA_Networking_2019.SPF.11.0 release, the feature for multicast dynamic rate adjustment (DRA) algorithm is introduced on IPQ807xA platforms only, which attempts to choose the highest common legacy rate for multicast.
g_mcast_rc_stale_period	**cfg80211tool athX g_mcast_rc_stale_period**	Retrieves the configured multicast rate-control stale period.
he_min_frag_size **get_he_min_frag_size**	**cfg80211tool athX he_min_frag_size** **<0-3>** **cfg80211tool athX get_he_min_frag_size**	Current value advertised (one) sets minimum fragment size to 128 bytes. Increasing the minimum fragment size can increase throughput by decreasing overhead in recombining the fragments. ■ 0: no minimum (not allowed) ■ 1: 128-bytes minimum ■ 2: 256-bytes minimum ■ 3: 512-bytes minimum
he_max_frag_msdu <0, 7> **get_he_max_frag_msdu**	**cfg80211tool athX he_max_frag_msdu <0,7>** **cfg80211tool athX get_he_max_frag_msdu**	Current advertised value is 0, which indicates pow(2,0)=1 fragment. Until HE Dynamic Fragmentation Support is > level 1, there can only be 1 fragment. Value of 7 indicates no restriction on number of fragments. Only one valid value until dynamic fragmentation capability level changes, and this command has capability check included. Set maximum number of fragmented MSDUs that a STA can receive at once where the number = power(2,value) and a value of 7 indicates no max number. The get command is used to retrieve current value of he_dynamic_frag_max_num.
he_omi <0,7> **get_he_omi**	**cfg80211tool athX he_omi <0,1>** **get_he_omi**	Only for STA mode. Enable or disable reception of OM Control subfield. The get command is used to retrieve current status of he_sta_omi field.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
he_multi_tid_aggr_rx **get_he_multi_tid_aggr_rx**	**cfg80211tool athX he_multi_tid_aggr_rx <0,7>** **cfg80211tool athX get_he_multi_tid_aggr_rx**	Set number of Traffic Identifier (TID) minus 1 Quality of Service (QoS) data frames that a HE STA can receive in a multi TID A-MPDU. The get command is used to retrieve the current value of he_multi_tid_aggr_rx.
he_multi_tid_aggr_tx **get_he_multi_tid_aggr_tx**	**cfg80211tool athX he_multi_tid_aggr_tx <0,7>** **cfg80211tool athX get_he_multi_tid_aggr_tx**	Set number of Traffic Identifier (TID) minus 1 Quality of Service (QoS) data frames that a HE STA can transmit in a multi TID A-MPDU. The get command is used to retrieve current value of he_multi_tid_aggr_tx.
he_max_ampdu_len_exp **get_he_max_ampdu_len_exp**	**cfg80211tool athX he_max_ampdu_len_exp <0,3>** **cfg80211tool athX get_he_max_ampdu_len_exp**	Set 2-bit value of the maximum AMPDU exponent extension value. The get command is used to retrieve current value of he_max_ampdu_len.
twt_responder **get_twt_responder**	**cfg80211tool athX twt_responder <0,1>** **cfg80211tool athX get_twt_responder**	Enable (1) or disable (0) twt_responder HE MAC Capability The get command is used to retrieve current value of twt_responder
he_bfee_sts_supp <0,7> <0,7> **get_he_bfee_sts_supp** **he_max_nc <0,7>** **get_he_max_nc**	**cfg80211tool athX he_bfee_sts_supp <0,7><0,7>** **cfg80211tool athX get_he_bfee_sts_supp** **cfg80211tool athX he_max_nc <0,7>** **cfg80211tool athX get_he_max_nc**	For HE Tx HBF, configure beamformee sts <=80 and > 80 HE PHY CAP fields. Argument 1 is beamformee sts <=80 and argument 2 is beamformee sts > 80. The get command is used to retrieve hex values for both fields. Set HE Max Number of Columns (NC) for HE compressed beamforming report. The get command is used to retrieve value of he_max_nc.
he_su_ppdu_1x_ltf_800ns_gi **get he_su_ppdu_1x_ltf_800ns_gi**	**cfg80211tool athX he_su_ppdu_1x_ltf_800ns_gi** **<0,1>** **cfg80211tool athX get he_su_ppdu_1x_ltf_800ns_gi**	Enable or disable reception of HE SU PPDU with 1x HE-LTF and 800ns GI in beacon. Get current status of reception of HE SU PPDU with 1x HE-LTF and 800ns GI in beacon. These commands will disable the 800ns capabilities for the associated Rx capabilities in the beacons only. Note that these commands do not disable the capability in HW. The commands for these capabilities will update the appropriate fields in the beacon/assoc response frames only. The AP's ability to receive fames containing the 800ns GI will not change, only the advertised capabilities will be updated with these commands. These are bits 14 and 58 in HE PHY capabilities field.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
he_su_mu_ppdu_4x_ltf_800ns_gi **get_he_su_mu_ppdu_4x_ltf_800ns_gi**	**cfg80211tool athX he_su_mu_ppdu_4x_ltf_800ns_gi <0,1>** **cfg80211tool athX get_he_su_mu_ppdu_4x_ltf_800ns_gi**	Enable or disable reception of HE SU MU PPDU with 1x HE-LTF and 800ns GI in beacon. Get current status of reception of HE SU MU PPDU with 1x HE-LTF and 800ns GI in beacon. These commands will disable the 800ns capabilities for the associated Rx capabilities in the beacons only. Note that these commands do not disable the capability in HW. The commands for these capabilities will update the appropriate fields in the beacon/assoc response frames only. The AP's ability to receive fames containing the 800ns GI will not change, only the advertised capabilities will be updated with these commands.
he_er_su_ppdu_1x_ltf_800ns_gi **get_he_er_su_ppdu_1x_ltf_800ns_gi**	**cfg8021tool athX he_er_su_ppdu_1x_ltf_800ns_gi** **cfg8021tool athX get_he_er_su_ppdu_1x_ltf_800ns_gi**	Enable or disable reception of HE ER SU PPDU with 1x HE-LTF and 800ns GI in beacon. Get current status of reception of HE ER SU PPDU with 1x HE-LTF and 800ns GI in beacon.
he_er_su_ppdu_4x_ltf_800ns_gi **get_he_er_su_ppdu_4x_ltf_800ns_gi**	**cfg8021tool athX he_er_su_ppdu_4x_ltf_800ns_gi** **cfg8021tool athX get_he_er_su_ppdu_4x_ltf_800ns_gi**	Enable or disable reception of HE ER SU PPDU with 4x HE-LTF and 800ns GI in beacon. Get current status of reception of HE ER SU PPDU with 4x HE-LTF and 800ns GI in beacon.
he_ndp_4x_ltf_3200ns_gi **get_he_ndp_4x_ltf_3200ns_gi**	**cfg8021tool athX he_ndp_4x_ltf_3200ns_gi** **cfg8021tool athX get_he_ndp_4x_ltf_3200ns_gi**	Enable or disable reception of NDP with 4x HE-LTF and 3200ns GI in beacon. Get current status of reception of NDP with 4x HE-LTF and 3200ns GI in beacon.

set_sa_rateX	**cfg80211tool wifiX set_sa_rateX**	To set Smart Antenna rate, enter the cfg80211tool wifiX set_sa_rateX command. Here, 01 = Radio ID, 00 = Reserved.
		`cfg80211tool wifiX set_sa_rateX 0x01000003 0x7f48c707`
		`[Here, 01 = Radio ID , 00 = Reserved]`
		`MAC: 00:03:7f:48:c7:07`

set_sa_rxant	**cfg80211tool wifiX set_sa_rxant**	To specify radio Rx antenna update, enter the cfg80211tool wifiX set_sa_rxant command.
		`cfg80211tool wifiX set_sa_rxant 0x01000003 0x7f48c707`
		`[Here, 01 = Radio ID , 00 = Reserved]`
		`MAC: 00:03:7f:48:c7:07`

set_sa_txant	**cfg80211tool wifiX set_sa_txant**	To specify radio Tx antenna update, enter the cfg80211tool wifiX set_sa_txant command.
		`cfg80211tool wifiX set_sa_rateX 0x01000003 0x7f48c707`
		`[Here, 01 = Radio ID , 00 = Reserved]`
		`MAC: 00:03:7f:48:c7:07`

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description

get_sa_nodeinfo	**cfg80211tool wifiX get_sa_nodeinfo**	To retrieve the Smart Antenna node information, enter the cfg80211tool wifiX get_sa_nodeinfo command.
		`cfg80211tool get_sa_nodeinfo 0x01000003 0x7f48c707`
		`[Here, 01 = Radio ID , 00 = Reserved]`
		`MAC: 00:03:7f:48:c7:07`

get_sa_rxant	**cfg80211tool wifiX get_sa_rxant**	To retrieve radio Rx antenna, enter the cfg80211tool wifiX get_sa_rxant command.
		`cfg80211tool set_sa_rateX 0x01000003 0x7f48c707`
		`[Here, 01 = Radio ID , 00 = Reserved]`
		`MAC: 00:03:7f:48:c7:07`

get_sa_txant	**cfg80211tool wifiX get_sa_txant**	To retrieve radio Tx antenna, enter the cfg80211tool wifiX get_sa_txant command.
		`cfg80211tool set_sa_rateX 0x01000003 0x7f48c707`
		`[Here, 01 = Radio ID , 00 = Reserved]`
		`MAC: 00:03:7f:48:c7:07`

set_sa_param	**cfg80211tool wifiX set_sa_param**	To enable SA debug prints, enter the cfg80211tool wifiX set_sa_param command.
multi_group_key <1 | 0>	**cfg8021tool ath0 multi_group_key <1 | 0>**	Enable (1) or disable (0) the capability for multiple passphrase with preshared keys on VLANs for WPA2. This capability is supported in WDS mode only. If VLAN bridges are created in root AP and repeater AP that functions as an extender AP, ping from root AP to the repeater AP/ext AP fails, which is expected.
mcast_rate_ control	**cfg80211tool athX mcast_rate_control** **cfg80211tool athX g_ mcast_rate_control**	Enable (1) or disable (0) FW multicast DRA functionality. The get command retrieves the configured setting.
oob_enable arg	**cfg80211tool wifiX oob_enable arg**	Enable (1) or disable (0) OOB feature for lower band APs to advertise collocated 6 GHz APs. Starting with QCA_Networking_2020.SPF.11.3 release, it is recommended to use the cfg80211tool wifiX 6ghz_rnr_collocated <enable/disable/driver mode> <frame_type> command instead of the oob_enable command, the usage details of which are described in the *11.0 WLAN driver parameters* section in this chapter.
bcast_prbrsp_en arg1 arg2	**cfg80211tool athX bcast_prbrsp_en arg1 arg2**	Enable/disable 20TU probe response with user defined period. Enabling 20TU PrbRsp disables FILS. arg1-{1/0} arg2 - {interval}
enable_fils arg1 arg2	**cfg80211tool athX enable_fils arg1 arg2**	Switch back to FILS from 20TU probe response mode arg1-{1/0} arg2 - {interval}
bcast_probersp_ en <value>	**cfg80211tool athX bcast_probersp_en <value>**	Enable 20 TU Probe Response

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
bcast_prb_rsp_intvl <value>	**cfg80211tool athX bcast_prb_rsp_intvl <value>**	Configure broadcast probe response timer
fils_en <value>	**cfg80211tool athX fils_en <value>**	Enable (1) or disable (0) FILS frame
oob_enable <value> **g_oob_enable**	**cfg80211tool wifiX oob_enable <value>** **cfg80211tool wifiX g_oob_enable**	Enable (1) or disable (0) OOB advertisement The get parameter retrieves the configured setting of OOB advertisement.
he_muedca_mode **get_he_muedca_mode**	**cfg80211tool wifiX he_muedca_mode** **cfg80211tool wifiX get_he_muedca_mode**	Enable (1) or disable (0) high efficiency (HE) enhanced distributed channel access (EDCA) mode Legacy EDCA favors the single-user (SU) operations in which the STA contends for channel for uplink (UL) transmissions. However, MU EDCA favors the multi-user (MU) operations in which the AP schedules UL transmissions for the STAs using a trigger. MU EDCA can be enabled only for the virtual device (vdev) that supports UL MU. ■ he_muedca_mode 0: Manual mode. User can manually specify the MU-EDCA parameters to use using other MU-EDCA commands (set_muedcaparams, muedca_ecwmin, muedca_ecwmax, etc) ■ he_muedca_mode 1: Host mode. The host dynamically controls the MU-EDCA configuration (deprecated). ■ he_muedca_mode 2: FW mode. The FW dynamically controls the MU-EDCA configuration (default). The get parameter retrieves the configured setting of HE EDCA mode.
acs_chan_grade_algo 0/1 **get_acs_chan_grade_algo**	**cfg80211tool wifiX acs_chan_grade_algo 0/1** **cfg80211tool wifiX get_acs_chan_grade_algo**	Starting with QCA_Networking_2020.SPF.11.1 release, the ACS optimization using target-provided channel grading metrics feature is implemented on IPQ401x, IPQ806x, IPQ807x/IPQ817x/IPQ807xA, and IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx/QCN90xx or QCN602x + IPQ807xA/IPQ60xx/IPQ50xx platforms. IPQ807x and later platform target SW might optionally provide channel characterization information via WMI to host at load time. This information will consist of a grade from 0-100 for each channel. The grading will be decided on a variety of PHY level factors applicable to the channel, and will give a relative indication to the host regarding expected performance degradation if the channel is used. Currently, this information will apply only if the channel is to be used as primary 20 MHz. If this advertisement is sent by target, the host auto channel selections algorithms (ACS and ICM) will factor in this information during channel selection (in addition to other metrics already being used). This will currently apply only to selection of primary 20 MHz. Default value is 1.
set_acs_chan_eff_var <value>	**cfg80211tool wifiX set_acs_chan_eff_var <value>**	Set the ACS channel efficiency variable. Value should be 0 to 1000. Default is 100.
get_acs_chan_eff_var	**cfg80211tool wifiX get_acs_chan_eff_var**	Retrieve the ACS channel efficiency variable:
set_acs_near_range_weightage <value>	**cfg80211tool wifiX set_acs_near_range_weightage <value>**	Set the ACS near-range weightage value. Value should be 0 to 100. Default is 50.

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
get_acs_near_range_weightage	**cfg80211tool wifiX get_acs_near_range_weightage**	Retrieve the ACS near-range weightage value.
set_acs_mid_range_weightage <value>	**cfg80211tool wifiX set_acs_mid_range_weightage <value>**	Set the ACS middle-range weightage value. Value should be 0 to 100. Default is 50.
get_acs_mid_range_weightage	**cfg80211tool wifiX get_acs_mid_range_weightage**	Retrieve the ACS middle-range weightage value.
set_acs_far_range_weightage <value>	**cfg80211tool wifiX set_acs_far_range_weightage <value>**	Set the ACS far-range weightage value. Value should be 0 to 100. Default is 50.
get_acs_far_range_weightage	**cfg80211tool wifiX get_acs_far_range_weightage**	Retrieve the ACS far-range weightage value.
scan_over_cac_en <1 | 0>	**cfg80211tool wifiX scan_over_cac_en <1 | 0>**	Before the implementation of this capability, during DFS wait time, if a scan is initiated, the DFS search will stop and then restart again after the scan is complete. The feature will be enabled only when this command is entered. By default, it is disabled. When the 'scan over CAC' feature is enabled for a dependent repeater scenario make sure that the channel switch option 'CSwOpts' 4 is set. This is enabled because when the scan is done over CAC, all the VAPs go down, scan happens and the VAPs are brought up. To ensure that CAC is done in the dependent repeater, the CSwOpts must be enabled.
tx_monitor 1	**cfg80211tool wifi0 tx_monitor 1**	Runtime flag on enable, tx monitor feature will turned on dynamically.
get_mbss_en	**cfg80211tool wifi0 get_mbss_en**	Displays whether the MBSS ID capability is enabled or not. The 802.11ax multiple basic service set ID (MBSS) information element (IE) feature adds MBSS Information Element to beacon and broadcast probe response frames. This feature does not support AP-STA connection.
hw_mode_omn_timer **get_hw_mode_omn_timer**	**cfg80211tool wifi0 hw_mode_omn_timer** **cfg80211tool wifi0 get_hw_mode_omn_timer**	Enable (1) or disable (0) the OMN timer. Once beacons are updated for all VAPs, start the OMN timer. The default value is 2 seconds. For the duration of this timeout period, OMN IE will be broadcast in the Beacon frame. When the timer fires, the callback function traverses through all the VAPs and removes the OMN IE, forcing another beacon template update. The get parameter retrieves the configured setting.
hw_mode_omn_enable **get_hw_mode_omn_enable**	**cfg80211tool wifi0 hw_mode_omn_enable** **cfg80211tool wifi0 get_hw_mode_omn_enable**	Enable (1) or disable (0) the Operating Mode Notification (OMN) capability. An AP must notify all associated STAs when the Spatial Stream and/or Channel Width configuration is changed. The notification should occur prior to a decrease in the maximum number of spatial streams and following an increase in the maximum number of spatial streams. The get parameter retrieves the configured setting.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
hw_mode_ primary **get_hw_mode_ primary**	**cfg80211tool wifi0 hw_mode_primary** **cfg80211tool wifi0 get_hw_mode_ primary**	Enable (1) or disable (0) the interface to be used as the primary interface during hardware mode change. Primary interface is an interface that stays up and does no go down when HW mode is changed. The FAST method needs a primary radio interface. The primary interface will stay UP while switching HW modes. For example, when wifi2 is the primary radio and we switch from DBS_SBS to DBS, wifi0 is brought DOWN and wifi2 will stay UP. New config called dynamic_hw_mode_primary_if describes the primary radio. In absence of this string in INI, wifi0 will be the used as primary interface. The get parameter retrieves the configured setting.
non_ht_dup_en **get_non_ht_ dup_en**	**cfg80211tool wifi0 non_ht_dup_en** **cfg80211tool wifi0 get_non_ht_dup_en**	Enable (1) or disable (0) the non-duplicate HT transmission. If the TXVECTOR parameter FORMAT is NON_HT and the TXVECTOR parameter NON_HT_MODULATION is NON_HT_DUP_OFDM, the transmitted PPDU is a non-HT duplicate. Non-HT duplicate transmission is used to transmit to non-HT OFDM STAs, HT STAs, VHT STAs and HE STAs that may be present in a part of a 40 MHz, 80 MHz, or 160 MHz channel. The RL-SIG, HE-SIG-A, HE-SIG-B, HESTF, and HE-LTF fields are not transmitted. The get parameter retrieves the configured setting.
mcs12_13_supp **g_mcs12_13_ supp**	**cfg80211tool athX mcs12_13_supp** **cfg80211tool athX g_ mcs12_13_supp**	Enable (1) or disable (0) the MCS12 and MCS13 support. When this parameter is enabled, 4K QAM support is activated on QCN90xx chipsets When this parameter is disabled, 4K QAM support is disabled on QCN90xx chipsets. The get parameter retrieves the configured setting.
rrm_capie **g_rrm_capie**	**cfg80211tool athX rrm_capie** **cg80211tool athX g_ rrm_capie**	Enable (1) or disable (0) the radio resource management (RRM) capability IE. The get parameter retrieves the configured setting.
map_sta_vlan **get_map_sta_ vlan**	**cfg80211tool athX map_sta_vlan** **cfg80211tool athX get_map_sta_vlan**	Enable (1) or disable (0) the Multi-AP capability on STA VLAN. The get parameter retrieves the configured setting.
set_opclass_tbl <value> **get_opclass_tbl**	**cfg80211tool wifi[x] set_opclass_tbl <value>** **cfg80211tool wifi[x] get_opclass_tbl**	To set the opclass table value (indicated by environment bit in country IE in beacons), the command used was "setCountry" where the third character following the first two country ISO characters represented this value. i.e., to set the opclass table value as 4 and country as US, the command to be used is - cfg80211tool wifi[x] setCountry US4. However, when this command is used in a device with country information flashed in memory, this command fails because setting country is not allowed by FW when the country value is already present in memory. A new command ("set_opclass_tbl") is introduced to set the opclass table index. Current use-case is to set the environment bit value in country IE in devices where the country information is already present in flash memory. Note: This command is to be immediately followed by "setCountryID" command for the opclass table value change to take its effect. Any "setCountry" after will overwrite the opclass table value with the value of the third character (null or otherwise).

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
rCACEn 1/0 **get_rCACEn**	**cfg80211tool wifiX rCACEn 1/0** **cfg80211tool wifix get_rCACEn**	To enable and disable Rolling CAC, enter cfg80211tool wifiX rCACEn 1/0 command. To get Rolling CAC (RCAC) config, enter cfg80211tool wifix get_ rCACEn command.
rcac_freq <value> **get_rCACfreq**	**cfg80211tool wifiX rcac_freq <value>** **cfg80211tool wifix get_rCACfreq**	To configure a RCAC frequency, enter cfg80211tool wifiX rcac_freq <value> command with value in MHz. To get RCAC configured frequency, enter cfg80211tool wifix get_ rCACfreq command.
setNxtRadarFreq <freq_value>	**cfg80211tool wifiX setNxtRadarFreq <freq_value>**	Configure the fallback frequency for advanced CAC implementation for MAPr1. The Controller is informed whenever there is a radar detection within the mesh so that it can take appropriate action in selecting another appropriate fronthaul or backhaul channel for the mesh. Wireless backhaul connectivity is maintained during radar detection on DFS channel. The fallback frequency must be configured separately for the driver and HYD.
igmpmcasten <1 | 0>	**cfg80211tool athX igmpmcasten <1 | 0>**	Enter the following command to enable the feature on a VAP: cfg80211tool athX igmpmcasten 1 Note: The feature can be enabled only when the general multicast enhancement enable command is enabled using the command: cfg80211tool athX mcastenhance <value> If general multicast enhancement feature is disabled, igmp multicast enhancement feature will also be disabled by default. Enter the following command to disable the feature on a VAP: cfg80211tool athX igmpmcasten 0
setpostNOLfreq **setpostNOLcfreq 2** **setpostNOLmode**	**cfg80211tool wifi0 setpostNOLfreq** **cfg80211tool wifi0 setpostNOLcfreq2** **cfg80211tool wifi0 setpostNOLmode**	Configure the settings for returning to the NOL expired channel To specify the primary frequency of the target channel to switch to, enter the cfg80211tool wifi0 setpostNOLfreq command To specify the secondary center frequency of the target channel to switch to, enter the cfg80211tool wifi0 setpostNOLcfreq2 command. To specify the mode (Bandwidth) in value (i.e., 20, 40, 80) of the target channel to switch to, enter the cfg80211tool wifi0 setpostNOLmode command.
rpt_max_phy 1/0 **get_mode**	**cfg80211tool wifiX rpt_max_phy 1/0** **cfgtool80211 athX get_mode**	Enter the cfg80211tool wifiX rpt_max_phy 1/0 command to enable or disable the capability of the AP side of a repeater to use the max PHY capabilities, although the repeater STA connects to a lower PHY-capable root AP. This functionality is also supported for DFS channels. By default, this feature is disabled. When the feature is enabled, all the VAPs are restarted. Enter the cfg80211tool wifiX g_rpt_max_phy command to display whether this capability is enabled or not. This feature changes the stats interpretation such as cfgtool80211 athX get_mode displays the max PHY mode configured among VAPs.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
set_innetwork_ 2g [mac] channel **get_innetwork_ 2g [0/1] [num_ entries/channel]**	**cfg80211tool athx set_innetwork_2g [mac] channel** **cfg80211tool athx get_innetwork_2g [0/1] [num_ entries/channel]**	To set in network information cfg80211tool athx set_innetwork_2g [mac] channel where x is 2.4G front haul interface, mac is the 2.4G MAC address and channel is 2.4G channel number. Starting with the QCA_Networking_2020.SPF.11.3 release, the set_ innetwork_2g [mac] channel parameter is available with the cfg80211tool, instead of the wifitool, where it was supported until the QCA_Networking_2020.SPF.11.2 release. However, the get command for this setting continues to exist with the wifitool. To get in network information There are two ways to get the in-network information. cfg80211tool athx get_innetwork_2g [0/1] [num_entries/channel] where x is 2.4G front haul interface By number of entries – user specifies the number of entries that needs to be displayed: wifitool athx get_innetwork_2g 0 num_entries By channel – displays all entries corresponding to the 2.4G channel specified: wifitool athx get_innetwork_2g 1 channel
he_6g_min_rate [1,206] **g_he_6g_min_ rate**	**cfg80211tool athX he_6g_min_rate [1,206]** **cfg80211tool athX g_ he_6g_min_rate**	Set/Unset 6G HE minimum rate in units of 1Mbps in 6G info of HE Operation IE Command applicable only for 6G VAPs Minimum rate value should be less than NSS 3 and MCS 3 setting. Considering this rate lmit per spec it should exceed 206 Mbps on 20 MHz BW. This command is to be applied to all VAPs (Non Tx VAPs followed by Tx VAP)
rnr_unsolicited _ prb_resp_en <1 | 0> **g_rnr_ unsolicited _prb_ resp_en**	**cfg80211tool wifiX rnr_unsolicited _prb_ resp_en <1 | 0>** **cfg80211tool wifiX g_ rnr_unsolicited _prb_ resp_en**	Set/Unset unsolicited probe response field in BSS params of TBTT infoset of RNR IE Command applicable only for 6 GHz radio only, which is collocated with legacy bands
rnr_member_ ess_collocated_ en <1 | 0> **g_rnr_member_ ess_collocated_ en**	**cfg80211tool wifiX rnr_member_ess_ collocated_en <1 | 0>** **cfg80211tool wifiX g_ rnr_member_ess_ collocated_en**	Set/Unset member ESS collocated field in BSS parameters of TBTT infoset of RNR IE Command applicable only for 6 GHz radio only, which is collocated with legacy bands

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description

selective_rnr_nontx 1/0	**cfg80211tool wifiX selective_rnr_nontx 1/0**	In 6 GHz frames RNR advertisement in “6Ghz-only AP” case in bcn/prb/fils frames. This advertisement to be selective advertisement of Non Tx VAPs. Only those Non Tx APs not in the MBSS IE of that frame to be present in RNR IE.
		This default behavior can be overridden by enabling “rnr_6ghz_driver_override=1” in ini/global.ini. Alternatively, use the following in /etc/config/wireless:
		`config qcawifi 'qcawifi’`
		`option rnr_6ghz_driver_override ‘1’`
		`wifi unload`
		`wifi load`
		This config will enable RNR IE advertisement in “Non 6Ghz-only AP” case.

6ghz_selective_nontx <1 | 0>	**cfg80211tool athX 6ghz_selective_nontx <1 | 0>**	Enable selective Non Tx AP inclusion in 6 GHz AP beacon based on MBSSID IE contents Command for 6 GHz radio only. Reject for lower bands In default mode, the get command returns 4 -> 100, i.e, B2 (3rd bit) is used for the selective RNR feature flag.

Qualcomm
Confidential – May Contain Trade Secrets
2021-05-13 03:57:44 PDT
pan.yafeng@zte.com.cn

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
6ghz_rnr_ collocated <enable/disable/ driver mode> <frame_type>	**cfg80211tool wifiX 6ghz_rnr_collocated <enable/disable/drive r mode> <frame_ type>**	Configure RNR IE based on frame type where: ■ 0 – Disable ■ 1 – Enable ■ 2 – Driver mode ■ frame_type: B0 - beacon, B1 - probe response (prbRsp), B2 - FILS/20TU probe response; Frame_type can be 1 to 7 (1 to 111) To enable beacon and probe response, enter the cfg80211tool wifiX 6ghz_rnr_collocated 1 3 command To disable beacon and probe response, enter the cfg80211tool wifiX 6ghz_rnr_collocated 0 3 command To enable all frames, enter the cfg80211tool wifiX 6ghz_rnr_collocated 1 7 command To switch back to driver based decision on RNR advertisement use, enter the cfg80211tool wifiX 6ghz_rnr_collocated 2 command To enable RNR IE advertisement in Beacon(B0) and Probe Resp(B1), enter the cfg80211tool wifiX rnr_6ghz_collocated 1 3 command To disable RNR in beacon and Probe response, enter the cfg80211tool wifiX rnr_6ghz_collocated 0 3 command cfg80211tool wifiX g_rnr_6ghz_collocated => 0x17 (In User mode, RNR enabled for all frames) cfg80211tool wifiX g_rnr_6ghz_collocated => 0x00 (In Driver mode) Upper 4 bits – 0- Driver mode, 1-Usr en/dis mode Lower 4 bits – Frame_type When the set command is issued as "cfg80211tool wifiX rnr_6ghz_ collocated 1 *frame_type*", The command retains the previous frame type config in driver. The set command applies new config in addition to keeping previous config in the driver. If the user requires only new config to be applied then old config has to be cleared Using disable command giving the old frame type to be disabled or using arg2 as 7 (clear all frame type cfg80211tool wifiX rnr_6ghz_collocated 0 7) before issuing new set command.
mbss_tx_vdev <1 | 0>	**cfg80211tool athX mbss_tx_vdev <1 | 0>**	To designate VDEV as Tx VDEV, use the following commands: cfg80211tool athX mbss_tx_vdev 1 -- to set VDEV as Tx VDEV cfg80211tool athX mbss_tx_vdev 0 -- to clear VDEV as Tx VDEV, This command is allowed, if that VDEV was configured as Tx VDEV and none of the VDEVs' netdev iface flag is set with IFF_UP. The Tx VDEV config also gets cleared on "ifconfig athX down" of the Tx VDEV.
mbss_auto <1 | 0>	**cfg80211tool wifiX mbss_auto <1 | 0>**	Auto mode can be enabled/disabled with the following commands cfg80211tool wifiX mbss_auto 1 -- to set AUTO mode to select Tx VDEV cfg80211tool athX mbss_auto 0 -- to clear AUTO mode This parameter is not thoroughly tested and not recommended to be used.
non_inherit_ enable 0/1	**cfg80211tool wifiN non_inherit_enable 0/1**	Enable/disable non-inheritance feature for EMA AP.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
dcs_wideband_ policy <0,1>	**cfg80211tool athX dcs_wideband_policy <0,1>**	Configure the DCS wideband policy. 0: DCS will select channels in the current band only. 1: DCS will select channels across both 5GHz and 6GHz channels.
set_triplets_bw <max_bw>	**cfg80211tool wifiX set_triplets_bw <max_bw>**	In case of wideband, continue to advertise sub-band triplets for 5Ghz channels. Use US4 for 2G, 5G and 6G in Country IE. Currently, 1 subband triplet sequence will be displayed as all the 59 channels are present in the current default 6G regdomain. The number of subband triplet sequence may vary according to the availability of the channels in the future 6G regdomains, Advertise additional operating triplets under API knob as per chip capability using the cfg80211tool wifiX set_triplets_bw <max_bw> command.
g_vdev_tsf	**cfg80211tool athX g_ vdev_tsf**	After bringing up radio interface (VDEV must be in 'VDEV_UP' state) and after the CAC timeout occurs (this command fails if it is run while timeout has not occurred), enter the cfg80211tool athX g_vdev_tsf command. If interface is up, the TSF is displayed; else, an error message is displayed. If interface does not exist, a user is returned to the console prompt. If no FW response is received within 50 ms of sending the event, then a message is displayed stating that the command failed.
he_er_su_ disable <1/0>	**cfg80211tool athX he_er_su_disable <1/0>**	Enable/disable ‘HE ER SU Disable’ subfield in HE Operation is available using the cfg80211tool athX he_er_su_disable <1/0> command. Enabling the capability knob will be allowed only on supported target platforms. Update ‘HE ER SU PPDU Rx’ from HE capabilities of ext service ready event to IC and VAP HE structures.
disable_fw_ inact_probe 1	**cfg80211tool athX disable_fw_inact_ probe 1**	FW Inactivity Probing can be disabled using the cfg80211tool athX disable_fw_inact_probe 1 command. Currently, Inactivity Timer implementation handles data frames only. Extend the Inactivity tracking to include the additional Keep-alive frames (according to the Max BSS Idle requirement). A command to configure ‘Protected Keep-Alive’ subfield for optional-PMF scenario, and to disable FW probing behavior are introduced. When Max BSS Idle Time feature is enabled, ‘inact’ command will be disallowed.
he_ul_mu_data_ dis_rx <1/0>	**cfg80211tool athX he_ul_mu_data_dis_ rx <1/0>**	Enable/disable this bit in HE capability IE is supported using the cfg80211tool athX he_ul_mu_data_dis_rx <1/0> command. Enabling the capability knob will be allowed only in supported target platforms.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
enable_mesh_ mode <enable/diable: 0..1> <color: 1..63>	**cfg80211tool wifiN enable_mesh_mode <enable/diable: 0..1> <color: 1..63>**	Two user-level arguments are introduced to pass the selected BSSCOLOR to host from user-level, and to inform host that mesh mode is active. Enter the following command to configure SR in mesh mode: cfg80211tool wifiN enable_mesh_mode <enable/diable: 0..1> <color: 1..63> Argument 1 sets wifiN radio in mesh mode if set to 1; disable otherwise. Argument 2 is optional and sets the BSS Color to chosen value in the range 1-63. Host will inform uCode to enable WAR that will disable BSSID mismatch based Spatial Reuse. Host will transmit beacon with Non-SRG OBSSPD disable for clients. Host will change the BSSCOLOR of the BSS to the BSSCOLOR value provided by user. Host SW on each Mesh node will set the “Non-SRG OBSS PD SR Disallowed” to 1. Based on 11ax standard, if “Non-SRG OBSS PD SR Disallowed” = 1, then “Non-SRG Offset Present” is NA.
print_rl <number_of_ prints> <time_ insecs>	**cfg80211tool athx print_rl <number_of_ prints> <time_ insecs>**	The QDF trace rate limit options are enhanced to consider no of prints and time as a parameter, such as the kernel implementation. Rate limiting is not impacted by the prints from other subsystem of the kernel. An appropriate threshold is reached for number of prints and time for rate limiting. Also, the number of prints and time for ratelimiting are displayed. Enter the cfg80211tool athx print_rl <number_of_prints> <time_insecs> command to specify the number of prints and the time for rate limiting in seconds.
set_chan_mode_ cfreq2_band <Channel-number> <phymode> <cfreq2> <band>	**cfg80211tool athX set_chan_mode_ cfreq2_band <Channel-number> <phymode> <cfreq2> <band>**	For disabling the auto-selection of PHY mode for invalid user space configuration, a new configuration parameter, "strict_channel_mode", is made available in the /ini/global.ini file. Setting this to "1" will enable the feature. When the feature is set, the driver does not implicitly change phymode when the user issues a channel change command (or vice-versa) With the feature set, when the user tries to change country (or regdomain) where the new country (or regdomain) does not support the existing channel/phymode, driver will reject the country and go back to the original country/channel. When all VAPs are turned down, the feature implicitly turns off. The recommended way to change country (or regdomain) when the feature is set, is to bring down all the VAPs, then change country (or regdomain) and bring up the VAPs. Also, the cfg80211tool athX set_chan_mode_cfreq2_band <Channel-number> <phymode> <cfreq2> <band> command is introduced, which accepts channel-number, phymode, cfreq2 and band in one command and set everything automatically.
low_latency_ mode <0 | 1>	**cfg80211tool wifi<X> low_latency_mode <0 | 1>**	Enable/disable latency mode in firmware

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
config_mesh_tid 0x<PQ>	**cfg80211tool athX config_mesh_tid 0x<PQ>**	Configure VDEV latency mesh TID configuration: cfg80211tool ath0 config_mesh_tid 0x<PQ> (Input in hex format) P = TID value Q = 1 (DL enable), 2(UL Enable) cfg80211tool ath0 config_mesh_tid 0x61 -------- > TID is 6, DL TID is enabled cfg80211tool ath0 config_mesh_tid 0x52 -------- > TID is 5, UL TID is enabled Note: “config_mesh_latency” must be applied after “config_mesh_tid” to send VDEV config to FW.
hlos_tidoverride <0 | 1>	**cfg80211tool athX hlos_tidoverride <0 | 1>**	Enable or disable TID override.
config_mesh_ latency <service interval> <burst size>	**cfg80211tool ath0 config_mesh_latency <service interval> <burst size>**	Set up VDEV latency configuration. cfg80211tool ath0 config_mesh_latency <service interval> <burst size> Service Interval of 10 miliseconds Burst Size of 100 bytes
cfg80211tool wifiX enable_ ema <0 | 1>	**cfg80211tool wifiX enable_ema <0 | 1>**	To disable EMA for 6GHz target band, execute the following command: cfg80211tool wifiX enable_ema 0 To enable EMA for 6GHz target band, execute the following command: cfg80211tool wifiX enable_ema 1 The command will be allowed for wideband radios only. Alternatively, to disable or e nable EMA, add 'option enable_ema 0/1' in the radiodevice section of the /etc/config/wireless file.
upload_pktlog 1	**cfg80211tool wifiX upload_pktlog 1**	The mechanism is introduced to collect the pktlog.dat file during a crash at the TFTP location itself. When system resets, either from firmware assert or a host crash, the log data of pktlog will transfer to host and parsed by host script to generate a log file, which is the same as copied file from pktlog’s proc/sys file for post processing. This can be implemented with minimum core-dump (minidump). Minidump, by specifying the memory virtual address and size of block, transfers the pktlog data in buffer and pktlog control block which maintain read and write position to host. The entire core dump with the pktlog data in pktlog buffer is loaded to host when system crashed. The host sends pktlog buffer start address, buffer size, read position and write position to FW. Core dump includes that information in firmware QDSS record. After processing, that information is used to generate pktlog file from full core dump. Set the upload pktlog option to 1 before starting the traffic.
user_rnr_frame_ sel **g_user_rnr_ frame_sel**	**cfg80211tool wifiX user_rnr_frame_sel** **cfg80211tool wifiX g_ user_rnr_frame_sel**	Configure frame enable/disable control for enhanced RNR for 6 GHz. Bit 0 for beacon Bit 1 for probe response Example: To enable in beacon, enter the cfg80211tool wifiX user_rnr_ frame_sel 1 command. To enable in beacon and probe, enter the cfg80211tool wifiX user_rnr_ frame_sel 3 command.

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
enable_mesh_mode <enable/disable: 0..1> <color: 1..63>	**cfg80211tool wifiN enable_mesh_mode <enable/disable: 0..1> <color: 1..63>**	Configure spatial reuse (SR) using two user-level attributes introduced to pass the selected BSSCOLOR to host from user-level, and to inform host that mesh mode is active. Argument 1 sets wifiN radio in mesh mode if set to 1; disable otherwise. Argument 2 is optional and sets the BSS Color to chosen value in the range 1-63.
dcs_random_chan_en <0/1> **g_dcs_random_chan_en**	**cfg80211tool athX dcs_random_chan_en <0/1>** **cfg80211tool athX g_dcs_random_chan_en**	Enable (1) or disable (0) selection between random channel and ACS.This command will control enable/disable random channel selection in the event of AWGN interference detection. If enabled, it will simply select a random channel. If disabled, it will force full ACS/ICM/CBS scanning and channel selection to take place. This command, currently, impacts AWGN interference only. To display whether the capability to select a random channel or ACS is configured, enter the cfg80211tool athX g_dcs_random_chan_en command.
dcs_csa_tbtt <val> **g_dcs_csa_tbtt**	**cfg80211tool athX dcs_csa_tbtt <val>** **cfg80211tool athX g_dcs_csa_tbtt**	To configure CSA beacon notification, enter the following command: cfg80211tool athX dcs_csa_tbtt <val>—This command controls the CSA TBTT (number of beacon advertisements to STAs before switching channel), if DCS triggers a CSA channel change to the new channel. Currently, this configuration impacts AWGN interference detection. No validation is performed on the value entered, which is less than a minimum of 2 beacons (that is, setting 0 or 1 will default to 2, and a value less than 2 cannot be processed. There is no upper limit). To display the configured CSA beacon notification, enter the cfg80211tool athX g_dcs_csa_tbtt command.
g_mbssid_ntx_max_pfl_size	**cfg80211tool wifiX g_mbssid_ntx_max_pfl_size**	Returns the maximum space available for the vendor and optional IEs of a non-tx profile in a beacon. These values will be used for determining the space free for vendor IEs and optional IEs to be populated by the user for the corresponding non-Tx profile.
vap_resource_profile <*vendor_ie_size*> <*optional_ie_size*>	**cfg80211tool athX vap_resource_profile <*vendor_ie_size*> <*optional_ie_size*>**	Allows user to configure vendor_ie and optional_ie section sizes per VAP. The WLAN driver honors the command only when all VAPs are in down state. Otherwise, the command is rejected. Here, the user will set their desired sizes for the Vendor IEs and Optional IEs for the given non-TX VAP. The sum of the sizes provided for this API must be less than the value returned by g_mbssid_non_tx_profile_size. This command provides the flexibility to update per-VAP resource profile configuration dynamically (when all VAPs are in down state). The values provided to this API will overwrite the existing values for the vendor and optional IE sizes for that non-Tx VAP.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-59 11.0 WLAN driver parameters (cont.)

Parameter	Command	Description
g_vap_resource_pfl <*ven_ie(0)/opt_ie(1)*>	**cfg80211tool athX g_vap_resource_pfl <*ven_ie(0)/opt_ie(1)*>**	Returns per-VAP resource profile by providing the current reservations of vendor_ie or optional_ie sections, based on user request.
g_mbssid_max_pp	**cfg80211tool wifiX g_mbssid_max_pp**	If VAPs are up on a given radio, this API will return the current max_pp for that radio. If no VAPs are up but TxVAP has been chosen, this API will calculate the new max_pp and return that value. If no VAPs are up and no TxVAP has been chosen, this API will return error saying TxVAP needs to be chosen before max_pp can be calculated.

3.64 Applying iwpriv commands using cfg80211tool

For cfg80211-based drivers, no generic applications such as iwpriv or iwlist are present to configure private commands. Therefore, Qualcomm Technologies developed an nl80211-based cfg80211tool to configure private commands. Input to the cfg80211tool is aligned with existing iwpriv commands; for example, command form and the number of arguments are as same as existing iwpriv commands.

NOTE Starting with WLAN driver, version 11.0, WEXT mode support is deprecated. Use the Linux 802.11 configuration API, cfg80211tool mode, instead. If a certain proprietary wireless tool command, such as `wlanconfig`, `wifitool`, or `apstats`, is entered, the driver checks the mode that is being currently used—WEXT or cfg80211—on the system, and processes the command accordingly. Parameters that are newly introduced, starting with the WLAN driver, version 11.0, can be configured only with cfg80211tool mode; only parameters that existed in WLAN driver, version 10.4 continue to be supported in WLAN driver, version 11.0, have iwpriv equivalent syntaxes.

Mapping of iwpriv commands with cfg80211tool commands

WEXT command	cfg80211 command
iwpriv athN <command> < arguments>	cfg80211tool athN <command> < arguments >
iwpriv wifiN <command> <arguments>	cfg80211tool wifiN <command> < arguments >

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

3.65 Additional protocol and radio layer parameters

Table 3-60 Additional protocol and radio layer commands

Parameter	Command	Description
addbaresp AID AC status	iwpriv athN addbaresp AID AC status	Sends an add block acknowledgment (addba) response frame on the indicated association ID (AID) and access controller (AC). The Association ID is the value shown under the AID column when using the wlanconfig list command. The status value is an 8 bit value indicating the status field of the response. This is normally used only during testing of the aggregation interface. The command does not have a corresponding get command, and also does not have a default value.
addbaresp AID AC	iwpriv athN addbaresp AID AC	Configures to send single VHT MPDU AMSDUs.
get_ countrycode	iwpriv wifiN get_ countrycode	Displays the current setting of the country code. The country code values are listed in *Appendix A, Country Code Definitions*.
extprotspac channelSpacing get_extprotspac (CWM mode)	iwpriv athN extprotspac channelSpacing iwpriv athN get_ extprotspac (CWM mode)	Sets the channel spacing for protection frames sent on the extension (secondary) channel when using 40 MHz channels. The following are the supported values. The default value is 20 MHz. 1: 20 MHz 2: 25 MHz This command is supported in DA mode only.
noedgech get_noedgech (IQUE)	iwpriv athN noedgech iwpriv athN get_noedgech (IQUE)	Forces the AP to avoid band edge channels while selecting a channel.
ampdulimit Byte Limit get_ampdulimit	iwpriv athN ampdulimit Byte Limit iwpriv athN get_ampdulimit	This is the same command as used in the radio layer; it affects all VAPs that are attached to the same radio. This parameter limits the number of bytes included in an AMPDU aggregate frame. Frames add to an aggregate until either the transmit duration is exceeded, the number of subframes is exceeded, the maximum number of bytes is exceeded, or the corresponding queue is empty. The subframe causing excess conditions is not included in the aggregate frame, but queues up to be transmitted with the next aggregate frame. The default value is 50000. This command is supported in DA mode only.
ampdusframes numFrames get_ ampdusframes	iwpriv athN ampdusframes numFrames iwpriv athN get_ ampdusframes	This is the same command as used in the radio layer. This command will affect all VAPs that are attached to the same radio. This command will set the maximum number of subframes to place into an AMPDU aggregate frame. Frames are added to an aggregate until a) the transmit duration is exceeded, b) the number of subframes is exceeded, c) the maximum number of bytes is exceeded, or d) the corresponding queue is empty. The subframe that causes excess conditions is not included in the aggregate frame, but will be queued up to be transmitted with the next aggregate frame. The default value is 32. This command has a corresponding get command.
amsdu isEnable get_amsdu	iwpriv athN amsdu isEnable iwpriv athN get_amsdu	Reception of ASMDU is supported by default. Transmission and reception of AMPDU is also supported by default. 0 - Disable AMSDU transmission 1 - Enable AMSDU transmission

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-60 Additional protocol and radio layer commands (cont.)

Parameter	Command	Description
dbgLVL Bitmask	iwpriv athN dbgLVL Bitmask	Another debug control. This parameter controls the debug level of the VAP based debug print statements. It is normally set to zero, eliminating all prints. **Symbolic name Bit Value Description** IEEE80211_MSG_11N 0x80000000 11n mode debug IEEE80211_MSG_DEBUG 0x40000000 IFF_DEBUG equivalent IEEE80211_MSG_DUMPPKTS 0x20000000 IFF_LINK2 equivalent IEE80211_MSG_CRYPTO 0x10000000 crypto work IEE80211_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 0x00000 10011.h IEEE80211_MSG_INACT 0x00000080 inactivity handling IEEE80211_MSG_ROAM 0x00000040 STA-mode roaming IEEE80211_MSG_ACTION 0x00000020 action management frames
rtscts_rcode (chainmask) get_rtscts_ rcode	iwpriv athN rtscts_rcode (chainmask) iwpriv athN get_rtscts_ rcode	This command set and get request to send (RTS) rate codes.
htprot 1|0 get_htprot	iwpriv athN htprot 1|0 iwpriv athN get_htprot	HT protection modes are defined in the 802.11n specification. Depending on conditions, various protection modes are implemented. This command will override automatic protection settings and enable protection for all modes. A value of 1 indicates all protection enabled, while a value of 0 indicates dynamically calculated protection levels. This command has a corresponding get command, and its default value is 0.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-60 Additional protocol and radio layer commands (cont.)

Parameter	Command	Description
vap_contryie isEnable (beacon config) get_vapcontryie	iwpriv athN vap_contryie isEnable (beacon config) iwpriv athN get_vapcontryie	Enables (1) or disables (0) country IE support of this VAP. This command is not supported in offload mode.
wdsdetect 1|0 (WDS) get_wdsdetect	iwpriv athN wdsdetect 1|0 (WDS) iwpriv athN get_wdsdetect	Enables (1) or disables (0) WDS. Due to a hardware bug in early 11n chips, a workaround for WDS frames was implemented between Qualcomm stations. For AR9000 series or later 802.11n products, this workaround is not required. This value enables (1) or disables (0) the AR5416 workaround for WDS. When the workaround is enabled, aggregation is not enabled for the WDS link. This command has a corresponding get command, and its default value is 1.
reset	iwpriv athN reset	Forces a reset on the VAP and its underlying radio layer. Note that any VAP connected to the same radio in mBSSID configuration will be affected. This is an action command that has no get command or default value.
mcastcipher cipher (security) get_ mcastcipher	iwpriv athN mcastcipher cipher (security) iwpriv athN get_ mcastcipher	Used mainly by the hostapd daemon, this command will set the cipher used for multicast. The iwpriv command sets the cipher type for the VAP. This is required to support operation of the host_apd authenticator. It has no default value, and the command has a corresponding get command. The value of cipher is one of the following:
		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
scanvalid period get_scanvalid	iwpriv athN scanvalid period iwpriv athN get_scanvalid	This command sets the period that scan data is considered value for roaming purposes. If scan data is older than this period, a scan will be forced to determine if roaming is required. A corresponding get command exists. This command is only supported for STA. Enter a value greater than 3000.
rssi11a rssiThreshold get_rssi11a	iwpriv athN rssi11a rssiThreshold iwpriv athN get_rssi11a	Sets roaming RSSI threshold for 11a BSS.
rssi11b rssi11g get_rssi11b get_rssi11g	iwpriv athN rssi11b iwpriv athN rssi11g iwpriv athN get_rssi11b iwpriv athN get_rssi11g	These commands set the RSSI threshold for roaming in 11g and 11b 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 units of db. These commands have corresponding get commands. The default value for both is 24 dBm.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-60 Additional protocol and radio layer commands (cont.)

Parameter	Command	Description
rate11a rate11b rate11g get_rate11a get_rate11b get_rate11g	iwpriv athN rate11a iwpriv athN rate11b iwpriv athN rate11g iwpriv athN get_rate11a iwpriv athN get_rate11b iwpriv athN get_rate11g	These commands set 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 500kbps increments, so a value of 48 indicates a rate of 24 Mbps. This command has a corresponding get command, and its default value is 48 for A band and 18 for B/G band.
inact_auth inactivity period get_inact_auth	iwpriv athN inact_auth inactivity period iwpriv athN get_inact_auth	This sets the TSPEC inactivity period for the AP AUTH state. This is an 802.11e mechanism that allows for allocating QoS priority to certain traffic types. The inactivity period is a timer that counts the seconds that a QoS stream is inactive during AUTH state. This timer will delete a traffic stream after the indicated number of seconds elapse. The default value is 180 seconds, and this command has a corresponding get command.
inact inactivity period get_inact	iwpriv athN inact inactivity period iwpriv athN get_inact	This sets the TSPEC inactivity period for the AP RUN state. This is an 802.11e mechanism that allows for allocating QoS priority to certain traffic types. The inactivity period is a timer that counts the seconds that a QoS stream is inactive during RUN state. This timer will delete a traffic stream after the indicated number of seconds elapse. The default value is 300 seconds, and this command has a corresponding get command. Traffic Specification (TSpec) allows an 802.11e client to signal its traffic requirements to the AP. In the 802.11e MAC definition, two mechanisms provide prioritized access. These are the contention-based EDCF option and the controlled access option provided by the transmit opportunity (TXOP). When describing TSPEC features where a client can specify its traffic characteristics, it is easy to assume that this would automatically result in the use of the controlled access mechanism, and have the client granted a specific TXOP to match the TSPEC request. However, this does not have to be the case; a TSpec request can be used to control the use of the various ACs in EDCF. Before a client can send traffic of a certain priority type, it must have requested to do so via the TSpec mechanism. For example, a WLAN client device wanting to use the voice AC must first make a request for use of that AC. Whether or not AC use is controlled by TSpec requests is configurable with voice and video ACs controlled by TSpec requests, and best-effort and background ACs can be open for use without a TSpec request. The use of EDCF ACs, rather than the 802.11e Hybrid Coordinated Channel Access (HCCA), to meet TSpec requests is possible in many cases because the traffic parameters are sufficiently simple to allow them to be met by allocating capacity, rather than creating a specific TXOP to meet the application requirements.
sleep 1|0 get_sleep	iwpriv athN sleep 1|0 iwpriv athN get_sleep	This test command forces a STA VAP into (1) or out of (0) sleep mode. This is useful only for station mode. When coming out of sleep, a null data frame will be sent. This command has a corresponding get command that returns the power management state (1 enabled 0 disabled). This has no default value.
qosnull	iwpriv athN qosnull	This command forces a QoS Null for testing.
shpreamble 1|0 get_shpreamble	iwpriv athN shpreamble 1|0 iwpriv athN get_ shpreamble	This command enables (1) or disables (0) short preamble. Short preamble will disable the use of a barker code at the start of the preamble. This command affects ALL VAPs connected to the same radio. This command has a corresponding get command, and its default value is 0.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-60 Additional protocol and radio layer commands (cont.)

Parameter	Command	Description
stafwd 1|0 get_stafwd	iwpriv athN stafwd 1|0 iwpriv athN get_stafwd	This command enables/disables station forwarding mode. In this mode, a client VAP will act as a surrogate interface as an Ethernet device, using the MAC address of the surrogate as its own, allowing a non-WiFi device to use a dongle to provide WiFi access without modification to the non-WiFi device. Setting to 1 will enable this mode, where setting to 0 will disable this mode. Note that the proper wlanconfig command must be used to set up the VAP in the first place. This command has a corresponding get command, and its default value is 0.
roaming mode\n get_roaming	iwpriv athN roaming mode\n iwpriv athN get_roaming	The roaming mode defines how state transitions are controlled in the AP, and what will cause a scan to happen. The default value is ROAMING_AUTO when in STA mode. This parameter has no meaning when operating in AP mode. The command has a corresponding get command. The roaming mode can take the following values: Value Definition 0 ROAMING_DEVICE. Scans are started in response to management frames coming in from the WLAN interface, and the driver starts the scan without intervention 1 ROAMING_AUTO. Scan algorithm is controlled by the 802.11 layer in the AP. Similar to ROAMING_DEVICE, additional algorithms are applied to the decision of when to scan/reassociate/roam. 2 ROAMING_MANUAL: Roaming decisions will be driven by IOCTL calls by external applications, such as the wpa_ supplicant.
ar isEnable get_ar	iwpriv athN ar isEnable iwpriv athN get_ar	Enable/disable Advanced Radar support, which is related to Dynamic Turbo.
scanprsleep value (Unassociated power consumption) get_ scanprsleep	iwpriv athN scanprsleep value (Unassociated power consumption) iwpriv athN get_ scanprsleep	Set the value of scan pre-sleep.
sleepprscan value get_ sleepprscan	iwpriv athN sleepprscan value iwpriv athN get_ sleepprscan	Sets the value of sleep pre-scan.
basicrates *basicRates*	iwpriv athN basicrates *basicRates*	Marks certain rates as basic rates according to user request.
ignore11d isEnable get_ignore11d	iwpriv athN ignore11d isEnable iwpriv athN get_ignore11d	Sets support to not process 11d beacons. 0—Disable the capability to not process 11d beacons 1—Enable the capability to not process 11d beacons

Table 3-60 Additional protocol and radio layer commands (cont.)

Parameter	Command	Description
vhtmcs <mcsindex> get_vhtmcs	iwpriv athN vhtmcs <mcsindex> iwpriv athN get_vhtmcs	This command specifies the VHT MCS Index to be used with data frame transmissions. Note that invoking this command with valid MCS Index (0-9) enables "fixed rate" and Invalid index disables fixed rate setting. Make sure that the chwidth and NSS parameters are set properly prior to invoking this command. MCS Index takes the following values: <0-9>—Valid Index; Enables Fixed Rate (>9)—Invalid index; Disables Fixed Rate For QCA9984, make sure that zero wait DFS is completed or disabled before the iwpriv athX vhtmcs command is issued. iwpriv ath1 nss 1 # Sets number of spatial streams iwpriv ath1 vhtmcs 0x1 -->VHT MCS1 iwpriv ath1 vhtmcs 0x2 -->VHT MCS2 iwpriv ath1 vhtmcs 0x3 -->VHT MCS3 iwpriv ath1 vhtmcs 0x4 -->VHT MCS4 iwpriv ath1 vhtmcs 0x5 -->VHT MCS5 iwpriv ath1 vhtmcs 0x6 -->VHT MCS6 iwpriv ath1 vhtmcs 0x7 -->VHT MCS7 iwpriv ath1 vhtmcs 0x8 -->VHT MCS8 iwpriv ath1 vhtmcs 0x9 -->VHT MCS9
nss <spatial_ streams> get_nss	iwpriv athN nss <spatial_ streams> iwpriv athN get_nss	This command specifies the number of Spatial Streams to be enabled. The Spatial Stream Count is 1-3. The default value is 3. For 11ax devices, the number of spatial streams can range from 1-8.
tx_stbc <0|1> (HT/VHT) get_tx_stbc	iwpriv athN tx_stbc <0|1> (HT/VHT) iwpriv athN get_tx_stbc	This command allows enabling/disabling of Tx STBC. Sets/unsets B18 of HE PHY CAP indicating support for the transmission of an HE PPDU that has a bandwidth less than or equal to 80 MHz using STBC. 0—Disable transmission of STBC 1—Enable transmission of STBC The default value is 1. For 11ax devices, sets/unsets B18 of HE PHY CAP indicating support for the transmission of an HE PPDU that has a bandwidth less than or equal to 80 MHz using STBC.
rx_stbc <0|1|2|3> get_rx_stbc	iwpriv athN rx_stbc <0|1|2|3> iwpriv athN get_rx_stbc	This command allows enabling/disabling of Rx STBC. Sets/unsets B19 of HE PHY CAP indicating support for the reception of an HE PPDU that has a bandwidth less than or equal to 80 MHz using STBC. 0—Disable Recv STBC 1—Enable Recv STBC (1) 2—Enable Recv STBC (2) 3—Enable Recv STBC (3) The default value is 1. For 11ax devices, sets/unsets B19 of HE PHY CAP indicating support for the reception of an HE PPDU that has a bandwidth less than or equal to 80 MHz using STBC.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-60 Additional protocol and radio layer commands (cont.)

Parameter	Command	Description
vht_txmcsmap <mcsmap> get_vht_ txmcsmap	iwpriv athN vht_txmcsmap <mcsmap> iwpriv athN get_vht_ txmcsmap	This command specifies the VHT TX MCS map to be used with VHT CAP advertisements. The 16 bits used to represent the map should be consistent with the Draft 3.1 11ac specification (Section 8.4.2.160.3 Figure 8.401bu-RX MCS MAP and TX MCS MAP)
vht_rxmcsmap <mcsmap> get_vht_ rxmcsmap	iwpriv athN vht_rxmcsmap <mcsmap> iwpriv athN get_vht_ rxmcsmap	This command specifies the VHT RX MCS map to be used with VHT CAP advertisements. The 16 bits used to represent the map should be consistent with the Draft 3.1 11ac specification (Section 8.4.2.160.3 Figure 8.401bu-RX MCS MAP and TX MCS MAP). The RX MCS Map parameter can be specified with one of the following values: ■ 0xfffc: NSS=1 MCS 0-7, NSS=2 not supported, NSS=3 not supported ■ 0xfff0: NSS=1 MCS 0-7, NSS=2 MCS 0-7, NSS=3 not supported ■ 0xffc0: NSS=1 MCS 0-7, NSS=2 MCS 0-7, NSS=3 MCS 0-7 ■ 0xfffd: NSS=1 MCS 0-8, NSS=2 not supported, NSS=3 not supported ■ 0xfff5: NSS=1 MCS 0-8, NSS=2 MCS 0-8, NSS=3 not supported ■ 0xffd5: NSS=1 MCS 0-8, NSS=2 MCS 0-8, NSS=3 MCS 0-8 ■ 0xfffe: NSS=1 MCS 0-9, NSS=2 not supported, NSS=3 not supported ■ 0xfffa: NSS=1 MCS 0-9, NSS=2 MCS 0-9, NSS=3 not supported ■ 0xffea: NSS=1 MCS 0-9, NSS=2 MCS 0-9, NSS=3 MCS 0-9 ■ 0xffda: NSS=1 MCS 0-9, NSS=2 MCS 0-9, NSS=3 MCS 0-8 ■ 0xffca: NSS=1 MCS 0-9, NSS=2 MCS 0-9, NSS=3
addba AID AC BufSize	iwpriv athN addba AID AC BufSize	These test commands are used to manually add or delete Block Acknowledge Aggregation streams. Note that automatic addba/delba processing must be turned off prior to using these commands (see setaddbaoper). Both commands require the AID (association ID) and the AC specified. The Association ID is the value shown when using the wlanconfig list command. When adding an aggregation link with addba, the BufSize parameter must be set to the maximum number of subframes that will be sent in an aggregate. When deleting an aggregation link, the initiator field indicates whether this link was initiated by the AP (1) or the remote station (0). The reason code is an 8-bit value indicating the reason the link was shut down. These commands have no corresponding get commands, nor do they have default values.
delba AID AC initiator reason	iwpriv athN delba AID AC initiator reason	These test commands are used to manually add or delete Block Acknowledge Aggregation streams. Note that automatic addba/delba processing must be turned off prior to using these commands (see setaddbaoper). Both commands require the AID (association ID) and the AC specified. The Association ID is the value shown when using the wlanconfig list command. When adding an aggregation link with addba, the BufSize parameter must be set to the maximum number of subframes that will be sent in an aggregate. When deleting an aggregation link, the initiator field indicates whether this link was initiated by the AP (1) or the remote station (0). The reason code is an 8-bit value indicating the reason the link was shut down. These commands have no corresponding get commands, nor do they have default values.

Table 3-60 Additional protocol and radio layer commands (cont.)

Parameter	Command	Description
cwmin AC Mode value get_cwmin	iwpriv athN cwmin AC Mode value iwpriv athN get_cwmin AC Mode value	This command sets the CWmin WMM parameter for either the AP or station parameter set. The cwmax command is a WMM command that must have the AC and Mode specified. The value is CWmin in units as described in the Access Categories and Modes table. This command has a corresponding get command, which requires the AC and mode to be specified.
cwmax AC Mode value get_cwmax	iwpriv athN cwmax AC Mode value iwpriv athN get_cwmax AC Mode value	This command sets the CWmax WMM parameter for either the AP or station parameter set. The cwmax command is a WMM command that must have the AC and Mode specified. The value is CWmax in units as described in the Access Categories and Modes table. This command has a corresponding get command.
aifs AC Mode Value get_aifs	iwpriv athN aifs AC Mode Value iwpriv athN get_aifs AC Mode value	This WMM command sets the AIFSN WMM parameter for either the AP or Station parameter set. This parameter controls the frame spacing in WMM operations. The command takes 3 parameters: The first value, AC, is the access class value. The second value indicates whether the command is to be applied to the AP or Station tables, which are kept separately. Finally, the third parameter is the AIFSN value (see the Access Categories and Modes table). This parameter has a corresponding get command, which requires the AC and Mode as arguments.
iwpriv athN txoplimit AC Mode limit iwpriv athN get_ txoplimit	iwpriv athN txoplimit AC Mode limit iwpriv athN get_txoplimit AC Mode value	This command will set the TXOP limit, described in the Access Categories and Modes table. The AC and Mode is as described at the beginning of this section. This command has a corresponding get command.
acm AC Mode value get_acm AC Mode value	iwpriv athN acm AC Mode value iwpriv athN get_acm AC Mode value	The ACM value for each access category is set using the acm command. The AC and Mode values must be set for this command. The value is the ACM value (see Access Categories and Modes table) for the specific access category. This command has a corresponding get command that returns the current setting for the indicated AC and mode.
noackpolicy AC Mode 0|1 get_noackpolicy	iwpriv athN noackpolicy AC *Mode <0|1>* iwpriv athN get_ noackpolicy AC Mode value	This command sets the No ACK policy bit in the WMM parameter set for either the AP or station. Where: ■ AC - Access category ■ 0 - The second parameter indicates BSS/local and has to be 0 for this command ■ X - 0 disable, 1 enable The noackpolicy command is a WMM command that must have the AC and mode specified. The value either sets the policy to no ACK sent (1) or send ACK (0). This command has a corresponding get command. The *Mode* parameter indicates BSS/local and has to be 0 for this command.
chainmasksel 1|0 get_ chainmasksel	iwpriv wifiN chainmasksel 1|0 iwpriv wifiN get_ chainmasksel	This command enables (1) automatic chainmask selection. This feature allows the system to select between 2 and 3 transmit chains, depending on the signal quality on the channel. For stations that are distant, using 3 chain transmit allows for better range performance. The default value of this parameter is 0 (disabled). This command has a corresponding get command. This command is supported in DA mode only.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-60 Additional protocol and radio layer commands (cont.)

Parameter	Command	Description
AMPDU 1|0 get_AMPDU	iwpriv wifiN AMPDU 1|0 iwpriv wifiN get_AMPDU	This is used to enable/disable transmit AMPDU aggregation for the entire interface. Receiving of aggregate frames will still be performed, but no aggregate frames will be transmitted if this is disabled. This has a corresponding get command, and the default value is 1 (enabled).
AMPDUFrames numFrames get_ AMPDUFrames	iwpriv wifiN AMPDUFrames numFrames iwpriv wifiN get_ AMPDUFrames	This command will set the maximum number of subframes to place into an AMPDU aggregate frame. Frames are added to an aggregate until either a) the transmit duration is exceeded, b) the number of subframes is exceeded, c) the maximum number of bytes is exceeded, or d) the corresponding queue is empty. The subframe that causes the excess conditions will not be included in the aggregate frame, but will be queued up to be transmitted with the next aggregate frame. The default value is 32. This command has a corresponding get command. This command is supported in DA mode only.
AMPDULim Byte Limit get_AMPDULim	iwpriv wifiN AMPDULim Byte Limit iwpriv wifiN get_ AMPDULim	This parameter will limit the number of bytes included in an AMPDU aggregate frame. Frames are added to an aggregate until either a) the transmit duration is exceeded, b) the number of subframes is exceeded, c) the maximum number of bytes is exceeded, or d) the corresponding queue is empty. The subframe that causes the excess conditions will not be included in the aggregate frame, but will be queued up to be transmitted with the next aggregate frame. The default value of this parameter is 50000. This command has a corresponding get command. This command is supported in DA mode only.

Qualcomm Confidential May Contain Trade Secrets 2021-05-13 pan.yafeng@zte.com.cn

Table 3-60 Additional protocol and radio layer commands (cont.)

Parameter	Command	Description
HALDbg debug level get_HALDbg	iwpriv wifiN HALDbg debug level iwpriv wifiN get_HALDbg	Used to set the debug level in the HAL code. This can be modified on the fly 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 to select from. Note that some categories will produce copious amounts of output, and should be used sparingly for a few seconds. HAL Debug Flags Symbolic Name Enable Bit Description and initialization HAL_DBG_RESET 0x00000001 Information pertaining to reset processing and initialization HAL_DBG_PHY_IO 0x00000002 PHY read/write states HAL_DBG_REG_IO 0x00000004 Register I/O, including all register values. Use with caution HAL_DBG_RF_PARAM 0x00000008 RF Parameter information, and table settings. HAL_DBG_QUEUE 0x00000010 Queue management for WMM support HAL_DBG_EEPROM_DUMP 0x00000020 Large dump of EEPROM information. System must be compiled with the EEPROM_DUMP conditional variable defined 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

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-60 Additional protocol and radio layer commands (cont.)

Parameter	Command	Description
HALDbg debug level get_HALDbg	iwpriv wifiN HALDbg debug level iwpriv wifiN get_HALDbg	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_UNMASKABLE 0xFFFFFFFF Will be printed in all cases if AH_DEBUG is defined. This command has a corresponding get command. Its default value is 0 (no debugging on), but this does not disable the unmaskable prints.
TXPwrOvr overrideLimit get_TXPwrOvr	iwpriv athN TXPwrOvr overrideLimit iwpriv athN get_TXPwrOvr	This command is used to override the transmit power limits set by iwconfig or the TXPowLim command. This is used for testing only, and is still limited by the regulatory power. A corresponding get command exists, and its default value is 0. This command is supported in DA mode only.
dyntxchain 1|0 get_dyntxchain	iwpriv wifiN dyntxchain 1|0 iwpriv wifiN get_dyntxchain	This command is used to override the dynamic txChainmask function on/off. When this function is turned on, Tx beamforming and STBC take priority. A corresponding get command exists, and its default value is 0.
bcnbwnssmap <1 | 0> get_ bcnbwnssmap	iwpriv athX bcnbwnssmap <1 | 0> iwpriv athX get_ bcnbwnssmap	Enable (1) or disable (0) bandwidth-NSS mapping in beacons. Use the get command to retrieve the configured value.
blbwnssmap <1 | 0> get_ blbwnssmap	iwpriv athX blbwnssmap <1 | 0> iwpriv athX get_ blbwnssmap	Enable (1) or disable (0) the advertisement of bandwidth-NSS mapping in STA mode. Use the get command to retrieve the configured value.

3.66 FTM debug parameters

Table 3-61 FTM debug parameters

Parameter	Command	Description
dl_modon 37 dl_modoff 37	iwpriv wifiX dl_modon 37 iwpriv wifiX dl_modoff 37	The FTM debug module can be turned on / off with the following commands. The debug module number for FTM is 37. iwpriv wifiX dl_modon 37 iwpriv wifiX dl_modoff 37 Logging debug statements in mission mode is available, in addition to the debugging in FTM in the form of firmware debug board with UART support and for IPQ401x chipsets. Also, the logging of debug statements is extended to FTM. With this functionality of WMI support for debug print statements in unified test firmware (UTF), DBGLOG_RECORD_LOG statements can be added in UTF in the same way as in mission mode so that the debug prints can be viewed on the same console. The method of operation is the same as in mission mode. The dbglog library is not included for UTF compilation at present; therefore, it has been included for this feature.
dl_loglevel <*level*>	iwpriv wifiX dl_loglevel <*level*>	Controls the firmware logging level. In the iwpriv wifiX dl_loglevel <0-6> command, 0 is the most verbose and 6 is the least verbose. The verbosity levels are the following: 0:NONE 1:FATAL 2:ERROR 3:WARN 4:INFO 5:INFO_HIGH 6:INFO_MED 7:INFO_LOW 8:DEBUG 9:ALL

g_qdf_cv_lvl,	iwpriv athX g_qdf_cv_lvl,	Retrieves the current debug levels. The command output will be in dmesg and not as part of the iwpriv output. The following is a sample output:
		`root@OpenWrt:/# iwpriv ath0 g_qdf_cv_lvl`
		`[ 815.029795] Verbose Level Legend - 0:NONE 1:FATAL 2:ERROR 3:WARN 4:INFO`
		`[ 815.029795] 5:INFO_HIGH 6:INFO_MED 7:INFO_LOW 8:DEBUG 9:ALL`
		`[ 815.029844] ----------`
		`[ 815.044265] | Module Name | Module Idx | Enabled | Verbose Level |`
		`[ 815.052323] ----------`
		`[ 815.451952] | tlshim 48(0x0030) 0 |`
		`[ 815.460079] | WMI 49(0x0031) 0 |`
		`[ 815.468234] | HTT 50(0x0032) 0 |`
		`[ 815.476423] | HDD 51(0x0033) 0 |`
		`[ 815.484553] | SME 52(0x0034) 0 |`
		`[ 815.492717] | PE 53(0x0035) 0 |`

3.67 Extended NSS signaling parameters

The existing 802.11 standard does not support negotiation of NSS as a function of channel bandwidth. As a result, certain workarounds are required to be used to advertise separate NSS capabilities for 80 MHz and 160 MHz modes. Recently, IEEE has made certain amendments to the standard called Extended NSS, which refers to signaling support for 160 and 80+80 MHz channel bandwidths with NSS support that is different from the maximum NSS signaled to legacy 80 MHz devices. OMN switching to and from 160/80+80 MHz is also a part of extended NSS signaling.

For more information on usage guidelines and the functionality regarding these parameters, see the *Extended NSS signaling* section of the *Networking offload and acceleration of link and multilink features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-62 Extended NSS signaling parameters

Parameter	Command	Description
ext_nss 0 g_ext_nss.	iwpriv athX ext_nss 0 iwpriv athX g_ext_nss.	Disables extended NSS signaling on the APs or STAs for the designated frequency used in the network topology. When extended NSS signaling is disabled, the "VHT Extended NSS BW Capable" Bit in the "Supported VHT-MCS and NSS Set" field of the VHT Capabilities element in the Beacon and Probe Response frame are equal to zero. The get command retrieves the configured value. EXT NSS signaling support becomes disabled if driver is not extended NSS-signaling capable.
ext_nss 1 g_ext_nss.	iwpriv athX ext_nss 1 iwpriv athX g_ext_nss.	Enables extended NSS signaling on the APs or STAs for the designated frequency used in the network topology. When extended NSS signaling is enabled, the "VHT Extended NSS BW Capable" Bit in the "Supported VHT-MCS and NSS Set" field of the VHT Capabilities element in the Beacon and Probe Response frame are equal to one. The get command retrieves the configured value. EXT NSS signaling support becomes disabled if driver is not extended NSS-signaling capable.
ext_nss_sup <1/0> g_ext_nss_sup	iwpriv athX ext_nss_sup <1/0> iwpriv athX g_ext_nss_sup	Enables/disables extended NSS-signaling support. The iwpriv athX g_ext_nss_sup command retrieves the extended NSS-signaling support; 1 if enabled, 0 if disabled.

3.68 Uncategorized protocol and radio layer parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *IEEE 802.11 features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-63 Uncategorized protocol and radio layer commands

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Parameter	Command	Description
addbaresp AID AC status	iwpriv athN addbaresp AID AC status	Sends an add block acknowledgment (addba) response frame on the indicated association ID (AID) and access controller (AC). The Association ID is the value shown under the AID column when using the wlanconfig list command. The status value is an 8 bit value indicating the status field of the response. This is normally used only during testing of the aggregation interface. The command does not have a corresponding get command, and also does not have a default value.
addbaresp AID AC	iwpriv athN addbaresp AID AC	Configures to send single VHT MPDU AMSDUs.
extprotspac channelSpacing get_extprotspac (CWM mode)	iwpriv athN extprotspac channelSpacing iwpriv athN get_ extprotspac (CWM mode)	Sets the channel spacing for protection frames sent on the extension (secondary) channel when using 40 MHz channels. The following are the supported values. The default value is 20 MHz. 1: 20 MHz 2: 25 MHz
noedgech get_noedgech (IQUE)	iwpriv athN noedgech iwpriv athN get_noedgech (IQUE)	Forces the AP to avoid band edge channels while selecting a channel.
ampdulimit Byte Limit get_ampdulimit	iwpriv athN ampdulimit Byte Limit iwpriv athN get_ampdulimit	This is the same command as used in the radio layer; it affects all VAPs that are attached to the same radio. This parameter limits the number of bytes included in an AMPDU aggregate frame. Frames add to an aggregate until either the transmit duration is exceeded, the number of subframes is exceeded, the maximum number of bytes is exceeded, or the corresponding queue is empty. The subframe causing excess conditions is not included in the aggregate frame, but queues up to be transmitted with the next aggregate frame. The default value is 50000. This command is supported in DA mode only.
ampdusframes numFrames get_ ampdusframes	iwpriv athN ampdusframes numFrames iwpriv athN get_ ampdusframes	This is the same command as used in the radio layer. This command will affect all VAPs that are attached to the same radio. This command will set the maximum number of subframes to place into an AMPDU aggregate frame. Frames are added to an aggregate until a) the transmit duration is exceeded, b) the number of subframes is exceeded, c) the maximum number of bytes is exceeded, or d) the corresponding queue is empty. The subframe that causes excess conditions is not included in the aggregate frame, but will be queued up to be transmitted with the next aggregate frame. The default value is 32. This command has a corresponding get command.
amsdu isEnable get_amsdu	iwpriv athN amsdu isEnable iwpriv athN get_amsdu	Reception of aggregated MSDUs (ASMDUs) is supported by default. Transmission and reception of AMPDU is also supported by default. 0 - Disable AMSDU transmission 1 - Enable AMSDU transmission Example: #iwpriv wifi0 amsdu 0—Disables AMSDU transmission #iwpriv wifi0 amsdu 1—Enables the AMSDUs transmission setting:

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Parameter	Command	Description
dbgLVL Bitmask	iwpriv athN dbgLVL Bitmask	Another debug control. This parameter controls the debug level of the VAP based debug print statements. It is normally set to zero, eliminating all prints. **Symbolic name Bit Value Description** IEEE80211_MSG_11N 0x80000000 11n mode debug IEEE80211_MSG_DEBUG 0x40000000 IFF_DEBUG equivalent IEEE80211_MSG_DUMPPKTS 0x20000000 IFF_LINK2 equivalent IEE80211_MSG_CRYPTO 0x10000000 crypto work IEE80211_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 0x00000 10011.h IEEE80211_MSG_INACT 0x00000080 inactivity handling IEEE80211_MSG_ROAM 0x00000040 STA-mode roaming IEEE80211_MSG_ACTION 0x00000020 action management frames
rtscts_rcode (chainmask) get_rtscts_ rcode	iwpriv athN rtscts_rcode (chainmask) iwpriv athN get_rtscts_ rcode	This command set and get request to send (RTS) rate codes.
htprot 1|0 get_htprot	iwpriv athN htprot 1|0 iwpriv athN get_htprot	HT protection modes are defined in the 802.11n specification. Depending on conditions, various protection modes are implemented. This command will override automatic protection settings and enable protection for all modes. A value of 1 indicates all protection enabled, while a value of 0 indicates dynamically calculated protection levels. This command has a corresponding get command, and its default value is 0.
vap_contryie isEnable (beacon config) get_vapcontryie	iwpriv athN vap_contryie isEnable (beacon config) iwpriv athN get_vapcontryie	Enables (1) or disables (0) country IE support of this VAP. This command is not supported in offload mode.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Parameter	Command	Description
wdsdetect 1|0 (WDS) get_Wdsdetect	iwpriv athN wdsdetect 1|0 (WDS) iwpriv athN get_Wdsdetect	Enables (1) or disables (0) WDS. Due to a hardware bug in early 11n chips, a workaround for WDS frames was implemented between Qualcomm stations. For AR9000 series or later 802.11n products, this workaround is not required. This value enables (1) or disables (0) the AR5416 workaround for WDS. When the workaround is enabled, aggregation is not enabled for the WDS link. This command has a corresponding get command, and its default value is 1.
reset	iwpriv athN reset	Forces a reset on the VAP and its underlying radio layer. Note that any VAP connected to the same radio in mBSSID configuration will be affected. This is an action command that has no get command or default value.
mcastcipher cipher (security) get_ mcastcipher	iwpriv athN mcastcipher cipher (security) iwpriv athN get_ mcastcipher	Used mainly by the hostapd daemon, this command will set the cipher used for multicast. The iwpriv command sets the cipher type for the VAP. This is required to support operation of the host_apd authenticator. It has no default value, and the command has a corresponding get command. The value of cipher is one of the following:
		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
rssi11a rssiThreshold get_rssi11a	iwpriv athN rssi11a rssiThreshold iwpriv athN get_rssi11a	Sets roaming rssi threshold for 11a bss.
rssi11b rssi11g get_rssi11b get_rssi11g	iwpriv athN rssi11b iwpriv athN rssi11g iwpriv athN get_rssi11b iwpriv athN get_rssi11g	These commands set the RSSI threshold for roaming in 11g and 11b 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 units of db. These commands have corresponding get commands. The default value for both is 24 dBm.
rate11a rate11b rate11g get_rate11a get_rate11b get_rate11g	iwpriv athN rate11a iwpriv athN rate11b iwpriv athN rate11g iwpriv athN get_rate11a iwpriv athN get_rate11b iwpriv athN get_rate11g	These commands set 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 500kbps increments, so a value of 48 indicates a rate of 24 Mbps. This command has a corresponding get command, and its default value is 48 for A band and 18 for B/G band.
inact_auth inactivity period get_inact_auth	iwpriv athN inact_auth inactivity period iwpriv athN get_inact_auth	This sets the TSPEC inactivity period for the AP AUTH state. This is an 802.11e mechanism that allows for allocating QoS priority to certain traffic types. The inactivity period is a timer that counts the seconds that a QoS stream is inactive during AUTH state. This timer will delete a traffic stream after the indicated number of seconds elapse. The default value is 180 seconds, and this command has a corresponding get command.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Parameter	Command	Description
inact inactivity period get_inact	iwpriv athN inact inactivity period iwpriv athN get_inact	This sets the TSPEC inactivity period for the AP RUN state. This is an 802.11e mechanism that allows for allocating QoS priority to certain traffic types. The inactivity period is a timer that counts the seconds that a QoS stream is inactive during RUN state. This timer will delete a traffic stream after the indicated number of seconds elapse. The default value is 300 seconds, and this command has a corresponding get command. Traffic Specification (TSpec) allows an 802.11e client to signal its traffic requirements to the AP. In the 802.11e MAC definition, two mechanisms provide prioritized access. These are the contention-based EDCF option and the controlled access option provided by the transmit opportunity (TXOP). When describing TSPEC features where a client can specify its traffic characteristics, it is easy to assume that this would automatically result in the use of the controlled access mechanism, and have the client granted a specific TXOP to match the TSPEC request. However, this does not have to be the case; a TSpec request can be used to control the use of the various ACs in EDCF. Before a client can send traffic of a certain priority type, it must have requested to do so via the TSpec mechanism. For example, a WLAN client device wanting to use the voice AC must first make a request for use of that AC. Whether or not AC use is controlled by TSpec requests is configurable with voice and video ACs controlled by TSpec requests, and best-effort and background ACs can be open for use without a TSpec request. The use of EDCF ACs, rather than the 802.11e Hybrid Coordinated Channel Access (HCCA), to meet TSpec requests is possible in many cases because the traffic parameters are sufficiently simple to allow them to be met by allocating capacity, rather than creating a specific TXOP to meet the application requirements.
sleep 1|0 get_sleep	iwpriv athN sleep 1|0 iwpriv athN get_sleep	This test command will force a STA VAP into (1) or out of (0) sleep mode. This is useful only for station mode. When coming out of sleep, a null data frame will be sent. This command has a corresponding get command that returns the power management state (1 enabled 0 disabled). This has no default value.
qosnull	iwpriv athN qosnull	This command forces a QoS Null for testing.
shpreamble 1|0 get_shpreamble	iwpriv athN shpreamble 1|0 iwpriv athN get_ shpreamble	This command will enable (1) or disable (0) short preamble. Short preamble will disable the use of a barker code at the start of the preamble. This command affects ALL VAPs connected to the same radio. This command has a corresponding get command, and its default value is 0.
stafwd 1|0 get_stafwd	iwpriv athN stafwd 1|0 iwpriv athN get_stafwd	This command enables/disables station forwarding mode. In this mode, a client VAP will act as a surrogate interface as an Ethernet device, using the MAC address of the surrogate as its own, allowing a non-WiFi device to use a dongle to provide WiFi access without modification to the non-WiFi device. Setting to 1 will enable this mode, where setting to 0 will disable this mode. Note that the proper wlanconfig command must be used to set up the VAP in the first place. This command has a corresponding get command, and its default value is 0.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Parameter	Command	Description
roaming mode\n get_roaming	iwpriv athN roaming mode\n iwpriv athN get_roaming	The roaming mode defines how state transitions are controlled in the AP, and what will cause a scan to happen. The default value is ROAMING_AUTO when in STA mode. This parameter has no meaning when operating in AP mode. The command has a corresponding get command. The roaming mode can take the following values: Value Definition 0 ROAMING_DEVICE. Scans are started in response to management frames coming in from the WLAN interface, and the driver starts the scan without intervention 1 ROAMING_AUTO. Scan algorithm is controlled by the 802.11 layer in the AP. Similar to ROAMING_DEVICE, additional algorithms are applied to the decision of when to scan/reassociate/roam. 2 ROAMING_MANUAL: Roaming decisions will be driven by IOCTL calls by external applications, such as the wpa_ supplicant.
ar isEnable get_ar	iwpriv athN ar isEnable iwpriv athN get_ar	Enable/disable Advanced Radar support, which is related to Dynamic Turbo.
get_vap_ind	iwpriv athN get_vap_ind	Denotes the status of the VAP WDS independence set. 0 disables wds independence set, and 1 enables wds independence set. The default value is 0.
scanprsleep value (Unassociated power consumption) get_ scanprsleep	iwpriv athN scanprsleep value (Unassociated power consumption) iwpriv athN get_ scanprsleep	Set the value of scan pre-sleep.
sleepprscan value get_ sleepprscan	iwpriv athN sleepprscan value iwpriv athN get_ sleepprscan	Sets the value of sleep pre-scan.
basicrates *basicRates*	iwpriv athN basicrates *basicRates*	Marks certain rates as basic rates according to user request.
ignore11d isEnable get_ignore11d	iwpriv athN ignore11d isEnable iwpriv athN get_ignore11d	Sets support to not process 11d beacons. 0—Disable the capability to not process 11d beacons 1—Enable the capability to not process 11d beacons
vhtmcs <mcsindex> get_vhtmcs	iwpriv athN vhtmcs <mcsindex> iwpriv athN get_vhtmcs	This command specifies the VHT MCS Index to be used with data frame transmissions. Note that invoking this command with valid MCS Index (0-9) enables "fixed rate" and Invalid index disables fixed rate setting. Make sure the chwidth and NSS parameters are set properly prior to invoking this command. MCS Index takes the following values: <0-9>—Valid Index; Enables Fixed Rate (>9)—Invalid index; Disables Fixed Rate For 11ax devices, the range is from 0-11. For QCA9984, make sure that zero wait DFS is completed or disabled before the iwpriv athX vhtmcs command is issued.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Parameter	Command	Description
nss <spatial_ streams> get_nss	iwpriv athN nss <spatial_ streams> iwpriv athN get_nss	This command specifies the number of Spatial Streams to be enabled. The Spatial Stream Count is 1-3. The default value is 3. For 11ax devices, the range is from 1-8.
tx_stbc <0|1> (HT/VHT) get_tx_stbc	iwpriv athN tx_stbc <0|1> (HT/VHT) iwpriv athN get_tx_stbc	This command allows enabling/disabling of Tx STBC. 0—Disable transmission of STBC 1—Enable transmission of STBC The default value is 1.
rx_stbc <0|1|2|3> get_rx_stbc	iwpriv athN rx_stbc <0|1|2|3> iwpriv athN get_rx_stbc	This command allows enabling/disabling of Rx Space Time Coding Block (STBC). 0—Disable Recv STBC 1—Enable Recv STBC (1) 2—Enable Recv STBC (2) 3—Enable Recv STBC (3) The default value is 1. This command allows enabling/disabling of Rx STBC. Sets/unsets B19 of HE PHY CAP indicating support for the reception of an HE PPDU that has a bandwidth less than or equal to 80 MHz using STBC. For 11ax devices, sets/unsets B19 of HE PHY CAP indicating support for the reception of an HE PPDU that has a bandwidth less than or equal to 80 MHz using STBC.
vht_txmcsmap <mcsmap> get_vht_ txmcsmap	iwpriv athN vht_txmcsmap <mcsmap> iwpriv athN get_vht_ txmcsmap	This command specifies the VHT TX MCS map to be used with VHT CAP advertisements. The 16 bits used to represent the map should be consistent with the Draft 3.1 11ac specification (Section 8.4.2.160.3 Figure 8.401bu-RX MCS MAP and TX MCS MAP)
vht_rxmcsmap <mcsmap> get_vht_ rxmcsmap	iwpriv athN vht_rxmcsmap <mcsmap> iwpriv athN get_vht_ rxmcsmap	This command specifies the VHT RX MCS map to be used with VHT CAP advertisements. The 16 bits used to represent the map should be consistent with the Draft 3.1 11ac specification (Section 8.4.2.160.3 Figure 8.401bu-RX MCS MAP and TX MCS MAP). The RX MCS Map parameter can be specified with one of the following values: ■ 0xfffc: NSS=1 MCS 0-7, NSS=2 not supported, NSS=3 not supported ■ 0xfff0: NSS=1 MCS 0-7, NSS=2 MCS 0-7, NSS=3 not supported ■ 0xffc0: NSS=1 MCS 0-7, NSS=2 MCS 0-7, NSS=3 MCS 0-7 ■ 0xfffd: NSS=1 MCS 0-8, NSS=2 not supported, NSS=3 not supported ■ 0xfff5: NSS=1 MCS 0-8, NSS=2 MCS 0-8, NSS=3 not supported ■ 0xffd5: NSS=1 MCS 0-8, NSS=2 MCS 0-8, NSS=3 MCS 0-8 ■ 0xfffe: NSS=1 MCS 0-9, NSS=2 not supported, NSS=3 not supported ■ 0xfffa: NSS=1 MCS 0-9, NSS=2 MCS 0-9, NSS=3 not supported ■ 0xffea: NSS=1 MCS 0-9, NSS=2 MCS 0-9, NSS=3 MCS 0-9 ■ 0xffda: NSS=1 MCS 0-9, NSS=2 MCS 0-9, NSS=3 MCS 0-8 ■ 0xffca: NSS=1 MCS 0-9, NSS=2 MCS 0-9, NSS=3

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Parameter	Command	Description
addba AID AC BufSize	iwpriv athN addba AID AC BufSize	These test commands are used to manually add or delete Block Acknowledge Aggregation streams. Note that automatic addba/delba processing must be turned off prior to using these commands (see setaddbaoper). Both commands require the AID (association ID) and the AC specified. The Association ID is the value shown when using the wlanconfig list command. When adding an aggregation link with addba, the BufSize parameter must be set to the maximum number of subframes that will be sent in an aggregate. When deleting an aggregation link, the initiator field indicates whether this link was initiated by the AP (1) or the remote station (0). The reason code is an 8-bit value indicating the reason the link was shut down. These commands have no corresponding get commands, nor do they have default values.
delba AID AC initiator reason	iwpriv athN delba AID AC initiator reason	These test commands are used to manually add or delete Block Acknowledge Aggregation streams. Note that automatic addba/delba processing must be turned off prior to using these commands (see setaddbaoper). Both commands require the AID (association ID) and the AC specified. The Association ID is the value shown when using the wlanconfig list command. When adding an aggregation link with addba, the BufSize parameter must be set to the maximum number of subframes that will be sent in an aggregate. When deleting an aggregation link, the initiator field indicates whether this link was initiated by the AP (1) or the remote station (0). The reason code is an 8-bit value indicating the reason the link was shut down. These commands have no corresponding get commands, nor do they have default values.
cwmin AC Mode value get_cwmin	iwpriv athN cwmin AC Mode value iwpriv athN get_cwmin	This command sets the CWmin WMM parameter for either the AP or station parameter set. The cwmax command is a WMM command that must have the AC and Mode specified. The value is CWmin in units as described in the Access Categories and Modes table. This command has a corresponding get command, which requires the AC and mode to be specified.
cwmax AC Mode value get_cwmax	iwpriv athN cwmax AC Mode value iwpriv athN get_cwmax	This command sets the CWmax WMM parameter for either the AP or station parameter set. The cwmax command is a WMM command that must have the AC and Mode specified. The value is CWmax in units as described in the Access Categories and Modes table. This command has a corresponding get command.
aifs AC Mode Value get_aifs	iwpriv athN aifs AC Mode Value iwpriv athN get_aifs	This WMM command sets the AIFSN WMM parameter for either the AP or Station parameter set. This parameter controls the frame spacing in WMM operations. The command takes 3 parameters: The first value, AC, is the access class value. The second value indicates whether the command is to be applied to the AP or Station tables, which are kept separately. Finally, the third parameter is the AIFSN value (see the Access Categories and Modes table). This parameter has a corresponding get command, which requires the AC and Mode as arguments.
iwpriv athN txoplimit AC Mode limit iwpriv athN get_ txoplimit	iwpriv athN txoplimit AC Mode limit iwpriv athN get_txoplimit	This command will set the TXOP limit, described in the Access Categories and Modes table. The AC and Mode is as described at the beginning of this section. This command has a corresponding get command.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Parameter	Command	Description
acm AC Mode value	iwpriv athN acm AC Mode value	The ACM value for each access category is set using the acm command. The AC and Mode values must be set for this command. The value is the ACM value (see Access Categories and Modes table) for the specific access category. This command has a corresponding get command that returns the current setting for the indicated AC and mode.
get_acm	iwpriv athN get_acm	This command is used to check whether ACM is enabled or disabled.
noackpolicy AC Mode 0|1 get_noackpolicy	iwpriv athN noackpolicy AC Mode 0|1 iwpriv athN get_ noackpolicy	This command sets the No ACK policy bit in the WMM parameter set for either the AP or station. The noackpolicy command is a WMM command that must have the AC and mode specified. The value either sets the policy to no ACK sent (1) or send ACK (0). This command has a corresponding get command.
chainmasksel 1|0 get_ chainmasksel	iwpriv wifiN chainmasksel 1|0 iwpriv wifiN get_ chainmasksel	This command enables (1) automatic chainmask selection. This feature allows the system to select between 2 and 3 transmit chains, depending on the signal quality on the channel. For stations that are distant, using 3 chain transmit allows for better range performance. The default value of this parameter is 0 (disabled). This command has a corresponding get command. This command is supported in DA mode only.
AMPDU 1|0 get_AMPDU	iwpriv wifiN AMPDU 1|0 iwpriv wifiN get_AMPDU	This is used to enable/disable transmit AMPDU aggregation for the entire interface. Receiving of aggregate frames will still be performed, but no aggregate frames will be transmitted if this is disabled. This has a corresponding get command, and the default value is 1 (enabled).
AMPDUFrames numFrames get_ AMPDUFrames	iwpriv wifiN AMPDUFrames numFrames iwpriv wifiN get_ AMPDUFrames	This command will set the maximum number of subframes to place into an AMPDU aggregate frame. Frames are added to an aggregate until either a) the transmit duration is exceeded, b) the number of subframes is exceeded, c) the maximum number of bytes is exceeded, or d) the corresponding queue is empty. The subframe that causes the excess conditions will not be included in the aggregate frame, but will be queued up to be transmitted with the next aggregate frame. The default value is 32. This command has a corresponding get command. This command is supported in DA mode only.
AMPDULim Byte Limit get_AMPDULim	iwpriv wifiN AMPDULim Byte Limit iwpriv wifiN get_ AMPDULim	This parameter will limit the number of bytes included in an AMPDU aggregate frame. Frames are added to an aggregate until either a) the transmit duration is exceeded, b) the number of subframes is exceeded, c) the maximum number of bytes is exceeded, or d) the corresponding queue is empty. The subframe that causes the excess conditions will not be included in the aggregate frame, but will be queued up to be transmitted with the next aggregate frame. The default value of this parameter is 50000. This command has a corresponding get command. This command is supported in DA mode only.

Parameter	Command	Description
HALDbg debug level get_HALDbg	iwpriv wifiN HALDbg debug level iwpriv wifiN get_HALDbg	Used to set the debug level in the HAL code. This can be modified on the fly 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 to select from. Note that certain categories will produce copious amounts of output, and should be used sparingly for a few seconds. HAL Debug Flags Symbolic Name Enable Bit Description and initialization HAL_DBG_RESET 0x00000001 Information pertaining to reset processing and initialization HAL_DBG_PHY_IO 0x00000002 PHY read/write states HAL_DBG_REG_IO 0x00000004 Register I/O, including all register values. Use with caution HAL_DBG_RF_PARAM 0x00000008 RF Parameter information, and table settings. HAL_DBG_QUEUE 0x00000010 Queue management for WMM support HAL_DBG_EEPROM_DUMP 0x00000020 Large dump of EEPROM information. System must be compiled with the EEPROM_DUMP conditional variable defined 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

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Parameter	Command	Description
HALDbg debug level get_HALDbg	iwpriv wifiN HALDbg debug level iwpriv wifiN get_HALDbg	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_UNMASKABLE 0xFFFFFFFF Will be printed in all cases if AH_DEBUG is defined. This command has a corresponding get command. Its default value is 0 (no debugging on), but this does not disable the unmaskable prints.
TXPwrOvr overrideLimit get_TXPwrOvr	iwpriv athN TXPwrOvr overrideLimit iwpriv athN get_TXPwrOvr	This command is used to override the transmit power limits set by iwconfig or the TXPowLim command. This is used for testing only, and is still limited by the regulatory power. A corresponding get command exists, and its default value is 0. This command is supported in DA mode only.
dyntxchain 1|0 get_dyntxchain	iwpriv wifiN dyntxchain 1|0 iwpriv wifiN get_dyntxchain	This command is used to override the dynamic txChainmask function on/off. When this function is turned on, Tx beamforming and STBC take priority. A corresponding get command exists, and its default value is 0.
bcnbwnssmap <1 | 0> get_ bcnbwnssmap	iwpriv athX bcnbwnssmap <1 | 0> iwpriv athX get_ bcnbwnssmap	Enable (1) or disable (0) bandwidth-NSS mapping in beacons. Use the get command to retrieve the configured value.
blbwnssmap <1 | 0> get_ blbwnssmap	iwpriv athX blbwnssmap <1 | 0> iwpriv athX get_ blbwnssmap	Enable (1) or disable (0) the advertisement of bandwidth-NSS mapping in STA mode. Use the get command to retrieve the configured value.

3.69 Miscellaneous radio layer parameters

The different radio layer (wifiX) parameters discussed in this section apply to both non-offload and offload models.

Table 3-64 Miscellaneous radio layer parameters

Parameter	Command	Description
get_countrycode	iwpriv wifiN get_ countrycode	Displays the current setting of the country code. The country code values are listed in *Appendix A, Country Code Definitions*.
getHALparam	iwpriv wifiX getHALparam	Retrieves the hardware abstraction layer attributes. This command is not supported in offload mode.
setHALparam	iwpriv wifiX setHALparam	Sets the hardware abstraction layer attributes. Value must be be+F6:F33tween 0-65536. This command is not supported in offload mode.
PCIEPwrSvEn	iwpriv wifiX PCIEPwrSvEn	Enable or disable power-saving for peripheral component interconnect express. This command is not supported in offload mode.
GetPCIEPwrSvEn	iwpriv wifiX GetPCIEPwrSvEn	Displays the status of power-saving for peripheral component interconnect express. This command is not supported in offload mode.
PCIEL1SKPEn	iwpriv wifiX PCIEL1SKPEn	Enable or disable skip symbol for link L1 state of peripheral component interconnect express. This command is not supported in offload mode.
GetPCIEL1SKPEn	iwpriv wifiX GetPCIEL1SKPEn	Displays the status of skip symbol for link L1 state of peripheral component interconnect express. This command is not supported in offload mode.
PCIEClkReq	iwpriv wifiX PCIEClkReq	Enable or disable clock request for peripheral component interconnect express. This command is not supported in offload mode.
GetPCIEClkReq	iwpriv wifiX GetPCIEClkReq	Displays the status of clock request for peripheral component interconnect express. This command is not supported in offload mode.
PCIEWAEN	iwpriv wifiX PCIEWAEN	Enable or disable workload accelerator for peripheral component interconnect express. This command is not supported in offload mode.
GetPCIEWAEN	iwpriv wifiX GetPCIEWAEN	Displays the status of workload accelerator for peripheral component interconnect express. This command is not supported in offload mode.
PCIEDETACH	iwpriv wifiX PCIEDETACH	Enable or disable detachment of device to peripheral component interconnect express. This command is not supported in offload mode.
GetPCIEDETACH	iwpriv wifiX GetPCIEDETACH	Displays the status of detachment for peripheral component interconnect express. This command is not supported in offload mode.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-64 Miscellaneous radio layer parameters (cont.)

Parameter	Command	Description
PCIEPwRset	iwpriv wifiX PCIEPwRset	Enable or disable the reset of power for peripheral component interconnect express. This command is not supported in offload mode.
GetPCIEPwRset	iwpriv wifiX GetPCIEPwRset	Displays the configuration of reset of power for peripheral component interconnect express. This command is not supported in offload mode.
PCIERestore	iwpriv wifiX PCIERestore	Enable or disable the restoration of power for peripheral component interconnect express. This command is not supported in offload mode.
GetPCIERestore	iwpriv wifiX GetPCIERestore	Displays the configuration of restoration of power for peripheral component interconnect express. This command is not supported in offload mode.
DisTurboG	iwpriv wifiX DisTurboG	Disables turbo G mode. This command is not supported in offload mode.
GetDisTurboG	iwpriv wifiX GetDisTurboG	Displays whether turbo G mode is disabled. This command is not supported in offload mode.
FIRStepLvl	iwpriv wifiX FIRStepLvl	Configures the first step level. This command is not supported in offload mode.
GetFIRStepLvl	iwpriv wifiX GetFIRStepLvl	Displays the configuration of first step level. This command is not supported in offload mode.
DisPACal	iwpriv wifiX DisPACal	Disables periodic power amplifier calculation. This command is not supported in offload mode.
GetDisPACal	iwpriv wifiX GetDisPACal	Displays the status of periodic power amplifier calculation. This command is not supported in offload mode.
RegRead_base	iwpriv wifiX RegRead_base	Configures register read base operation. This command is not supported in offload mode.
GetRegReads	iwpriv wifiX GetRegReads	Displays the output of the register read base operation. This command is not supported in offload mode.
keycache	iwpriv wifiX keycache	Configures the key cache for independent BSS or adhoc network. The key cache must be updated after association according to the remote capability of TxBF. This command is not supported in offload mode.
get_keycache	iwpriv wifiX get_keycache	Displays the key cache for independent BSS or adhoc network. This command is not supported in offload mode.
printBBDebug	iwpriv wifiX printBBDebug	Configures busybox debugging capability. This command is not supported in offload mode.
g_printBBDebug	iwpriv wifiX g_ printBBDebug	Displays the configured busybox debugging capability. This command is not supported in offload mode.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-64 Miscellaneous radio layer parameters (cont.)

Parameter	Command	Description
txchmaskleg	iwpriv wifiX txchmaskleg	Configures legacy Tx chainmask. For MIMO devices, Tx/Rx chainmasks indicate the number of Tx/Rx streams, and which chains are used. For some Qualcomm Technologies devices, up to 3 chains can be used, others are restricted to 3, 2 or 1. For dual chain devices, chain 2 is not available. Single chain devices only support chain 0. This command is not supported in offload mode.
get_txchmaskleg	iwpriv wifiX get_ txchmaskleg	Displays legacy Tx chainmask. This command is not supported in offload mode.
rxchmaskleg	iwpriv wifiX rxchmaskleg	Configures legacy Rx chainmask. This command is not supported in offload mode.
get_rxchmaskleg	iwpriv wifiX get_ rxchmaskleg	Displays legacy Rx chainmask. This command is not supported in offload mode.
AMPDULim	iwpriv wifiX AMPDULim	Configures the aggregated MAC protocol data unit (AMPDU) limit. This command is supported in DA mode only.
getAMPDULim	iwpriv wifiX getAMPDULim	Retrieves the configured AMPDU limit. This command is supported in DA mode only.
AMPDUFrames	iwpriv wifiX AMPDUFrames	Configures the maximum number of AMPDU frames. This command is supported in DA mode only.
getAMPDUFrames	iwpriv wifiX getAMPDUFrames	Retrieves the configured maximum number of AMPDU frames. This command is supported in DA mode only.
AMPDURxBsize	iwpriv wifiX AMPDURxBsize	Configures the AMPDU Rx buffer size. This command is supported in DA mode only.
getAMPDURxBsize	iwpriv wifiX getAMPDURxBsize	Retrieves the configured the AMPDU Rx buffer size. This command is supported in DA mode only.
burst_dur	iwpriv wifiX burst_dur	Configures the burst duration.
get_burst_dur	iwpriv wifiX get_burst_dur	Retrieves the configured burst duration.
TXPwrOvr	iwpriv wifiX TXPwrOvr	Configures Tx power override capability. This command is not supported in offload mode.
getTXPwrOvr	iwpriv wifiX getTXPwrOvr	Retrieves the configured Tx power override capability. This command is not supported in offload mode.
DisASPMWk	iwpriv wifiX DisASPMWk	Disables Active State Power Management (ASPM) wake. ASPM is a power management protocol used to manage PCI Express-based (PCIe) serial link devices. This command is not supported in offload mode.
getDisASPMWk	iwpriv wifiX getDisASPMWk	Retrieves the status of disabling Active State Power Management (ASPM) wake. This command is not supported in offload mode.
EnaASPM	iwpriv wifiX EnaASPM	Enables Active State Power Management (ASPM) for PCIe. This command is not supported in offload mode.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-64 Miscellaneous radio layer parameters (cont.)

Parameter	Command	Description
getEnaASPM	iwpriv wifiX getEnaASPM	Displays whether Active State Power Management (ASPM) for PCIe is enabled. This command is not supported in offload mode.
ATHDebug	iwpriv wifiX ATHDebug	Configures debugging for the ath or protocol layer. This command is not supported in offload mode.
getATHDebug	iwpriv wifiX getATHDebug	Displays whether debugging is configured for the ath or protocol layer. This command is not supported in offload mode.
tpscale	iwpriv wifiX tpscale	Configures transmit power scaling.
get_tpscale	iwpriv wifiX get_tpscale	Displays whether transmit power scaling is configured.
get_rxq_info	iwpriv wifiX get_rxq_info	Displays Rx queue information. This command is not supported in offload mode.
ctlpwscale	iwpriv wifiX ctlpwscale	Configures control power scaling. tpscale reduces power from regulatory limit, whereas ctlpwscale reduces from the final power.
get_ctlpwscale	iwpriv wifiX get_ctlpwscale	Displays control power scaling.
rximt_first	iwpriv wifiX rximt_first	Configures Rx interleaved multithreading for first packet. This command is not supported in offload mode.
get_rximt_first	iwpriv wifiX get_rximt_first	Displays configuration of Rx interleaved multithreading for first packet. This command is not supported in offload mode.
rximt_last	iwpriv wifiX rximt_last	Configures Rx interleaved multithreading for last packet. This command is not supported in offload mode.
get_rximt_last	iwpriv wifiX get_rximt_last	Displays configuration of Rx interleaved multithreading for last packet. This command is not supported in offload mode.
tximt_first	iwpriv wifiX tximt_first	Configures Tx interleaved multithreading for first packet. This command is not supported in offload mode.
get_tximt_first	iwpriv wifiX get_tximt_first	Displays configuration of Tx interleaved multithreading for first packet. This command is not supported in offload mode.
tximt_last	iwpriv wifiX tximt_last	Configures Tx interleaved multithreading for last packet. This command is not supported in offload mode.
get_tximt_last	iwpriv wifiX get_tximt_last	Displays configuration of Tx interleaved multithreading for last packet. This command is not supported in offload mode.
retrydur	iwpriv wifiX retrydur	Configures retry duration for frames. This command is not supported in offload mode.
get_retrydur	iwpriv wifiX get_retrydur	Displays the configuration of retry duration for frames. This command is not supported in offload mode.
immunity	iwpriv wifiX immunity	Configures the toggling of immunity level. This command is not supported in offload mode.

Table 3-64 Miscellaneous radio layer parameters (cont.)

Parameter	Command	Description
get_immunity	iwpriv wifiX get_immunity	Displays whether toggling of immunity level is set. This command is not supported in offload mode.
limit_legacy	iwpriv wifiX limit_legacy	Configures limit for legacy frames. This command is not supported in offload mode.
g_limit_legacy	iwpriv wifiX g_limit_legacy	Displays configured limit for legacy frames. This command is not supported in offload mode.
get_ledcustom	iwpriv wifiX get_ledcustom	Retrieves the custom LED of general purpose I/O pin. This command is not supported in offload mode.
set_ledcustom	iwpriv wifiX set_ledcustom	Retrieves the custom LED of general purpose I/O pin. This command is not supported in offload mode.
get_swapled	iwpriv wifiX get_swapled	Displays whether the swapping of default WLAN LED is configured. This command is not supported in offload mode.
set_swapled	iwpriv wifiX set_swapled	Configures the swapping of default WLAN LED. This command is not supported in offload mode.
getVowExt	iwpriv wifiX getVowExt	Displays whether the usage of additional buffers for video over wireless extension functionality is enabled. This command is not supported in offload mode.
setVowExt	iwpriv wifiX setVowExt	Specifies the usage of additional buffers for video over wireless extension functionality. In compile-time configuration, enable ATH_SUPPORT_VOWEXT in build/scripts/<board>/config.<board>. For run-time configuration, check the current configuration of the VoW. Check the current configuration of the VoW before setting video over wireless extension functionality for retry reordering and transmit buffer operations. This command is not supported in offload mode.
get_vsp_enable	iwpriv wifiX get_vsp_enable	Displays whether VSP is enabled or disabled. This command is not supported in offload mode.
set_vsp_enable	iwpriv wifiX set_vsp_enable	Enable or disable video stream protection (VSP). The default value is 1 (Enabled) if VoW is compiled. VSP offers QoS enhancements, such as latency, jitter, and packet loss rate. This command is not supported in offload mode.
get_vsp_thresh	iwpriv wifiX get_vsp_thresh	Retrieves the goodput threshold value, which is used by the VSP algorithm to differentiate between bad and good video streams. This command is not supported in offload mode.
set_vsp_thresh	iwpriv wifiX set_vsp_thresh	Sets goodput threshold value, which is used by the VSP algorithm to differentiate between bad and good video streams. The default value is 4096000 (40 Mbps converted in Kbps). This configuration parameter is not a trivial one, and is not meant for end users to configure. It should be used only by the developers. This command is not supported in offload mode.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-64 Miscellaneous radio layer parameters (cont.)

Parameter	Command	Description
get_vsp_evaldur	iwpriv wifiX get_vsp_ evaldur	Gets the current value of VSP Algo Eval Duration value (in ms). This command is not supported in offload mode.
set_vsp_evaldur	iwpriv wifiX set_vsp_evaldur	Specifies the VSP algorithm evaluation interval in milliseconds. This is the time after which video streams are (re)evaluated to be bad or good. Default value is 100 ms (unit is ms) This command is not supported in offload mode.
getVowExtStats	iwpriv wifiX getVowExtStats	Retrieves the collected video over wireless extension statistical counters.
setVowExtStats	iwpriv wifiX setVowExtStats	Configures the collection of video over wireless extension statistical counters.
get_txdutycycle	iwpriv wifiX get_txdutycycle	Retrieves the configured WLAN Tx duty cycle. This command is not supported in offload mode.
set_txdutycycle	iwpriv wifiX set_txdutycycle	Configures WLAN Tx duty cycle. This command is not supported in offload mode.
get_buff_thresh	iwpriv wifiX get_buff_thresh	Retrieves the buffer threshold for Hy-Fi networks.
set_buff_thresh	iwpriv wifiX set_buff_thresh	Configures the buffer threshold for Hy-Fi networks.
get_cvtimeout	iwpriv wifiX get_cvtimeout	Retrieves the channel vector timeout value. This command is not supported in offload mode.
set_cvtimeout	iwpriv wifiX set_cvtimeout	Configures the channel vector timeout value. This command is not supported in offload mode.
setRegdomain	iwpriv wifiX setRegdomain	Sets the regulatory domain to define transmission specifications for a country, regardless of the country code specified.
getRegdomain	iwpriv wifiX getRegdomain	Retrieves the configured regulatory domain.
setCH144	iwpriv wifiX setCH144	Sets the channel 144 for transmission purposes for countries in FCC3 and FCC6 domain. All the144 channels are skipped when regulatory domain (RD) bit 5 is equal to 0 or when channel 144 is not allowed as primary channel.
getCH144	iwpriv wifiX getCH144	Displays whether channel 144 is configured for transmission purposes.
enable_macreq	iwpriv wifiX enable_macreq	Enables MAC address request. MAC ID statically generated by specifying the VAP ID in wlanconfig CLI command for VAP creation. This feature works if enable_macreq is set to 1 for a particular radio. By default, if a user requests for a VAP ID in the range of 0-7, this ID is mapped to 8-15 internally. This behavior is unchanged to maintained backward compatibility.
setHwaddr	iwpriv wifiX setHwaddr	Set MAC address of the interface. For example iwpriv wifi1 setHwaddr 8c:fd:f0:77:78:79. This command is used to set the hardware address for the radio. VAPs must not be brought up to set this value.
amemPrint	iwpriv wifiX amemPrint	Displays the address memory of Atheros Service Framework (ASF). This command is not supported in offload mode.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-64 Miscellaneous radio layer parameters (cont.)

Parameter	Command	Description
g_PhyRestartWar	iwpriv wifiX g_ PhyRestartWar	Displays whether the workaround to perform a restart of PHY is configured. This command is not supported in offload mode.
s_PhyRestartWar	iwpriv wifiX s_ PhyRestartWar	Configures the workaround to perform a restart of PHY. This command is not supported in offload mode.
g_KeySrchAlways	iwpriv wifiX g_ KeySrchAlways	Displays whether the workaround to always perform a key-search operation is configured. This command is not supported in offload mode.
s_KeySrchAlways	iwpriv wifiX s_ KeySrchAlways	Configures the workaround to always perform a key-search operation. This command is not supported in offload mode.
noisespuropt	iwpriv wifiX noisespuropt	Configures the option for spurs with Adaptive Noise Immunity (ANI). This command is not supported in offload mode.
disp_tpc	iwpriv wifiX disp_tpc	Configures the display of transmit power control values. For each available transmission rate, it shows whether it supports TxBF/STBC/1-chain/2-chain/3-chain. In case of TxBF or STBC, it also shows the Tx power limit in txpower_txbf[] and txpower_ stbc[] respectively.
cntrx	iwpriv wifiX cntrx	Configures the counter to compute the number of data bytes successfully received by the AP. This command is not supported in offload mode.
get_cntrx	iwpriv wifiX get_cntrx	Displays the number of data bytes successfully received by the AP. This command is not supported in offload mode.
setctsrate	iwpriv wifiX setctsrate	Specifies the rate for receive-to-send packets. Argument for the command / Rate Index / Rate 1 / 5 / 24 2 / 6 / 12 3 / 7 / 6 4 / 8 / 54
get_ctsrate	iwpriv wifiX get_ctsrate	Displays the receive-to-send and clear-to-send rates.
rxintr_delay	iwpriv wifiX rxintr_delay	Configures the Rx loop limit. The Rx loop limit is required for Rx processing because IOS runs in a single context and it is up to the tasks to relinquish the CPU for other tasks. For IOS, if a process is running and a WLAN interrupt occurs, the ISR routine is called to process the interrupts and returns. Enable the compilation flag ATH_RX_LOOPLIMIT_TIMER (which is disabled by default) for this command to work.
g_rxintr_delay	iwpriv wifiX g_rxintr_delay	Retrieves the configured Rx loop limit. Enable the compilation flag ATH_RX_LOOPLIMIT_TIMER (which is disabled by default) for this command to work.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-64 Miscellaneous radio layer parameters (cont.)

Parameter	Command	Description
disablestats	iwpriv wifiX disablestats	Disables collection of statistical counters to deactivate debugging. In certain scenarios, it might be necessary to disable troubleshooting to avoid performance impact and degradation of traffic-handling capacity.
get_disstats	iwpriv wifiX get_disstats	Displays whether the debugging capability is disabled. This command is not supported in offload mode.
get_if_path	iwpriv wifiX get_if_path	Retrieves the interface identifier. This command is not supported in offload mode.
allowpromisc	iwpriv wifiX allowpromisc	Enables or disables the promiscuous mode on device. This command is not supported in offload mode.
getallowpromisc	iwpriv wifiX getallowpromisc	Displays the status of promiscuous mode on device. This command is not supported in offload mode.
get_radio_type	iwpriv wifiX get_radio_type	Retrieves the configured radio type.
set_enable_rate	iwpriv wifiX set_enable_rate	Configures the receive-to-send processing. This command is not supported in offload mode.
get_enable_rate	iwpriv wifiX get_enable_rate	Displays whether the receive-to-send processing is configured. This command is not supported in offload mode.
set_en_sel_rts	iwpriv wifiX set_en_sel_rts	Configures the selective receive-to-send processing. This command is not supported in offload mode.
get_en_sel_rts	iwpriv wifiX get_en_sel_rts	Displays whether the selective receive-to-send processing is configured. This command is not supported in offload mode.
blockdfschlist	iwpriv wifiX blockdfschlist	Configures the blocking of channels in Dynamic Frequency Selection (DFS) scheduler list. This command is supported for DA mode only; no cfg80211tool equivalent.
g_blkdfschlist	iwpriv wifiX g_blkdfschlist	Displays the status of blocked channels in Dynamic Frequency Selection (DFS) configuration. This command is supported for DA mode only; no cfg80211tool equivalent.
atf_p_tput	iwpriv wifiX atf_p_tput	Sets the adjustment of possible throughput with airtime fairness module. This command is not supported in offload mode.
g_atf_p_tput	iwpriv wifiX g_atf_p_tput	Displays the adjustment of possible throughput with airtime fairness module. This command is not supported in offload mode.
atf_tput	iwpriv wifiX atf_tput	Sets the adjustment of throughput with airtime fairness module. This command is not supported in offload mode.
g_atf_tput	iwpriv wifiX g_atf_tput	Displays the adjustment of throughput with airtime fairness module. This command is not supported in offload mode.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-64 Miscellaneous radio layer parameters (cont.)

Parameter	Command	Description
atf_tput_m	iwpriv wifiX atf_tput_m	Sets the adjustment of more throughput with airtime fairness module. This command is not supported in offload mode.
g_atf_tput_m	iwpriv wifiX g_atf_tput_m	Displays the adjustment of more throughput with airtime fairness module. This command is not supported in offload mode.
g_phycckerr	iwpriv wifiX g_phycckerr	Displays the Complementary Code Keying (CCK) timing errors.
Ena_BTCOEX	iwpriv wifiX Ena_BTCOEX	Enables Bluetooth coexistence with WLAN. This command is supported on direct attach chipsets only.
get_BTCOEX	iwpriv wifiX get_BTCOEX	Displays whether Bluetooth coexistence with WLAN is configured. This command is supported on direct attach chipsets only.
set_BT_WEIGHT	iwpriv wifiX set_BT_ WEIGHT	Configures the Bluetooth weight using a 3-wire interface. This command is supported on direct attach chipsets only.
get_BT_WEIGHT	iwpriv wifiX get_BT_ WEIGHT	Displays the Bluetooth weight using a 3-wire interface. This command is supported on direct attach chipsets only.
set_WL_WEIGHT	iwpriv wifiX set_WL_ WEIGHT	Configures the WLAN weight using a 3-wire interface. This command is supported on direct attach chipsets only.
get_WL_WEIGHT	iwpriv wifiX get_WL_ WEIGHT	Displays the WLAN weight using a 3-wire interface. This command is supported on direct attach chipsets only.
queue_map	iwpriv wifiX queue_map	Configures the traffic identifier (TID) override queue mapping to enable the content-aware routing feature. This command is not supported on platforms earlier than IPQ807x/IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx/QCN90xx or QCN602x + IPQ807xA/IPQ60xx/IPQ50xx.
get_queue_map	iwpriv wifiX get_queue_map	Displays the configured traffic identifier (TID) override queue mapping. This command is not supported on platforms earlier than IPQ807x/IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx/QCN90xx or QCN602x + IPQ807xA/IPQ60xx/IPQ50xx.
no_vlan	iwpriv wifiX no_vlan	Disables the VLAN operation.
get_no_vlan	iwpriv wifiX get_no_vlan	Displays whether VLAN operation is disabled.
chswitchcnt	iwpriv wifiX chswitchcnt	Configures the count for channel-switching.
g_chswitchcnt	iwpriv wifiX g_chswitchcnt	Displays the channel-switch count.
fast_lane	iwpriv wifiX fast_lane	Enables the creation of a fast lane between the STA VAP of one 5 GHz radio and the AP VAP of another 5 GHz radio in a tri-radio board. To enable this feature, user has to enable fast_lane option on both the radios that are intended to create a fast lane.
get_fast_lane	iwpriv wifiX get_fast_lane	Displays whether the configuration of fast lane on tri-radios is enabled.

Table 3-64 Miscellaneous radio layer parameters (cont.)

Parameter	Command	Description
pref_uplink	iwpriv wifiX pref_uplink	Configures the preferred uplink interface to be used with fast lane configuration. The radios on which the fast_lane option is enabled function as an enhanced independent repeater.
get_pref_uplink	iwpriv wifiX get_pref_uplink	Displays the preferred uplink interface that is configured for fast lane.
sec_offsetie	iwpriv wifiX sec_offsetie	Configures the secondary offset information element (IE).
g_sec_offsetie	iwpriv wifiX g_sec_offsetie	Displays the configured secondary offset IE.
wb_subelem	iwpriv wifiX wb_subelem	Configures the wideband subelement to be used.
g_wb_subelem	iwpriv wifiX g_wb_subelem	Displays whether the wideband subelement is configured.

3.70 Miscellaneous protocol layer parameters

Table 3-65 Miscellaneous protocol layer parameters

Parameter	Command	Description
vow_dbg_cfg	iwpriv athX vow_dbg_cfg	Specifies the Video over Wireless (VoW) debugging configuration setting in the range of 0-3.
inact_init	iwpriv athX inact_init	Sets the node inactivity timer to ensure that stale nodes are removed from the BSS.
get_inact_init	iwpriv athX get_inact_init	Retrieves and displays the configured inactivity timer value.
session	iwpriv athX session	Configures the session timeout value for mesh local peers. In Mesh mode, Mesh VAP will be continuously receiving beacons from all mesh peers in range, after getting the beacons, if it's a new mesh peer, then it will be added locally to mesh VAPs node table.
get_session	iwpriv athX get_session	Retrieves and displays the session timeout value for mesh local peers.
sifs_trigger	iwpriv athX sifs_trigger	Configures Short Interframe Space (SIFS), which is the amount of time in microseconds required for a wireless interface to process a received frame and to respond with a response frame. It is the difference in time between the first symbol of the response frame in the air and the last symbol of the received frame in the air. SIFS spaced bursting is done by sending AMPDUs in SIFS period following the Block Ack (BA) from a station. This ensures that medium is occupied by transmission more and increases the MAC efficiency. This command is supported on chipsets other than IPQ807x/IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx/QCN90xx or QCN602x + IPQ807xA/IPQ60xx/IPQ50xx.
g_sifs_trigger	iwpriv athX g_sifs_trigger	Retrieves and displays the configured SIFS setting. This command is supported on chipsets other than IPQ807x/IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx/QCN90xx or QCN602x + IPQ807xA/IPQ60xx/IPQ50xx.

Table 3-65 Miscellaneous protocol layer parameters (cont.)

Parameter	Command	Description
wds_policy	iwpriv athX wds_policy	Enables (1) or disables (0) the Wireless Distribution System (WDS) policy mechanism. This command is not applicable for cfg80211 and for iwpriv command, it is under compilation flag WDS_VENDOR_EXTENSION, which is disabled by default.
get_wds_poli	iwpriv athX get_wds_poli	Retrieves and displays the WDS policy setting. This command is not applicable for cfg80211 and for iwpriv command, it is under compilation flag WDS_VENDOR_EXTENSION, which is disabled by default.
get_ips_pspoll	iwpriv athX get_ips_pspoll	Retrieves and displays the configured IPS PS-Poll value.
wnmsleepmode	iwpriv athX wnmsleepmode	Enables (1) or disables (0) the WNM sleep mode. The cfg80211tool ath0 wnm_sleep 1 must be set before using this command. This command is supported in STA mode only.
g_wnmsleepmode	iwpriv athX g_wnmsleepmode	Retrieves and displays the configured WNM sleep mode setting. The cfg80211tool ath0 wnm_sleep 1 must be set before using this command. This command is supported in STA mode only.
wnmsmenter	iwpriv athX wnmsmenter	Starts the sleep mode for wireless network management (WNM) notification requests.
wnmsmexit	iwpriv athX wnmsmexit	Terminates the sleep mode for WNM notification request frames.
get_mcsmode	iwpriv athX get_mcsmode	Retrieves and displays the configured modulation coding scheme (MCS) mode.
get_channf	iwpriv athX get_channf	Retrieves and displays the configured channel noise floor.
bandwidth	iwpriv athX bandwidth <mode value>	Sets the operating mode (for example, 11ACVHT80) according to the desired bandwidth according to the following values: <mode value>: NONHT = 0x0C, HT20 = 0x0, HT40 = 0x1, VHT20 = 0x2, VHT40 = 0x3, VHT80 = HE20 = 0x7, HE40 = 0x8, HE80 = 0x09, HE160 = 0xA, HE80_80 = 0xB The following is an example for VHT80: iwpriv ath0 bandwidth 0x4 ifconfig ath0 down ifconfig ath0 up This command is supported both on OL and DA radios.
freq_band	iwpriv athX freq_band	Specifies the frequency band of the channel.
extchan	iwpriv athX extchan	Specifies the extension channel.
get_bandwidth	iwpriv athX get_bandwidth	Retrieves and displays the IEEE 802.11 channel bandwidth.
get_freq_band	iwpriv athX get_freq_band	Retrieves and displays the frequency band of the channel.
get_extchan	iwpriv athX get_extchan	Retrieves and displays the extension channel.
setrcparams	iwpriv athX setrcparams	Sets the rate control (RC) parameters.

Table 3-65 Miscellaneous protocol layer parameters (cont.)

Parameter	Command	Description
rtparams	iwpriv athX rtparams	Specifies the Rate Table (RT) parameters. The Rate Table is a table that stores information about the PHY Rates for a given mode (802.11na, 802.11ng, 802.11a, 802.11g, 802.11b). The Rate Table contains information for all rates for a given mode (802.11na, 802.11ng, and so on).
setratemask	iwpriv athX setratemask	Sets the rate control (RC) mask.
scanvalid period	iwpriv athN scanvalid period	Specifies scan for valid channels. This command is supported for STA only.
get_scanvalid	iwpriv athN get_scanvalid	Retrieves and displays the scan functionality for valid channels.
dbgreq	iwpriv athX dbgreq	Sets the debug request. For MU measurement, the application issues ioctl IEEE80211_IOCTL_DBGREQ with type set to IEEE80211_DBGREQ_MU_SCAN. This has the channel on which MU has to be computed and MU parameters such as the duration and the algorithm to use.
dbgLVLmac	iwpriv athX dbgLVLmac	Displays debug information level of MAC to be used. For example, iwpriv ath0 dbgLVLmac 0x80. Enable the compilation flag DBG_LVL_MAC_FILTERING (which is disabled by default) for this command to work.
getdbgLVLmac	iwpriv athX getdbgLVLmac	Retrieves and displays the configured debug information level of MAC. Enable the compilation flag DBG_LVL_MAC_FILTERING (which is disabled by default) for this command to work.
dbgLVL_high	iwpriv athX dbgLVL_high	Enables (1) or disables (0) the high level of debugging.
getdbgLVL_high	iwpriv athX getdbgLVL_high	Retrieves and displays the configured high level of debugging information.
vbLVL	iwpriv athX vbLVL	Specifies the verbose level of UMAC.
no_HT_TxAmsdu	iwpriv athX no_HT_TxAmsdu	Enables (1) or disables (0) the non-high throughput Tx for AMSDU.
g_no_HT_TxAmsdu	iwpriv athX g_no_HT_ TxAmsdu	Retrieves and displays the non-high throughput Tx for AMSDU.
ctsprt_dtmbcn	iwpriv athX ctsprt_dtmbcn	Specifies the Delivery Traffic Indication Map (DTIM) beacon for clear-to-send (CTS) protection.
g_ctsprt_dtmbcn	iwpriv athX g_ctsprt_dtmbcn	Retrieves and displays the Delivery Traffic Indication Map (DTIM) beacon for clear-to-send (CTS) protection.
vsp_enable	iwpriv athX vsp_enable	Video service protection (VSP) can be enabled/disabled using this iwpriv. Default value is 1 (Enabled) if VoW is compiled. Enter iwpriv wifiX set_vsp_enable 1 to enable VSP. Alternatively, enter iwpriv wifix set_vsp_enable 0 To disable VSP.
g_vsp_enable	iwpriv athX g_vsp_enable	Retrieves the configured VSP value.
get_mixedmode	iwpriv athX get_mixedmode	Retrieves and displays the mixed mode setting. Since training with network traffic has disadvantages like dependency on the network traffic, which cannot be controlled, a mix of both training with network traffic and proprietary packet generation will be used to achieve best results. This mode is called as mixed mode training.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-65 Miscellaneous protocol layer parameters (cont.)

Parameter	Command	Description
noIBSSCreate	iwpriv athX noIBSSCreate	Enables (1) or disables (0) the capability to not create an independent BSS (IBSS). This command is not applicable for cfg80211 and for iwpriv command it is under compilation flag UMAC_SUPPORT_IBSS.
g_noIBSSCreate	iwpriv athX g_noIBSSCreate	Retrieves and displays the setting for not creating an IBSS. This command is not applicable for cfg80211 and for iwpriv command it is under compilation flag UMAC_SUPPORT_IBSS.
no_wradar	iwpriv athX no_wradar	Enables (1) or disables (0) Weather Radar.
get_no_wradar	iwpriv athX get_no_wradar	Retrieves and displays the Weather Radar setting.
wepkeycache	iwpriv athX wepkeycache	Enables (1) or disables (0) the wireless encryption protocol (WEP) key cache.
get_wepkeycache	iwpriv athX get_wepkeycache	Displays the configured WEP key cache.
htweptkip	iwpriv athX htweptkip	Configures the TKIP for high throughput WEP. Disable this option to remove the Temporal Key Integrity Protocol (TKIP) software compression algorithm from the driver. In that case, only hardware encryption will be available.
get_htweptkip	iwpriv athX get_htweptkip	Displays the configured TKIP for high throughput WEP.
pure11ac	iwpriv athX pure11ac	Enables (1) or disables (0) pure 802.11ac.
get_pure11ac	iwpriv athX get_pure11ac	Retrieves and displays pure 802.11ac.
strictbw	iwpriv athX strictbw	Restricts association of clients that do not support same bandwidth when pure11ac mode is enabled. Enables (1) or disables (0) strict bandwidth. This command is supported on both OL and DA radios.
get_strictbw	iwpriv athX get_strictbw	Retrieves and displays strict bandwidth.
basicrates	iwpriv athX basicrates	Specifies 802.11b basic rates or Complementary Code Keying (CCK) rates.
stafwd	iwpriv athX stafwd	Enables (1) or disables (0) the capability to add the Station Forwarding feature into the UMAC. This option allows the connection of a PC behind the station.
get_stafwd	iwpriv athX get_stafwd	Retrieves and displays the Station Forwarding setting.
clrappoptie	iwpriv athX clrappoptie	Clears the application and option IEs.
vap_contryie	iwpriv athX vap_contryie	An enable/disable control that determines if the country IE is to be sent out as part of the beacon for each VAP. 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 vap_countryie 1 #iwpriv ath0 get_vap_countryie ath0 get_vap_countryie:1 This command is not supported in offload mode.
get_vapcontryie	iwpriv athX get_vapcontryie	Displays the country IE for each VAP.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-65 Miscellaneous protocol layer parameters (cont.)

Parameter	Command	Description
mfptest	iwpriv athX mfptest	Specifies the Management Frame Protection (MFP) test. MFP provides security for the management messages passed between access point (AP) and Client stations.
get_mfptest	iwpriv athX get_mfptest	Displays the configured MFP parameter.
osen	iwpriv athX osen	Configures Online Signup Server (OSU) Server-Only Authenticated L2 Encryption Network (OSEN).
get_osen	iwpriv athX get_osen	Displays the configured OSEN functionality.
hcbssload	iwpriv athX hcbssload	Configures the high channel BSS load.
get_hcbssload	iwpriv athX get_hcbssload	Displays the high channel BSS load.
oce_wan_mtr	iwpriv athX oce_wan_mtr	Configures the Optimized Connectivity Experience (OCE) WAN metrics.
g_oce_wan_mtr	iwpriv athX g_oce_wan_mtr	Displays the configured OCE WAN metrics.
mgmt_rate	iwpriv athX mgmt_rate	Configures the management rate of frames.
g_mgmt_rate	iwpriv athX g_mgmt_rate	Displays the management rate of frames.
rrm	iwpriv athX rrm	Configures IEEE 802.11k radio resource measurement.
get_rrm	iwpriv athX get_rrm	Displays IEEE 802.11k radio resource measurement.
rrmstats	iwpriv athX rrmstats	Configures IEEE 802.11k radio resource measurement statistics.
get_rrmstats	iwpriv athX get_rrmstats	Displays IEEE 802.11k radio resource measurement statistics.
rrmslwin	iwpriv athX rrmslwin	Configures IEEE 802.11k radio resource measurement service load window.
get_rrmslwin	iwpriv athX get_rrmslwin	Displays IEEE 802.11k radio resource measurement service load window.
rrmdbg	iwpriv athX rrmdbg	Configures IEEE 802.11k radio resource measurement debugging.
get_rrmdbg	iwpriv athX get_rrmdbg	Displays IEEE 802.11k radio resource measurement debugging.
wnm	iwpriv athX wnm	Enables (1) or disables (0) wireless network management (WNM) capability.
get_wnm	iwpriv athX get_wnm	Retrieves and displays the configured WNM setting.
wnm_bss	iwpriv athX wnm_bss	Enables (1) or disables (0) WNM basic service set.
get_wnm_bss	iwpriv athX get_wnm_bss	Retrieves and displays the configured WNM BSS setting.
wnm_tfs	iwpriv athX wnm_tfs	Enables (1) or disables (0) WNM traffic filter service.
get_wnm_tfs	iwpriv athX get_wnm_tfs	Retrieves and displays the configured WNM TFS setting.
wnm_tim	iwpriv athX wnm_tim	Enables (1) or disables (0) WNM traffic indication map.
get_wnm_tim	iwpriv athX get_wnm_tim	Retrieves and displays the configured WNM TIM setting.
wnm_sleep	iwpriv athX wnm_sleep	Enables (1) or disables (0) WNM sleep.
get_wnm_sleep	iwpriv athX get_wnm_sleep	Retrieves and displays the configured WNM sleep setting.
wnm_fms	iwpriv athX wnm_fms	Enables (1) or disables (0) WNM flexible multicast service.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-65 Miscellaneous protocol layer parameters (cont.)

Parameter	Command	Description
get_wnm_fms	iwpriv athX get_wnm_fms	Retrieves and displays the configured WNM FMS setting.
sendmgmt	iwpriv athX sendmgmt	Specifies the sending of management frames.
gap_dbgprint	iwpriv athX gap_dbgprint	Enables (1) or disables (0) Green AP debug prints.
gap_g_dbgprint	iwpriv athX gap_g_dbgprint	Retrieves and displays Green AP debug prints.
setwapi	iwpriv athX setwapi	Enable this option to support WLAN authentication and privacy infrastructure (WAPI).
wapi_rkupkt	iwpriv athX wapi_rkupkt	Configures rekeying for WAPI unicast session key. Station is disassociated after rekeying. Rekeying must happen after 60 seconds and ping packets must transmit continuously. This is a known limitation. The WAPI Rekey problem involves the passing of the M2 packet to the WAPID and then the WAPID concludes the rekey as successful.
get_wapi_rkupkt	iwpriv athX get_wapi_rkupkt	Displays the configured rekeying for WAPI unicast session key.
wapi_rkmpkt	iwpriv athX wapi_rkmpkt	Configures rekeying for WAPI multicast session key.
get_wapi_rkmpkt	iwpriv athX get_wapi_rkmpkt	Displays the configured rekeying for WAPI multicast session key.
wapi_rkupdate	iwpriv athX wapi_rkupdate	Configures rekeying update for WAPI.
ccmpSwSelEn	iwpriv athX ccmpSwSelEn	Enable or disable the CCMP software encryption. Disable this option to remove the CCMP encryption software from the driver. In that case, only hardware encryption (into the WLAN chip) will be available.
get_ccmpSwSelEn	iwpriv athX get_ccmpSwSelEn	Retrieves the configured CCMP software encryption.
s_scan_flush	iwpriv athX s_scan_flush	Clears the current scan table in the driver. Uses wlan_scan_table_flush UMAC API.
sw_wow	iwpriv athX sw_wow	Enables the wake on wireless (WoW) feature. When enabled, the driver builds the “WoW” kernel module. Enable the compilation flag ATH_SW_WOW (which is disabled by default) for this command to work.
get_sw_wow	iwpriv athX get_sw_wow	Retrieves the configured WoW setting. Enable the compilation flag ATH_SW_WOW (which is disabled by default) for this command to work.
wdsaddr	iwpriv athX wdsaddr	Configures the WDS address.
no_disassoc	iwpriv athX no_disassoc	Instructs the driver not to send a disassociation frame while stopping the VAP.
get_no_disassoc	iwpriv athX get_no_disassoc	Retrieves the configured setting to not send a disassociation request for connection of a STA with AP.
dbgcfg	iwpriv athX dbgcfg	Configures the debugging capability.
getdbgcfg	iwpriv athX getdbgcfg	Displays the configured debugging capability.
dbgrestart	iwpriv athX dbgrestart	Restarts the debugging capability.
getdbgrestart	iwpriv athX getdbgrestart	Displays whether the restart of debugging capability is activated.

Table 3-65 Miscellaneous protocol layer parameters (cont.)

Parameter	Command	Description
getrxdropstats	iwpriv athX getrxdropstats	Displays whether the mechanism to drop Rx frames is enabled or not.
setibssdfsparam	iwpriv athX setibssdfsparam	Enable this option to support DFS channel in AdHoc mode. If this option is disabled, only non-DFS channels are allowed in IBSS. This command does not have a cfg80211tool equivalent.
getibssdfsparam	iwpriv athX getibssdfsparam	Displays whether the option to support DFS channel in independent BSS is set. This command does not have a cfg80211tool equivalent.
s_ibssrssiclass	iwpriv athX s_ibssrssiclass	Specifies the RSSI class in independent BSS netlink notification. This command is not applicable for cfg80211 and for iwpriv command it is under compilation flag ATH_SUPPORT_ IBSS_NETLINK_NOTIFICATION, which is disabled by default.
strtibssrssimon	iwpriv athX strtibssrssimon	Starts the RSSI monitor in independent BSS netlink notification. This command is not applicable for cfg80211 and for iwpriv command it is under compilation flag ATH_SUPPORT_ IBSS_NETLINK_NOTIFICATION, which is disabled by default.
setibssrssihyst	iwpriv athX setibssrssihyst	Specifies the RSSI hysteresis in independent BSS netlink notification. This command is not applicable for cfg80211 and for iwpriv command it is under compilation flag ATH_SUPPORT_ IBSS_NETLINK_NOTIFICATION, which is disabled by default.
autocvupdate	iwpriv athX autocvupdate	Specifies the automatic complex-valued (CV) update in TX beamforming. The beamforming mechanism uses a software timer to initial CV update.
g_autocvupdate	iwpriv athX g_autocvupdate	Retrieves and displays the configured automatic complex-valued (CV) update in TX beamforming.
cvupdateper	iwpriv athX cvupdateper	Specifies the CV update packet error rate.
get_cvupdateper	iwpriv athX get_cvupdateper	Retrieves the configured CV update packet error rate.
maxsta	iwpriv athX maxsta	Configures the maximum number of stations to connect to AP.
get_maxsta	iwpriv athX get_maxsta	Displays the maximum number of stations to connect to AP.
scanband	iwpriv athX scanband	Configures the scan band.
get_scanband	iwpriv athX get_scanband	Retrieves the configured scan band.
scanchevent	iwpriv athX scanchevent	Configures the scan channel event.
get_scanchevent	iwpriv athX get_scanchevent	Retrieves the configured scan event.
txcorrection	iwpriv athX txcorrection	Configures the correction field of Tx frames for Wi-Fi positioning. This command is under compilation flag ATH_SUPPORT_ LOWI, which is disabled by default.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-65 Miscellaneous protocol layer parameters (cont.)

Parameter	Command	Description
g_txcorrection	iwpriv athX g_txcorrection	Displays the correction field of Tx frames for Wi-Fi positioning. This command is under compilation flag ATH_SUPPORT_LOWI, which is disabled by default.
rxcorrection	iwpriv athX rxcorrection	Configures the correction field of Rx frames for Wi-Fi positioning. This command is under compilation flag ATH_SUPPORT_LOWI, which is disabled by default.
g_rxcorrection	iwpriv athX g_rxcorrection	Displays the correction field of Rx frames for Wi-Fi positioning. This command is under compilation flag ATH_SUPPORT_LOWI, which is disabled by default.
rejoingtime	iwpriv athX rejoingtime	Configures the rejoining time for WPA supplicant. This command is not applicable for cfg80211 and for iwpriv command, it is under compilation flag ATH_SUPPORT_WPA_SUPPLICANT_CHECK_TIME which is disabled by default.
get_rejoingtime	iwpriv athX get_rejoingtime	Displays the configured rejoining time for WPA supplicant. This command does not have a cfg80211 equivalent.
senddeauth	iwpriv athX senddeauth	Configures the sending of deauthentication frames.
get_senddeauth	iwpriv athX get_senddeauth	Displays the sending of deauthentication frames.
siwtxpwradjust	iwpriv athX siwtxpwradjust	Specifies Tx power adjustment.
aponly	iwpriv athX aponly	Enable this option to remove all station-specific code from the UMAC and decrease its size.
get_aponly	iwpriv athX get_aponly	Displays whether the AP-only mode is enabled.
txrx_dbg	iwpriv athX txrx_dbg	Enables the debugging for Tx and Rx packets.
vap_txrx_stats	iwpriv athX vap_txrx_stats	Enables the collection of Tx and Rx statistics for VAPs. Valid values for this command are 14, 16, and 17. This command is supported on QCA9886, QCA9984, and IPQ4019 platforms only; this command is not supported on IPQ807x/IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx/QCN90xx or QCN602x + IPQ807xA/IPQ60xx/IPQ50xx platforms.
vap_txrx_st_rst	iwpriv athX vap_txrx_st_rst	Resets the collection of Tx and Rx statistics for VAPs.
txrx_fw_mstats	iwpriv athX txrx_fw_mstats	Enables the collection of firmware Tx and Rx statistics.
tx_ppdu_log_cfg	iwpriv athX tx_ppdu_log_cfg	Enables the protocol packet data unit logging configuration.
vhtmcs	iwpriv athX vhtmcs	Enables the very high throughput modulation coding scheme. For QCA9984, make sure that zero wait DFS is completed or disabled before the iwpriv athX vhtmcs command is issued.
get_vhtmcs	iwpriv athX get_vhtmcs	Displays the configured very high throughput modulation coding scheme.
nss	iwpriv athX nss	Enables the network switching subsystem.

Table 3-65 Miscellaneous protocol layer parameters (cont.)

Parameter	Command	Description
get_nss	iwpriv athX get_nss	Displays the status of the network switching subsystem.
get_sta_count	iwpriv athX get_sta_count	Displays the number of stations associated with AP.
novap_reset	iwpriv athX novap_reset	Specifies that the VAP must not be reset.
get_novap_reset	iwpriv athX get_novap_reset	Displays whether the capability to not reset VAP is configured.
vht80_rate	iwpriv athX vht80_rate	Specifies the VHT80 channel rate.
get_vht80_rate	iwpriv athX get_vht80_rate	Retrieves the configured VHT80 channel rate.
bw_nss_rate	iwpriv athX bw_nss_rate	Sets the bandwidth NSS rate mask.
ldpc	iwpriv athX ldpc	Specifies the low density parity check for rate control.
get_ldpc	iwpriv athX get_ldpc	Displays the configured low density parity check.
tx_stbc	iwpriv athX tx_stbc	Configures space time block coding for Tx.
get_tx_stbc	iwpriv athX get_tx_stbc	Displays space time block coding for Tx.
rx_stbc	iwpriv athX rx_stbc	Configures space time block coding for Rx.
get_rx_stbc	iwpriv athX get_rx_stbc	Displays space time block coding for Rx.
opmode_notify	iwpriv athX opmode_notify	Specifies the operating mode notification request is set to be transmitted.
g_opmod_notify	iwpriv athX g_opmod_notify	Displays the configured operating mode notification request.
set_cactimeout	iwpriv athX set_cactimeout	Specifies the timeout period for channel availability check. CAC (Channel Availability Check) time is the fixed interval during which a system shall monitor a channel for the presence of radar prior to initiating a communication link on that channel. The option to disable the CAC waiting time during channel change is provided.
get_cactimeout	iwpriv athX get_cactimeout	Displays the configured timeout period for CAC.

Table 3-65 Miscellaneous protocol layer parameters (cont.)

Parameter	Command	Description
enablertscts	iwpriv athX enablertscts <val>	Enable/disable use of RTS/CTS from rate control logic. <val> indicates the combination of RTS flag (lower nibble) and RTS profile (upper nibble). **RTS Flag:** 0: no RTC/CTS, 1: RTS-CTS enabled 2: CTS2Self **RTS Profile:** /* Disable the RTS-CTS from Rate Control */ IEEE80211_RTSCTS_FOR_NO_RATESERIES = 0, /* Enable the RTS-CTS from Rate Control for each PPDU */ IEEE80211_RTSCTS_FORCED = 1, /* Enable the RTS-CTS from Rate Control Only for the PPDUs that are being retransmitted */ IEEE80211_RTSCTS_ACROSS_SW_RETRIES = 2, /*RTS/CTS used for ERP protection for every PPDU.*/ IEEE80211_RTSCTS_ERP = 3, /* Enable RTS-CTS for all rate series */ IEEE80211_RTSCTS_FOR_ALL_RATESERIES = 4, /* To add new profiles before this profile */ IEEE80211_RTSCTS_PROFILE_MAX = 15 **Examples:** To enable the RTS from rate control only for Retransmitted PPDUs, enter `iwpriv <VAP> enablertscts 0x21` command. To completely disable the RTS from rate control logic, enter `iwpriv <VAP> enablertscts 0` command.
rts <val>	iwconfig athX rts <val>	Enable RTS during PPDU construction, based on PPDU size. 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). Example usage: `iwconfig <VAP> rts 256`

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-65 Miscellaneous protocol layer parameters (cont.)

Parameter	Command	Description
g_enablertscts	iwpriv athX g_enablertscts	Status of use of RTS/CTS when first transmission attempt fails.
rc_retries	iwpriv athX rc_retries	Configures the number of retries for rate control. For a direct attach data path, software must select a rate series, which consists of four rates and the number of attempts (retries) at each rate. The rate series is used by the WLAN MAC hardware. The MAC hardware first attempts to transmit at the rate of the first entry in the series. If it fails, the transmission is repeated at that rate for a total number of attempts equal to the number of retries specified in the series. If all attempts at that rate fail, the MAC hardware drops down to the next rate in the series and repeats.
get_rc_retries	iwpriv athX get_rc_retries	Displays the number of retries for rate control.
bcast_rate	iwpriv athX bcast_rate	Specifies the broadcast rate of transmission.
get_bcast_rate	iwpriv athX get_bcast_rate	Displays the configured broadcast rate of transmission.
vht_txmcsmap	iwpriv athX vht_txmcsmap	Specifies the Tx MCS Map field of VHT Capabilities IE.
g_vht_txmcsmap	iwpriv athX g_vht_txmcsmap	Displays the Tx MCS Map field of VHT Capabilities IE.
vht_rxmcsmap	iwpriv athX vht_rxmcsmap	Specifies the Rx MCS Map field of VHT Capabilities IE.
g_vht_rxmcsmap	iwpriv athX g_vht_rxmcsmap	Displays the Rx MCS Map field of VHT Capabilities IE.
get_parent	iwpriv athX get_parent	Retrieves the parent AP, which is the device to which the client is actually connected. Parent Wi-Fi of an AP is the wifi which the user originally configures as extender of the AP.
set_onetxchain	iwpriv athX set_onetxchain	Specifies that only one Tx chain be used for transmission.
set_cabq_maxdur	iwpriv athX set_cabq_maxdur	Specifies the maximum period of time for the content after beacon queue (CABQ).
get_acs_state	iwpriv athX get_acs_state	Retrieves the state of the automatic channel selection (ACS) reporting functionality.
get_cac_state	iwpriv athX get_cac_state	Retrieves the status of the channel availability check (CAC) mechanism.
implicitbf	iwpriv athX implicitbf	Enable (1) or disable (0) implicit beamforming.
get_implicitbf	iwpriv athX get_implicitbf	Retrieves the configuration of implicit beamforming.
vhtstscap	iwpriv athX vhtstscap	Enable (1) or disable (0) space time stream-capability (STS) for very high throughput bands.
get_vhtstscap	iwpriv athX get_vhtstscap	Displays the status of space time stream-capability (STS) for very high throughput bands.
vhtsounddim	iwpriv athX vhtsounddim	Enable (1) or disable (0) sounding packet delivery indication map (DIM) for very high throughput bands.
get_vhtsounddim	iwpriv athX get_vhtsounddim	Displays the status of sounding packet delivery indication map (DIM) for very high throughput bands.
ext_acs_prg	iwpriv athX ext_acs_prg	Configures the automatic channel selection in-progress for extender AP.
get_ext_acs_prg	iwpriv athX get_ext_acs_prg	Displays the automatic channel selection in-progress for extender AP.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-65 Miscellaneous protocol layer parameters (cont.)

Parameter	Command	Description
get_deschan	iwpriv athX get_deschan	Displays the configured desired channel.
get_desmode	iwpriv athX get_desmode	Displays the configured desired channel mode.
send_add_ies	iwpriv athX send_add_ies	Configures the transmission of additional information elements.
g_send_add_ies	iwpriv athX g_send_add_ies	Displays whether the configuration for transmission of additional information elements is set.
maxscanentry	iwpriv athX maxscanentry	Configures the maximum number of channel scan entries.
g_maxscanentry	iwpriv athX g_maxscanentry	Displays the maximum number of channel scan entries.
scanentryage	iwpriv athX scanentryage	Configures the duration for which the scan entry must be valid.
g_scanentryage	iwpriv athX g_scanentryage	Displays the duration for which the scan entry is valid.
scanmindwell	iwpriv athX scanmindwell	Configures the minimum dwell time for the channel scanning. The expected value should be greater than zero. For the given pool of channel the Min dwell time will keep constant for all the channel scanning. Starting the channel scan will return failure for invalid Min dwell time.
getscanmindwell	iwpriv athX getscanmindwell	Displays the configured minimum dwell time for channel scan.
scanmaxdwell	iwpriv athX scanmaxdwell	Configures the maximum dwell time for the channel scanning. The expected value should be greater than zero. For the given pool of channel the Max dwell time will keep constant for all the channel scanning. Starting the channel scan will return failure for invalid Max dwell time.
getscanmaxdwell	iwpriv athX getscanmaxdwell	Displays the configured maximum dwell time for channel scan.
vow_dbg	iwpriv athX vow_dbg	Configures the debugging capability for video over wireless functionality.
get_vow_dbg	iwpriv athX get_vow_dbg	Displays the debugging level for video over wireless functionality.
splitmac	iwpriv athX splitmac	Configures the split MAC functionality. WLAN driver does not respond to management frames, such as association requests. WLAN controller processes and sends the response to AP, which in turn forwards the response to the Wi-Fi client. WLAN controller calls splitmac API(add_ client) to add Wi-Fi client using specified information, such as rates, on the AP.
get_splitmac	iwpriv athX get_splitmac	Displays the configured split MAC functionality.
get_tso_stats	iwpriv athX get_tso_stats	Displays TCP segmentation offload statistics.
rst_tso_stats	iwpriv athX rst_tso_stats	Resets TCP segmentation offload statistics.
get_sg_stats	iwpriv athX get_sg_stats	Displays segmentation statistics.
rst_sg_stats	iwpriv athX rst_sg_stats	Resets segmentation statistics.
get_lro_stats	iwpriv athX get_lro_stats	Display large receive offload statistics.
rst_lro_stats	iwpriv athX rst_lro_stats	Resets large receive offload statistics.
get_csum_stats	iwpriv athX get_csum_stats	Displays Rx checksum error statistics.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-65 Miscellaneous protocol layer parameters (cont.)

Parameter	Command	Description
rst_csum_stats	iwpriv athX rst_csum_stats	Resets Rx checksum error statistics.
sta_fixed_rate	iwpriv athX sta_fixed_rate	Configures fixed rate of reception of frames for station. This command allows a different fixed rate to be specified for each associated station. The iwpriv commands (vhtmcs/he_mcs/set11NRates) fix the rate for all associated stations. STA fixed rate can only be used AFTER VDEV level fixed rate has been enabled. To enable VDEV level fixed rate, use the nss and vhtmcs/he_mcs/set11NRates commands shown above! where AAAA is the station AID, and RRRR is the 16 bit rate code. The 16 bit rate code format (0xRRRR) is: (preamble_type << 8) | (nss_minus_1 << 5) | (mcs) where preamble_type is defined as: 0 - OFDM 1 - CCK 2 - HT 3 - VHT 4 - HE For example, HE MCS 3, NSS=2: (4 << 8) | ((2-1) << 5) | 3 = 0xRRRR = 0x0423 VHT MCS 5, NSS=3: (3 << 8) | ((3-1) << 5) | 5 = 0xRRRR = 0x0345
get_signal_dbm	iwpriv athX get_signal_dbm	Displays the Rx signal power in dBm. This command is supported only for DA mode.
atf_max_buf	iwpriv athX atf_max_buf	Configures the maximum Tx buffer size for airtime fairness.
g_atf_max_buf	iwpriv athX g_atf_max_buf	Displays the maximum Tx buffer size for airtime fairness.
atf_min_buf	iwpriv athX atf_min_buf	Configures the minimum Tx buffer size for airtime fairness.
g_atf_min_buf	iwpriv athX g_atf_min_buf	Displays the minimum Tx buffer size for airtime fairness.
perunit	iwpriv athX perunit	Configures the airtime fairness for each unit.
get_perunit	iwpriv athX get_perunit	Displays the airtime fairness for each unit.
atf_tput_at	iwpriv athX atf_tput_at	Configures the airtime throughput for airtime fairness.
g_atf_tput_at	iwpriv athX g_atf_tput_at	Displays the airtime throughput for airtime fairness.
bss_chan_info	iwpriv athX bss_chan_info	Displays BSS channel information.
ssid_config	iwpriv athX ssid_config	Specifies whether SSID is public or private.
get_ssid_config	iwpriv athX get_ssid_config	Displays whether SSID is public or private.
neighbourfilter	iwpriv athX neighbourfilter	Configures Rx filter for neighboring peers monitor.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-65 Miscellaneous protocol layer parameters (cont.)

Parameter	Command	Description
enable_rtt	iwpriv athX enable_rtt	Enable round trip time for location Wi-Fi interfaces. Wi-Fi Location certification program is based on IEEE802.11mc where clients and AP use Fine Timing Measurements (FTMs) to do ranging (also known as RTT - round trip time). ■ value : 0 - Disable both FTM responder and initiator modes. ■ value : 1 - Enable FTM responder mode alone. ■ value : 2 - Enable FTM initiator mode alone. ■ value : 3 - Enable both FTM initiator and responder modes.
enable_lci	iwpriv athX enable_lci	Enable location civic information for ranging (also known as RTT - round trip time).
enable_lcr	iwpriv athX enable_lcr	Enable location civic report for ranging (also known as RTT - round trip time).
pause_scan	iwpriv athX pause_scan	Configures the pausing of VAP channel scan.
get_pause_scan	iwpriv athX get_pause_scan	Displays the configuration of the capability for pausing VAP channel scan.
random_gen_mode	iwpriv athX random_gen_mode	Configures the random number generation mode. All the implementation is kept under macro “ATH_GEN_RANDOMNESS”. By default, the mode is set to 0. This implementation helps to update the random number even in situation where the rssi is constant. This parameter is not applicable for offload chipsets.
get_random_gen	iwpriv athX get_random_gen	Displays the random number generation mode. This parameter is not applicable for offload chipsets.
ampduden_ovrd	iwpriv athX ampduden_ovrd	Configures the AMPDU density override. Enter a value in the range of 0-7.
g_ampduden_ovrd	iwpriv athX g_ampduden_ovrd	Displays the AMPDU density override.
smesh_cfg	iwpriv athX smesh_cfg	Specifies the smart mesh configuration.
get_smesh_cfg	iwpriv athX get_smesh_cfg	Displays the smart mesh configuration.
bcnbwnssmap	iwpriv athX bcnbwnssmap	Enable (1) or disable (0) bandwidth-NSS mapping in beacon.
get_bcnbwnssmap	iwpriv athX get_bcnbwnssmap	Displays the configured bandwidth-NSS mapping in beacon.
blbwnssmap	iwpriv athX blbwnssmap	Enable (1) or disable (0) the advertisement of bandwidth-NSS mapping in STA mode.
get_blbwnssmap	iwpriv athX get_blbwnssmap	Displays the status of the advertisement of bandwidth-NSS mapping in STA mode.
setmhdr	iwpriv athX setmhdr	Specifies the MAC header. This command works for mesh VAP only.
meshcap	iwpriv athX meshcap	Configures the mesh capabilities.
get_meshcap	iwpriv athX get_meshcap	Retrieves the configured mesh capabilities.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-65 Miscellaneous protocol layer parameters (cont.)

Parameter	Command	Description
rxinfo_perpkt	iwpriv athX rxinfo_perpkt	Configures the Rx information per packet. Enable the compilation flag ATH_DATA_RX_INFO_EN (which is disabled by default) for this command to work.
get_whc_wds	iwpriv athX get_whc_wds	Retrieves the wireless distribution system configuration for whole home coverage (WHC, also known as Wi-Fi SON).
get_whc_son	iwpriv athX get_whc_son	Retrieves the Wi-Fi SON information for whole home coverage.
set_whc_dist	iwpriv athX set_whc_dist	Sets the root AP distribution for whole home coverage.
get_whc_dist	iwpriv athX get_whc_dist	Retrieves the root AP distribution for whole home coverage.
set_whc_sfactor	iwpriv athX set_whc_sfactor	Sets the smoothness factor for whole home coverage.
get_whc_sfactor	iwpriv athX get_whc_sfactor	Retrieves the smoothness factor for whole home coverage.
get_whc_rate	iwpriv athX get_whc_rate	Retrieves the rate for whole home coverage.
set_whc_ul_rate	iwpriv athX set_whc_ul_rate	Sets the uplink rate for whole home coverage.
get_whc_ul_rate	iwpriv athX get_whc_ul_rate	Retrieves the uplink rate for whole home coverage.
get_whc_bssid	iwpriv athX get_whc_bssid	Retrieves the basic service set identifier for whole home coverage.
g_whc_cap_bssid	iwpriv athX g_whc_cap_bssid	Retrieves the basic service set identifier of central AP for whole home coverage.
g_best_ob_bssid	iwpriv athX g_best_ob_bssid	Retrieves the best uplink other band BSSID for whole home coverage.
g_whc_ob_bssid	iwpriv athX g_whc_ob_bssid	Retrieves the uplink other band BSSID for whole home coverage.
otherband_bssid	iwpriv athX otherband_bssid	Configures the uplink other band BSSID for whole home coverage.
assocwar160	iwpriv athX assocwar160	Configures association for 160 MHz for wireless access repeater.
get_assocwar160	iwpriv athX get_assocwar160	Retrieves the configured association for 160 MHz for wireless access repeater.
son	iwpriv athX son	Configures self-organizing network (SON) mode.
get_son	iwpriv athX get_son	Displays the configured self-organizing network (SON) mode.
rept_spl	iwpriv athX rept_spl	Configures the special multinode repeater.
g_rept_spl	iwpriv athX g_rept_spl	Retrieves the configured special multinode repeater.
rmode_pktsim	iwpriv athX rmode_pktsim	Configures the raw mode of packet subscriber identify module.
g_rmode_pktsim	iwpriv athX g_rmode_pktsim	Retrieves the raw mode of packet subscriber identify module.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-65 Miscellaneous protocol layer parameters (cont.)

Parameter	Command	Description
rawdwepind	iwpriv athX rawdwepind	Configures raw dynamic wired equivalent privacy for independent repeater. Dynamic WEP refers to the combination of 802.1x technology and the Extensible Authentication Protocol. Dynamic WEP changes WEP keys dynamically.
get_rawdwepind	iwpriv athX get_rawdwepind	Displays the configured raw dynamic wired equivalent privacy for independent repeater.
txrx_vap_stats	iwpriv athX txrx_vap_stats	Displays the Tx and Rx statistics of VAP.
mudeasoc	iwpriv athX mudeasoc	Configures the multi-user central AP association timer.
get_mudeasoc	iwpriv athX get_mudeasoc	Displays the multi-user central AP association timer.
get_RDG_enable	iwpriv athX get_RDG_enable	Displays the reverse direction grant capability.
get_DFS_support	iwpriv athX get_DFS_support	Displays the dynamic frequency selection support.
get_DFS_enable	iwpriv athX get_DFS_enable	Displays the dynamic frequency selection status.
get_ACS_support	iwpriv athX get_ACS_support	Displays the automatic channel selection support.
get_SSID_status	iwpriv athX get_SSID_status	Displays the subscriber service set identifier status.
get_DL_prisup	iwpriv athX get_DL_prisup	Displays the downlink queue priority support.
clear_mm_rssi	iwpriv athX clear_mm_rssi	Clears the minimum and maximum RSSI values.
clear_qos	iwpriv athX clear_qos	Clears the QoS setting.
mu_blklist_cnt	iwpriv athX mu_blklist_cnt	Configures the count of Tx multi-user blacklist entries.
get_mu_tx_count	iwpriv athX get_mu_tx_count	Retrieves the count of MU MIMO Tx to a peer.
rst_mu_tx_count	iwpriv athX rst_mu_tx_count	Resets the count of MU MIMO Tx to a peer.
get_mu_peer_pos	iwpriv athX get_mu_peer_pos	Retrieves the multi-user peer position.
vie_ena	iwpriv athX vie_ena	Enable (1) or disable (0) vendor IE.
g_vie_ena	iwpriv athX g_vie_ena	Retrieves the configured vendor IE.
nstswar	iwpriv athX nstswar	Configures the number of space time streams.
get_nstswar	iwpriv athX get_nstswar	Retrieves the configured number of space time streams.
csmode	iwpriv athX csmode	Configures the channel switch mode.
g_csmode	iwpriv athX g_csmode	Retrieves the configured channel switch mode.
enable_ecsa	iwpriv athX enable_ecsa	Configures the extended channel switch announcement.
g_enable_ecsa	iwpriv athX g_enable_ecsa	Retrieves the extended channel switch announcement.
ecsa_opclass	iwpriv athX ecsa_opclass	Configures the operating class for extended channel switch announcement.
g_ecsa_opclass	iwpriv athX g_ecsa_opclass	Displays the operating class for extended channel switch announcement.
backhaul	iwpriv athX backhaul	Configures the backhaul interface to connect to upstream devices.
get_backhaul	iwpriv athX get_backhaul	Retrieves the specified backhaul interface to connect to upstream devices.

Table 3-65 Miscellaneous protocol layer parameters (cont.)

Parameter	Command	Description
ul_hyst	iwpriv athX ul_hyst	Configures the best uplink hysteresis. **NOTE** In the componentized WLAN driver, version 11.0 driver, the best uplink hysteresis has been moved to the vdev object. As a result, the iwpriv command must be used in the VAP, instead of the radio interface. Also, the UCI setting must be configured on the wifi-iface section (of the backhaul STA VAP). **NOTE** If a unified configuration is required for 10.4 and 11.0 drivers, set the value for both the radio and VAP through UCI, and the appropriate setting is applied.
g_ul_hyst	iwpriv athX g_ul_hyst	Displays the configured best uplink hysteresis.
son_event_bcast	iwpriv athX son_event_bcast	Enables(1) or disables (0) the broadcasting of band steering events.
g_son_event_bcast	iwpriv athX g_son_event_ bcast	Retrieves the broadcasting of band steering events.
set_pcp_tid_map	iwpriv athX set_pcp_tid_map	Configures the traffic identifier (TID) map for Port Control Protocol (PCP). This parameter is applicable for IPQ401x/IPQ402x only and not for IPQ807x and QCA9984 platforms.
get_pcp_tid_map	iwpriv athX get_pcp_tid_map	Configures the traffic identifier (TID) map for Port Control Protocol (PCP). This parameter is applicable for IPQ401x/IPQ402x only and not for IPQ807x and QCA9984 platforms.
set_tidmap_prec	set_tidmap_prec	Configures the traffic map identifier precedence.
get_tidmap_prec	get_tidmap_prec	Retrieves the configured traffic map identifier precedence.

3.71 Miscellaneous radio layer parameters in offload mode only

The parameters discussed in this section apply to offload models only.

Table 3-66 Miscellaneous radio layer parameters in offload mode only

Parameter	Command	Description
LDPC	iwpriv wifiX LDPC	Specifies the low density parity check for rate control.
getLDPC	iwpriv wifiX getLDPC	Displays the low density parity check for rate control.
ps_report	iwpriv wifiX ps_report	Configures the change of state for power-save (PS) capability.
get_ps_report	iwpriv wifiX get_ps_report	Displays the status of change of state for power-save (PS) capability.

Table 3-66 Miscellaneous radio layer parameters in offload mode only (cont.)

Parameter	Command	Description
nonaggr_swretry	iwpriv wifiX nonaggr_ swretry	Configures software retry (SWR) for non-aggregate frames.
aggr_swretry	iwpriv wifiX aggr_swretry	Displays the status of software retry (SWR) for non-aggregate frames.
sta_kickout	iwpriv wifiX sta_kickout	Configures the capability to dissociate a station from an AP if the station loses connectivity with the AP.
LTR	iwpriv wifiX LTR	Configures long term reference (LTR) frame functionality.
getLTR	iwpriv wifiX getLTR	Displays whether the long term reference (LTR) frame functionality is configured.
LTRBE	iwpriv wifiX LTRBE	Configures long term reference (LTR) functionality for best effort frames.
getLTRBE	iwpriv wifiX getLTRBE	Displays the status of long term reference (LTR) functionality for best effort frames.
LTRBK	iwpriv wifiX LTRBK	Configures long term reference (LTR) functionality for background frames.
getLTRBK	iwpriv wifiX getLTRBK	Displays the status of long term reference (LTR) functionality for background frames.
LTRVO	iwpriv wifiX LTRVO	Configures long term reference (LTR) functionality for voice frames.
getLTRVO	iwpriv wifiX getLTRVO	Displays the status of long term reference (LTR) functionality for voice frames.
LTRVI	iwpriv wifiX LTRVI	Configures long term reference (LTR) functionality for video frames.
getLTRVI	iwpriv wifiX getLTRVI	Displays the status of long term reference (LTR) functionality for video frames.
LTRACTimeout	iwpriv wifiX LTRACTimeout	Configures the latency timeout of access controller for long term reference (LTR) functionality.
getLTRACTimeout	iwpriv wifiX getLTRACTimeout	Displays the latency timeout of access controller for long term reference (LTR) functionality.
LTRTxTimeout	iwpriv wifiX LTRTxTimeout	Configures the Tx inactivity timeout for long term reference (LTR) functionality.
getLTRTxTimeout	iwpriv wifiX getLTRTxTimeout	Displays the Tx inactivity timeout for long term reference (LTR) functionality.
LTRSLEEP	iwpriv wifiX LTRSLEEP	Configures the override of sleep for long term reference (LTR) functionality.
getLTRSLEEP	iwpriv wifiX getLTRSLEEP	Displays the override of sleep for long term reference (LTR) functionality.
LTRRX	iwpriv wifiX LTRRX	Configures the Rx override for long term reference (LTR) functionality.
getLTRRX	iwpriv wifiX getLTRRX	Displays the Rx override for long term reference (LTR) functionality.
L1SS	iwpriv wifiX L1SS	Configures spatial stream for link L1 state of peripheral component interconnect express.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-66 Miscellaneous radio layer parameters in offload mode only (cont.)

Parameter	Command	Description
getL1SS	iwpriv wifiX getL1SS	Displays spatial stream for link L1 state of peripheral component interconnect express.
DSLEEP	iwpriv wifiX DSLEEP	Enables or disables the deep sleep mechanism.
getDSLEEP	iwpriv wifiX getDSLEEP	Displays the deep sleep mechanism.
arpdhcp_ac	iwpriv wifiX arpdhcp_ac	Configures the override of Address Resolution Protocol and Dynamic Host Configuration Protocol for access controller.
g_arpdhcp_ac	iwpriv wifiX g_arpdhcp_ac	Displays the override of Address Resolution Protocol and Dynamic Host Configuration Protocol for access controller.
sIgmpDscpOvrid	iwpriv wifiX sIgmpDscpOvrid	Configures override of Differentiated Services code point (DSCP) values for Internet Group Management Protocol and Multicast Listener Discovery.
gIgmpDscpOvrid	iwpriv wifiX gIgmpDscpOvrid	Displays override of Differentiated Services code point (DSCP) values for Internet Group Management Protocol and Multicast Listener Discovery.
sIgmpDscpTidMap	iwpriv wifiX sIgmpDscpTidMap	Configures traffic identifier map of Differentiated Services code point (DSCP) values for Internet Group Management Protocol and Multicast Listener Discovery.
gIgmpDscpTidMap	iwpriv wifiX gIgmpDscpTidMap	Displays traffic identifier map of Differentiated Services code point (DSCP) values for Internet Group Management Protocol and Multicast Listener Discovery.
setdeauthcnt	iwpriv wifiX setdeauthcnt	Specifies the count of deauthentication frames.
getdeauthcnt	iwpriv wifiX getdeauthcnt	Displays the count of deauthentication frames.
dyn_grouping	iwpriv wifiX dyn_grouping	Configures dynamic grouping of clients.
g_dyngroup	iwpriv wifiX g_dyngroup	Displays dynamic grouping of clients.
dpd_enable	iwpriv wifiX dpd_enable	Configures digital predistortion (DPD) functionality. 0: Disables the DPD 1: Enable and trigger the DPD Note: DPD enable/disable is master controlled through BDF. If user loads the BDF with DPD disabled mode, then this command would not be taken into consideration and the DPD is always disabled.
g_dpdenable	iwpriv wifiX g_dpdenable	Displays digital predistortion functionality status in response to dpd_enable command CLI_DPD_STATUS_DISABLED = 0x0, /* DPD disabled via CLI dpd_enable */ CLI_DPD_STATUS_PASS = 0x1, /* DPD triggered via CLI dpd_enable and calibration passed */ CLI_DPD_CMD_INPROGRES = 0x2, /* DPD triggered via CLI dpd_enable and no response received yet from target */ CLI_DPD_NA_STATE = 0xFE, /* DPD not triggered via CLI command hence invalid/NA state */ CLI_DPD_STATUS_FAIL = 0xFF, /* DPD triggered via CLI dpd_enable & calibration failed OR DPD disabled BDF is loaded*/

Table 3-66 Miscellaneous radio layer parameters in offload mode only (cont.)

Parameter	Command	Description
dbglog_ratelim	iwpriv wifiX dbglog_ ratelim	Configures debugging logs for rate-limiting functionality.
cca_threshold	iwpriv wifiX cca_ threshold	Configures the threshold for Clear Channel Assessment (CCA).
g_cca_threshold	iwpriv wifiX g_cca_ threshold	Displays the threshold for Clear Channel Assessment (CCA).
ani_poll_len	iwpriv wifiX ani_poll_len	Configures the adaptive noise immunity (ANI) polling period.
ani_cck_level	iwpriv wifiX ani_cck_level	Sets the new desired CCK noise immunity level. When this parameter is set, a check is made for the first step noise level to see if it corresponds to the expected value. If it does not, it calls the ani_control() function to set the first step immunity level to the expected values. In addition, it checks if the MRC CCK detection needs to be turned on or off.
ani_listen_len	iwpriv wifiX ani_listen_len	Configures the effect time spent listening for PHY errors since the last PHY error counter check. It subtracts the time spent transmitting or receiving frames from the total cycle count.
ani_ofdm_level	iwpriv wifiX ani_ofdm_ level	Configures the new desired OFDM noise immunity level. A check is made for the current spur immunity level and the firstep noise level, and sees if they correspond to the expected values. If they do not, it calls the ani_control() function to set the spur immunity level and firstep immunity level to the expected values. In addition it checks if the OFDM weak signal detection needs to be turned on or off.
vow_extstats	iwpriv wifiX vow_extstats	Configures the collection of statistical counters for usage of additional buffers for video over wireless extension functionality.
g_vow_extstats	iwpriv wifiX g_vow_ extstats	Displays the collection of statistical counters for usage of additional buffers for video over wireless extension functionality.
arp_dbg_srcaddr	iwpriv wifiX arp_dbg_ srcaddr	Configures debugging of source address for ARP.
g_arp_dbgstats	iwpriv wifiX g_arp_ dbgstats	Displays debugging of source address for ARP.
arp_dbg_dstaddr	iwpriv wifiX arp_dbg_ dstaddr	Configures debugging of destination address for ARP.
arp_dbg_conf	iwpriv wifiX arp_dbg_conf	Specifies debugging configuration for ARP.
g_arp_dbg_conf	iwpriv wifiX g_arp_dbg_ conf	Displays debugging configuration for ARP.
setCH144EppOvrd	iwpriv wifiX setCH144EppOvrd	Specifies the override of EEPROM on channel 144 for transmission purposes for countries in FCC3 and FCC6 domain.
getCH144EppOvrd	iwpriv wifiX getCH144EppOvrd	Displays the override of EEPROM on channel 144 for transmission purposes for countries in FCC3 and FCC6 domain.

Table 3-66 Miscellaneous radio layer parameters in offload mode only (cont.)

Parameter	Command	Description
ol_stats_status	iwpriv wifiX ol_stats_ status	Displays whether the capability to collect offload statistics is enabled. Enter the iwpriv wifiX enable_ol_stats 1 command to enable the collection of offload-mode statistics. Only after the iwpriv wifi0 enable_ol_stats 1 command is issued to enable station statistics, the output of the apstats -s -m command displays information pertaining to AP statistical details. The reset_ol_stats parameter resets the collection of all VAP control path statistics and peer data path statistics for offload radios.
enable_macreq	iwpriv wifiX enable_ macreq	Configures the enabling of MAC requests.
dl_reportsize	iwpriv wifiX dl_reportsize	Configures the size of the debugging log report.
dl_tstamprez	iwpriv wifiX dl_tstamprez	Configures the resolution of timestamp for debugging log.
dl_reporten	iwpriv wifiX dl_reporten	Enables or disables reporting for debugging log.
dl_loglevel	iwpriv wifiX dl_loglevel	Configures the severity level of debugging log.
dl_vapon	iwpriv wifiX dl_vapon	Enables or turns on debugging log for VAPs.
dl_vapoff	iwpriv wifiX dl_vapoff	Disables or turns off debugging log for VAPs.
dl_modon	iwpriv wifiX dl_modon	Enables or turns on debugging log module.
dl_modoff	iwpriv wifiX dl_modoff	Disables or turns off debugging log module.
prof_id_enable	iwpriv wifiX prof_id_ enable	Enables the WLAN profile identifier. This command is supported on IPQ4019 and IPQ8064 platforms only.
prof_trigger	iwpriv wifiX prof_trigger	Configures the trigger for WLAN profile. This command is supported on IPQ4019 and IPQ8064 platforms only.
sHmmcDscpOvrid	iwpriv wifiX sHmmcDscpOvrid	Specifies the override of DSCP for host managed multicast tables.
gHmmcDscpOvrid	iwpriv wifiX gHmmcDscpOvrid	Displays the override of DSCP for host managed multicast tables.
sHmmcDscpTidMap	iwpriv wifiX sHmmcDscpTidMap	Specifies the traffic identifier map of DSCP for host managed multicast tables.
gHmmcDscpTidMap	iwpriv wifiX gHmmcDscpTidMap	Displays the traffic identifier map of DSCP for host managed multicast tables.
get_total_PER	iwpriv wifiX get_total_ PER	Displays the total number of packet error rate.
twice_antgain	iwpriv wifiX twice_antgain	Specifies twice the antenna gain.
set_rxfilter	iwpriv wifiX set_rxfilter	Specifies RX filter. Promiscuous mode must be supported with this API.
setCountry	iwpriv wifiX setCountry	Sets the AP to the regulatory requirements of the country. 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.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-66 Miscellaneous radio layer parameters in offload mode only (cont.)

Parameter	Command	Description
getCountry	iwpriv wifiX getCountry	Retrieves the regulatory domain that is configured.
gpio_config	iwpriv wifiX gpio_config	Specifies the general purpose input/output (GPIO) pin configuration for WLAN LED functionality. This command takes seven arguments as follows: iwpriv wifi0 gpio_config <gpio_num> <pin_dir> <pull_type> <intr_mode> <mux_config> <drive> <internal_config> Example: iwpriv wifi0 gpio_config 29 0 1 1 3 7 0
gpio_output	iwpriv wifiX gpio_output	Specifies the output of the general purpose input/output (GPIO) pin for WLAN LED functionality. This command is used to turn on and off the LED. This command accepts two arguments. Example: iwpriv wifi0 gpio_output 17 0 iwpriv wifi0 gpio_output 17 1
aggr_burst	iwpriv wifiX aggr_burst	Configures the aggregation of frames for short inter-frame space (SIFS) burst.
get_aggr_burst	iwpriv wifiX get_aggr_ burst	Displays the aggregation of frames for short inter-frame space (SIFS) burst.
getBlkReportFld	iwpriv wifiX getBlkReportFld	Displays blacklist report flooding in Hy-Fi networks.
setBlkReportFld	iwpriv wifiX setBlkReportFld	Configures blacklist report flooding in Hy-Fi networks.
getDropSTAQuery	iwpriv wifiX getDropSTAQuery	Displays the number of queries received on an AP from STAs that are dropped.
setDropSTAQuery	iwpriv wifiX setDropSTAQuery	Configures an AP to drop the queries received from STAs.
reset_ol_stats	iwpriv wifiX reset_ol_ stats	Resets the collection of all VAP control path statistics and peer data path statistics for offload radios.
set_fw_hang	iwpriv wifiX set_fw_hang	Specifies the identifier for scenarios in which firmware fails to respond or hangs.
set_fw_recovery	iwpriv wifiX set_fw_ recovery	Specifies the identifier for scenarios in which firmware recovery occurs.
get_fw_recovery	iwpriv wifiX get_fw_ recovery	Retrieves the identifier configured for firmware recovery.
blockdfslist	iwpriv wifiX blockdfslist	Configures the blocking of Dynamic Frequency Selection (DFS) scheduler list.
g_blockdfslist	iwpriv wifiX g_blockdfslist	Displays the scheduler list of blocked DFS.
sifs_frmtype	iwpriv wifiX sifs_frmtype	Configures the frame type for SIFS.
sifs_uapsd	iwpriv wifiX sifs_uapsd	Configures Unscheduled Automatic Power Save Delivery (UAPSD) power save mechanism for SIFS. SIFS frame support is necessary for this command to be supported. This command is specific to QCA9984 platforms and it is not supported on IPQ807x/IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx/QCN90xx or QCN602x + IPQ807xA/IPQ60xx/IPQ50xx platforms.
disable_reset	iwpriv wifiX disable_reset	Disables the reset of firmware.

Table 3-66 Miscellaneous radio layer parameters in offload mode only (cont.)

Parameter	Command	Description
msdu_ttl	iwpriv wifiX msdu_ttl	Configures the time-to-live for MAC service data unit.
ppdu_duration *X*	iwpriv wifiX ppdu_ duration *X*	Configures the time period for packet protocol data units, where *X* is the duration in microseconds (default is 2000)
txbf_snd_int	iwpriv wifiX txbf_snd_int	Configures the period of transmission for Tx beamforming.
get_txbf_snd	iwpriv wifiX get_txbf_snd	Displays the period of transmission for Tx beamforming.
promisc	iwpriv wifiX promisc	Enables or disables the promiscuous mode on offload radios.
get_promisc	iwpriv wifiX get_promisc	Displays the status of promiscuous mode on offload radios.
burst_mode	iwpriv wifiX burst_mode	Configures the burst mode.
obss_rssi_th	iwpriv wifiX obss_rssi_th	Configures RSSI threshold for other BSS.
gobss_rssi_th	iwpriv wifiX gobss_rssi_th	Displays the configured RSSI threshold for other BSS.
obss_rxrssi_th	iwpriv wifiX obss_rxrssi_ th	Configures Rx RSSI threshold for other BSS.
gobss_rxrssi_th	iwpriv wifiX gobss_rxrssi_ th	Displays the configured Rx RSSI threshold for other BSS.
print_ratelim	iwpriv wifiX print_ratelim	Configures the capability to display or print rate limit.
gprint_ratelim	iwpriv wifiX gprint_ratelim	Displays whether the capability to display or print rate limit is configured.
pdev_reset	iwpriv wifiX pdev_reset	Configures the reset of physical device (PDEV).
set_nohostcrash	iwpriv wifiX set_ nohostcrash	Specifies the creation of firmware dump for no-host crashes.
get_nohostcrash	iwpriv wifiX get_ nohostcrash	Displays the status of creation of firmware dump for no-host crashes.
obss_longslot	iwpriv wifiX obss_longslot	Considers OBSS for non-effective radiated power (ERP) stations using long slot time.
g_obss_longslot	iwpriv wifiX g_obss_ longslot	Displays whether OBSS is considered for non-effective radiated power (ERP) stations using long slot time.
fc_stats_global	iwpriv wifiX fc_stats_ global	Configures the collection of per-radio global TID queue statistics.
fc_peer_stats	iwpriv wifiX fc_peer_stats	Configures the collection of per-peer global TID queue statistics.
fc_flush_intv	iwpriv wifiX fc_flush_intv	Configures the flush interval as part of Peer Flow control and Queue Map parameters.
g_fc_flush_intv	iwpriv wifiX g_fc_flush_ intv	Displays the configured queue flush interval for peer flow control and queue map.
fc_buf_max	iwpriv wifiX fc_buf_max	Configures the maximum size of the buffers for peer flow control and queue map.
fc_buf0_max	iwpriv wifiX fc_buf0_max	Configures the maximum size of the buffer 0 for peer flow control and queue map. This command is used to set the Max Buffer Queue Size for Wi-Fi Tx Queue (used in peer based flow control). In NSS Offload mode, this prevents NSS hoarding buffers greater than the number configured for a given peer range.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-66 Miscellaneous radio layer parameters in offload mode only (cont.)

Parameter	Command	Description
fc_buf1_max	iwpriv wifiX fc_buf1_max	Configures the maximum size of the buffer 1 for peer flow control and queue map.
fc_buf2_max	iwpriv wifiX fc_buf2_max	Configures the maximum size of the buffer 2 for peer flow control and queue map.
fc_buf3_max	iwpriv wifiX fc_buf3_max	Configures the maximum size of the buffer 3 for peer flow control and queue map.
g_fc_buf_max	iwpriv wifiX g_fc_buf_ max	Retrieves the maximum size of the buffer for peer flow control and queue map.
fc_q_max	iwpriv wifiX fc_q_max	Configures the maximum size of the queue for peer flow control and queue map.
g_fc_q_max	iwpriv wifiX g_fc_q_max	Retrieves the maximum size of the queue for peer flow control and queue map.
fc_buf_min	iwpriv wifiX fc_buf_min	Configures the minimum size of the queue for peer flow control and queue map.
g_fc_buf_min	iwpriv wifiX g_fc_buf_min	Retrieves the minimum size of the queue for peer flow control and queue map.
fc_q_min	iwpriv wifiX fc_q_min	Configures the minimum size of the queue for peer flow control and queue map.
g_fc_q_min	iwpriv wifiX g_fc_q_min	Retrieves the minimum size of the queue for peer flow control and queue map.
fc_cong_timer	iwpriv wifiX fc_cong_ timer	Configures the flow control configuration timer interval.
g_fc_cong_timer	iwpriv wifiX g_fc_cong_ timer	Displays the flow control configuration timer interval.
fc_stats_tmr	iwpriv wifiX fc_stats_tmr	Specifies the flow control statistics timer interval.
g_fc_stats_tmr	iwpriv wifiX g_fc_stats_ tmr	Displays the flow control statistics timer interval.
fc_rot_timer	iwpriv wifiX fc_rot_timer	Specifies the flow control rotting timer interval.
g_fc_rot_timer	iwpriv wifiX g_fc_rot_ timer	Displays the flow control rotting timer interval.
fc_latency_on	iwpriv wifiX fc_latency_on	Enables or disables latency for flow control.
fc_hostq_dump	iwpriv wifiX fc_hostq_ dump	Specifies the generation of host queue dump for flow control.
fc_tidq_map	iwpriv wifiX fc_tidq_map	Specifies the generation of traffic identifier queue dump for flow control.
st_ht_noamsdu	iwpriv wifiX st_ht_ noamsdu	Disables AMSDU for STA VAP for high throughput. Disabling AMSDU in STA for uplink traffic is only applicable to QCA980 and QCA9984 chipsets, and not in IPQ807x in any Lithium-based chipsets. Use the rx_amsdu_ptid parameter for disabling AMSDU in STA for uplink traffic on IPQ807x chipsets.
g_st_ht_noamsdu	iwpriv wifiX g_st_ht_ noamsdu	Displays whether AMSDU for STA VAP for high throughput is disabled or not.
staDFSEn	iwpriv wifiX staDFSEn	Enables or disables dynamic frequency selection of STA.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-66 Miscellaneous radio layer parameters in offload mode only (cont.)

Parameter	Command	Description
get_staDFSEn	iwpriv wifiX get_ staDFSEn	Displays the status of dynamic frequency selection of STA.
CSwOpts	iwpriv wifiX CSwOpts	Specifies the configuration of channel switch options. ■ CSwOpts 0x1 - Random channel selection chooses only from non-DFS channels present in the channel list. ■ CSwOpts 0x2 - If the CSA received by the repeater has a DFS target channel, ignore the CSA and choose a random channel locally. ■ CSwOpts 0x4 - When the STA vap of the repeater brings up the AP vap, the AP vap should do CAC. ■ CSwOpts 0x8 - When the STA vap of the repeater brings up the AP vap, the AP vap should send CSA. ■ CSwOpts 0x10 - Repeater sends RCSA to uplink when it detects a RADAR or it receives RCSA. ■ CSwOpts 0x20 - Process the received RCSA. ■ CSwOpts 0x40 - On RADAR detect go the apriori channel. ■ CSwOpts 0x80 - When bringing up the Repeater AP vaps, skip the CAC, if the RootAP's new channel is subset of old channel. A user can enter more than one value to combine different channel switch options.
get_CSwOpts	iwpriv wifiX get_CSwOpts	Displays whether configuration of Channel switch options is enabled.
tpdbscale	iwpriv wifiX tpdbscale	Configures the Tx power db scaling.
emiwar80p80	iwpriv wifiX emiwar80p80	Configures electromagnetic interference workaround for 80 + 80 MHz channel. Enable the compilation flag QCA_EMIWAR_80P80_ CONFIG_SUPPORT (which is disabled by default) for this command to work.
get_emiwar80p80	iwpriv wifiX get_ emiwar80p80	Displays the configured electromagnetic interference workaround for 80 + 80 MHz channel. Enable the compilation flag QCA_EMIWAR_80P80_ CONFIG_SUPPORT (which is disabled by default) for this command to work.
batch_mode	iwpriv wifiX batch_mode	Configures batch mode for offload optimized data path. This parameter enables dynamic software rate based interrupt mitigation for incoming packets, coming from the target to host. The design for batch mode is implemented in the firmware. On the host, the macro is used for enabling or disabling the feature.
packaggrdelay	iwpriv wifiX packaggrdelay	Configures packet aggregation delay.
CalCheck	iwpriv wifiX CalCheck	Specifies the verification of calibration mechanism version.
get_CalCheck	iwpriv wifiX get_ CalCheck	Displays whether the verification of calibration mechanism version is specified.

Table 3-66 Miscellaneous radio layer parameters in offload mode only (cont.)

Parameter	Command	Description
set_pcp_tid_map	iwpriv wifiX set_pcp_tid_map	Configures the traffic identifier (TID) map for Port Control Protocol (PCP). This parameter is applicable for IPQ401x/IPQ402x only and not for IPQ807x and QCA9984 platforms.
get_pcp_tid_map	iwpriv wifiX get_pcp_tid_map	Configures the traffic identifier (TID) map for Port Control Protocol (PCP). This parameter is applicable for IPQ401x/IPQ402x only and not for IPQ807x and QCA9984 platforms.
set_tidmap_prec	iwpriv wifiX set_tidmap_prec	Configures the traffic map identifier precedence.
get_tidmap_prec	iwpriv wifiX get_tidmap_prec	Retrieves the configured traffic map identifier precedence.
bsta_fixed_idmask	iwpriv wifiX bsta_fixed_idmask	Configures a fixed ID mask for backhaul STA.
enable_amsdu	enable_amsdu	Enables (1) or disables (0) Tx aggregated MAC service data unit (AMSDU) aggregation.
enable_ampdu	enable_ampdu	Enables (1) or disables (0) Tx aggregated MAC protocol data unit (AMPDU) aggregation.
g_activity_f	g_activity_f	Retrieves the flow activity.
mark_subchan	mark_subchan	Enables (1) or disables (0) subchannel marking to add only radar affected subchannels to the NOL; else, all subchannels will be added to the current channel to NOL, regardless of the frequency offset provided.
g_mark_subchan	iwpriv wifiX g_mark_subchan	Retrieves the status of subchannel marking configuration.
bw_reduceEn	iwpriv wifiX bw_reduceEn	Enables or disables reduced bandwidth mechanism.
get_bw_reduceEn	iwpriv wifiX get_bw_reduceEn	Displays whether reduced bandwidth mechanism is enabled.
get_he_ul_ppdu_dur	iwpriv wifiX get_he_ul_ppdu_dur	Retrieves the PPDU duration in uplink direction for high efficiency (HE) functionality.
he_ru_alloc	iwpriv wifiX he_ru_alloc	Enables or disables RU allocation for HE.
get_he_ru_alloc	iwpriv wifiX get_he_ru_alloc	Displays the current status of RU allocation for HE.
flush_rate_stats	iwpriv wifiX flush_rate_stats	Clears the Tx and Rx rate statistical information.
set_dcs_enable_timer	iwpriv wifiX set_dcs_enable_timer	Specifies the timer for dynamic channel selection.
get_dcs_enable_timer	iwpriv wifiX get_dcs_enable_timer	Retrieves the timer value for dynamic channel selection.
wifi_down_ind	iwpriv wifiX wifi_down_ind	Brings down the Wi-Fi interface on independent repeater.
enable_wmi_dis_dump	iwpriv wifiX enable_wmi_dis_dump	Enables the WMI distribution dump.
acs_chan_grade_algo	iwpriv wifiX acs_chan_grade_algo	Specifies the channel grade algorithm for ACS.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-66 Miscellaneous radio layer parameters in offload mode only (cont.)

Parameter	Command	Description
get_acs_chan_grade_ algo	iwpriv wifiX get_acs_ chan_grade_algo	Displays the configured channel grade algorithm for ACS.
set_acs_chan_eff_var	iwpriv wifiX set_acs_ chan_eff_var	Configures the ACS channel efficiency variable.
get_acs_chan_eff_var	iwpriv wifiX get_acs_ chan_eff_var	Displays the configured ACS channel efficiency variable.
set_acs_near_range_ weightage	iwpriv wifiX set_acs_ near_range_weightage	Specifies the ACS near-range weightage value.
get_acs_near_range_ weightage	iwpriv wifiX get_acs_ near_range_weightage	Displays the configured ACS near-range weightage value.
set_acs_mid_range_ weightage	iwpriv wifiX set_acs_mid_ range_weightage	Specifies the ACS mid-range weightage value.
get_acs_mid_range_ weightage	iwpriv wifiX get_acs_ mid_range_weightage	Displays the configured ACS mid-range weightage value.
set_acs_far_range_ weightage	iwpriv wifiX set_acs_far_ range_weightage	Specifies the ACS far-range weightage value.
get_acs_far_range_ weightage	iwpriv wifiX get_acs_far_ range_weightage	Displays the configured ACS far-range weightage value.
set_mgmt_ttl	iwpriv wifiX set_mgmt_ttl	Specifies the time-to-live value for management frames.
get_mgmt_ttl	iwpriv wifiX get_mgmt_ttl	Displays the time-to-live value for management frames.
set_prb_rsp_ttl	iwpriv wifiX set_prb_rsp_ ttl	Specifies the time-to-live value for probe responses.
get_prb_rsp_ttl	iwpriv wifiX get_prb_rsp_ ttl	Displays the time-to-live value for probe responses.
set_mu_ppdu_dur	iwpriv wifiX set_mu_ ppdu_dur	Specifies the multi PPDU duration.
get_mu_ppdu_dur	iwpriv wifiX get_mu_ ppdu_dur	Displays the configured multi-user PPDU duration.
set_tbtt_ctrl	iwpriv wifiX set_tbtt_ctrl	Enables/disables calculation of target beacon transmission time (TBTT) offset for control frames.
get_tbtt_ctrl	iwpriv wifiX get_tbtt_ctrl	Displays the target beacon transmission time (TBTT) offset value for control frames.
get_rchwidth	iwpriv wifiX get_rchwidth	Displays the receive channel width.
nav_config	iwpriv wifiX nav_config	Specifies the network allocation vector configuration.
get_nav_config	iwpriv wifiX get_nav_ config	Displays the current state of the network allocation vector configuration.

3.72 MBO parameters

Multiband operation (MBO) facilitates efficient use of multiple frequency bands and channels that are available to access points (APs), and the wireless devices that associate with them (stations). MBO offers solutions for band steering, load balancing, and other related operational procedures. The features defined for MBO are band-independent and apply across all available unlicensed

bands where Wi-Fi systems operate. With exchange of information between APs and STAs, intelligent decisions are made to collectively drive the network towards more efficient use of the available spectrum.

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Wi-Fi Alliance MBO functions* section of the *Beacons and frames transmission* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

3.72.1 MBO Information Element (IE)

It is included by the AP in beacon frame, probe response frame, BTM request frames and (re-) association frames to indicate support for MBO. The element ID is 0xDD which is included in the first byte of the IE. If the receiving STA does not support MBO, it will not recognize the element ID and it can use the length field value contained in the second byte of the IE to get to the end of the MBO IE and skip it altogether. This IE is included in the frames before including any MBO attribute IE.

3.72.2 MBO Attributes IE

All Attribute IEs have a common format: First is the Attribute ID field which is unique for each attribute, next is the length field which denotes the number of bytes more that are part of that particular attribute's IE and lastly there is (are) the byte(s) that denote some specific value for that attribute. This third field is named differently depending on the attribute.

3.72.3 MBO Capability Indication Attribute included in beacon, probe response, and reassociation frames

The IE related to this attribute allows an MBO AP to denote that it is MBO capable and which features are enabled. The valid values for the third field of this IE (otherwise known as capability indication in this case) are as follows. For this particular IE, values are given bit position wise and compatibility with both big endian and little endian systems has been ensured:

The following list describes the bit positions and their corresponding usages:

- 0-Reserved bit.
- 1-If this bit is equal to 1, Non-Preferred Channel Report capability is enabled.
- 2-If this bit is equal to 1, MBO Cellular Capabilities is enabled.
- 3-If this bit is equal to 1, Cellular Data Connection Preference is enabled.
- 4-If this bit is equal to 1, Cellular Data Link Request is enabled.
- 5-7-Reserved bits.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

3.72.4 MBO Association Disallowed Attributes included in beacon, probe response, and reassociation frames

The presence of the MBO IE with the Association Disallowed Attribute in any of the beacon, probe response, or ressociation Response frames are interpreted as an indication that the MBO AP is currently not accepting associations. Sending this IE along with the MBO IE implies that an MBO STA receiving it, must not send a ressociation Request frame or a unicast Probe Request frame to that AP until it receives a Beacon or Probe Response frame from that AP without the Association Disallowed Attribute.

Valid values for reason code of this IE are as follows:

- 0-Reserved
- 1-Unspecified Reason
- 2-Maximum number of associated STAs reached
- 3-Air interface overloaded
- 4-Authentication server overloaded
- 5-Insufficient RSSI
- 6-255-Reserved

Table 3-67 MBO parameters

Parameter	Command	Description
mbo 1	iwpriv athX mbo 1	This command is used to enable MBO. After this command is configured, MBO IEs are observed in the beacon frames, probe response frames, (re-)association frames and in BTM req frames (solicited or unsolicited). The MBO IE consists of the following fields: ■ Elem id: 0xDD ■ Length: number of bytes till the end of MBO IE including the IE of all the attributes. If address of this byte is added to the value it holds then we obtain the address to the last byte of the MBO IE. ■ OUI: 0x50-6F-9A ■ OUI Type: 0x16 The MBO IE must be necessarily enabled to send MBO attribute IEs. Otherwise, the attribute IEs is not seen in the frames that are supposed to carry MBO related information.
mbo 0	iwpriv athX mbo 0	This command is used to disable MBO. MBO IEs and all MBO related attribute IEs are no more sent out in the frames.
g_mbo	iwpriv athX g_mbo	This command is used to obtain set value of MBO. Ether 0 or 1 is returned depending on whether MBO is disabled or enabled respectively.

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafei@qualcomm.com.cn

Table 3-67 MBO parameters (cont.)

Parameter	Command	Description
mbocap 1	iwpriv athX mbocap 1	Enable and send out MBO Capability IEs in the beacon, probe response, and reassociation frames. The MBO Capability IE consists of the following fields: ■ Attribute ID: 0x01 ■ Length: 0x01 ■ Capability indication value: The default value of 0x70 is sent out.
mbocap 0	iwpriv athX mbocap 0	Disable MBO Capability IE.
mbocap <value>	iwpriv athX mbocap <value>	This command also serves to send out MBO Capability IEs in the beacon, probe response, and reassociation frames. However, the default value of 0x70 for capability indication field is replaced with the user set value. If the user set value is invalid (that is, if reserved bits are set), then an error message is displayed stating the valid and invalid bits.
mbo_asoc_dis <0 | 1-5>	iwpriv athX mbo_asoc_dis <value>	Enable (1) or disable (0) association disallowed attribute and set a reason code value. This IE appears in beacon, probe response and (re-)association frames as follows: ■ Attribute ID: 0x04 ■ Length: 0x01 ■ Reason code: Set to <0 | 1-5> if the value specified is valid. Otherwise, an error message is displayed prompting the user to set correct values. If value is 0, the association disallowed attribute is disabled and this IE is not sent out along with the frames.
g_mbo_asoc_dis	iwpriv athX g_mbo_asoc_dis	Obtain reason code value set for association disallowed attribute. If association disallowed attribute has not been enabled, it returns zero.
mbo_cel_pref <value>	iwpriv athX mbo_cel_pref <value>	The BTM Request frame may contain the Cellular Data Connection Preference Attribute, alone or in addition to the Neighbor Report List. Use of the Cellular Data Connection Preference Attribute enables the MBO AP to indicate its preference to the MBO STA for the MBO STA to move its data traffic from WLAN to cellular. If included with a Neighbor Report List, the MBO AP can indicate the relative preference for various base station subsystems (BSSs), channels and the cellular data connection. The values of the Cellular Data Connectivity field are defined as follows: ■ 0—Excluded the attribute because the MBO AP does not want the MBO STA to use the cellular data connection ■ 1—Note use the attribute; the MBO AP prefers the MBO STA must not use cellular. ■ 2-254—Reserved ■ 255—Use the attribute; the MBO AP prefers the MBO STA must use cellular.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-67 MBO parameters (cont.)

Parameter	Command	Description
mbo_trans_rs <value>	iwpriv athX mbo_trans_rs <value>	The MBO Reason Code Attribute is provided by an MBO AP in the BSS Transition Request frame to indicate the reason for the transition request. This attribute is mandatory in the BSS Transition Request and may be used by an MBO STA to determine its next action after receiving the BSS Transition Request from the MBO AP. The values of the transition reason code are defined as follows: ■ 0—Unspecified ■ 1—Excessive frame loss rate ■ 2—Excessive delay for current traffic stream ■ 3—Insufficient bandwidth for current traffic stream ■ 4—Load balancing ■ 5—Low RSSI ■ 6—Received excessive number of retransmissions ■ 7—High interference ■ 8—Gray zone Imbalance between the PHY operating margin in the downlink direction compared to the uplink direction can result in a “gray” zone of coverage in which MBO STAs can become “stalled” in certain states such as: □ Mobile is 802.11 authenticated, but not associated □ Mobile is 802.11 associated, but not EAP authenticated □ Mobile is unable to obtain an IP address via DHCP □ Mobile is unable to perform name resolution via DNS ■ If the serving MBO AP can detect that an MBO STA is in a gray zone, it should try to have the device transition. ■ 9—Transitioning to a Premium AP ■ 10-255—Reserved
mbo_asoc_ret <value>	iwpriv athX mbo_asoc_ret <value>	If an MBO STA has received an unsolicited BTM Request frame that contains an MBO IE with an Association Retry Delay Attribute indicating a non-zero Re-association Delay field, the prescribed MBO STA shall not attempt to associate with this BSS, after it has been disassociated, for the duration of the specified period. The value that you define using this command indicates the Reassociation Delay field of the Association Retry Delay Attribute. This value specifies the number of seconds which the MBO STA must wait before trying to (re-)associate with the BSS. This timer starts when the BTM Request containing this field is received.

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:55:41 UTC pan.yafeng@zte.com.cn

Table 3-67 MBO parameters (cont.)

Parameter	Command	Description
g_mbo_cell_pref g_mbo_cel_pref	iwpriv athX g_mbo_cell_pref cfg80211tool athX g_mbo_cel_pref	Retrieves the Cellular Data Connection Preference attribute. This command is used to obtain value for cellular data connection preference set by the user in the iwpriv athX mbo_cel_pref command. If a value is not set, it returns zero. Consider a scenario in which the MBO STA informs its serving AP whether or not it has cellular support and updates the serving AP of the same in case there is a change in such a situation, the MBO AP will include or exclude MBO - cellular data connection preference IE from the BTM request frame for the recipient of that frame.
g_mbo_trans_rs	iwpriv athX g_mbo_trans_rs	Retrieves the Transition Reason Code attribute. This command is used to obtain value for transition reason code that the user sets in the iwpriv athX mbo_trans_rs command. If a value is not set, it returns zero.
g_mbo_asoc_ret	iwpriv athX g_mbo_asoc_ret	Retrieves the Association Retry Delay attribute. This command is used to obtain value for association retry delay that the user sets in the iwpriv athX mbo_asoc_ret command. If a value is not set, it returns zero.

3.73 Neighbor report parameters

A response is sent only for the neighbor report (NR) information that matches SSID in NR request from STA. If no SSID is presented in the NR request, AP filters the current SSID with which the STA associates and responds to STA. NR response must include NR information collected from all the radios with same SSID.

For example, if the NR response is for all three radios in a tri-radio, the scan table of each VAP of each radio is iterated, NR information is populated, and this derived NR statistic is sent to STA.

If shared flag is set and if no SSID presented in the NR request, then AP filters with current SSID with which the STA is associated and responds to STA. The maximum NR response packet length check is implemented, based on the maximum management frame length (MAX_TX_RX_PACKET_SIZE) to avoid packet boundary overflow. NR response fills until the maximum packet length size.

For more information on usage guidelines and the functionality regarding these parameters, see the *Customized neighbor report* section of the *Beacons and frames transmission* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-68 Neighbor report parameters

Parameter	Command	Description
nrshareflag	iwpriv athN nrshareflag FLAG	Specify how the neighbor report (NR) flag is shared between the wifi interfaces (that is, the two or three radios for which the NR response applies). Bit [0~2] indicates the scan table of the corresponding radio (wifi0 ~ wifi2); denotes whether it is shared or not. 0, 1, 2, 4: Not shared. 3: Shared between wifi0 and wifi1 5: Shared between wifi0 and wifi2 6: Shared between wifi1 and wifi2 7: Shared cross wifi0, wifi1 and wifi2 >7 && < 0xff Invalid. Input will be refused : 0xff Disable the FR functionality
get_neshareflag	iwpriv athN get_ nrshareflag	Get current NR share flag.

3.74 Target core dump parameter

The functionality to generate runtime target core dump is available. At runtime of an AP, users can generate a dump of the target core into a file without impacting the working of the AP. The core dump is used for troubleshooting errors and diagnosis purposes. The core dump at runtime is exactly the same as when a target assert happens.

For more information on usage guidelines and the functionality regarding these parameters, see the *Generate target core dump* section of the *Using logs and diagnostic tools for debugging* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-69 Target core dump parameter

Parameter	Command	Description
dump_target 1	iwpriv wifi0 dump_target 1	Dump target core.

3.75 Connection state logging parameter

For more information on usage guidelines and the functionality regarding these parameters, see the *Connection state logging functionality* section of the *Using logs and diagnostic tools for debugging* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Connection state logging (CSL) is a print-level debug framework to aid to detect connection level issues automatically without human intervention. A user-configurable (enable/disable control for CSL) is introduced. Enter the `iwpriv ath0 csl <1/0>` command to enable (1) or disable (0) this logging mechanism. Enabling CSL should enable IEEE80211_AUTH_MSG and IEEE80211_ASSOC_MSG debug flags.

CSL is supported for the following operations:

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

- Probe-related (Request and Response)
- Authentication-related (Request and Response)
- Association-related (Request and Response)
- Security-related (Handshake, Failure, Pass)

All WLAN frames involved in connection (probe requests and responses, authentication requests and responses, and association requests and responses) are logged.

Table 3-70 CSL parameter

Parameter	Command	Description
csl <1 | 0>	iwpriv ath0 csl <1 |0>	Enable (1) or disable (0) the connection state logging mechanism for troubleshooting and diagnosis purposes. Enabling CSL should enable IEEE80211_AUTH_MSG and IEEE80211_ASSOC_MSG debug flags.

3.76 Tx data packet capture parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Tx data packet captures in OL mode* section of the *Beacons and frames transmission* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:00:00 PDT pan.yafeng@zte.com.cn

Table 3-71 Tx data packet capture parameters

Parameter	Command	DA	OL	Description
tx_capture_da <1 | 0>	iwpriv ath*X* tx_capture_da <1 | 0>	Y	N	Enable (1) or disable (0) the support on the monitor VAP to mirror Tx packets in DA mode. When interception of Tx packets or monitoring of Tx packets is enabled, Tx packets of all data VAPs are mirrored to the monitor VAP. Besides Tx packets, the monitor VAP also captures Rx packets in promiscuous mode. If a user starts capturing the packets using the Tcpdump utility and then saves them in a .pcap file for viewing it later using a Wireshark tool, captured packets can be viewed on a remote Linux PC running Wireshark tool as and when packets are being received in network stack of AP. **NOTE** Before you configure the capability to mirror Tx packets sent from the AP VAP to the monitor VAP, configure regular DATA VAPs (such as ath0, ath1, and ath2) and configure the monitor VAP (such as athx) on the same radio.
tx_capture <1 | 0> g_tx_capture	iwpriv ath*X* tx_capture <1 | 0> iwpriv athX g_tx_capture	N	Y	Enable (1) or disable (0) the feature to mirror Tx data packets in OL mode. When interception of Tx packets or monitoring of Tx packets is enabled, Tx packets of all data VAPs are mirrored to the monitor VAP. Besides Tx packets, the monitor VAP also captures Rx packets in promiscuous mode. In association, dissociation and reassociation responses, and authentication and reauthentication responses, the mgmt completion handler gets the completion message with the PPDU_ID contained in it. Examine the frame type to confirm that the received message frame is the Deauth response frame. The pktlog TX_STAT message carries an event with the metadata information and the same PPDU_ID. The get command is used to check whether tx capture is enabled or not.

3.77 QCN broadcast probe response parameters

The broadcast probe response capability is supported for Qualcomm (QCN)-based access points (APs) and stations. The following are the input/output controls (IOCTLs) introduced in the form of iwpriv commands to support the QCN broadcast probe response functionality.

For more information on usage guidelines and the functionality regarding these parameters, see the *QCN broadcast probe response* section of the *Beacons and frames transmission* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-72 QCN broadcast probe response parameters

Parameter	Command	Description
set_bpr_enable 0/1	iwpriv athX set_bpr_enable 0/1	Enable the BPR feature and ensure for QCN STA, either broadcast response or beacon is sent and not unicast response. The default is 1 (enabled). 0 denotes disabling BCAST probe response feature and 1 denotes enabling BCAST probe response
get_bpr_enable	iwpriv athX get_bpr_enable	Retrieves the BPR feature as set by the user.
bpr_delay [1-255]	iwpriv athX bpr_delay [1-255]	Denotes the duration, in ms, for which the AP holds BCAST probe response. The default is 50 ms; the range is 1–255 ms.
get_bpr_delay	iwpriv athX get_bpr_delay	Retrieve the bpr_delay value
set_bpr_latency [1-10]	iwpriv athX set_bpr_latency [1-10]	Denotes before how much time (in ms) should the AP send the broadcast response before the channel time of STA expires. This parameter is configured per radio because it is the same across all VAPs per radio. The default is 5ms; the range is between 1 and 10.
get_bpr_latency	iwpriv athX get_bpr_latency	Retrieve the bpr_latency value.
set_bcn_latency [5-20]	iwpriv athX set_bcn_latency [5-20]	Denotes the time (in ms) to compensate for the delay caused due to beacon sent from the driver to firmware. This parameter is configured per radio because it is the same across all VAPs per radio. The default is 5ms; the range is between 5 and 20.
get_bcn_latency	iwpriv athX get_bcn_latency	Retrieve the bcn_latency value

clr_bpr_stats	iwpriv athX clr_bpr_stats	Clears all the timer-related counters.
		The sample output is as follows:
		`BPR timer start count - 0`
		`BPR timer resize count - 0`
		`BPR timer callback count - 0`
		`BPR timer cancel count - 0`

get_bpr_stats	iwpriv athX get_bpr_stats	Prints the summary of all the data
		The sample output is as follows:
		`BPR feature disabled - 0`
		`BPR Latency compensation - 5 ms`
		`Beacon Latency compensation- 5 ms`
		`BPR delay - 50 mS`
		`Current Timestamp - 16179117804480`
		`Next beacon Timestamp - 16179126788320`
		`BPR timer start count - 20`
		`BPR timer resize count - 2`
		`BPR timer callback count - 10`
		`BPR timer cancel count - 7`

3.78 Copy Engine logging parameters

The copy engine (CE) logging capability enables logging of information relating to CE descriptors and associated buffers. In addition, this functionality also supports framework for Smart Logging in which fatal events from FW are recorded or dumped.

This CE logging functionality provides a mechanism to enable and disable the Smart Logging feature. A user can collect and store the history of CE descriptors and fixed 'N' bytes of associated buffers. Also, FW fatal event messages are recorded when the Smart Logging functionality is enabled.

The sysfs file interface is used for dumping FW fatal logs and current CE descriptors and associated data buffers. In addition, the sysfs file interface is used for dumping CE descriptors and fixed 'N' bytes of associated data buffers.

For CE debugging, user can save all the history of CE descriptors and the fixed n bytes of the buffer pointed by it on to the console. For Smart logging, the FW logs and current CE descriptors with N bytes of data buffer pointed by it can be dumped by the user on the console.

The FW logs are stored in a Smart Log memory area, whenever a Smart Log event comes from the FW, and can be dumped on the console by the user.

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *CE logging functionality* section of the *Using logs and diagnostic tools for debugging* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 13:13:13 PDT zhan.yafeng@zte.com.cn

Table 3-73 CE logging parameters

Parameter	Command	Description

smartLogSkbsz <size>	iwpriv wifiX smartLogSkbsz <size>	For dumping all smart logs that contain FW logs and SmartLog events, set the SKB size of the associated buffer that needs to be stored by entering the following commands:
		`$iwpriv wifiX smartLogSkbsz <size>`
		`$cat /sys/class/net/wifiX/smartlogs_dump`

ce_debug_en <1/0>	iwpriv wifiX ce_debug_en <1/0>	For dumping history of CE descriptors and the fixed N bytes of buffer pointed by it on console, enable CE debug logs and dump the CE buffer history on the console by entering the following commands:
		`$iwpriv wifiX ce_debug_en <1/0>`
		`$cat /sys/class/net/wifiX/celogs_dump`

g_ce_debug_en	iwpriv wifiX g_ce_debug_en	Displays whether the capability to generate copy engine (CE) debug logs is enabled or not.

Table 3-73 CE logging parameters (cont.)

Parameter	Command	Description
smartLogEn <1/0>	iwpriv wifiX smartLogEn <1/0>	Enables (1) or disables (0) the smart logging mechanism. An event-based logging framework is introduced to reduce the effort and time consumed in reproducing problems or defects for test teams and customers by collecting all the necessary logs for various possible error conditions. These events can be scenarios such as copy engine failure, Tx/Rx not working, connection failure, and ping failure. The trigger-based logging or smart logging framework (also called scenario-based logging) is responsible for the following: Identification of the trigger conditions Collection of necessary debug information, for the trigger condition, by interacting with Host/FW Flushing of any residual log (in the internal log buffers) in the Host/FW
get_smartLogEn	iwpriv wifiX get_smartLogEn	Displays whether he smart logging mechanism is enabled or not.
smartLogEvent <1 | 0>	iwpriv wifiX smartLogEvent <1 | 0>	Enables (1) or disables (0) the logging of fatal events using the smart logging mechanism
arp_dbg_srcaddr <addr> arp_dbg_dstaddr <addr>	iwpriv wifi0 arp_dbg_srcaddr <addr> iwpriv wifi0 arp_dbg_dstaddr <addr>	Analyzing the history of P1 ping failures, it has been concluded that a debugging mechanism is necessary to log all the ARP requests and its response. This could be implemented on both the AP and STA cases for all peers. Currently a logging mechanism for certain ARP packets exist which uses iwpriv commands to enable logging. The existing implementation is a filtered logging, which would only be done for configured source and destination address pair: Design enhancements are made to the existing framework in the case of P1 ping failure for smart logging. As this feature involves packet inspection, it can affect normal performance. Since this logging will be enabled only at the time of debugging, the performance dip does not have much impact.
slPingFailLog 1 | 0 g_slPingFailLog	iwpriv wifiX slPingFailLog 1 | 0 iwpriv wifiX g_slPingFailLog	Command to start logging of P1 ping failures: iwpriv wifiX slPingFailLog 1 Command to stop logging of P1 ping failures: iwpriv wifiX slPingFailLog 0 Command to get status of he configuration of P1 ping failures: iwpriv wifiX g_slPingFailLog

3.79 FILS parameter

Support has been introduced for compatibility with the IEEE 802.11ai protocol to enable the implementation of fast initial link setup (FILS) capability on the proprietary Wi-Fi driver and the hostapd application.

FILS protocol defined in 802.11ai specification is helpful to reduce the connection delay. WLAN initial connection delay is reduced without any compromise on security. FILS uses EAP-RP protocol for authentication and key derivation. Association to a FILS-capable AP always occurs through a full extensible authentication protocol (EAP) exchange. The subsequent connections use the FILS authentication algorithm to minimize the delay in connection.

For more information on usage guidelines and the functionality regarding these parameters, see the *FILS capability* and *FILS Discovery frames* sections of the *Performance enhancement capabilities* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-74 FILS parameter

Parameter	Command	Description
enable_fils <1 | 0> g_enable_fils	iwpriv athX enable_fils <1 | 0> iwpriv athX g_enable_fils	To enable FILS capability, enter the iwpriv athX enable_fils 1 command. To disable FILS capability, enter the iwpriv athX enable_fils 0 command. To retrieve the configured value and display whether FILS capability is enabled or not, enter the iwpriv athX g_enable_fils command. This option or parameter is added in the driver to provide backward compatibility support. In certain scenarios, the old driver (without FILS capability) is used with new hostapd (with FILS support). When the enable_fils parameter is entered, hostapd queries the driver for FILS support and stops performing FILS specific operations such as IEEE80211_MLME_AUTH_FILS for drivers that do not support FILS. This command is recommended to be used when FILS is configured for the VAP.

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57 pan.yafeng@zte.com.cn

Table 3-74 FILS parameter (cont.)

Parameter	Command	Description
enable_fils <1 | 0> <period> g_enable_fils	iwpriv athX enable_fils <1 | 0> <period> iwpriv athX g_enable_fils	Enable FILS capability and also configure the period or time units (TUs) of FILS discovery (FD) frames. To configure the period (TUs) of FD frames, enter the `iwpriv <vap> enable_fils <1/0> <period>` command. ■ vap—VAP name ■ 1/0—1 to enable, 0 to disable ■ Period—Time period between each FD frames in TUs. The following are examples of FD period configuration: ■ `#iwpriv ath0 enable_fils 1 20`—Command to enable FILS FD frame with period set to 20 TUs ■ `#iwpriv ath0 enable_fils 0 20`—Command to disable FILS FD frame To retrieve the configured period of FD frames, enter the `iwpriv athX g_enable_fils` command. FILS Discovery frame will be transmitted if following conditions are met. ■ Bursted beacon mode is enabled. ■ FILS Discovery interval must be greater than or equal to 20 TUs. ■ FILS Discovery interval must be less than Beacon interval. ■ Beacon interval must be divisible by FILS Discovery interval. ■ This command is recommended to be used when FILS is configured for the VAP.

3.80 Target recovery parameter

To avoid the internal RAM (IRAM) write mechanism to overwrite the firmware (FW) dump area, the FW recovery feature, after dumping the target RAM, waits for the user to continue the recovery process.

For more information on usage guidelines and the functionality regarding these parameters, see the *Early detection of bit corruption during firmware assert* section of the *WLAN AP driver operations* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:41 pan.yafeng@zte.com.cn

Parameter	Command	DA	OL	Description
set_fw_ recovery <value>	*iwpriv wifiX set_fw_ recovery <value>*	N	Y	Enable the feature to detect bit corruption. The <value> option can be one of the following values: ■ Value = 0, no FW recovery takes place. On target assert, after dumping data to RAM, the board is rebooted. ■ Value = 1, (default) FW recovery restores all VAPs after RAM dump. ■ Value = 2, FW recovery performs a RAM dump and waits for the user to trigger the complete recovery. If this 'wait for user' option is used, enter the following command for the recover process to continue: `ifconfig wifiX up`

3.81 Soft-blocking parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Soft-blocking for safe connections* section of the *Wi-Fi SON: Steering and APS* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

NOTE The soft-blocking feature adds and clears ACL entries. Therefore, this feature must not be used in conjunction with the Multi-AP SIG feature (also called as Easy Mesh) or with the Wi-Fi SON capability. If a combination of soft-blocking feature and Easy Mesh feature, or a combination of soft-blocking feature and Wi-Fi SON capability are configured, an unexpected behavior might occur because soft-blocking and Wi-Fi SON (or Easy Mesh) have independent logic of adding and clearing the ACL entries.

- Enter the wifitool athX softblocking <dstmac> <0/1> and wifitool athX softblocking_get <dstmac> commands to set/get the enable flag of the dstmac. Setting the flag resets the timer to the initial value.

Table 3-75 Soft-blocking parameters

Parameter	Command	Description
softblock_wait <0-100000> g_sftblk_wait	iwpriv athX softblock_ wait<0-100000> iwpriv athx g_sftblk_ wait	Set/get the parameter of the wait time in microsecond, default is 10000.
allow_time <0-100000> g_sftblk_allow	iwpriv athX softblock_ allow <0-100000> iwpriv athx g_sftblk_ allow	Set/get the parameter of the allow time in microsecond, default is 15000.
addmac_sec <mac>	iwpriv athX addmac_ sec <mac>	Adds specified MAC addresses to the secondary access control list (ACL).
maccmd_sec 2	iwpriv athX maccmd_ sec 2	Instructs how the ACL is used to limit access the AP.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-75 Soft-blocking parameters

Parameter	Command	Description
softblock_wait <0-100000> g_sftblk_wait	iwpriv athX softblock_wait <0-100000> iwpriv athx g_sftblk_wait	Set/get the parameter of the wait time in microsecond, default is 10000.
softblock_allow <0-100000> g_sftblk_allow	iwpriv athX softblock_allow <0-100000> and iwpriv athx g_sftblk_allow	Set/get the parameter of the allow time in microsecond, default is 15000.

3.82 Protected management frames (PMF) parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *802.11w protected management frames (PMF)* section of the *Power management techniques* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

NOTE The following information regarding the timeout parameter applies to QCA_Networking_2020.SPF.11.1 and QCA_Networking_2020.SPF.11.2 releases and later:

timeoutie is particularly useful when enterprise mode is used, where enterprise applications ask driver to set this IE in driver frames, and SA Query procedure is managed by enterprise applications OR enterprise authenticators. When hostapd is used as authenticator, the timeout and retry mechanisms shall be used by authenticator. This configuration is provided to hostapd through following UCI configuration. It is discouraged to use iwpriv in hostapd case. This need to be limited to validation of timeout IE validation and not the SA Query timeout and retries.

```
option ieee80211w_max_timeout 'value' (same as timeoutie value)
```

```
option ieee80211w_retry_timeout 'value'
```

Table 3-76 Protected management frames (PMF) parameters

Parameter	Command	Description
timeoutie “value” g_timeoutie	iwpriv athX timeoutie “value” iwpriv athX g_timeoutie	Enter the iwpriv athX timeoutie “value” command to specify the timeout interval after which hostapd starts the Security Association (SA) Query procedure upon receiving the association response event from the driver. To display the configured SA query timeout interval, enter the iwpriv athX g_timeoutie command.

pmf_assoc <1 | 0>
g_pmf_assoc
	iwpriv athX pmf_assoc <1 | 0>
	iwpriv athX g_pmf_assoc
		Enable (1) or disable (0) PMF using the timeout interval mechanism for Security Association (SA) query.
		To check whether PMF using SA query timeout interval is configured, enter the iwpriv athX g_pmf_assoc command. Alternatively, enter the following UCI commands to enable this feature:
		`uci set wireless.@wifi-iface[X].pmf_assoc=1`
		`uci commit wireless`

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

3.83 Tx chain-mask parameter

For enabling power consumption in the most optimal manner, a capability to apply Tx chain-masks without STAs being disconnected/reconnected, when required, is introduced. This feature enables the user to configure the Tx chain-masks on the fly in the AP board without disconnecting the associated clients. Rx chain mask must remain intact. Because dynamically changing Tx chain-masks affects the MU functionality, it is necessary to disable the MU feature when this Tx-chain masks feature is enabled and also the key performance indicator (KPI) for the throughput cannot be guaranteed. This application of Tx chain-mask functionality is not supported on 11ax chipsets.

For more information on usage guidelines and the functionality regarding these parameters, see the *Modify TX chain-mask without reconnecting clients* section of the *Power management techniques* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-77 Tx chain-mask parameter

Parameter	Command	Description
txchainsoft 0xN get_txchainsoft	iwpriv wifiX txchainsoft 0xN iwpriv wifiX get_txchainsoft	Modify the Tx chain masks dynamically to enable optimal power consumption. Run ping or IxChariot traffic in single user (SU) or multiple user (MU) mode. The chain mask can be changed ensuring that ping is not stopped, although ping traffic might be impacted slightly and clients continue to remain connected. Change the Tx chain mask in ascending order and then in descending order (or vice versa). Also, a user can change the chain mask randomly to assure its stability. These changes can be performed in both SU mode and MU mode. Increasing or decreasing the Tx chain appropriately impacts the NSS count listed under Firmware stats 6 section of the txrx_fw_stats command. The get_txchainsoft parameter is used to display whether the Tx chainmask capability for optimal power consumption is enabled or not.

3.84 ICM parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Intelligent Channel Manager* section of the *Advanced channel selection capabilities* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-78 ICM parameters

Parameter	Format	DA	OL	Description	
set_icm_active	**cfg80211tool wifiX set_icm_active <value>**	Y	Y	Set whether ICM is active. Note: This setting is not available with iwpriv command. This command line parameter is only for specific debug purposes and not for regular operation, since this state is supposed to be set by the ICM application. Using this command line parameter during regular operation might result in unpredictable results depending on the circumstances.	
				0	ICM is not active
				1	ICM is active
get_icm_active	**cfg80211tool wifiX get_icm_active**	Y	Y	Get whether ICM is active. Note: This setting is not available with iwpriv command.	

3.85 TR-181 data elements for carrier networks

The Broadband Forum defines the TR-181 standard, which represents the device data model for TR-069, for carrier and service provider networks. The device:2 data model applies to all types of TR-069-enabled devices, including end devices, residential gateways, and other network infrastructure devices. It represents a next generation evolution that supersedes both Device:1 and InternetGatewayDevice:1.

For details regarding the supported TR-181 data elements for WLAN access points, and the corresponding in-put/output controls (IOCTLs) for configuring these elements, refer to the *TR-181 Data Elements for WLAN Access Points Reference Guide* (80-YA955-1).

3.86 Wi-Fi medium utilization statistics parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Crypto service module and data structures* and *Retrieve Wi-Fi medium utilization statistics* sections in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-79 Wi-Fi medium utilization statistics parameters

Parameter	Command	Description
g_ch_util_ap_rx	iwpriv wifiX g_ch_ util_ap_rx	Get AP Rx channel utilization percentage.
g_ch_util_free	iwpriv wifiX g_ch_ util_free	Get free medium available percentage.
g_ch_util_ap_tx	iwpriv wifiX g_ch_ util_ap_tx	Get AP Tx channel utilization percentage.
g_ch_util_obss	iwpriv wifiX g_ch_ util_obss	Get OBSS channel utilization.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

3.87 Parameter APIs for carrier networks

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Support for reporting associated and steering stations for carrier networks* section of the *Wi-Fi SON: Additional features* chapter in the Wireless LAN Access Point (Driver Version 11.0) Programming Guide (80-YA728-6).

Starting with QCA_Networking_2016.SPF.6.0, support is implemented for generating reports of associated and non-associated stations, and steered and non-steered stations, which are beneficial for collecting statistics and computing usage of subscriber services to aid troubleshooting and diagnosis. Users can enable or disable the collection of parameter APIs, which refer to the associated, non-associated, steered, and non-steered stations using the iwpriv wifiX set_papi <0 | 1> command for the WLAN driver to transmit or not transmit the report to user space.

Associated station report and non-associated station report are sent directly from driver to user space application. Steering events are sent from LBD to other user space application using the SON extension library.

By default, the driver will not send the report to user space, and the feature has to be enabled in driver to send the reports. The following iwpriv commands are available to enable/disable the feature.

iwpriv wifiX set_papi 0/1 (0 - disable, 1- enable)

iwpriv wifiX get_papi (get the current configuration)

Extended BSS Load support and information parameters

Support is introduced to enable or disable extended BSS load information element (IE) in beacon using iwpriv, by adding new ioctl IEEE80211_PARAM_XBSS_LOAD. This support

FW provides secondary 20, 40, and 80 utilization as part of WMI_INTERFERENCE_EVENT_ ID, which are saved in _wmi_host_dcs_im_tgt_stats structure and further saved to the ic structure.

New fields added in ic structure are as follows:

```
u_int16_t                    ic_ss_uu;
u_int8_t                     ic_sec_20u;
u_int8_t                     ic_sec_40u;
u_int8_t                     ic_sec_80u;
```

- When extended BSS load IE is enabled, ieee80211_ie_ext_bssload is filled with values in ic structure and added to beacons.
- When extended BSS load IE is enabled, values in beacon IE can be fetched using iwpriv commands as follows:

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-80 Parameter APIs for carrier networks

Parameter	Command	Description
set_papi 0/1 (	iwpriv wifiX set_papi 0/1 (	Enables (1) or disables (0) the parameter API functionality for carrier networks. By default, the driver will not send the report of associated stations, non-associated stations, steered stations, and non-steered stations to user space, and the feature has to be enabled in driver to send the reports. Associated station report and non-associated station report are sent directly from driver to user space application. Steering events are sent from LBD to other user space application using the SON extension library.
get_papi	iwpriv wifiX get_papi	Retrieve the current configuration of parameter API feature.
xbssload 0/1 get_xbssload	**set** : iwpriv athX xbssload 0/1 **get** : iwpriv athX get_ xbssload	XBSSLoadMUStaCount in the Reporting Extended BSS Load Element Enhanced iv structure to hold mu capable sta count on every join request if vht capabilities have IEEE80211_VHTCAP_MU_BFORMEE and IEEE80211_VHTCAP_MU_BFORMER bits and decremented on leave event.
xbssload 0/1 get_xbssload	**set** : iwpriv athX xbssload 0/1 **get** : iwpriv athX get_ xbssload	The following values in beacon IE can be fetched using iwpriv commands: ■ XBSSLoadSSUnderUtil in the Reporting Extended BSS Load Element ■ XBSSLoad20SecUtil in the Reporting Extended BSS Load Element ■ XBSSLoad40SecUtil in the Reporting Extended BSS Load Element ■ XBSSLoad80SecUtil in the Reporting Extended BSS Load Element Calculated with values received as part of WMI_DCS_INTERFERENCE_ EVENT_ID

3.88 CBS for VoW parameters

To provide IPTV services in real-time using Wi-Fi connections, the quality must be guaranteed from interference. The AP must be able to scan its full band periodically, and change the channel without any deterioration of service quality. Starting with QCA_Networking_2017.SPF.6.0 release, support is implemented for continuous background scan (CBS) to resolve the issues of legacy scan for video over wireless (VoW) scanning and switching channels without impacting services.

For more information on usage guidelines and the functionality regarding these parameters, see the *Support for CBS for VoW switching and scanning channels* section of the *Video and multimedia features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 3-81 CBS for VoW parameters

Parameter	Command	Description
cbs_resttime <ms>	iwpriv wifiX cbs_ resttime <ms>	Command to modify scan rest interval (msec). The default value is 500 ms. If a user attempts to change rest time while CBS is running, the scan is canceled, rest time is changed and the scan is restarted.
cbs_dwellrest <ms>	iwpriv wifiX cbs_ dwellrest <ms>	Command to modify dwell rest time (msec), that is, the time between scans on same channel with different scan offset. The default value is 100 ms. Dwell rest times can be only integral multiples of 100 ms (beacon interval). If a non-integral multiple of beacon interval is passed it is adjusted to next multiple of the beacon interval. The default value is set to be 100 ms. Also, 100 ms is the least value we can set. If a user tries to change dwell rest time, while CBS is running, the scan is canceled, dwell rest is changed and then the scan is restarted.
cbs_waittime <ms>	iwpriv wifiX cbs_ waittime <ms>	Command to modify wait time (msec), that is, the time to wait after all channels are scanned and before starting a new scan. Set this to 0 to stop after one complete scan. The default value is 1000 ms. Wait time can only be integral multiples of 100 ms (beacon interval); for example, iwpriv or cfg80211tool wifi0 cbs_waittime 200/300/400 If given number is not an integral multiple of the beacon interval, it is adjusted to next multiple of the beacon interval. If configured to 0, the scan is done on all channels in the channel list once and stopped. If a user attempts to change wait time while CBS is running, the scan is canceled, the wait time is changed and the scan is restarted.
cbs_bkscanen 1/0	iwpriv wifiX cbs_ bkscanen 1/0	Command to enable/disable CBS background scan.
g_cbs_ bkscanen	iwpriv wifi0 g_cbs_ bkscanen	Command to retrieve the configured CBS background scan.
cbs_dwellspt <value>	iwpriv wifiX cbs_ dwellspt <value>	Command to configure the dwell split time. This is command can be used to test different scan offset passed to FW. The value can be 25, 50, 75. The default value is 50 ms. Returns an error if we set to any other value. If a user attempts to change dwell split time while CBS is running, the scan is canceled, split time is changed and the scan is restarted.
setchanlist ch1 ch2 .....n	wifitool athX setchanlist ch1 ch2 .....n	Command to set user desired channel list to scan.
g_cbs_resttime	iwpriv wifi0 g_cbs_ resttime	Command to retrieve the configured scan rest interval.
g_cbs_dwellrest	iwpriv wifi0 g_cbs_ dwellrest	Command to retrieve the configured dwell rest time.
g_cbs_dwellspt	iwpriv wifi0 g_cbs_ dwellspt	Command to retrieve the configured dwell split time.
g_cbs_waittime	iwpriv wifi0 g_cbs_ waittime	Command to retrieve the configured scan wait time.

3.89 TDMA parameters

Time division multiple access (TDMA) is a channel-access methodology for point-to-multipoint within intra-backhaul network. TDMA deployment is controlled and managed by OEMs as a proprietary ecosystem. No interoperability is necessary with other Wi-Fi-based TDMA devices.

TDMA is beneficial in reducing interferences between multiple APs operating within the intra-backhaul network. STAs connected to AP start data-transfer only after the reception of trigger frames from an AP. The AP coordinates and sends trigger frames to connected STAs in a round-robin manner.

This feature enables the user to configure TDMA support in the AP/STA at runtime. After TDMA is configured, AP sends trigger frames to STAs to indicate STA to start transmission. STA sends a burst of traffic and sends end-trigger to AP.

AP coordinates best effort (BE) traffic of connected clients by sending trigger frames to each of the clients in round-robin scenario. STA starts transmitting only after receiving the trigger. Therefore, collisions are avoided in the medium because only one STA can start transmission at a point in time.

The design involves triggers being sent from both AP and STA in the following two ways:

- Trigger sent from AP to STA is BAR frame.
- Trigger sent from STA to AP is PSPOLL frame.

For more information on usage guidelines and the functionality regarding these parameters, see the *TDMA method* section of the *WLAN AP modes* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-82 TDMA parameter

Parameter	Command	Description
qboost_enable <1| 0>	iwpriv wifiX qboost_enable <1| 0>	Issue the iwpriv wifiX qboost_enable 1 command on all the STA and AP boards to enable TDMA functionality. Issue the iwpriv wifiX qboost_enable 0 command on all the STA and AP boards to disable TDMA functionality.

3.90 Strict passive scan parameters

With the hidden SSID support in strict passive scan, an enhancement was required to resolve scenarios in which a repeater AP failed to initiate active scanning with root AP's hidden SSID in DFS channel. In addition, support is introduced for hidden SSID in strict passive scan in QWRAP/ExtAP STA-VAP mode.

In general, APs broadcast beacons periodically, and in the beacon, SSID string is present. However, for security purposes, sometimes APs do not send the SSID information in the beacon. It is known as a hidden SSID. For a client to be able to connect to a hidden SSID AP, the client needs to perform an active scan (sending probe in the channel where the AP is present). However, there

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

are channels which are passive channels (where one can send a probe request only after seeing a beacon in that channel). Therefore, even if an AP is in a passive channel, connection to it is possible.

Certain users require strict passive scan, where the rule is not to send any probe requests in that channel at all even after seeing beacon. This strict passive scan is used, perhaps, to reduce power consumption and/or to reduce the traffic in the BSS or some other regulatory restrictions. Now, when strict passive scan is used (no probe requests at all), connection to a hidden SSID AP is not possible using the traditional method of candidate AP selection. The traditional method of selection collects beacons or probe responses and add them in a list and then select the AP with the appropriate SSID.

However, in case of strict passive scan, although the beacon from the AP is present, the SSID is not available; therefore, it is not possible to choose the candidate AP to which the station must connect. The station is unable to connect to hidden-SSID AP with strict passive scan.

To overcome the hidden SSID issue with strict passive scan, an enhancement is implemented. Because the beacon information from the hidden SSID AP is available, all the information about the AP is available, except the SSID. Also, it is possible that there are more than one hidden SSID APs. A list of all the beacons/probe-response from various APs are available and the AP to which the connection must be made is not clear at this stage.

This problem can be resolved by adding one extra piece of information in the client software. The extra piece of information is the BSSID. Therefore, instead of setting SSID only, a combination of SSID and BSSID must be set.

For more information on usage guidelines and the functionality regarding these parameters, see the *Hidden SSID support in strict passive scan* section of the *Beacons and frames transmission* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-83 Strict passive scan parameter

Parameter	Command	Description
conf_bssid 'bssid of the hidden AP' get_conf_bssid	cfg80211tool /iwpriv athx conf_bssid 'bssid of the hidden AP' cfg80211tool athx get_conf_bssid	Set/retrieve the BSSID of hidden SSID AP for which a client or STA needs to connect. Example: `cfg80211tool/iwpriv athx conf_bssid 8C:FD:F0:00:D7:E3` Note: To test this behavior, strict passive scan also must be enabled.
pas_scanen 1/0 g_pas_scanen	iwpriv /cfg80211 wifix pas_scanen 1/0 iwpriv /cfg80211 wifix g_pas_scanen	Enable strict passive scan. Use the get command to display whether strict passive scan is enabled or not.

3.91 Periodic channel frequency response dump capture parameters

iwpriv and wifitool utilities are used to configure the capability to generate the channel frequency response (CFR) dump. To generate periodic CFR capture, use the iwpriv command. To generate per-peer CFR capture, use the wifitool command. All user input is sent through iwpriv/wifitool commands from user space to host. The start and stop operation of periodic CFR capture is issued per PHY. All other user input is entered per station. User space populates all the per-peer inputs in driver entry, whenever it is requesting a CFR dump. Otherwise, the host ignores the request.

For more information on usage guidelines and the functionality regarding these parameters, see the *Generate channel frequency response dump for IPQ4019 platforms and Generate channel frequency response dump for IPQ807xA platforms* sections of the *Core channel selection capabilities* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 3-84 Periodic CFR dump generation parameter

Parameter	Command	Description
cfr_timer <1/0>	iwpriv wifix cfr_timer <1/0>	Start or stop the generation of periodic CFR dump. Periodic CFR capture enable = 1, Periodic CFR capture disable = 0 At any point of time, either one-shot (per-peer) capture or periodic capture is taken. Both per-peer capture and periodic capture cannot be enabled simultaneously. An error is returned that ioctl failed if per-peer capture is requested when periodic CFR capture is enabled.
get_cfr_timer	iwpriv wifix get_cfr_timer	Display whether the capture of periodic CFR dump is enabled or disabled.

3.92 Changes to cfg80211tool commands for 6 GHz support

The behavior is expected to be same in 6 GHz as the behavior for 2.4 GHz and 5 GHz bands. An optional new argument at end is added for chan band. If specified, channel no is considered from specified band else its assumed that command is applied for 2G/5G band. Here, <chan band> refers to 2 GHz, 5 GHz, or 6 GHz, and <chan no> refers to the channel number. Band value 1 for 2 GHz, 2 for 5 GHz and 3 for 6 GHz should be used. If band field is not specified, it is assumed that "channel" is from 2 GHz or 5 GHz band. The <chan band> argument is necessary for 6 GHz, while it is optional for 2 and 5 GHz.

- cfg80211tool ath0 channel <chan no> [<chan band>]
- cfg80211tool ath0 doth_chanswitch <chan no> <tbtt> [<chan band>]
- cfg80211tool ath0 doth_ch_chwidth <chan no> <tbtt> <new chwidth> [<chan band>]
- cfg80211tool ath0 cfreq2 <secondary 80 MHz centre channel index> [<chan band>]

NOTE: Currently, issuing this command on a VAP which is on a 5 GHz current channel with a 6 GHz channel and band identifier as input arguments will bring up the VAP in the 6 GHz channel in co-hosted mode considering the VAPs are being hosted on wide-band

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

radio. This usage of the command is invalid and not recommended as co-hosted mode is not allowed as per 6 GHz standard.

- Starting with QCA_Networking_2021.SPF.11.4 release, the cfg80211tool wifiX setRegdomain command is enhanced to set 6 GHz super domain ID. It will be backward compatible. The regulatory structure, which contains country and regulatory domain, is expanded to include 6 GHz superdomain. Before this enhancement, only the first 16 bits of the 32bit regdomain code sent to the target were used. To accommodate the 6G superdomain in the existing 32-bit regdomain code, use the first 16 bits for the existing 2GHz-5GHz regdomain pair, and the next 16 bits for the 6 GHz super domain ID.

  The 6G super domain ID is represented by the upper 16 bits of the regulatory domain hex code in the "cfg80211tool wifiX setRegdomain" command.

  Syntax: cfg80211tool wifiX setRegdomain <reg domain hex code (First 16 bits represent 6G superdomain code and the next 16 bits represent the 2G_5G regdomain pair code)>

  Example: The super domain code should be 65536 (0x00010016) for US Country super-domain {6g_FCC1_CLIENT_LPI_REGULAR, 6g_FCC1_AP_LPI, 6g_FCC1_CLIENT_SP} is 0x1 and FCC8_FCCA is 0x16.

  Therefore, the command should be executed as: cfg80211tool wifiX setRegdomain 0x00010016

  It is possible to execute cfg80211tool wifiX setRegdomain 0x16 to set only 2.4 GHz and 5 GHz paircode.

3.92.1 Guidelines for 6 GHz out-of-band functionality

- In response to ANQP request with Neighbor report query, AP includes information of all co-located APs across all bands supported. This is because request cannot query for specific 6 GHz AP info
- Advertisement of ANQP support in beacon to be configurable and used for WFA test case only through sigma-dut/manual selection
- 20 TU Probe Response will be a full broadcast probe response (All IE included)
- All tests to be carried out in MBSSID mode enabled in 6 GHz
- TBTT offset in RNR to have accurate value only for same beacon interval case
- Few beacons in lower band may not contain RNR of co-located 6 GHz AP upon coming up
- OCT and related Out of band tunneling requirements not supported.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Command	Arguments	Description
cfg80211tool wifiX oob_enable *arg*	{1/0}	Enable/disable OOB feature for lower band APs to advertise collocated 6 GHz APs. Starting with QCA_Networking_ 2020.SPF.11.3 release, it is recommended to use the cfg80211tool wifiX 6ghz_rnr_collocated <enable/disable/driver mode> <frame_ type> command instead of the oob_ enable command, the usage details of which are described in the *11.0 WLAN driver parameters* section in this chapter.
cfg80211tool athX bcast_prbrsp_en *arg1 arg2*	arg1-{1/0} arg2 - {interval}	Enable/disable 20TU probe response with user defined period. Enabling 20TU PrbRsp disables FILS
cfg80211tool athX enable_fils *arg1 arg2*	arg1-{1/0} arg2 - {interval}	Switch back to FILS from 20TU probe response mode

3.92.2 CLI enhancements for in-band and OOB features

Command	Argument	Functionality
cfg80211tool athX bcast_ prbrsp_en <0 | 1> <interval>	bcast_prbrsp_en <0 | 1> <interval>	Enable 20 TU Probe Response and configure interval
cfg80211tool athX bcast_ prb_rsp_intvl <value>	bcast_prb_rsp_intvl <value>	Configure bcast Probe Response timer
cfg80211tool athX fils_en <0 | 1> <interval>	fils_en <0 | 1> <interval>	Enable FILS frame
cfg80211tool wifiX oob_ enable <value>	oob_enable <value>	Enable OOB advertisement Starting with QCA_Networking_ 2020.SPF.11.3 release, it is recommended to use the cfg80211tool wifiX 6ghz_rnr_ collocated <enable/disable/driver mode> <frame_type> command instead of the oob_enable command, the usage details of which are described in the *11.0 WLAN driver parameters* section in this chapter.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

3.93 Extending wifistats command for displaying control path statistics

Starting with QCA_Networking_2020.SPF.11.3 release, this feature is implemented on IPQ807x/IPQ817x/IPQ807xA, IPQ50xx, IPQ60xx, and QCN90xx/QCN60xx attached to IPQ807xA, IPQ60xx platforms. Starting with the IPQ5018.ILQ.11.4 release, this feature is extended to QCN61xx chipsets attached to IPQ50xx platforms.

For analyzing problems and troubleshooting them, wifistats as (wifistats wifiX <stats_id>) is used. These wifistats use host target transport (HTT) as an interface to send statistics request and get response between host and target. HTT interface is expected to be used for data path purposes. The infrastructure that uses wireless messaging interface (WMI) interface is leveraged to also print control path statistics. The corresponding wifistats command is of the following syntax:

```
wifistats wifiX <stats_id> <optional_arg>  --wmi
```

where --wmi is used to indicate host that, command is expected to send over WMI.

The primary objective of this feature is to bring up wifistats infra for control path statistics, which will be applicable on all IPQ807x/IPQ807xA/IPQ817x, IPQ50xx and QCN61xx + IPQ50xx, IPQ60xx, and QCN90xx/QCN60xx platforms.

The WMI interface will be used to send control path statistics request and event, and corresponding wifistats command is as follows:

wifistats wifiX <stats_id> <optional_arg> --wmi

- stats_id = statistics id defined in wmi_ctrl_path_stats_id; /*1 - pdev tx stats*/
- optional_arg = optional arguments are dependent on statistics type, it can be vdev_id or mac address
  - --vdev_id=y / --mac=aa:bb:cc:dd:ee:ff
  - --action = defined in wmi_ctrl_path_stats_action. /*1 - get stats*/ /* 2- Reset stats*/ If action field is not passed, by default it will be considered as get statistics.
- --wmi is used to indicate host that, command is expected to send over wmi.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

4 wifitool utility commands

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

4.1 802.11k

For more information on usage guidelines and the functionality regarding these parameters, see the *Voice Enterprise* section in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6). Also, see the *Radio resource management (802.11k) parameters* section in the i*wpriv and cfg80211tool parameters* chapter.

Table 4-1 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: Channel number for which stats are to be determined ■ reg_class: Regulatory 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: Regulatory class of operating channel. ■ destmac: MAC address of associated station. ■ channel: Channel number on which we want station to calculate channel load.
sendstastats	wifitool *interface_name sendstastats dst_mac duration gid*	■ dst mac: MAC address of associated client ■ duration: Interval for which we want to take this statistics. Value is in ms. ■ gid: group id, this value is taken from 802.11k specification.
sendnhist	wifitool *interface_name sendnhist dstmac duration regclass channel*	■ dst mac: MAC address of associated client. ■ duration: Interval for which we want to take this statistics. Value is in msec. ■ regclass: Regulatory class of operating channel. ■ channel: Channel number on which station will calculate channel load will be calculated.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

4.2 Channel loading

For more information on usage guidelines and the functionality regarding these parameters, see the *UMAC auto channel selection (scanning)* section of the *Advanced channel selection capabilities* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 4-2 Wifitool channel loading parameters

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

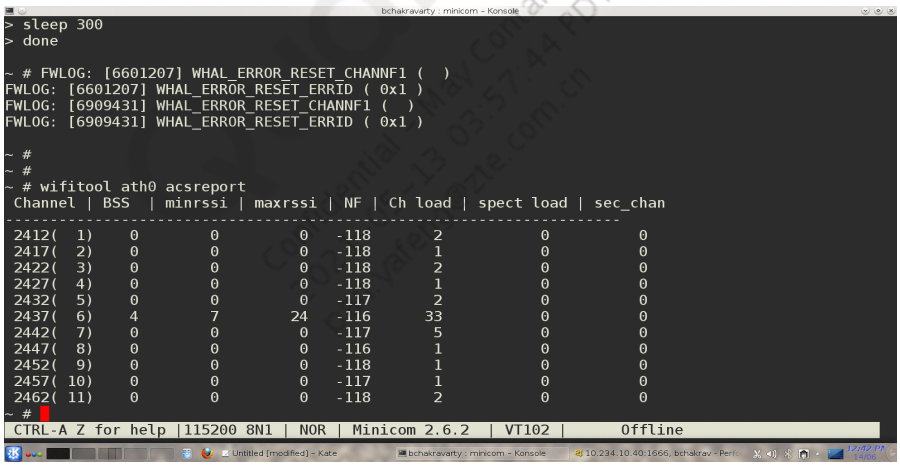


Figure 4-1 Channel loading

```
root@OpenWrt:/# wifitool ath2 acsreport


Legend: SC: Secondary Channel, WR: Weather Radar,DFS: DFS Channel,
        HN: High Noise, RSSI: Low RSSI,CL: High Channel Load
        RP: Regulatory Power,N2G: Not selected 2G, P80X: Primary 80X80
        NS80X: Only for primary 80X80, NP80X: Only for Secondary 80X80

 The number of channels scanned for acs report is:  12

 Channel | BSS  | minrssi | maxrssi | NF | Ch load | spect load | sec_chan | Ranking
| Unused
-------------------------------------------------------------------------------------
------------
```

```
 5580(116)    0        0        0   -254       0           0          0          7
()
 5600(120)    0        0        0   -254       0           0          0          8
()
 5620(124)    0        0        0   -254       0           0          0          9
()
 5640(128)    0        0        0   -254       0           0          0         10
()
 5660(132)    0        0        0   -254       0           0          0         11
()
 5680(136)    0        0        0   -254       0           0          0         12
()
 5700(140)    0        0        0   -254       0           0          0         13
()
 5745(149)    0        0        0   -254       0           0          0         15
()
 5765(153)    0        0        0   -254       0           0          0          0
(SC )
 5785(157)    0        0        0   -254       0           0          0         14
()
 5805(161)    0        0        0   -254       0           0          0          1
()
 5825(165)    0        0        0   -254       0           0          0          2
()
ACS_FAILED_NBSS: Current channel is selected based on the min bss count
wifitool: acs->nchans: 12

Chan|chpool|noisefloor|radarNF|maxrssi|!80211NF|util|utiltotal|busy|busytx|busyrx|
busyself|busyext
-----------------------------------------------------------------------------------
-------------------
  116       1    -254       0       0       0       0       0       0        0
0        0        0
  120       1    -254       0       0       0       0       0       0        0
0        0        0
  124       1    -254       0       0       0       0       0       0        0
0        0        0
  128       1    -254       0       0       0       0       0       0        0
0        0        0
  132       1    -254       0       0       0       0       0       0        0
0        0        0
  136       1    -254       0       0       0       0       0       0        0
0        0        0
  140       1    -254       0       0       0       0       0       0        0
0        0        0
  149       1    -254       0       0       0       0       0       0        0
0        0        0
  153       1    -254       0       0       0       0       0       0        0
0        0        0
  157       1    -254       0       0       0       0       0       0        0
0        0        0
  161       1    -254       0       0       0       0       0       0        0
0        0        0
  165       1    -254       0       0       0       0       0       0        0
0        0        0
-----------------------------------------------------------------------
```

Qualcomm Confidential - May Contain Trade Secrets 2021-09-13 23:57:44 PDT pan.yafeng@zte.com.cn

```
Additonal Channel Stat
----------------------------------------------------------------------
 Index | Channel | NBSS | SSID     | BSSID       | rssi | Phymode
----------------------------------------------------------------------
 1              149     3      sit-tst-bbss      8c:fd:f0:0f:90:72          36
19
 2              149     3      MapOTA-tst-bbss      8c:fd:f0:0f:90:78          9
19
 3              149     3      MapOTA-tst-bbss      8c:fd:f0:0f:90:1a          6
19
```

4.3 Block channel list

For more information on usage guidelines and the functionality regarding these parameters, see the *Channel blocking* section of the *Core channel selection capabilities* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

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

Table 4-3 Block channel list

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

Max value for N is 255.

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

“Following channels are blocked from ACS”

[channel 1] [channel2]…….[channel]

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

e.g

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

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

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

Will block only channel 4, 5, and 6

4.4 FIPS validation

For more information on usage guidelines and the functionality regarding these parameters, see the *QCA99xx and IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx/QCN90xx or QCN602x/IPQ807x/IPQ807xA/IPQ817x FIPS-compliant encryption function* section of the *WLAN AP driver operations* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

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

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

The following is an example text file:

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

Run the following command

```
#wifitool ath0 fips input_file
```

Wifitool implements sample test functionality using the interfaces mentioned in this document. It accepts FIPS test data in a text file. Each line in this file is a test vector for the FIPS command. The format of the file is specified as shown in this section; WifiTool expects the text file to be in UNIX file format; the tool will not process any other file formats.

```
{CommandID}{space}{Mode}{space}{KeyLength}{space}{InputDataLength}{Key}{
InputData}{ExpectedOutput}{space}{IV}{newline}
```

Where:

CommandID	0	Encrypt
	1	Decrypt (not supported)
Mode	0	AES_CTR
	1	AES_MIC (not supported)
KeyLength	(16|32)	
DataLength	16..1488 bytes of data	
Key	Actual AES Key of *KeyLength*	
InputData	Data of length *DataLength*	
ExpectedOutput	Encrypted data of *DataLength*	
IV	Initialization counter (not used)	

The access point must be properly configured and up and running to use this tool. As FIPS is a test mode, it is expected that the test is conducted when there are no stations associated. Stations may be disconnected at the time of a FIPS test.

Use this command to run the FIPS test on the access point. A driver load and reload is suggested after a test.

```
$ wifitool ath0 fips input_file
```

If the test passes, this command prints that the test is successful. The test file can contain multiple lines of data, provided that each of these lines is one test data set. When provided with multiple lines, the **wifitool/fips** command executes all test data present in the file and wifitool conducts sanity and boundary checks on the data before continuing with the data.

Sample test data in a file is provided below:

```
0 1 16 16 2b7e151628aed2a6abf7158809cf4f3c
6bc1bee22e409f96e93d7e117393172a 3ad77bb40d7a3660a89ecaf32466ef97
f0f1f2f3f4f5f6f7f8f9fafbfcfdfeff
```

4.5 Chainmask per client

For more information on usage guidelines and the functionality regarding these parameters, see the *Dynamic Transmit Chainmask Selection (DTCS)* section of the *Beacons and frames transmission* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

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

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

Run the following command

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

4.6 Set antenna switch

For more information on usage guidelines and the functionality regarding these parameters, see the *Advanced Enterprise* section of the *Power management techniques* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

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

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

Run the following command

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

Some Qualcomm Technologies chipsets have an RF switch that provides the option to dynamically select the antenna for the radio. For example, the external antenna 5 GHz radio can either use uniband 5 GHz antenna or share the same dual-band antenna with 2.4 GHz radio. At load time, default SWCOM values will be set based on EEPROM values. During runtime, use the wifitool athX set_antenna_switch command to adjust the RF switch state and select antenna combinations is provided. The related WMI command is WMI_PDEV_SET_ANTENNA_SWITCH_TABLE_CMDID.

4.7 Set user control table

For more information on usage guidelines and the functionality regarding these parameters, see the *Advanced Enterprise* section of the *Power management techniques* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

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

User-defined control (CTL) tables are usually hardcoded in EEPROM for each device. And currently, only FCC is supported, Japan and ETSI in QCA CTL table structure. Advance Enterprise has more combinations, which makes it hard to store everything in EEPROM and confront with Qualcomm Technologies CTL format. Instead, the CTL API which accepts the QCA compatible CTL format from customers is provided. All the conversions of CTL format and domain mapping are supposed to be done by customer itself.

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

Run the following command

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

4.8 Block acknowledge

For more information on usage guidelines and the functionality regarding these parameters, see the *802.11n features* section in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Qualcomm Confidential May Contain Trade Secrets 2021-05-13 05:37:44 PDT pan.yafeng@zte.com.cn

Table 4-4 Block acknowledge

Parameter	Format	Description
refusealladdbas X	wifitool athN refusealladdbas X	Enables/disables refusal of all incoming ADD BA Req. where X is 0 - disable, 1 - enable This wifitool command must be issued only after setting the BA module to manual mode (iwpriv athN setaddbaoper X command).

sendaddba
senddelba
	wifitool ath*N* sendaddba AID *AC BufSize*
	wifitool ath*N* senddelba AID AC initiator reason
		Send ADD BA Req to a particular associated peer (chosen by AID), on a particular transaction identifier (chosen by TID) with a specified buffer size (chosen by BufSize).
		where:
		■ AID - Association ID
		■ TID - Transaction ID
		■ BufSize - Maximum number of subframes sent in an aggregation
		Send DEL BA Req to a particular associated peer (chosen by AID), on a particular transaction identifier (chosen by TID) initiated by initiator with reasoncode reason.
		where,
		■ AID - Association ID
		■ TID - Transaction ID
		■ initiator - 1 (AP), 0 (STA)
		■ reason - 16-bit code indicating the reason for deleting BA session.
		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 16-bit code indicates the reason the link shut down. No corresponding get parameters or default values.
		`#wifitool ath0 sendaddba 1 0 32`
		`#wifitool ath0 senddelba 1 0 1 36`
		These wifitool commands must be issued only after setting the BA module to manual mode (iwpriv athN setaddbaoper X command).

setaddbaresp	wifitool ath*N* *set*addbaresp AID *AC status*	Sends an addba response frame on the indicated AID and AC with a particular status code status. 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. `#wifitool ath0 setaddbaresp 1 0 25` This wifitool command must be issued only after setting the BA module to manual mode (iwpriv athN setaddbaoper X command).

Table 4-4 Block acknowledge

Parameter	Format	Description	

getaddbastats	wifitool ath*N* getaddbastats AID *TID status*	Gets the ADDBA (Add Block Acknowledgment) status for AID (Association Identifier) and TID (Traffic Identifier).
		`#wifitool ath0 getaddbastats aid tid status`
		`#wifitool ath0 ath0 getaddbastats: 1 0 25`	

		aid	AID number of STA
		tid	TID number between 0-15

4.9 TR-69 parameters

Table 4-5 describes the list of 'wifitool' commands.

Table 4-5 wifitool commands

Parameter	Format	DA	OL	Description
tr069_get_ success_retrans	wifitool athN tr069_get_ success_retrans	Y	Y	Command to get successful retransmission count from the VAP Usage: `#wifitool ath0 tr069_get_success_retrans`
tr069_get_fail_ retrans	wifitool athN tr069_get_fail_ retrans	Y	Y	Command to get failed retransmission count from the VAP Usage: `#wifitool ath0 tr069_get_fail_retrans`
tr069_get_ success_mul_ retrans	wifitool athN tr069_get_ success_mul_ retrans	Y	Y	Command to get successful retries after multiple attempts from the VAP Usage: `#wifitool ath0 tr069_get_success_mul_retrans`
tr069_get_ack_ failures	wifitool athN tr069_get_ack_ failures	Y	Y	Command to get ACK failures count from the VAP Usage: `#wifitool ath0 tr069_get_ack_failures`
tr069_get_ retrans	wifitool athN tr069_get_ retrans	Y	Y	Command to get total re-transmission count from the VAP Usage: `#wifitool ath0 tr069_get_retrans`
tr069_get_ aggr_pkts	wifitool athN tr069_get_aggr_ pkts	Y	Y	Command to get aggregate packets from the VAP. Usage: `#wifitool ath0 tr069_get_aggr_pkts`
tr069_get_sta_ bytes_sent	wifitool athN tr069_get_sta_ bytes_sent <STA MAC>	Y	Y	Command to get bytes sent to the specific station from the VAP. Usage: `#wifitool ath0 tr069_get_sta_bytes_sent <STA MAC>`
tr069_get_sta_ bytes_rcvd	wifitool athN tr069_get_sta_ bytes_rcvd <STA MAC>	Y	Y	Command to get bytes received from the specific station in the VAP. Usage: `wifitool ath0 tr069_get_sta_bytes_rcvd <STA MAC>`

4.10 Preserve regulatory settings after HLOS replacement

For more information on usage guidelines and the functionality regarding these parameters, see the *Preserve regulatory settings after HLOS replacement* section of the *Regulatory compliance of WLAN APs* in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

The functionality to support regulatory offload to firmware enables securing the FCC/US rules for the device operates with these rules. This enables OEMs to allow HLOS replacement to support after-market usage and upgrade or replacement of the router software. The following are the salient benefits of this feature:

- Prevent operation on channels 12-14 in 2.4GHz and any invalid channels on 5GHz.
- Prevent power levels greater than maximum allowed by FCC.
- Allow replacement of the HLOS with preservation of the regulatory parameters.
- Check flags such as DFS, quarter rate, and half rate for proper usage.

This feature helps in maintaining regulatory information for US country and relevant FCC domains. With the regulatory data stored in Firmware, this functionality prevents misuse of power, prevents antenna gain or invalid operation on DFS channels, and disallows unsupported frequencies.

Support is provided only for US country code rules to be stored by Firmware due to memory limitations.

When any problem is observed with the feature, enter the following command to check the current regulatory that is used:

```
wifitool athX beeliner_fw_test 132 1
```

4.11 MBO parameters

Multiband Operation (MBO) facilitates efficient use of multiple frequency bands and channels that are available to access points (APs), and the wireless devices that associate with them (stations). MBO offers solutions for band steering, load balancing, and other related operational procedures. The features defined for MBO are band-independent and apply across all available unlicensed bands where Wi-Fi systems operate. With exchange of information between APs and STAs, intelligent decisions are made to collectively drive the network towards more efficient use of the available spectrum.

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Wi-Fi Alliance MBO functions* section of the *Beacons and frames transmission* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Qualcomm
Confidential – May Contain Trade Secrets
2021-05-13 03:57:44 PDT
pan.yafeng@zte.com.cn

Table 4-6 wifitool MBO parameters

Parameter	Command	Description
ath0 sendbstmreq <associated_sta_mac> <candidate list> <disassoc timer> <validity interval> [disassoc imminent] [bss termination bit] [bss terminaton tsf] [bss termination duration] [abridged] <VSIE byte1> <VSIE byte2><VSIE byte3> <VSIE byte4> <VSIE byte5> <VSIE byte6> <VSIE byte7><VSIE byte8> <VSIE byte9> <VSIE byte10>	wifitool ath0 sendbstmreq <associated_sta_mac> <candidate list> <disassoc timer> <validity interval> [disassoc imminent] [bss termination bit] [bss terminaton tsf] [bss termination duration] [abridged] <VSIE byte2><VSIE byte3> <VSIE byte4> <VSIE byte5> <VSIE byte6> <VSIE byte7><VSIE byte8> <VSIE byte9> <VSIE byte10>	The sendbstmreq command is used to specify the parameters or fields set during the sending of BSS transition management (BTM) request frames. ■ <associated_sta_mac>—MAC address of associated station in hexadecimal format. ■ <candidate list>—Preferred Candidate List Included bit. The AP responds with a BTM Request frame that includes a BSS Transition Candidate List Entries, which contains one or more Neighbor Report elements, each of which shall contain the BSS Transition Candidate Preference sub-element. ■ <disassoc timer>—Disassociation Timer field. The MBO AP sends a Disassociation Frame to the STA after expiry of the Disassociation Timer and no later than 2 seconds after the Disassociation Timer expires, only when the Disassociation Imminent field is set as one. The AP is not expected to send a Disassociation Frame if the STA has sent a Disassociation Frame to the AP beforehand. ■ <validity interval>—Time period for which the BTM is valid. ■ [disassoc imminent]—An MBO AP shall, prior to disassociating an MBO STA, send an unsolicited BTM Request frame with Disassociation Imminent field set to one to that STA in the unsolicited BTM Request frame. When the Disassociation Imminent field is set to one, the MBO AP sets the Disassociation Timer field to the number of TBTTs that will occur prior to the MBO AP disassociating the MBO STA. When the Disassociation Imminent bit is set to one and/or the BSS Termination Included field is set to one, the STA shall not reject the Transition Management Request. ■ [bss termination bit]—An MBO AP shall, prior to terminating a BSS, send an unsolicited BTM Request frame to all associated MBO STAs by setting the BSS Termination Included field to one. ■ [bss terminaton tsf]—When the BSS Termination Included field is set to one, the MBO AP sets the BSS Termination TSF field of the BSS Termination Duration field according to the time until the BSS is terminated, and also terminates the BSS after the BSS Termination TSF is reached. ■ [bss termination duration]—Time until which the BSS transition management is not terminated. ■ [abridged]— Abridged mode bit ■ <VSIE byte*N*>—These keywords are introduced in QCA_Networking_2020.SPF.11.2 release and are optional arguments that are appended to the frame (vendor specific IE, for example). To specify this IE, first the 5 optional args need to be provided, and then the IE bytes can be provided as a sequence of numbers. Users must make sure the IE is correctly formatted. Backward compatibility is retained for users upgrading to QCA_Networking_2020.SPF.11.2 release from earlier releases, and therefore, no adverse impact is caused for users that prefer to not configure these keywords.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 4-6 wifitool MBO parameters (cont.)

Parameter	Command	Description
wifitool athX sendbstmreq_target <associated_sta_mac> [-di <disassoc_duration> <disassoc_imminent>] (optional) [<candidate_bssid> <candidate channel> <candidate_preference> <operating class> <PHY type> ...]	wifitool athX0sendbstmreq_target <associated_sta_mac> [-di <disassoc_duration> <disassoc_imminent>] (optional) [<candidate_bssid> <candidate channel> <candidate_preference> <operating class> <PHY type> ...] Example: wifitool ath0 sendbstmreq_target a8:db:03:32:60:2d -di 200 1 00:03:7F:12:4E:C7 1 255 81 9	The sendbstmreq_target command is used to specify the parameters or fields set during the sending of BSS transition management (BTM) request frames. ■ <associated_sta_mac>-MAC address of associated station in hexadecimal format. ■ <disassoc timer>-Disassociation Timer field. The MBO AP sends a Disassociation Frame to the STA after expiry of the Disassociation Timer and no later than 2 seconds after the Disassociation Timer expires, only when the Disassociation Imminent field is set as one. The AP is not expected to send a Disassociation Frame if the STA has sent a Disassociation Frame to the AP beforehand. ■ <disassoc imminent>-An MBO AP shall, prior to disassociating an MBO STA, send an unsolicited BTM Request frame with Disassociation Imminent field set to one to that STA in the unsolicited BTM Request frame. When the Disassociation Imminent field is set to one, the MBO AP sets the Disassociation Timer field to the number of TBTTs that will occur prior to the MBO AP disassociating the MBO STA. When the Disassociation Imminent bit is set to one and/or the BSS Termination Included field is set to one, the STA shall not reject the Transition Management Request. ■ <candidate_bssid>- BSSID (mac address) of the neighbor AP. ■ <candidate_channel>- Neighbor AP's channel number. ■ <candidate_preference>- The Preference field value of a Neighbor Report element used in a BSS Transition Management Query frame shall be between 1 and 255. The value of 0 is reserved. The values between 1 and 255 provide the indication of order, with 255 indicating the most preferred BSS within the given candidate list, decreasing numbers representing decreasing preference relative only to entries with lower values of the Preference field, and equal numbers representing equal preference. ■ <operating class>- Operating class of the neighbor AP. ■ <PHY type>-The PHY Type field indicates the PHY type of the neighbor AP. It is an integer value coded according to the values as follows: PHY_type : value dsss: 2 ofdm: 4 hrdsss: 5 erp: 6 ht: 7 dmg: 8 vht: 9 tvht: 10

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 4-6 wifitool MBO parameters (cont.)

Parameter	Command	Description
athN sendbcnrpt <STA MAC> <regclass> <channum> <rand_ivl> <duration> <mode> <req_ssid> <rep_cond> <rpt_detail> <req_ie> <chanrpt_mode> [specific_bssid] [AP channel numbers]	wifitool athN sendbcnrpt <STA MAC> <regclass> <channum> <rand_ivl> <duration> <mode> <req_ssid> <rep_cond> <rpt_detail> <req_ie> <chanrpt_mode> [specific_bssid] [AP channel numbers]	The sendbcnrpt command is used to specify the parameters or fields for the Beacon Report. ■ <STA MAC>—MAC address of the station in aa:bb:cc:dd:ee:ff format. The MAC address parameter is used to specify the BSSID of an AP that the user wants to include in the neighbor report table. ■ <regclass>—Operating class contains a value, which specifies the operating class in which the Channel List is valid. An MBO STA informs the AP of its channel/band capabilities by including the Sup-ported Operating Classes element in the (Re)Association Request frame. ■ <channum>—•Channel number can be set to a specific channel (along with the appropriate operating class), 0 (along with an operating class), or 255 (along with one or more AP Channel Report sub-elements) ■ <rand_ivl>—Random interval, see OCE specification for details. ■ <duration>—Measurement duration, refer to OCE specification for definition. ■ <mode>—Measurement mode set to Active, Passive, or 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 OCE Technical 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 OCE Technical Specification for details. ■ <req_ie>—For current implementation, this should be set to zero. ■ <chanrpt_mode>—Reporting condition for beacon report. Refer to OCE Technical Specification for details. ■ [specific_bssid]—BSSID specified in the xx:xx:xx:xx:xx:xx format in the beacon report request. Instead of a wildcard BSSID, a specific BSSID is defined. ■ [AP channel numbers]— The Channel List contains a variable number of octets, where each octet describes a single channel number. An empty Channel List field indicates that the indicated Preference applies to all channels in the Operating Class. Channel numbering is dependent on Operating Class.

4.12 Using wifitool commands for cfg80211 mode: examples

For cfg80211 based drivers, no generic applications such as iwpriv or iwlist are present to configure private commands. Therefore Qualcomm Technologies developed an nl80211-based

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

cfg80211tool to configure private commands. Input to the cfg80211tool is aligned with existing iwpriv commands; for example, command form and the number of arguments are as same as existing iwpriv commands.

wifitool

Add the new switch **–cfg80211** to support cfg80211-based drivers. Addition of the **-cfg80211** switch is optional. If a certain proprietary wireless tool command, such as wifitool, is entered without the **-cfg80211** switch, the driver checks the mode that is being currently used—WEXT or cfg80211—on the system, and processes the command accordingly.

For example, for the WEXT command of `wifitool athX chmask_persta <mac_addr><nss>`, the corresponding cfg80211 command is `wifitool ath0 chmask_persta <mac_address>[-cfg80211]`.

4.13 Set data path tracing

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Optimization of OS abstraction layer* section of the *WLAN AP driver layers* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Wifitool command is used for setting parameters related to data path trace operations (DP_TRACE).

Protocol bitmap

Protocol Bitmap is used to indicate what protocol packets need to be recorded.

Command format: wifitool <interface> dp_trace pbitmap <value>

Example: wifitool ath0 dp_trace pbitmap 32

- QDF_NBUF_PKT_TRAC_TYPE_EAPOL 0x02
- QDF_NBUF_PKT_TRAC_TYPE_DHCP 0x04
- QDF_NBUF_PKT_TRAC_TYPE_MGMT_ACTION 0x08
- QDF_NBUF_PKT_TRAC_TYPE_ARP 0x10
- QDF_NBUF_PKT_TRAC_TYPE_ICMP 0x20

By default, **ARP, DHCP**, and **EAPOL** are logged.

Interval/Number of packets

This is used to indicate at what frequency, the packets should be recorded. For ex. Value of 10 indicates that every 10th packet will be recorded.

Command format: wifitool <interface> dp_trace nrecords <value>

Example: wifitool ath0 dp_trace nrecords 1

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Verbosity

Verbosity is used to determine the extent of tracing required.

Command format: wifitool <interface> dp_trace verbosity <value>

Example: wifitool ath0 dp_trace verbosity 3

The default verbosity is 3.

Live mode

Live mode displays all packets while they are being recorded.

Command format: wifitool <interface> dp_trace livemode

Example: wifitool ath0 dp_trace livemode

Clear DP trace buffer

Used to clear the trace buffer.

Command format: wifitool <interface> dp_trace clear_buf

Example: wifitool ath0 dp_trace clear_buf

QDF_TRACE_LEVEL

DP_TRACE prints come under QDF_MODULE_ID_QDF. TRACE must be enabled for this module to view the dp_trace prints. The following command is used to enable QDF DP trace level:

iwpriv ath0 qdf_cv_lvl 0x00380008

4.14 ACS channel score parameter

For more information on usage guidelines and the functionality regarding these parameters, see the *Channel-ranking information in ACS reports* section of the *Advanced channel selection capabilities* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

The following table describes the commands related to the channel score functionality for ACS.

Table 4-7 ACS channel score parameter

Parameter	Command	DA	OL	Description
acsreport	wifitool athX acsreport	Y	Y	Get the report based on last scanning.

The following is a sample output of an ACS report with all the channels ranked:

```
wifitool athx acsreport

Legend: SC: Secondary Channel, WR: Weather Radar,DFS: DFS Channel, HN:
High Noise, RSSI: Low RSSI,CL: High Channel Load
```

```
RP: Regulatory Power,N2G: Not selected 2G, P80X: Primary 80X80 NS80X:
Only for primary 80X80, NP80X: Only for Secondary 80X80

The number of channels scanned for acs report is:24

Channel | BSS| minrssi | maxrssi | NF | Ch load | spect load | sec_chan | Ranking | Unused
-------------------------------------------------------------------------------------

5180( 36) 13    12      71     -107   45       0          1         24       ()
5200( 40) 1     51      51     -107   1        0          1         19       ()
5220( 44) 1     37      37     -107   3        0          1         16       ()
5240( 48) 0     0       0      -107   1        0          1         6        ()
5260( 52) 0     0       0      -107   1        0          1         1        ()
5280( 56) 0     0       0      -107   1        0          1         13       ()
5300( 60) 0     0       0      -106   1        0          1         17       ()
5320( 64) 10    10      64     -106   13       0          0         23       ()
5500(100) 1     15      15     -102   5        0          0         5        ()
5520(104) 0     0       0      -104   0        0          0         2        ()
5540(108) 0     0       0      -104   0        0          1         3        ()
5560(112) 0     0       0      -101   51       0          1         4        ()
5580(116) 1     8       8      -104   1        0          1         7        ()
5600(120) 0     0       0      -103   1        0          1         9        ()
5620(124) 1     59      59     -104   1        0          0         11       ()
5640(128) 1     45      45     -105   1        0          0         14       ()
5660(132) 0     0       0      -104   1        0          1         12       ()
5680(136) 2     45      54     -103   1        0          0         15       ()
5700(140) 0     0       0      -103   1        0          1         8        ()
5745(149) 8     6       52     -103   3        0          0         21       ()
5765(153) 3     39      39     -104   4        0          1         22       ()
5785(157) 6     10      17     -104   6        0          1         20       ()
5805(161) 3     46      48     -104   3        0          1         18       ()
5825(165) 0     0       0      -105   1        0          0         10       ()
```

The following is a sample output of an ACS report with only some of the channels ranked and the remaining channels disregarded from ranking, along with reason codes:

```
wifitool athx acsreport
Legend: SC: Secondary Channel, WR: Weather Radar,DFS: DFS Channel, HN:
High Noise, RSSI: Low RSSI,CL: High Channel Load
RP: Regulatory Power,N2G: Not selected 2G, P80X: Primary 80X80 NS80X:
Only for primary 80X80, NP80X: Only for Secondary 80X80

The number of channels scanned for acs report is:11

Channel    | BSS   | minrssi | maxrssi | NF | Ch load | spect load | sec_chan |
Ranking | Unused
-------------------------------------------------------------------------------------
------------
5180( 36)12  29    71     -107  4      0            1      0      (SC )
5220( 44)1   37    37     -107  1      0            1      0      (SC )
5260( 52)0   0     0      -107  1      0            1      4      (Random SC )
5300( 60)0   0     0      -105  1      0            1      0      (SC )
5500(100)1   15    15     -102  4      0            0      1      ()
5540(108)0   0     0      -103  0      0            1      0      (SC )
```

Qualcomm Confidential – May Contain Trade Secrets 2014-05-19 03:57:44 PDT pan.yafeng@zte.com.cn

```
5580(116)1   8      8      -103  1      0            1      0      (SC )
5620(124)1   59     59     -104  1      0            0      2      ()
5660(132)0   0      0      -104  1      0            1      0      (SC )
5745(149)8   6      52     -103  10     0            0      3      ()
5785(157)5   10     16     -104  4      0            1      0      (SC )
```

4.15 Softblocking wifitool parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Soft-blocking for safe connections* section of the *Wi-Fi SON: Steering and APS* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

- Enter the iwpriv athX addmac_sec <mac> command to add specified MAC addresses to the secondary access control list (ACL).
- Enter the iwpriv athX maccmd_sec 2 command to instruct how the ACL is used to limit access the AP. The value of 2 denies association with any MAC address on the ACL.
- Enter the iwpriv athX wait_time <0-100000> and iwpriv athx g_wait_time commands to set/get the parameter of the wait time in microsecond, default is 10000.
- Enter the iwpriv athX allow_time <0-100000> and iwpriv athx g_allow_time commands to set/get the parameter of the allow time in microsecond, default is 15000.
- Enter the wifitool athX softblocking <dstmac> <0/1> and wifitool athX softblocking_get <dstmac> commands to set/get the enable flag of the dstmac. Setting the flag resets the timer to the initial value.

Table 4-8 Softblocking parameters

Parameter	Command	Description
softblocking <dstmac> <0/1> softblocking_get <dstmac>	wifitool athX softblocking <dstmac> <0/1> wifitool athX softblocking_get <dstmac>	Set/get the enable flag of the destination MAC address (dstmac). Setting the flag will reset the timer to initial value. Similar to the SSID steering mechanism, the soft-blocking capability is to force private users to connect to private SSID, instead of a public SSID. The soft-blocking capability is a safety mechanism; it tries to allow the client to connect, if the public SSID is continued to be connected in a period. If client associates on Open/EAP Public SSID, check blacklist for client's MAC address. If MAC address is on the blacklist, perform a 'Soft Block' to deny the client to authenticate with Public SSID. MAC address will remain in blacklist of Open/EAP SSID until it is manually cleared.

4.16 Retrieve Radio Resource Management utilities for parameter APIs in carrier networks

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Support for reporting associated and steering stations for carrier*

networks section of the *Wi-Fi SON: Additional features* chapter in the Wireless LAN Access Point (Driver Version 11.0) Programming Guide (80-YA728-6).

Starting with QCA_Networking_2016.SPF.6.0, support is implemented for generating reports of associated and non-associated stations, and steered and non-steered stations, which are beneficial for collecting statistics and computing usage of subscriber services to aid troubleshooting and diagnosis. Users can enable or disable the collection of parameter APIs, which refer to the associated, non-associated, steered, and non-steered stations using the iwpriv wifiX set_papi <0 | 1> command for the WLAN driver to transmit or not transmit the report to user space.

- New structure ieee80211_rrmutil_t , corresponding to the following values, is added that is sent as a data pointer for ioctl IEEE80211_DBGREQ_TR069 (sub-ioctl: TR069_GET_RRM_UTIL).
- UCI configuration is enhanced to serve request and handle support for sub ioctl
- Wifitool is enhanced by adding new command and relavent code for help and command handling.

Name	High Level Implementation Details	Usage Details
Device.WiFi.Radio.{i}.X_<OEM>_COM.RRM.Util.		
Channel	Reutilized Current channel in ic structure	wifitool athX get_rrmutil
ChanUtilization	Reutilized channel utilization in ic which is derived from WMI_DCS_INTERFERENCE_EVENT_ID	wifitool athX get_rrmutil
OBSSUtilization	Enhanced ic to save bss utilization which is derived from WMI_DCS_INTERFERENCE_EVENT_ID	wifitool athX get_rrmutil
NoiseFloor	Reutilizaed ic pointer to get noise floor on current channel.	wifitool athX get_rrmutil
RadarDetected	Reutilized ic flags to check for IEEE80211_CHAN_CSA_RECEIVED	wifitool athX get_rrmutil

4.17 CBS for VoW parameters

To provide IPTV services in real-time using Wi-Fi connections, the quality must be guaranteed from interference. The AP must be able to scan its full band periodically, and change the channel without any deterioration of service quality. Starting with QCA_Networking_2017.SPF.6.0 release, support is implemented for continuous background scan (CBS) to resolve the issues of legacy scan for video over wireless (VoW) scanning and switching channels without impacting services.

For more information on usage guidelines and the functionality regarding these parameters, see the *Support for CBS for VoW switching and scanning channels* section of the *Video and multimedia features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 4-9 CBS for VoW parameters

Parameter	Command	Description
setchanlist ch1 ch2 .....n	wifitool athX setchanlist ch1 ch2 .....n	Command to set user desired channel list to scan. Pass all the channels as space separated values. Negative and invalid channel numbers also get accepted but can stop the scan. Channels are unsigned integers so any value given is truncated to the unsigned 8-bit number. If this is valid (even though the passed value may not be) scan runs without any issues. Example: #wifitool ath0setchanlist36 40 -212 will be translated to channel configuration of [36, 40, 44] although -212 is invalid.
getchanlist	wifitool athX getchanlist	Command to retrieve the user-desired channel list to scan.

4.18 Debug commands for AC-based ATF

For more information on usage guidelines and the functionality regarding these parameters, see the *Access category based-ATF for non-offload chipsets* section of the *Memory and bandwidth optimization features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Access category (AC)-based ATF is supported for non-offload radios or chipsets. With the implementation of the AC-based ATF, the ATF module supports following configurations:

1. SSID Based Configuration
   a. User can assign airtime to a specific SSD/VAP
2. Peer Based Configuration
   a. User can assign airtime to a specific peer. The percentage would be computed based on the percentage allocated to the VAP it's connected to. For example, a peer configured with 10% airtime connected to a VAP with 80% airtime. This peer would then be assigned 8% of the airtime (10% of 80).
3. SSID Group configuration
   a. User can create a GROUP and map multiple SSID to the same group. The group can then be assigned an airtime which would be shared across all SSID's in this group

Starting with QCA_Networking_2017.SPF.6.0 release, one more ATF configuration type, Access Category based ATF, is introduced.

The following rules apply for AC-based ATF:

1. User must be able to configure access-based rule for an SSID or SSID group.
2. The AC-based airtime is applicable only for unconfigured clients in that SSID or SSSID group.

   The unconfigured airtime for an SSID/SSID group must be shared as per the AC configuration provided by the user.
3. AC based configuration does not have any effect on Configured Clients. In other words, all ACs of a configured client will share the airtime allocated to that specific client (as it exists before the implementation of this functionality)

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

A debug command is added to dump the token allocation per subgroup. Set the debug log size before starting the test. The debug log can be dumped towards the end of the test.

```
wifitool <vap> atf_debug_size  128 // Set debug log size to 128
wifitool <vap> atf_dump_debug     // Dump debug logs for all subgroup.
```

A debug command is added to display the node pointer to subgroup mapping & also the TID state.

```
wifitool <vap > atf_debug_nodestate <node mac>
```

4.19 Configure allowed channels and ACS-allowed channels

For more information on usage guidelines and the functionality regarding these parameters, see the *Configure allowed channels and ACS-allowed channels* section of the *Advanced channel selection capabilities* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

The functionality to enable the AP VAP to select primary channel from the user-configured primary-allowed channel list is implemented. A compile time macro is introduced to enable/disable this feature to avoid increasing object file size. The macro name is **ATH_ SUPPORT_PRIMARY_ALLOWED_CHAN.**

ATH_SUPPORT_PRIMARY_ALLOWED_CHAN is enabled by default in config.wlan.unified.perf, config.wlan.unified.profile. A flag is added to enable/disable this feature dynamically in struct ieee802com **ic_primary_allowed_enable;** this flag is disabled by default. This flag is enabled when a user sets primary allowed channel list.

A wifitool command is added to set/get primary allowed channel list. To set primary allowed channel list, enter the **wifitool athX setprimarychans ch1 ch2 ….. chn** command. Each channel number is separated by a space. To get primary allowed channel list, enter the **wifitool athX getprimarychans** command.

The **wifitool athX setprimarychans ch1 ch2 ….. chn** command adds the user-configured primary allowed channels to **ieee80211_primary_allowed_chanlist_t *ic_primary_chanlist** and enables **ic_primary_allowed_enable** flag. To clear the previously stored channel list, run the **wifitool athX setprimarychans** command without any channel value.

The **wifitool athX getprimarychans** command lists all the primary allowed channels, which is present in the **ieee80211_primary_allowed_chanlist_t *ic_primary_chanlist** if **ic_primary_ allowed_enable** flag is enabled. Otherwise, it returns with the console error message.

4.19.1 Guidelines

The following guidelines apply to the usage of this functionality to configure primary-allowed list of channels:

- If the athX current channel is not one among the primary allowed channel, it restarts all VAPs that belong to the same radio and it selects the best channel from the primary-allowed channel list.
- When ic_primary_allowed_enable flag is enabled, if the primary channel selected by ACS is not present in the primary allowed channel list then AP prints error message and it continues to

select the next channel. At the end, the best channel is selected from the primary allowed channel list.

- If ic_primary_allowed_enable flag is enabled, the channel specified by iwconfig athN channel chan_num is ignored with a console error message if the chan_num is not present in the primary allowed channel list.
- If ic_primary_allowed_enable flag is enabled, the channel specified by iwpriv athN doth_ chanswitch chan_num tbtt is ignored with the console error message if the chan_num is not present in the primary allowed channel list.
- If ic_primary_allowed_enable flag is enabled while AP detecting radar, it chooses a channel from the primary allowed channel list to operate. If all primary allowed channels are in NOL list, AP does not send a beacon for next 30 mins. To avoid this problem, a user can enter at least one non-DFS primary channel while configuring primary allowed channel list in 5 GHz.

4.19.2 Limitations

- The `iwpriv wifiX acs_2g_allchan` command must be enabled while configuring the primary allowed channel list in 2.4 GHz.

  If this option is enabled, AP can select one of the overlapping channels with ACS because ACS provides a higher precedence to the channels 1, 6, and 11.

  2.4 GHz AP VAP is configured in HT40+ mode and primary allowed channels given at runtime is 8 9 10 11. ACS has a higher precedence for the htmode. ACS uses the first available channel in that mode to operate. The `iwconfig athX channel <no>` command provides a higher precedence to the channel number. It is not possible to implement changes in ACS to downgrade HT40+ mode. In a subsequent release, support might be added to bring down the VAP in such a case.

- 2.4 GHz AP VAP is configured in HT40- mode and primary allowed channels that are entered at runtime are 1 2 3 4.

  ACS provides a higher precedence to the htmode. ACS uses the first available channel in that mode to operate. It is not possible to implement changes in ACS to downgrade HT40- mode.

  In a subsequent release, support might be added to bring down the VAP in such a case.

  If a user configures primary-allowed channels that are heavily loaded and other channels are free, the AP VAP does not pick up free channels (which is not present in the primary allowed channel list) to operate. Instead, the AP VAP chooses the best channel from the primary allowed channel list.

- ACS report lists all the channels and while selecting primary channel, the best channel is selected from the primary allowed channel list.
- This feature does not support Dynamic Channel Selection (DCS).
- This feature is implemented only for root AP and repeater mode is not currently supported. The configuration fails with a console error message if a user attempts to configure primary allowed channel list in repeater mode.
- The iwpriv command support is currently not added in this feature; instead wifitool support is available to get/set primary allowed channel list.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

- If a user provides only DFS channels in primary allowed channel list and if all the channels are added to NOL, AP prints a message saying that no channel is available and stops functioning for next 30 minutes. Therefore, a user has to provide at least one non-DFS channel in the primary allowed channel list.

4.20 Extended VoW statistics

For more information on usage guidelines and the functionality regarding these parameters, see the *Extended VoW statistics* section of the *Video and multimedia features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Support for VoW statistics is extended. Debug counters are added. The host-change involves the TID value that was being overwritten in Tx path of host driver. It was caused as part of Mesh Mode changes, which unconditionally modified TID using SKB priority field. The fix is to retain TID as sent by transmitter.

Debug counters are added at following points in Wi-Fi host:

- Counters to keep track of number of video packets
- Counters to calculate interframe delay between video packets.

4.20.1 FW changes

- Excessive Retry drop threshold is customized per AC and configured to 40 by wifitool command.
- Fallback Retry threshold wifitool command added and set to 30 by default
- Fallback Rates configuration option added for MCS/NSS/Preamble/BW
  - By default, VHT MCS0 VHT20 set as fallback rate
  - The option to modify this value is provided by wifitool command.
- SGI enable for all rate option provided. By default it is disabled
- Change to pause/unpause traffic when continuous retry crosses threshold done when RTS is enabled
- RTS is suggested to be enabled by default for tests (hidden node scenario tests)
- RSSI reporting improvements done to report RSSI for data and MGMT frames separately

4.20.2 FW debug changes

- wifitool ath0 beeliner_fw_test 119 1—Rx path counters for peer ID
- wifitool ath0 beeliner_fw_test 120 1—Tx path counters per peer ID
- wifitool ath0 beeliner_fw_test 122 1—MSDU counters for peer ID
- wifitool ath0 beeliner_fw_test 123 1—Number of MSDU error counters global

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

- wifitool ath0 beeliner_fw_test 170 0—Latency histogram measurements done in FW enqueue path
- wifitool ath0 beeliner_fw_test 171 0—Continuous retry histogram done in FW
- wifitool ath0 beeliner_fw_test 172 0—Data RSSI report along with SNR and NF
- wifitool ath0 beeliner_fw_test 175 0—Management RSSI report along with SNR and NF

Beacon RSSI and Data RSSI can be seen only when traffic runs.

4.20.3 Wi-Fi firmware

From the firmware (FW) side, the packet counters for video can be printed using the following command:

```
wifitool ath0 beeliner_fw_test 191 2
```

A sample output of the preceding command is as follows:

```
[58427.105000] [wifi0] FWLOG: [59783627] WAL_DBGID_TX_AC_BUFFER_SET (
0x1b037, 0x1b037, 0xa06307f, 0xa06307f )
[58427.114715] [wifi0] FWLOG: [59783627] WAL_DBGID_TX_AC_BUFFER_SET (
0x0, 0x0, 0x0, 0x2 )
[58427.122713] [wifi0] FWLOG: [59783627] WAL_DBGID_TX_AC_BUFFER_SET (
0x0, 0x0, 0x0, 0xdb0 )
[58427.130867] [wifi0] FWLOG: [59783627] WAL_DBGID_TX_AC_BUFFER_SET (
0x0, 0x0, 0x0, 0x0 )
[58427.138864] [wifi0] FWLOG: [59783627] WAL_DBGID_TX_AC_BUFFER_SET (
0x1, 0x0 )
```

The preceding command prints five lines of logs. The packet counters are only present in the first line. Rest of the prints are other counters from FW, which are not relevant to the packet counters.

The format is as follows:

```
FWLOG: [59783627] WAL_DBGID_TX_AC_BUFFER_SET ( RX_FROMAIR, RX_SENTTOHOST,
TX_FROMHOST, TX_SENTONAIR )
```

In the case of AP, observe the **TX_FROMHOST** and **TX_SENTONAIR** counters. In the case of STA, observe the **RX_FROMAIR** and **RX_SENTTOHOST** counters.

To find the correct number of packets during the test, run this command before and after traffic is run and find the delta for each of the counters.

The AP-side TX_FROMHOST must match the STA-side RX_SENTTOHOST. Any difference in this would mean packets are lost between AP side FW and STA side FW components.

NOTE For IPQ4019 chipsets, the interfaces used are ath2 and ath21. Therefore, change ath*X* interface accordingly for this command.

4.21 wifitool commands for distributed VoW

For more information on usage guidelines and the functionality regarding these parameters, see the *Distributed VoW* section of the *Video and multimedia features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Distributed Video over Wireless (VoW) requires that VoW does not have any single frame loss with the tests that are run under distributed VoW use-cases. This mechanism ensures that video streaming functionality works well with less jitter, lower latency, and zero loss in video stream.

4.21.1 Design enhancements

Based on distributed VoW requirements, the following design enhancements are implemented:

1. Retry threshold for the transmitted frames is increased from 30 to 200 so that frames are not dropped with less retries.
2. Bursting of best effort traffic is set to be higher than 2 ms to prioritize video traffic to reduce latency. Video traffic is bursted. With no burst for best effort traffic, latency of video traffic is reduced.
3. Module parameters, ‘carrier_vow_config’ and ‘carrier_vow_stats’, are added to enable VoW configuration in firmware and enable Vow statistics in firmware.
4. Several statistics are added to assist in debugging VoW issues related to frame loss, latency. MSDU level Tx enqueue time statistics, MSDU level Rx completion statistics, MSDU level transmit/ receive statistics per access category are added.
5. Statistics, per peer, are included up to 8 clients to assist in debugging multi client VoW issues.
6. Software RTS, instead of hardware RTS, is added in security mode to address the issue that packets reaching to firmware without decryption from hardware.

Several wifitool commands were added for VoW enhancements. The commands are used to enable VoW features and VoW statistics, and print the statistics, which are useful for debugging. The following are the newly added wifitool commands:

1. wifitool command 191 <AC> – Per-AC statistics for the packet counters and error counters in Tx and Rx paths
2. wifitool command 192 – Enable decrypt error logs
3. wifitool commands 192/194/196/199 – Enable PPDU truncation for RTS mode and configure truncation size
4. wifitool command 195 <1/2/3/4/5>
   a. Histogram of PHY rates used
   b. Software retries
   c. Tx path delay factor latency
   d. Inter-packet histogram
   e. Rx path latency DF
5. wifitool command 197 – BE queue Tx timeout avoid waldev_reset

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

6. wifitool command 198 <AC> - Per AC set Rx reorder timeout
7. wifitool command 200 – Enable RSSI based ANI control
8. wifitool command 201 – Specify the option for host to indicate Host side peer_q full condition to FW
9. wifitool command 212 – Specify the option to configure SW_RTS generation
10. wifitool command 214 - Set Rx reorder flush skip mask to control behavior per AC
11. wifitool command 223 – Configure MSDU enqueue time feature that will enable DF test latency stats
12. wifitool command 224 – Configure option to enable all the histogram based stats
13. wifitool command 228 - Specify the per-peer stats for the packet counters and error counters in Tx and Rx path for ACs 0/1/2/3
14. wifitool command 229 – Specify the nodal VoW enable command which is used to set all other wifitool commands to the required configuration for VoW configuration
15. wifitool command 233 - Configure the dynamic pktlog feature to start a dynamic pktlog with DF latency threshold so that pktlog stops capturing when the threshold latency is reached

4.22 wifitool commands to configure AP TWT

For more information on usage guidelines and the functionality regarding these parameters, see the *Configure AP TWT* section of the *Beacons and frames transmission* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Negotiation of implicit Target Wake Time (TWT) sessions, including handling of TWT Information elements is supported. Host-side enhancements are implemented for AP TWT.

To initiate a TWT session, a negotiation phase occurs at first in which the AP and the target station agree on a common set of parameters. The significant parameters are as follows:

- *Target Wake Time (TWT)*: The next point in time, in microseconds, at which the station participating in the TWT-based communication must wake up for the TWT session.
- *TWT Wake interval*: The time interval between subsequent TWT sessions for the station; the value is higher than 0 when the TWT is periodic.
- *Minimum TWT awake duration*: Minimum time duration a station must stay awake since the starting time of the TWT session so as to be able to receive frames from the other stations.
- *TWT Channel*: The channel a station can use temporarily as the primary one.
- *TWT Protection*: Mechanism employed to protect a TWT session from transmissions of external stations, such as the RTS/CTS.

The TWT timing can be explicit or implicit: while the former requires specification of the TWT parameters before each new session, the latter allows to set periodic sessions by relying on the first set of parameters until a new set is received. After the TWT parameters are agreed, the stations can go to sleep until the next TWT session is reached. In practice, each station can establish up to 8 TWT sessions with the AP.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

To modify TWT-enabled configuration parameters, enter the following commands:

```
wifitool athX ap_twt_add_dialog <peer_mac> <dialog id> <wake intvl (us)>
<wake int mantis> <wake dur(us)> <sp offset(us)> <cmd (0-7)> <flags>
wifitool athX ap_twt_del_dialog <dialog id>
wifitool athX ap_twt_pause_dialog <dialog id>
wifitool athX ap_twt_resume_dialog <dialog id> <sp offset(us)>
```

4.23 Configure off-channel Rx using wifitool command

For more information on usage guidelines and the functionality regarding these parameters, see the *Off-channel Rx for 20, 40, 80, and 160 MHz channels* section of the *Beacons and frames transmission* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Off-channel Rx is extended to channels with BW of 40 MHz, 80 MHz, and 160 MHz, in addition to the support that previously existed for channels with BW of 20 MHz. Support for end-to-end 160 MHz depends on HALPHY support.

Enter the `iwpriv wifiX widebw_scan 1` command for wideband scan (40, 80, and 160). This command must be specified before enabling off-channel Rx.

```
wifitool athX  offchan_rx_test <chan> <scan_dur> <bw_mode>(optional)
```

- bw_mode – integer takes one of the following values:
  - 20, 40-, 40+,40, 80, 160 are the supported bandwidth values
  - If not specified, the default is 0, which represents 20 MHz.

Driver design

- The wifitool is supported with entirely new API, which updates the scan parameters.
- *ieee80211_update_scan_channel_phymode* is reused to update the right PHY mode for the wide band scan.

Generic: wifitool athX offchan_rx_test <chan> <scan_dur> <bw_mode> (optional)

1. wifitool ath0 offchan_rx_test 149 80 160 – To scan in off-channel 149 for 80ms in 160 MHz bandwidth.
2. wifitool ath0 offchan_rx_test 149 80 80 – To scan in off-channel 149 for 80ms in 80 MHz bandwidth.
3. wifitool ath0 offchan_rx_test 149 80 40 – To scan in off-channel 149 for 80ms in 40 MHz bandwidth.
4. wifitool ath0 offchan_rx_test 149 80 40- – To scan in off-channel 149 for 80ms in 40 MHz bandwidth with secondary channel offset below.
5. wifitool ath0 offchan_rx_test 149 80 40+ – To scan in off-channel 149 for 80ms in 40 MHz bandwidth with secondary channel above.
6. wifitool ath0 offchan_rx_test 149 80 20 – To scan in off-channel 149 for 80ms in 20 MHz bandwidth.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

7. wifitool ath0 offchan_rx_test 149 80 – To do normal scan in off-channel 149 for 80 ms, which is 20MHz bandwidth.

4.24 Changes to wifitool commands for 6 GHz

- wifitool athN setchanlist ch1 ch2….chN.

  This command is changed to accept optional band argument as follows:

  wifitool athN setchanlist [-band B1] ch1 ch2 [-band B2] chA chB chC [-band B3] chX chY chZ.

  Here, ch1, ch2 are in band B1, chA, chB, chC are in band B2 and remaining chX, chY, chZ are in band B3. If channels from only one band are required to be specified, one -band option is adequate.

  Example:

```
root@OpenWrt:/# wifitool ath0 setchanlist -band 0 36 40 44 -band 2 100
104 108
```

- wifitool athN getchanlist:

  The following is a sample output:

```
root@OpenWrt:/# wifitool ath0 setchanlist -band 0 36 40 44 -band 2 100
104 108
root@OpenWrt:/# wifitool ath0 getchanlist
Following list used in channel load report
 -band 2 36  40  44  100  104  108
root@OpenWrt:/# wifitool ath0 setchanlist  36 40 44 100 104 108
root@OpenWrt:/# wifitool ath0 getchanlist
Following list used in channel load report
 -band 2 36  40  44  100  104  108
```

- wifitool athN block_acs_channel wifitool athN block_acs_channel [-band B1] ch1,ch2 [-band B2] chA,chB,chC [-band B3] chX,chY,chZ

  The following is a sample output:

```
root@OpenWrt:~# ./wifitool ath0 block_acs_channel 36,40 -band 2
44,48,52,56,100, 104,120,124
Following channels are blocked from Channel selection algorithm
 -band 0 [36] [40]  -band 2 [44] [48] [52] [56] [100] [104] [120] [124]
root@OpenWrt:~# ./wifitool ath0 block_acs_channel
36,40,44,48,52,56,100,104,120, 124
Following channels are blocked from Channel selection algorithm
 -band 0 [36] [40] [44] [48] [52] [56] [100] [104] [120] [124]
```

- wifitool athX custom_chan_list -a [-band B1] <number of channels> <ch1> <ch2> ..<chN> -n [-band B2] <number of channels> <ch1> <ch2> ..<chN> [-band B3] <number of channels> <ch1> <ch2> ..<chN>

  The following is a sample output:

```
root@OpenWrt:/# wifitool ath01 custom_chan_list -a 5 100 104 108 36 40
-n 4 52 5 6 44 48
root@OpenWrt:~# wifitool ath01 custom_chan_list -n 2 36 40 -band 2 2
44 48
root@OpenWrt:~# wifitool ath01 custom_chan_list -a -band 0 3 100 104
108 -band 2 2 36 40 -n 2 52 56 -band 2 2 44 48
```

- wifitool athX offchan_tx_test [-band B] chan dwell_time

  The following is a sample output:

```
root@OpenWrt:~# wifitool ath0 offchan_tx_test  44 45
root@OpenWrt:~# wifitool ath0 offchan_tx_test -band 2 100 45
```

- wifitool athX offchan_rx_test [-band B] chan dwell_time

  The following is a sample output:

```
root@OpenWrt:~# wifitool ath0 offchan_rx_test  44 45
root@OpenWrt:~# wifitool ath0 offchan_rx_test -band 2 100 45
```

- wifitool ath0 setprimarychans [-band B1] ch1 ch2 ..chn [-band Bn] ch1 ch2 ..chn

  The following is a sample output:

```
root@OpenWrt:~# wifitool ath0 setprimarychans 36 40 44 48 -band 2 100
-band 2 104 108
root@OpenWrt:~# wifitool ath0 setprimarychans 36 40 44 48 100 104 108
```

- wifitool ath0 getprimarychans

  The following is a sample output:

```
root@OpenWrt:~# wifitool ath0 setprimarychans 36 40 44 48 -band 2 100
-band 2 104 108
root@OpenWrt:~# wifitool ath0 getprimarychans
Allowed primary channel list
 -band 2    36    40    44    48    100    104    108
root@OpenWrt:~# wifitool ath0 setprimarychans 149 153
root@OpenWrt:~# wifitool ath0 getprimarychans
Allowed primary channel list
 -band 2    149    153
```

- wifitool ath0 acsreport: band info is added as part of channel number. Note the " – X Ghz".

```
root@OpenWrt:/# wifitool ath0 acsreport


Legend:
SC: Secondary Channel            WR: Weather Radar
DFS: DFS Channel                 HN: High Noise
RS: Low RSSI                     CL: High Channel Load
RP: Regulatory Power             N2G: Not selected 2G
P80X: Primary 80X80              NS80X: Only for primary 80X80
NP80X: Only for Secondary 80X80  SR: Spacial reuse

 The number of channels scanned for acs report is:  25

 Channel        | BSS  | minrssi | maxrssi | NF | Ch load | spect load | sec_chan | SR bss | SR load | Ch avil | Chan eff | NearBSS | Med BSS | Far BSS | Eff BSS | grade | Rank | Unused | Radar
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
 5180( 36 - 5 GHz)   0       0        0   -107      4          0          0        0        0      98      892       0       0       0     102     91     0        0     0
 5200( 40 - 5 GHz)   0       0        0   -107     13          0          0        0        0      97      932       0       0       0     103     96     0        0     0
 5220( 44 - 5 GHz)   0       0        0   -107      4          0          0        0        0      96      951       0       0       0     104     99     0        0     0
 5240( 48 - 5 GHz)   0       0        0   -107      5          0          0        0        0      94      933       0       0       0     106     99     0        0     0
 5260( 52 - 5 GHz)   0       0        0   -107      4          0          1        0        0      89      883       0       0       0     112     99     0        0     0
 5280( 56 - 5 GHz)   1      10       10   -107      3          0          0        0        0      80      800       0       0       1     125    100     0        0     0
 5300( 60 - 5 GHz)   0       0        0   -107      6          0          1        0        0      89      883       0       0       0     112     99     0        0     0
 5320( 64 - 5 GHz)   0       0        0   -107      6          0          1        0        0      94      933       0       0       0     106     99     0        0     0
 5500(100 - 5 GHz)   0       0        0   -107      2          0          0        0        0      96      961       0       0       0     104    100     0        0     0
 5520(104 - 5 GHz)   0       0        0   -107      4          0          0        0        0      94      933       0       0       0     106     99     0        0     0
 5540(108 - 5 GHz)   0       0        0   -107      4          0          1        0        0      89      883       0       0       0     112     99     0        0     0
 5560(112 - 5 GHz)   1      18       18   -107      3          0          0        0        0      80      784       0       0       1     125     98     0        0     0
 5580(116 - 5 GHz)   0       0        0   -107      6          0          0        0        0      89      875       0       0       0     112     98     0        0     0
 5600(120 - 5 GHz)   0       0        0   -107      5          0          0        0        0      94      886       0       0       0     106     94     0        0     0
 5620(124 - 5 GHz)   0       0        0   -107      8          0          0        0        0      96      923       0       0       0     104     96     0        0     0
 5640(128 - 5 GHz)   0       0        0   -107      8          0          0        0        0      97      951       0       0       0     103     98     0        0     0
 5660(132 - 5 GHz)   0       0        0   -107     12          0          0        0        0     100      970       0       0       0     100     97     0        0     0
 5680(136 - 5 GHz)   0       0        0   -107      6          0          0        0        0     100      990       0       0       0     100     99     0        0     0
 5700(140 - 5 GHz)   0       0        0   -107      4          0          0        0        0     100      990       0       0       0     100     99     0        0     0
 5720(144 - 5 GHz)   0       0        0   -107      3          0          0        0        0     100      990       0       0       0     100     99     0        0     0
 5745(149 - 5 GHz)   1       5        5   -107     11          0          0        0        0      80      792       0       0       1     125     99     0        0     0
 5765(153 - 5 GHz)   0       0        0   -107     13          0          1        0        0      89      875       0       0       0     112     98     0        0     0
 5785(157 - 5 GHz)   0       0        0   -107      1          0          0        0        0      94      905       0       0       0     106     96     0        0     0
 5805(161 - 5 GHz)   0       0        0   -107      7          0          0        0        0      96      951       0       0       0     104     99     0        0     0
 5825(165 - 5 GHz)   0       0        0   -107      4          0          0        0        0     100        0       0       0       0       0     99     0        0     0
ACS_SUCCESS: Current channel is selected by ACS algorithm
-------------------------------------------------------------------------------
Additonal Channel Statistics
-------------------------------------------------------------------------------
 Index | Channel | NBSS |        SSID      |            BSSID      |    RSSI | PHYMODE
-------------------------------------------------------------------------------
root@OpenWrt:/#
```

- root@OpenWrt:/# wifitool ath0 tr069_chanhist: New Output will have band info printed along with channel number as follows:

```
root@OpenWrt:/# wifitool ath0 tr069_chanhist
 Channel | band | Selection Time
-------------------------------------
   44       2       2019-09-16 15:46:07 UTC
  149       2       2019-09-16 15:44:57 UTC
root@OpenWrt:/#
```

- root@OpenWrt:/# wifitool ath0 tr069_chan_inuse: New Output will have band info printed along with channel number as follows:

```
root@OpenWrt:/# wifitool ath0 tr069_chan_inuse
 List of Channels In Use
-------------------------
 -band 2 36,40,44,48,52,56,60,64,100,104,108,112,116,120,124,128,132,136,140,144,149,153,157,161,165
root@OpenWrt:/#
```

4.25 Configure standalone sounding sequence

Starting with QCA_Networking_2020.SPF.11.2 release, this functionality is supported on IPQ807x/IPQ817x/IPQ807xA and IPQ60xx platforms. Starting with the IPQ5018.ILQ.11.3 release, this functionality is supported on IPQ50xx platforms.

The functionality to support standalone sounding sequence to stations with no data frames enqueued is introduced.Certain vendor-specific proprietary location module run in the context of RT thread in the FW and location module sends message to scheduler to start standalone sounding sequence.

Standalone SU sounding sequence will be supported in following configurations: VHT and HE, 2.4 and 5GHz

Standalone MU sounding sequence is not planned to be supported in the following scenarios:

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

- HE MU: reception of UL OFDMA frame not supported on the radio doing location measurement
- VHT MU: No strong requirement to support VHT MU sounding sequence from cisco
- Complexity in grouping users for MU sounding sequence

Trigger HE SU standalone sounding sequence

Prototype code available for standalone sounding sequence

Support for triggering scheduler thread to initiate SU sounding sequence using unit test command

```
# wifitool ath0 setUnitTestCmd 0x47 2 223 1
# wifitool ath0 setUnitTestCmd 0x47 6 264 0 2 0 0 0

wifitool ath0 setUnitTestCmd 0x47 6 264 <pdev_id> <sw_peer_id>
<tid no>  <codebook> <ng>

# wifistats wifi0 12 | grep ax_su
ax_su_ndpa_queued = 1
ax_su_ndpa_tried = 1
ax_su_ndp_queued = 1
ax_su_ndp_tried = 1
ax_su_ndp_err = 0
ax_su_ndp_flushed = 0
ax_su_ndpa_err = 0
ax_su_ndpa_flushed = 0
ax_su_ndpa_sch_status =  1:1, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
ax_su_ndp_sch_status =  1:1, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0,
```

Qualcomm Confidential – May Contain Trade Secrets 2021-12-13 03:57:44 PDT pan.yafei@zte.com.cn

Trigger standalone sounding sequence

Prototype code available for stand alone sounding sequence

Support for triggering scheduler thread to initiate SU sounding sequence using unit test command

```
# wifitool ath0 setUnitTestCmd 0x47 2 223 1
#wifitool ath0 setUnitTestCmd 0x47 6 264 <pdev_id> <sw_peer_id>
                <tid no>  <codebook> <ng>

# wifitool ath0 setUnitTestCmd 0x47 6 264 0 3 0 0 0

# wifistats wifi0 12 | grep ac_su
ac_su_ndpa_queued = 1
ac_su_ndpa_tried = 1
ac_su_ndp_queued = 1
ac_su_ndp_tried = 1
ac_su_ndp_err = 0
ac_su_ndp_flushed = 0
ac_su_ndpa_err = 0
```

ac_su_ndpa_flushed = 0

4.26 Disable probing in VHT20, MCS 9 rate in NSS 1 and 2 after a certain number of failures

With the following commands, the feature can be enabled/disabled/configured in all radios since our global structure is not per pdev, regardless of whatever values they give for X.

This command will help to configure disabling MCS after consecutive probe failures

wifitool athX setUnitTestCmd 10 7 28 <enable/disable> <Preamble> <bw> <mcs> <nss> <failure count>

athX - replace X with 0,1,2 depending upon the radio(eg : 1 for 2G)

<enable/disable> - 0: Disable, 1/non-zero: Enable

<Preamble> - 2 - HT, 3 - VHT, 4 - HE

<bw> - 0 - 20MHz, 1 - 40MHz, 2- 80MHz, 3 - 160MHz

<mcs> - mcs threshold for consecutive failures

<nss> - nss index value (not the index)

<failure count> - consecutive failure count

"This command will help to enable default invalid rates. (ie) VHT20 MCS9 and above in NSS1 and NSS2

wifitool athX setUnitTestCmd 10 2 29 <enable/disable>

athX - replace X with 0,1,2 depending upon the radio(eg : 1 for 2G)

<enable/disable> - 0: Disable, 1/non-zero: Enable

4.27 Command to configure TPE

This command is for LPI devices. Customers might be interested in having multiple TPE elements to able to determine different cell sizes.

```
wifitool athx add_tpe <tx_pwr_intrpt>  <tx_pwr_cnt> < tx_pwr_cat> <tx_pwr
…..>
```

If ‘tx_pwr_intrpt’ is 0 or 2 then ‘tx_pwr’ is expected in ‘EIRP’ format. If 1 or 3, then ‘PSD’ format

- Validate ‘tx_pwr’ against values sent by target in REG_CC
- ‘tx_pwr_intrpt’ > 3 will be ignored
- If ‘tx_pwr_intrpt’ is 0 or 2, then ‘tx_pwr_cnt’ should be in the range of <0, 3>.
- If ‘tx_pwr_intrpt’ is 1 or 3, then ‘tx_pwr_cnt’ should be in the range of <0, 4>.
- ‘tx_pwr_cat’ can either be 0 or 1 only
- ‘tx_pwr_val’ should be absolute Tx Power value. To set 5dbm/MHz, user should input 10.

```
wifitool athX del_tpe <intrpt> <cat>
```

- The command deletes the TPE IE corresponding to the interpretation and category values passed in command.
- This command deletes any TPE IE added by the user using the add_tpe command.
- This command displays an error if the TPE IE corresponding to the interpretation and category arguments passed to the command is not present.
- For MBSSID/EMA AP config, the `add_tpe` and `del_tpe` commands will be allowed only for Tx VAP.

4.28 Command for peer latency configuration

wifitool athX set_set_peer_latency_param_config <peer_mac> <tid> <dl/ul> <service interval> <burst size> <add/sub>

<dl/ul> = 1 for dl , 2 for ul

<add/sub> = 1 for addition, 2 for subtraction

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

5 wlanconfig utility commands

The Qualcomm Technologies **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 might cause undesired effects, not all such impacts might have been documented. Consider the nature of multiple VAP configurations that use multiple radios, and use caution while changing parameters.

5.1 Create a VAP

For more information on usage guidelines and the functionality regarding these parameters, see the *WLAN AP modes* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

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

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

Where:

Argument	Description	
ATH*[N]*	VAP name. If the number at the end of the name is omitted, the system will automatically use the next available interface number. The VAP name ATH is not required, any text string will do. Note that when the index is occupied by another VAP, **create VAP** will fail.	
create	Create action	
wlandev wifi*N*	Indicates to which interface the VAP will attach. The interface number is required for this argument. For dual concurrent operations, *N* indicates which radio to attach the VAP to.	
wlanmode *mode*	Indicates the mode to open the VAP into. The valid modes are:	
	ap	AP (infrastructure) mode
	sta	STA (client) mode
	wrap	AP mode to be used under special repeater mode called QWRAP

Argument	Description
bssid *-bssid*	Optional parameter indicating that the MAC address should be cloned from the first VAP for this interface. Not normally specified. **NOTE** bssid is not supported by wlanconfig, but is supported by the Qualcomm driver.
nosbeacon	Indicates that no beacons will be transmitted from this VAP. Used as part of STA mode.
mataddr	Original mac of wired/wireless station connected on QWRAP AP
wlanaddr	Virtual/changed mac address to be used for proxy sta with respect to wired/wireless clients in QWRAP mode
vapid N	Indicates the Vap id based on which MAC address will be generated. N ranges from 0 to 15.

5.2 List VAP parameters

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

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

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

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

NOTE In the output of the *wlanconfig athx list* command, the value displayed in the Supported Operating Classes field is per tag length in the frame. Based on tag length, the entire data is parsed and displayed in the Supported Operating Classes field of the output and it is the behavior of STA that determines whether the current operating class is included in operating classes list.

NOTE The values of the 'Minimum Tx Power' and Maximum Tx Power' fields are not available from clients for 2.4 GHz in the output of the wlanconfig athX list sta command.

5.2.1 AP list elements

Table 5-1 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
```

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT xanwafeng@zte.com.cn

```
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 5-1 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 Vendor IE	**RSN**	aRSN IE
	WME	WMM IE	**VEN**	Vendor-Specific IE	**???**	Unknown IE

5.2.2 STA list Elements

Table 5-2 describes the list elements for each STA associated with the indicated VAP. This listing is produced:

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

NOTE The data for the ACAPS element data is no longer reported. In the example output above, the data 0, 33, Q, and WME correspond to ERP, STATE, HTCAPS, and (no header) elements listed in Table 5-2.

Table 5-2 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

Table 5-2 STA list elements (cont.)

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

Associated Client STA Capabilities and Statistics for parameter APIs in carrier networks

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Support for reporting associated and steering stations for carrier networks* section of the *Wi-Fi SON: Additional features* chapter in the Wireless LAN Access Point (Driver Version 11.0) Programming Guide (80-YA728-6).

- ieee80211_sta_info structure is enhanced with required fields and wlanconfig is enhanced to display these details.

Name	High Level Implementation Details	Usage Details
Device.WiFi.AccessPoint.{i}.AssociatedDevice.{i}.		
X_<OEM>_COM_ExtCapabilities	ni structure is enhanced with all extended capabilities and enhanced wlanconfig to display extended capabilities.	wlanconfig athX list sta (field: XCAPS)
X_<OEM>_COM_PowerCapability	Enhanced in host driver to parse a power capabilities and enhanced ni and isi structures to support enhancement of wlanconfig.	wlanconfig athX list sta (Minimum Tx Power, Maximum Tx Power)
X_ARRIS_COM_OperatingClasses	isi structure enhanced to save current operating class and supported operating classes for ie received from sta.	wlanconfig athX list sta (Current Operating class)
X_<OEM>_COM_SupportedOperatingClasses	isi structure enhanced to save current operating class and supported operating classes for ie received from sta.	wlanconfig athX list sta (Supported Operating classes)
X_<OEM>_COM_HighestTxMCSRate	Enhanced wlanconfig to display value in isi_tx_ mcs_rate	wlanconfig athX list sta (Highest Tx MCS Rate)
X_<OEM>_COM_HTSupport	Enhanced wlanconfig to display ht support status	wlanconfig athX list sta (HT Capability)
X_<OEM>_COM_VHTSupport	Enhanced wlanconfig to display vht support status	wlanconfig athX list sta (VHT Capability)
X_<OEM>_COM_NumTxSS	Already existing	wlanconfig athX list sta (TXNSS) In 'wlanconfig athX list sta' command, "TXNSS" and "RXNSS" are supposed to mean maximum supported/capable NSS by station.
X_<OEM>_COM_NumRxSS	Already existing	wlanconfig athX list sta (RXNSS)
X_<OEM>_COM_MUCapable	Enhanced wlanconfig to display MU capability status	wlanconfig athX list sta (MU capable)

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

X_<OEM>_COM_SupportedChannels	Enhanced beacon parsing APIs to parse for supported channel ie and save back to info to ni and sta_info structure, further wlanconfig is enhanced to display info in sta_info	wlanconfig athX list sta (Channels supported)
X_<OEM>_COM_SupportedRates	Enhanced wlanconfig to display existing isi_rates values.	wlanconfig athX list sta (Supported Rates)
X_<OEM>_COM_ExtSupportedRates	Enhanced wlanconfig to display existing isi_rates values	wlanconfig athX list sta (Supported Rates)
Device.WiFi.AccessPoint.{i}.AssociatedDevice.{i}. X_<OEM>_COM_SteerStats.		
SNR	Enhanced wlanconfig to display SNR	wlanconfig athX list sta (SNR)
TrafficLevel	Already existing	wlanconfig athX list sta (TXRATE, RXRATE)

5.2.3 Channel list elements

Table 5-3 describes the channel list elements, listing available channels and frequencies followed by strings indicating specific VAP channel capabilities. This example lists channels with channel number and frequency in MHz:

```
# wlanconfig ath0 list chan
```

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

Table 5-3 Channel list elements

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

Table 5-3 Channel list elements

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

5.2.4 Capabilities list elements

Table 5-4 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 5-4 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		

5.2.5 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
```

5.2.6 Keys list elements

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

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

5.3 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
```

5.4 NAWDS configuration parameters

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *NAWDS configuration* section of the *Repeater access point functions* chapter in the Wireless LAN Access Point (Driver Version 11.0) Programming Guide (80-YA728-6).

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 5-5 as a separate entry.

NOTE If a NAWDS link breaks, it is necessary to add the peer again, although the peer entry is already available in the NAWDS table. For example, if the iwpriv ath0 vht_11ng 1 command is issued to enable 256QAM, the NAWDS link breaks. In this scenario, the entry must be added again to the table for NAWDS to work.

Table 5-5 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

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 5-5 Configure NAWDS parameters (cont.)

Parameter	Format	DA	OL	Description

defcaps	wlanconfig ath*N* nawds *defcaps caps*	Y	Y	0x0 — When a NAWDS AP is operating in learning mode, it must discover which capability the NAWDS AP peer has. In this situation, defcaps would be used. The CAPS is defined as follows:
				`#define NAWDS_REPEATER_CAP_DS 0x01`
				`#define NAWDS_REPEATER_CAP_TS 0x02`
				`#define NAWDS_REPEATER_CAP_4S 0x04`
				`#define NAWDS_REPEATER_CAP_HT20 0x0100`
				`#define NAWDS_REPEATER_CAP_HT2040 0x0200`
				`#define NAWDS_REPEATER_CAP_11ACVHT20 0x0400`
				`#define NAWDS_REPEATER_CAP_11ACVHT40 0x0800`
				`#define NAWDS_REPEATER_CAP_11ACVHT80 0x1000`
				`#define NAWDS_REPEATER_CAP_11ACVHT80_80 0x2000`
				`#define NAWDS_REPEATER_CAP_11ACVHT160 0x4000`
				If CAPS equals 0, the HT rate would be disabled. To enable NAWDS_REPEATER_CAP_DS, at least one of NAWDS_REPEATER_CAP_HT20 and NAWDS_REPEATER_CAP_HT2040 must be specified.
				Support will be added in a subsequent release for 5-8 streams (5 GHz <= 80 MHz), and 3-4 streams (5 GHz 160 MHz).
				The range of CAPS values are defined as follows:

NxN	HT20	HT40	VHT20	VHT40	VHT80	VHT80_80	VHT160
1x1	0x0100	0x0200	0x0400	0x0800	0x1000	0x2000	0x4000
2x2	0x0101	0x0201	0x0401	0x0801	0x01001	0x2004	0x4004
3x3	0x102	0x0202	0x0402	0x0802	0x01002	N/A	N/A
4x4	0x0102	0x0204	0x0404	0x0804	0x1004	N/A	N/A

Parameter	Format	DA	OL	Description
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.

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT nan.yafeng@zte.com.cn

Table 5-5 Configure NAWDS parameters (cont.)

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

5.4.1 Configuration examples

Static bridge and peer node supports HT20 rates

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

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

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

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

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

Static bridge and peer node supports VHT rates

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

Iwpriv ath0 wds 1
wlanconfig ath0 nawds mode 2
wlanconfig ath0 nawds add-repeater 00:03:7f:xx:xx:xx:xx 0x1002
```

VHT Example Rates: 3x3

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

VHT Example Rates: 2x2

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

VHT Example Rates: 1x1

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

5.5 HMWDS/HMMC commands

For more information on usage guidelines and the functionality regarding these parameters, see the *Hy-Fi WLAN* section of the *WLAN AP modes* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 5-6 Configure HMWDS/HMMC parameters

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

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan yafeng@zte.com.cn

5.6 ATF configuration commands

For more information on usage guidelines and the functionality regarding these parameters, see the *Airtime fairness* section of the *Memory and bandwidth optimization features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 5-7 Configure/show ATF parameters

Parameter	Format	D	O	Description
addssid	wlanconfig athX addssid *<ssid name>* <airtime percentage>	Y	Y	Assign percentage of airtime to the SSID. The airtime percentage value range is 0~100. Example: `#wlanconfig ath0 addssid BEE0 12`
delssid	wlanconfig athX delssid <ssid name>	Y	Y	Delete the SSID assigned. Example: `# wlanconfig ath0 delssid BEE0`

addsta	wlanconfig athX addsta
	<sta mac addr>
	<airtime percentage>
		Y	Y	Assign percentage of airtime to the STA.
				The airtime percentage value range is 0~100.
				Example:
				`#wlanconfig ath0 addsta 220011abef6660`
				Enter the wlanconfig athX addsta mac per command to assign percentage of airtime globally.
				Enter the wlanconfig athX addsta mac per [ssid1] command to assign percentage of airtime per SSID.
				Station-based configuration is not supported globally for ATF. Configuring a STA must have an SSID associated with it. Therefore, specify the SSID associated with the STA while assigning ATF percentage for the STA using the wlanconfig athX addsta command. The syntax of the command is:
				`wlanconfig athX addsta <mac addr> <airtime configured> <ssid to which it is applicable if associated>`
				First, configure the airtime for a VAP before configuring a airtime for a station (STA). The correct command sequence to assign airtime to a VAP and STA is as follows:
				1. Assign airtime to VAP
				`wlanconfig ath0 addssid vap1 100 // Assigns 100% airtime to VAP1`
				2. Assign airtime to STA
				`wlanconfig ath0 addsta 000102030405 10 vap1 //Assign 10% airtime to specified MAC when associated to VAP1`

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 5-7 Configure/show ATF parameters (cont.)

delsta	wlanconfig athX delsta <sta mac addr>	Y	Y	Delete the STA assigned. Example: `# wlanconfig ath0 delsta 220011abef66` Enter the wlanconfig athX delsta mac command to delete percentage of airtime globally.
showatftable	wlanconfig athX showatftable	Y	Y	Displays the ATF table. The SSIDs and STAs part of the ATF table will be listed Example: `#wlanconfig ath0 showatftable` Enter the *wlanconfig athx showatftable [index:-to show per_peer_table]* command to display the ATF percentage configured for a particular peer in a particular SSID. An index value of 0 is a global percentage. Indexes starting from 1 are for corresponding VAP IDs. In case of SSID and peer-based configuration, the showatftable command output continues to show the details as done currently. In case of AC based configuration, showatftable continues to show the details of the configured clients correctly. How-ever the unconfigured client airtime may not be the same as that shown in the showatftable, since the airtime allocation now happens in subgroup level. The actual allocation is shown in 'showatfsubgroup' output.
showairtime	wlanconfig athX showairtime	Y	Y	Lists all STAs and percentage of ATF. Example: `#wlanconfig ath0 showairtime` Displays STAs added in the ATF table. The airtime value shown is in terms of 1000.
flushatftable	wlanconfig athX flushatftable	Y	Y	This flushes all the configurations and data present in the atf table. Example `#wlanconfig ath0 flushatftable` When this command is issued, a reset of BSS occurs and the stations that were previously connected are terminated and reconnected.
addatfgroup	wlanconfig athX addatfgroup <group name> <ssid name>	Y	N	Creates a new atfgroup, if the group doesn't exist already and adds the SSID to the group created. If the group already exists, the SSID will be added to the group. Example: `#wlanconfig ath0 addatfgroup group1 myvap`

Table 5-7 Configure/show ATF parameters (cont.)

configatfgroup	wlanconfig athX configatfgroup <group name> <airtime percentage>	Y	N	Assigns/Configures airtime percentage to the group. Example: `#wlanconfig ath0 configatfgroup group1 80`
delatfgroup	wlanconfig athX delatfgroup <group name>	Y	N	Deletes a group. Example: `#wlanconfig ath0 delatfgroup group1`
showatfgroup	wlanconfig athX showatfgroup	Y	N	Displays all groups configured, SSIDs for each group and the airtime percentage assigned to each group Example: `#wlanconfig ath0 showatfgroup`

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

addtputsta	wlanconfig athX addtputsta ffffffffffff <tput in kbps (ignored)> <reserved airtime percentage>	Y	N	To configure reserved airtime as 5% Example: `#wlanconfig ath0 addtputsta ffffffffffff 25000 5`
deltputsta	wlanconfig athX addtputsta ffffffffffff <tput in kbps (ignored)> <reserved airtime percentage>	Y	N	To remove throughput configuration for a client Example: `#wlanconfig ath0 deltputsta 220011abef66`
deltputsta	wlanconfig athX deltputsta ffffffffffff	Y	N	To remove throughput configuration for all clients Example: `#wlanconfig ath0 deltputsta ffffffffffff`
showtputtbl	wlanconfig athX showtputtbl	Y	N	To show throughput configuration of all clients and the reserved airtime Example: `#wlanconfig ath0 showtputtbl`
atfaddac	wlanconfig <VAP> atfaddac <ssid/group> <ac>:<airtime> <ac>:<airtime>	Y	N	To add an access category (AC)-based ATF configuration Example: `#wlanconfig ath0 atfaddac ssid1 BE:30 BK:20 VI:10 VO:15`
atfdelac	wlanconfig <VAP> atfdelAC <ssid/group> <ac>:<airtime> <ac>:<airtime>	Y	N	To delete an AC-based ATF configuration Example: `#wlanconfig ath0 atfdelAC ssid1 BE:30 BK:20 VI:10 VO:15`

Table 5-7 Configure/show ATF parameters (cont.)

atfgroupsched	wlanconfig <VAP> atfgroupsched <group name> <sched policy>	Y	N	To configure per-group ATF scheduling policy for access category. Scheduling policy: 0/1/2 (0 - Fair 1- Strict 2- FAIR UB or restricted fair mode) Example: `wlanconfig ath0 atfgroupsched group1 2 // configure 'FAIR with UB' sched policy for group1` This command is applicable only when grouping is enabled. The group scheduling policy that is configured is displayed in the output of the wlanconfig ath0 showatfgroup command.
showatfsubgroup	wlanconfig athX showatfsubgroup	Y	N	To display the groups and subgroups for access category (AC)-based ATF. Example: `#wlanconfig ath0 showatfsubgroup` A group/VDEV is configured for restricted fair scheduling with ATF units allocated, where the nodes/peers within the group/VDEV are scheduled in fair mode. However, the cumulative airtime of the group/VDEV does not go beyond the configured ATF

5.6.1 ATF statistics commands

1. wlanconfig athX showatfstats

   Shows ATF stats per SSID or group along with list of associated peers. For group it shows entire list of configured groups and their corresponding peers. For SSID configurations, it shows the SSID corresponds to the VAP passed in command and that VAP-associated peers.

```
root@OpenWrt:/# wlanconfig ath21 showatfstats
                      SHOW  ATF STATS
   Total Airtime for Radio in us              171245
  SSID    Client(MAC Address)       Alloted Airtime(%)  Actual Airtime(%)
Borrowed(%)  Unused(%)  Assoc_Status(1--Assoc,0-No-Assoc)
  SKM_ATF2                                40.0          27.5          0
12.5

            00:03:7f:12:67:67      20.0          9.5       0          10.5
1

            e0:9d:31:22:34:1c      20.0          17.5      0          2.5
1

  Others                                  0             0.0           0
0
```

2. wlanconfig athX showatfacstats

   Shows ATF stats based on AC configurations with respect to each group.

```
root@OpenWrt:/# wlanconfig ath21 showatfacstats
                  SHOW  ATF AC STATS
```

```
Group/SSID      Group Airtime    AC      Alloted Airtime     Actual Airtime
Borrowed   Unused
SKM_ATF1           60.0
                                       BE          6.0               40.0
34.0         0
                                       BK          6.0               0.0
0           6.0
                                      VI          6.0               0.0
0           6.0
                                     VO          12.0               0.0
0           12.0
SKM_ATF2           40.0
                                      BE          2.0               59.3
57.3         0
                                 BK         4.0            0.0          0
4.0
                                 VI         6.0            0.0          0
6.0
                                VO         10.0            0.0          0
10.0
```

5.7 Using wlanconfig commands for cfg80211 mode: examples

For cfg80211 based drivers, no generic applications such as iwpriv or iwlist are present to configure private commands. Therefore Qualcomm Technologies developed an nl80211-based cfg80211tool to configure private commands. Input to the cfg80211tool is aligned with existing iwpriv commands; for example, command form and the number of arguments are as same as existing iwpriv commands.

wlanconfig

Add the new switch **–cfg80211** to support cfg80211-based drivers. ddition of the **-cfg80211** switch is optional. If a certain proprietary wireless tool command, such as wlanconfig, is entered without the **-cfg80211** switch, the driver checks the mode that is being currently used—WEXT or cfg80211—on the system, and processes the command accordingly.

For example, for the WEXT command of `wlanconfig ath0 nawds mode 3`, the equivalent cfg80211 command is `wlanconfig ath0 nawds mode 3 [-cfg80211]`.

5.7.1 Create and delete VAP using cfg80211tool

For VAP creation and deletion, use generic tool, *iw*, by performing the following procedure:

For VAP creation:

```
wlanconfig athN create wlandev wifiN wlanmode <mode> -cfg80211
```

```
iw phy phyN interface add <athN>  type  <__ap/managed/…>
```

VAP bringup: (only required for __ap mode)

To bring up VAP hostpad, run with configuration (that is, ssid, mode, channel)

The following is a sample hostap configuration:

```
driver=nl80211
interface=ath0
logger_syslog=127
logger_syslog_level=2
logger_stdout=127
logger_stdout_level=2
channel=36
ieee80211ac=1
ieee80211n=1
vht_oper_chwidth=0
ht_capab=[HT40+]
hw_mode=a
dtim_period=1
ignore_broadcast_ssid=0
ctrl_interface=/var/run/hostapd-wifi0
auth_algs=1
wpa=0
ssid=qca_cfg80211
bridge=br-lan
```

Run hostapd with configuration file: hostapd -dddd hostap-ath0.conf

For VAP destroy:

```
iw dev <athN>  del
```

For all configurations other than the configurations to create a VAP and delete a VAP, use the switch, **–cfg80211**, with the **wlanconfig** command to support cfg80211-based drivers.

5.8 Off-channel Tx/Rx without active WLAN

For more information on usage guidelines and the functionality regarding these parameters, see the *Off-channel Tx/Rx without active WLAN* section of the *Beacons and frames transmission* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

The capability for off-channel Tx/Rx without active WLAN is supported on IPQ807x/IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx/QCN90xx or QCN602x chipsets. To validate the off-channel Tx/Rx without active VLAN capability, do the following:

- Create AP VAP using the iw command (cfg80211mode) / wlanconfig commands (for WEXT).

  Commands for cfg80211:

  - wlanconfig ath0 create wlandev wifi0 wlanmode ap -cfg80211
  - iw dev wifi0 interface add ath0 type __ap

  Commands for WEXT:

  - wlanconfig ath0 create wlandev wifi0 wlanmode ap

- Send off-channel packets using wifitool (do not bring up the VAP so that an active WLAN is not present).

  Commands:

  wifitool ath0 offchan_tx_test 44 70

5.9 RSSI scan of NACs parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Scanning RSSI of NACs* section of the *Core channel selection capabilities* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

No new dedicated VAP is used for the NAC RSSI obtaining because VAP numbers are limited in certain customer deployments. The BSSID, MAC address of the client and channel are configured using CLI commands. If the configured channel is different from the working home channel on AP, an off-channel scan is performed to sniff the frames from the client to its associated AP. A wlanconfig CLI command to perform the scanning of RSSI of NACs by capturing the frames between the client and its associated. Also, a wlanconfig command to show the RSSI of the NAC is implemented.

If the channel that is specified as the input parameter is the home channel, the BSSID and MAC address of the client are set in to firmware (FW) to receive the frames from the client to the configured BSSID. The RSSI is recorded in FW. If the channel is not equal to the home channel, off-channel scan is implemented after setting BSSID and MAC address of NAC to FW.

No special VAP is dedicated to perform the scanning of NAC RSSI.

WMI_REQUEST_STATS_CMDID is leveraged to retrieve the NAC RSSI statistics from FW. NAC RSSI statistics information is integrated in the event WMI_UPDATE_STATS_EVENTID. New statistic type WMI_REQUEST_NAC_RSSI_STAT is added to represent the NAC RSSI from FW. On the host side, the NAC RSSI information is stored in vap->iv_nac_rssi for the VAP. If no traffic is present from the client to its associated AP (such as the clients disassociated, clients left, and power-save values), the average RSSI is not updated.

While receiving WMI command WMI_VDEV_SET_SCAN_NAC_RSSI_CMDID from host, FW configures the BSSID to filter register ((RXPCU_BSSID3_L32, RXPCU_BSSID3_U16 and RXPCU_RX_FILTER2, PROXY_STA_AD1_SEARCH_BSSID) to filter the frames and then the unicast frames targeted to the BSSID will be received in HW level. Client's MAC address is recorded for FW to retrieve the RSSI in SW level. Only the frames whose source MAC address matching the stored MAC address are used to retrieve the RSSI from their descriptors. The Avg RSSI is calculated with same method of associated clients and it's recorded in FW.

Switching of modes does not occur. If two APs work in different modes, the AP might not obtain the correct RSSI of NA.

Smart monitor uses the following configuration for scanning of RSSI for NACs:

```
wlanconfig athX nac add bssid <bssid address>
wlanconfig athX nac add client <client address>
wlanconfig athX nac list client
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

NOTE: There are two sets of commands—one is wlanconfig athX nac_rssi and the other is wlanconfig athX nac. When wlanconfig athX nac_rssi command is used to configure scanning of RSSI for NACs, use wlanconfig athX rssi_nac show_rssi command to display the scanned RSSI. When wlanconfig athX nac command is used to configure scanning of RSSI for NACs, smart monitor VAP must be created, at first, before using this command.

Table 5-1 RSSI scan of NACs parameters

Parameter	Format	Description
nac add/del	wlanconfig athX rssi_nac add/del bssid xx:xx:xx:xx:xx:xx client xx:xx:xx:xx:xx:xx channel value	Perform the scanning of RSSI of NACs. If the channel that is specified as the input parameter is the home channel, the BSSID and MAC address of the client are set in to firmware (FW) to receive the frames from the client to the configured BSSID. The RSSI is recorded in FW. If the channel is not equal to the home channel, off-channel scan is implemented after setting BSSID and MAC address of NAC to FW. If the channel that the client is working is not the home channel, an off-channel scan is performed. No special VAP is dedicated to perform the scanning of NAC RSSI. ■ If same BSSID and client MAC are specified when the **add** parameter is set with the wlanconfig command, and if the channel is different, only the channel value is changed if the channel is different from home channel. ■ The wlanconfig athX rssi_nac add command can be entered multiple times with same BSSID and client MAC continuously. When this command is entered, the first time, BSSID and client MAC are set in the FW. For all the subsequent entries of this command, only a change in channel is performed, if needed. However, if the BSSID or client MAC are different from the previous settings, an error is returned. ■ If the BSSID and client MAC are not specified when **del** parameter is set with the wlanconfig command, an error message is displayed.
nac list client	wlanconfig athX rssi_nac list client	Display the scanned RSSIs of NACs.

5.10 Configure the listing HW capabilities for parameter APIs in carrier networks

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Support for reporting associated and steering stations for carrier networks* section of the *Wi-Fi SON: Additional features* chapter in the Wireless LAN Access Point (Driver Version 11.0) Programming Guide (80-YA728-6).

Starting with QCA_Networking_2016.SPF.6.0, support is implemented for generating reports of associated and non-associated stations, and steered and non-steered stations, which are beneficial

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

for collecting statistics and computing usage of subscriber services to aid troubleshooting and diagnosis. Users can enable or disable the collection of parameter APIs, which refer to the associated, non-associated, steered, and non-steered stations using the iwpriv wifiX set_papi <0 | 1> command for the WLAN driver to transmit or not transmit the report to user space.

The wlanconfig application is enhanced as follows:

- Added new API to fetch hardware capabilities using ioctl IEEE80211_PARAM_MODE.
- Added new API list_hw_capabilities to parse hardware capabilities and frame a string.

Name	Usage Details
Device.WiFi.Radio.{i}.	
SupportedStandards	wlanconfig athX list hwcaps

5.11 Start or stop channel frequency response dump capture per peer for IPQ401x, IPQ807xA, IPQ60xx, and QCN90xx /QCN602x

For more information on usage guidelines and the functionality regarding these parameters, see the *Generate channel frequency response dump for IPQ4019 platforms, Generate channel frequency response dump for IPQ807xA platforms*, and *Generate channel frequency response dump for IPQ60xx and QCN90xx/QCN602x platforms* sections of the *Core channel selection capabilities* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

At any point of time, either one-shot (per-peer) channel frequency response (CFR) capture or periodic CFR capture is taken. Both per-peer capture and periodic capture cannot be enabled simultaneously. An error is returned that ioctl failed if per-peer capture is requested when periodic CFR capture is enabled.

Start per-peer CFR capture

Command to start capture: *wlanconfig athX cfr start <peer_mac_addr> <bandwidth> <periodicity> <capture_type>*

Start capture needs four arguments, namely, MAC address of peer, bandwidth of capture, periodicity and capture type. Bandwidth of capture cannot be greater than the channel/connection bandwidth. Periodicity must be entered in multiples of 10 milliseconds due to FW timer logic. For a single-shot capture, period must be configured as 0. Capture type decides the method to capture CFR data. Currently, only one method is supported, which is QoS NULL frame ACK method. Global cfr_timer must be enabled if periodic capture must be used. Enable timer using *cfg80211tool wifiX cfr_timer 0/1*.

Stop per-peer CFR capture

Command to stop capture: *wlanconfig ath0 cfr stop <peer_mac_addr>*

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Stop capture needs only one argument, the MAC address of the peer, whose capture must be stopped.

5.11.0.1 User application

cfrtool is the test utility for CFR capture testing. Customers can modify or customize this cfrtool for their requirements. For example, instead of logging the cfrcapture info to file, redirect/send it to a cloud server for live capture.

Issuing the `root # "cfrtool capture"` command starts the application and opens the socket with netlink CFR family. Any packet that is sent from driver for this family is logged to **/tmp/cfr_wifiX file**. The maximum size of this log file is ~8 MB with wraparound (new data will overwrite the old data). If CFR capture is not enabled for a particular radio, its corresponding file will be empty. If any known unexpected error occurs during data capture, the application will close the socket and exit with a debug and error code. If the application to capture CFR data is started and CFR capture (one-shot or periodic) is not enabled, application will continue to listen at socket and keep waiting with periodic debug messages indicating its state. Customer can parse the **/tmp/cfr_wifiX file** according to the prescribed format to obtain the metadata and also verify if the metadata is valid before processing CFR dump.

CFR is only enabled for OFDM rates. By default, in 2.4 GHz band, management frames go in CCK rates. Enable any of the OFDM rate (`acfg_tool acfg_set_op_support_rates wifiX 140`) or disable the CCK rates (i`wpriv athX dis_legacy 0x1f`) and then trigger the CFR for unassociated client in 2.4 GHz band.

Perform the following:

1. Configure the AP
2. Bring up a STA in the environment and note down the MAC address (STA should be up and sending probe requests to AP.)
3. Disable CCK rates on the AP.
4. Enable Per-peer one-shot CFR capture for the unassociated client (`wlanconfig athX cfr start <mac address of STA> 0 0 2`)
5. Parse the CFR dump collected using the python script and validate all the fields.

5.12 Configure Rx flow identification based on flow-type using user space commands

For more information on usage guidelines and the functionality regarding these parameters, see the *Rx flow identification based on flow type* section of the *WLAN AP driver operations* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

5.12.0.1 INI configuration

Four ini parameters will be configured in qcawificfg80211.sh as follows:

- update_internal_ini QCA8074V2_i.ini dp_rx_flow_tag_enable 1

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

- update_internal_ini QCA8074V2_i.ini dp_rx_flow_search_table_size 16384
- update_internal_ini QCA8074V2_i.ini dp_rx_per_pdev_flow_search 0
- update_internal_ini QCA8074V2_i.ini dp_rx_monitor_protocol_flow_tag_enable 1

5.12.0.2 wlanconfig commands

Add a flow

Usage: `wlanconfig <interface> rx_flow_tag_op 0 <ipver# 0:IPv4, 1:IPv6> <source_ip <dest_ip> <source_port> <dest_port> <protocol#: 0:TCP 1:UDP> <flow_metadata>`

NOTE All arguments are mandatory.

Example:

```
wlanconfig wlanX rx_flow_tag_op 0 0 192.168.1.2 192.168.1.1 15000
15001 1 1010
```

```
wlanconfig wifi0 rx_flow_tag_op 0 1 fe80::b5c5:e011:5b8f:8b9a
fe80::28d3:5eff:fed9:2ff8 17000 7000 0 3333
```

Delete a flow

Usage: `wlanconfig <interface> rx_flow_tag_op 1 <ipver# 0:IPv4, 1:IPv6> <source_ip <dest_ip> <source_port> <dest_port> <protocol#: 0:TCP 1:UDP>`

Example: `wlanconfig wlanX rx_flow_tag_op 1 0 192.168.1.2 192.168.1.1 15000 15001 1 1010`

Obtain statistics for a flow

Usage: `wlanconfig <interface> rx_flow_tag_op 2 <ipver# 0:IPv4, 1:IPv6>  <source_ip <dest_ip> <source_port> <dest_port> <protocol#: 0:TCP 1:UDP>`

Example: `wlanconfig wlanX rx_flow_tag_op 2 0 192.168.1.2 192.168.1.1 15000 15001 1`

5.13 Create DISA interface

Starting with QCA_Networking_2019.SPF.11.0 release, this functionality to perform CCM encryption and decryption is implemented for IPQ807xA platforms.

For defense information systems agency (DISA) certification, crypto engine must be tested to ensure that the crypto engine does AES encryption as expected (with the assumption that key, PN,

plain text and header the encrypted text must be the be same on any crypto engine - FIPS compliance). This avoids mangling of the encrypted text by the crypto engine vendors. Therefore, to validate it, an infrastructure is formulated to take plain text, assemble the necessary elements and pass that through crypto engine (HW block), encrypt it, and obtain the encrypted data. For decryption, the encrypted data is provided and plain text is retrieved.

Debug log is added on the code flow, which can be observed on the console in the console with cnss_diag (cnss_diag -c -p | grep -i disa &). The infrastructure is available to log the received header, plain data, PN, seq no, keyid octet, encrypted data, CRC in dword hex format.

By default, the wifi interface is brought up and DISA interface is created using the following commands.

```
wlanconfig disa0 create wlandev wifi0 wlanmode ap -cfg80211
iw dev wifi0 interface add disa0 type __ap
wifitool disa0 fips input.txt
```

where the input.txt is of the format:

<CommandID> <Mode> <KeyLength> <PNLength> <HeaderLength> <PayloadLength> <ExpOutputLength> <KeyIdx> <CipherSuite> <Key> <PN> <Header> <Payload> <ExpOT> <newline>

Example:

```
root@OpenWrt:/# cat input.txt
0 2 16 6 32 20 64  0 4 c97c1f67ce371185514a8a19f2bdd52f b5039776e70c 0848c32c0fd
2e128a57c5030f1844408abaea5b8fcba80330000000000000000 f8ba1a55d02f85ae967bb62fb6
cda8eb7e78a050 0848c32c0fd2e128a57c5030f1844408abaea5b8fcba80330ce70020769703b5f
3d0a2fe9a3dbf2342a643e43246e80c3c04d0197845ce0b16f976231d99f066
root@OpenWrt:/#
```

5.14 wlanconfig changes for 6 GHz

The following command is added:

`wlanconfig ath0 list band`: Lists frequency bands supported on underlying radio:

The following is a sample output:

```
root@OpenWrt:/# wlanconfig ath0 list band
Supported Frequency Bands:      2       5 GHz
root@OpenWrt:/# wlanconfig ath1 list band
Supported Frequency Bands:      1       2 GHz
root@OpenWrt:/# wlanconfig ath2 list band
Supported Frequency Bands:      1       2 GHz   2       5 GHz
```

The following command is modified:

```
wlanconfig ath0 list chan
```

Output format is changed as follows to display an extra field “(X GHz)” added just after channel number and flag “P” for PSC channels. The rest of the output is similar to the existing behavior.

The following output taken from 5 GHz + 6 GHz enabled radio:

```
OpenWrt:/# ./ulanconfig ath0 list chan
3.624444] ulan: [9148:E:ANY] ulan_cfg80211_list_chan: 7992: channel->ic_nchans = 84
3.624489] ulan: [9148:E:ANY] ulan_cfg80211_list_chan: 7995: ret = 0
3.631535] ulan: [9148:E:ANY] ulan_cfg80211_list_chan160: 8089: channel->ic_nchans = 84
3.637715] ulan: [9148:E:ANY] ulan_cfg80211_list_chan160: 8093: ret = 0
: 84, chans_160: 84, achans_6g: 57
el  36(5 GHz) : 5180     Mhz 11na C CU V VU V80- 42 V160- 50 H HU HL H80- 42 H160- 50   Channel  61(6 GHz) : 6245     Mhz H HU HL H80- 55 H160- 47
el  40(5 GHz) : 5200     Mhz 11na C CL V VL V80- 42 V160- 50 H HU HL H80- 42 H160- 50   Channel  65(6 GHz) : 6265   P Mhz H HU HL H80- 71 H160- 79
el  44(5 GHz) : 5220     Mhz 11na C CU V VU V80- 42 V160- 50 H HU HL H80- 42 H160- 50   Channel  69(6 GHz) : 6285     Mhz H HU HL H80- 71 H160- 79
el  48(5 GHz) : 5240     Mhz 11na C CL V VL V80- 42 V160- 50 H HU HL H80- 42 H160- 50   Channel  73(6 GHz) : 6305     Mhz H HU HL H80- 71 H160- 79
el  52(5 GHz) : 5260 *   Mhz 11na C CU V VU V80- 58 V160- 50 H HU HL H80- 58 H160- 50   Channel  77(6 GHz) : 6325     Mhz H HU HL H80- 71 H160- 79
el  56(5 GHz) : 5280 *   Mhz 11na C CL V VL V80- 58 V160- 50 H HU HL H80- 58 H160- 50   Channel  81(6 GHz) : 6345   P Mhz H HU HL H80- 87 H160- 79
el  60(5 GHz) : 5300 *   Mhz 11na C CU V VU V80- 58 V160- 50 H HU HL H80- 58 H160- 50   Channel  85(6 GHz) : 6365     Mhz H HU HL H80- 87 H160- 79
el  64(5 GHz) : 5320 *   Mhz 11na C CL V VL V80- 58 V160- 50 H HU HL H80- 58 H160- 50   Channel  89(6 GHz) : 6385     Mhz H HU HL H80- 87 H160- 79
el 100(5 GHz) : 5500 *   Mhz 11na C CU V VU V80-106 V160-114 H HU HL H80-106 H160-114   Channel  93(6 GHz) : 6405     Mhz H HU HL H80- 87 H160- 79
el 104(5 GHz) : 5520 *   Mhz 11na C CL V VL V80-106 V160-114 H HU HL H80-106 H160-114   Channel 101(6 GHz) : 6445     Mhz H
el 108(5 GHz) : 5540 *   Mhz 11na C CU V VU V80-106 V160-114 H HU HL H80-106 H160-114   Channel 105(6 GHz) : 6465     Mhz H HU HL
el 112(5 GHz) : 5560 *   Mhz 11na C CL V VL V80-106 V160-114 H HU HL H80-106 H160-114   Channel 109(6 GHz) : 6485     Mhz H HU HL
el 116(5 GHz) : 5580 *   Mhz 11na C CU V VU V80-122 V160-114 H HU HL H80-122 H160-114   Channel 113(6 GHz) : 6505   P Mhz H
el 120(5 GHz) : 5600 *   Mhz 11na C CL V VL V80-122 V160-114 H HU HL H80-122 H160-114   Channel 121(6 GHz) : 6545     Mhz H HU HL
el 124(5 GHz) : 5620 *   Mhz 11na C CU V VU V80-122 V160-114 H HU HL H80-122 H160-114   Channel 125(6 GHz) : 6565     Mhz H HU HL
el 128(5 GHz) : 5640 *   Mhz 11na C CL V VL V80-122 V160-114 H HU HL H80-122 H160-114   Channel 129(6 GHz) : 6585   P Mhz H HU HL H80-135 H160-143
el 132(5 GHz) : 5660 *   Mhz 11na C CU V VU V80-138 H HU HL H80-138                     Channel 133(6 GHz) : 6605     Mhz H HU HL H80-135 H160-143
el 136(5 GHz) : 5680 *   Mhz 11na C CL V VL V80-138 H HU HL H80-138                     Channel 137(6 GHz) : 6625     Mhz H HU HL H80-135 H160-143
el 140(5 GHz) : 5700 *   Mhz 11na C CU V VU V80-138 H HU HL H80-138                     Channel 141(6 GHz) : 6645     Mhz H HU HL H80-135 H160-143
el 144(5 GHz) : 5720 *   Mhz 11na C CL V VL V80-138 H HU HL H80-138                     Channel 145(6 GHz) : 6665   P Mhz H HU HL H80-151 H160-143
el 149(5 GHz) : 5745     Mhz 11na C CU V VU V80-155 H HU HL H80-155                     Channel 149(6 GHz) : 6685     Mhz H HU HL H80-151 H160-143
el 153(5 GHz) : 5765     Mhz 11na C CL V VL V80-155 H HU HL H80-155                     Channel 153(6 GHz) : 6705     Mhz H HU HL H80-151 H160-143
el 157(5 GHz) : 5785     Mhz 11na C CU V VU V80-155 H HU HL H80-155                     Channel 157(6 GHz) : 6725     Mhz H HU HL H80-151 H160-143
el 161(5 GHz) : 5805     Mhz 11na C CL V VL V80-155 H HU HL H80-155                     Channel 161(6 GHz) : 6745   P Mhz H HU HL H80-167 H160-175
el 165(5 GHz) : 5825     Mhz 11na C CU V VU H HU HL                                     Channel 165(6 GHz) : 6765     Mhz H HU HL H80-167 H160-175
el 169(5 GHz) : 5845     Mhz 11na C CL V VL H HU HL                                     Channel 169(6 GHz) : 6785     Mhz H HU HL H80-167 H160-175
el 173(5 GHz) : 5865     Mhz 11na C V H                                                 Channel 173(6 GHz) : 6805     Mhz H HU HL H80-167 H160-175
el   1(6 GHz) : 5945   P Mhz H HU HL H80-  7 H160- 15                                   Channel 177(6 GHz) : 6825   P Mhz H HU HL H80-183 H160-175
el   5(6 GHz) : 5965     Mhz H HU HL H80-  7 H160- 15                                   Channel 181(6 GHz) : 6845     Mhz H HU HL H80-183 H160-175
el   9(6 GHz) : 5985     Mhz H HU HL H80-  7 H160- 15                                   Channel 185(6 GHz) : 6865     Mhz H HU HL H80-183 H160-175
el  13(6 GHz) : 6005     Mhz H HU HL H80-  7 H160- 15                                   Channel 189(6 GHz) : 6885     Mhz H HU HL H80-183 H160-175
el  17(6 GHz) : 6025   P Mhz H HU HL H80- 23 H160- 15                                   Channel 193(6 GHz) : 6905   P Mhz H HU HL H80-199 H160-207
el  21(6 GHz) : 6045     Mhz H HU HL H80- 23 H160- 15                                   Channel 197(6 GHz) : 6925     Mhz H HU HL H80-199 H160-207
el  25(6 GHz) : 6065     Mhz H HU HL H80- 23 H160- 15                                   Channel 201(6 GHz) : 6945     Mhz H HU HL H80-199 H160-207
el  29(6 GHz) : 6085     Mhz H HU HL H80- 23 H160- 15                                   Channel 205(6 GHz) : 6965     Mhz H HU HL H80-199 H160-207
el  33(6 GHz) : 6105   P Mhz H HU HL H80- 39 H160- 47                                   Channel 209(6 GHz) : 6985   P Mhz H HU HL H80-215 H160-207
el  37(6 GHz) : 6125     Mhz H HU HL H80- 39 H160- 47                                   Channel 213(6 GHz) : 7005     Mhz H HU HL H80-215 H160-207
el  41(6 GHz) : 6145     Mhz H HU HL H80- 39 H160- 47                                   Channel 217(6 GHz) : 7025     Mhz H HU HL H80-215 H160-207
el  45(6 GHz) : 6165     Mhz H HU HL H80- 39 H160- 47                                   Channel 221(6 GHz) : 7045     Mhz H HU HL H80-215 H160-207
el  49(6 GHz) : 6185   P Mhz H HU HL H80- 55 H160- 47                                   Channel 225(6 GHz) : 7065   P Mhz H HU HL
el  53(6 GHz) : 6205     Mhz H HU HL H80- 55 H160- 47                                   Channel 229(6 GHz) : 7085     Mhz H HU HL
el  57(6 GHz) : 6225     Mhz H HU HL H80- 55 H160- 47                                   Channel 233(6 GHz) : 7105     Mhz H
OpenWrt:/#
OpenWrt:/#
```

The following output lists channels on 6 GHz of QCN90xx or QCN602x. 'P' indicates a 6 GHz Preferred Scanning Channel (PSC).

```
Channel  1(  6GHz) : 5945   Mhz H HU H80-  7
Channel  97(   6GHz) : 6425     Mhz H HU H80-103
Channel  5(  6GHz) : 5965  P Mhz H HL H80-  7
Channel 101(   6GHz) : 6445   P Mhz H HL H80-103
Channel  9(  6GHz) : 5985   Mhz H HU H80-  7
Channel 105(   6GHz) : 6465     Mhz H HU H80-103
Channel 13(  6GHz) : 6005   Mhz H HL H80-  7
Channel 109(   6GHz) : 6485     Mhz H HL H80-103
Channel 17(  6GHz) : 6025   Mhz H HU H80- 23
Channel 113(   6GHz) : 6505     Mhz H HU H80-119
Channel 21(  6GHz) : 6045  P Mhz H HL H80- 23
Channel 117(   6GHz) : 6525   P Mhz H HL H80-119
Channel 25(  6GHz) : 6065   Mhz H HU H80- 23
Channel 121(   6GHz) : 6545     Mhz H HU H80-119
Channel 29(  6GHz) : 6085   Mhz H HL H80- 23
Channel 125(   6GHz) : 6565     Mhz H HL H80-119
Channel 33(  6GHz) : 6105   Mhz H HU H80- 39
Channel 129(   6GHz) : 6585     Mhz H HU H80-135
```

```
Channel  37(  6GHz) : 6125  P Mhz H HL H80- 39
Channel 133(   6GHz) : 6605   P Mhz H HL H80-135
Channel  41(  6GHz) : 6145   Mhz H HU H80- 39
Channel 137(   6GHz) : 6625     Mhz H HU H80-135
Channel  45(  6GHz) : 6165   Mhz H HL H80- 39
Channel 141(   6GHz) : 6645     Mhz H HL H80-135
Channel  49(  6GHz) : 6185   Mhz H HU H80- 55
Channel 145(   6GHz) : 6665     Mhz H HU H80-151
Channel  53(  6GHz) : 6205  P Mhz H HL H80- 55
Channel 149(   6GHz) : 6685   P Mhz H HL H80-151
Channel  57(  6GHz) : 6225   Mhz H HU H80- 55
Channel 153(   6GHz) : 6705     Mhz H HU H80-151
Channel  61(  6GHz) : 6245   Mhz H HL H80- 55
Channel 157(   6GHz) : 6725     Mhz H HL H80-151
Channel  65(  6GHz) : 6265   Mhz H HU H80- 71
Channel 161(   6GHz) : 6745     Mhz H HU H80-167
Channel  69(  6GHz) : 6285  P Mhz H HL H80- 71
Channel 165(   6GHz) : 6765   P Mhz H HL H80-167
Channel  73(  6GHz) : 6305   Mhz H HU H80- 71
Channel 169(   6GHz) : 6785     Mhz H HU H80-167
Channel  77(  6GHz) : 6325   Mhz H HL H80- 71
Channel 173(   6GHz) : 6805     Mhz H HL H80-167
Channel  81(  6GHz) : 6345   Mhz H HU H80- 87
Channel 177(   6GHz) : 6825     Mhz H HU
Channel  85(  6GHz) : 6365  P Mhz H HL H80- 87
Channel 181(   6GHz) : 6845   P Mhz H HL
Channel  89(  6GHz) : 6385   Mhz H HU H80- 87
Channel 185(   6GHz) : 6865     Mhz H
Channel  93(   6GHz) : 6405     Mhz H HL H80- 87
```

The following output is from 5 GHz regular radio:

```
root@OpenWrt:/# wlanconfig ath0 list chan
Channel  36 (5 GHz) : 5180     Mhz 11na C CU V VU V80- 42 V160- 50 H HU H80- 42 H160- 50     Channel 120 (5 GHz) : 5600 *^  Mhz 11na C CL V VL V80-122 V160-114 H HL H80-122 H160-114
Channel  40 (5 GHz) : 5200     Mhz 11na C CL V VL V80- 42 V160- 50 H HL H80- 42 H160- 50     Channel 124 (5 GHz) : 5620 *^  Mhz 11na C CU V VU V80-122 V160-114 H HU H80-122 H160-114
Channel  44 (5 GHz) : 5220     Mhz 11na C CU V VU V80- 42 V160- 50 H HU H80- 42 H160- 50     Channel 128 (5 GHz) : 5640 *^  Mhz 11na C CL V VL V80-122 V160-114 H HL H80-122 H160-114
Channel  48 (5 GHz) : 5240     Mhz 11na C CL V VL V80- 42 V160- 50 H HL H80- 42 H160- 50     Channel 132 (5 GHz) : 5660 *^  Mhz 11na C CU V VU V80-138 H HU H80-138
Channel  52 (5 GHz) : 5260 *^  Mhz 11na C CU V VU V80- 58 V160- 50 H HU H80- 58 H160- 50     Channel 136 (5 GHz) : 5680 *^  Mhz 11na C CL V VL V80-138 H HL H80-138
Channel  56 (5 GHz) : 5280 *^  Mhz 11na C CL V VL V80- 58 V160- 50 H HL H80- 58 H160- 50     Channel 140 (5 GHz) : 5700 *^  Mhz 11na C CU V VU V80-138 H HU H80-138
Channel  60 (5 GHz) : 5300 *^  Mhz 11na C CU V VU V80- 58 V160- 50 H HU H80- 58 H160- 50     Channel 144 (5 GHz) : 5720 *^  Mhz 11na C CL V VL V80-138 H HL H80-138
Channel  64 (5 GHz) : 5320 *^  Mhz 11na C CL V VL V80- 58 V160- 50 H HL H80- 58 H160- 50     Channel 149 (5 GHz) : 5745     Mhz 11na C CU V VU V80-155 H HU H80-155
Channel 100 (5 GHz) : 5500 *^  Mhz 11na C CU V VU V80-106 V160-114 H HU H80-106 H160-114     Channel 153 (5 GHz) : 5765     Mhz 11na C CL V VL V80-155 H HL H80-155
Channel 104 (5 GHz) : 5520 *^  Mhz 11na C CL V VL V80-106 V160-114 H HL H80-106 H160-114     Channel 157 (5 GHz) : 5785     Mhz 11na C CU V VU V80-155 H HU H80-155
Channel 108 (5 GHz) : 5540 *^  Mhz 11na C CU V VU V80-106 V160-114 H HU H80-106 H160-114     Channel 161 (5 GHz) : 5805     Mhz 11na C CL V VL V80-155 H HL H80-155
Channel 112 (5 GHz) : 5560 *^  Mhz 11na C CL V VL V80-106 V160-114 H HL H80-106 H160-114     Channel 165 (5 GHz) : 5825     Mhz 11na C V H
Channel 116 (5 GHz) : 5580 *^  Mhz 11na C CU V VU V80-122 V160-114 H HU H80-122 H160-114

V80_80 capable. Pri20: Any V80 pri20 above. Sec80 centre: Centre of any V80 block discontiguous with primary 80.
H80_80 capable. Pri20: Any H80 pri20 above. Sec80 centre: Centre of any H80 block discontiguous with primary 80.
root@OpenWrt:/#
```

5.15 Configure repetitive channel capture using wlanconfig commands for IPQ60xx and QCN90xx/QCN602x

IPQ807xA support channel capture only on directed FTM frames and ACK frames. IPQ60xx hardware is enabled to support enhanced channel capture modes. Starting with QCA_Networking_2020.SPF.11.1 release, the functionality to extract the 20 MHz, 40 MHz, and 80 MHz Channel Frequency Response (CFR) and generate a dump of this information in a periodic manner is supported, based on the user configured mode, on IPQ60xx platforms. Starting with QCA_Networking_2020.SPF.11.1 release, the functionality to extract the 20 MHz, 40 MHz, 80 MHz,

and 160 MHz Channel Frequency Response (CFR) and generate a dump of this information in a periodic manner is supported, based on the user configured mode, on QCN90xx/QCN6020x platforms. This dump will be relayed to user space for further processing by customer's application. This capability is also called channel state information (CSI) collection.

Two asynchronous events are waited upon, before relaying the CFR data to user space.

Tx based captures

- WMI event to indicate completion of CFR data DMA to host DDR
- WMI event for Tx completion of the QOS NULL frame sent to the peer

Rx based captures (RCC)

- WMI event to indicate completion of CFR data DMA to host DDR
- WDI event posted from datapath monitor status ring processing

wlanconfig commands have been introduced to configure various RCC modes. User has to select a particular mode and its relevant configuration.Every command should be sent via wlanconfig using netlink call from user space to Host.

With every wlanconfig command issued by user, the configurations will be saved into a cfr component structure. To make those configurations effective, user has to issue "commit" command. Issuing "commit" command specifies that the information provided by user is the final configuration and "commit" command will act as a trigger to send the configuration to the FW.

5.15.0.1 RCC mode config commands

Table 5-1 RCC mode config commands

Parameter	Command	Description
Mode configuration:		
M_DIRECTED_FTM	wlanconfig athx cfr m_directed_ftm 1/0	Setting m_directed_ftm bit to 1 will filter in directed FTM and ACK as FP and setting it to 0, will drop directed FTM/ACK frame. Currently, this mode is not supported in IPQ60xx and QCN90xx/QCN602x.
M_DIRECTED_NDPA_NDP	wlanconfig athx cfr m_directed_ndpa_ndp 1/0	Set m_directed_ndpa_ndp to 1 and all directed NDPA and NDP will be filtered in as FP and setting it to 0, will drop directed NDPA and NDP.
M_TA_RA_Filter	wlanconfig athx cfr m_ta_ra_filter 1/0	Set m_ta_ra_filter to 1 and filter in packets with programmed Tx address/ Rx address/ Tx address mask/ Rx address mask/ BW/ NSS/ packet type and subtype as MO. Setting it to 0, will disable this mode.
M_ALL_FTM_ACK	wlanconfig athx cfr m_all_ftm_ack 1/0	Set m_all_ftm_ack to 1 and all FTM and ACK will be filtered in as FP. Setting it to 0 will drop non-directed FTM/ACK frame. This is not supported in IPQ60xx and QCN90xx/QCN602x.
M_NDPA_NDP_ALL	wlanconfig athx cfr m_ndpa_ndp_all 1/0	Set m_ndpa_ndp_all to 1 and all NDPA and NDP including non-directed will be filtered in as FP; whereas setting it to 0, will filter in NDPA and NDP as MO.

Table 5-1 RCC mode config commands

Parameter	Command	Description
M_ALL_PKT	wlanconfig athx cfr m_all_pkt 1/0	Setting m_all_pkt bit to 1, will filter in all packets as MO. Setting it to 0, will not filter in any type of frames. Currently, the support is not available for IPQ60xx and QCN90xx/QCN602x but in next phase this capture mode is going to be supported.
disable_all	wlanconfig athx cfr disable_all	This will disable all capture modes. Ex: wlanconfig ath0 cfr disable_all

5.15.0.2 TA_RA configuration commands

According to the design, a user must understand that group configuration is only applicable for m_ta_ra_filter mode out of aforementioned mentioned modes. In case of m_ta_ra_filter mode, hardware allows a user to configure 16 entries. In other words, hardware allows us to have 16 group of configurations (0 to 15).

User needs to specify the group id while providing group parameters. Every group should have following nine parameters : Tx address, Tx addr mask, Rx address, Rx address mask, bandwidth, nss, subtype (mgmt, ctrl, data). Users can provide inputs for upto 16 groups, but they have the option to either group configuration.

Table 5-2 TA_RA configuration commands

Parameter	Command	Description
In case of M_TA_RA_FILTER mode, RXPCU provides 16 sets of filter registers for TA/RA/type based captures. If a filter set is enabled and all conditions in the filter set match, then capture is performed. For TA/RA match, FW will program the following values which will be obtained from user provided configuration and passed on by Host to FW. **Input in case of M_TA_RA_FILTER Mode:**		
ta_ra_addr	wlanconfig athx cfr ta_ra_addr <group_id> <ta> <ta_mask> <ra> <ra_mask> EX: wlanconfig ath0 cfr ta_ra_addr 0 00:03:7f:12:42:4b ff:ff:ff:ff:ff:ff 00:00:00:00:00:00 00:00:00:00:00:00	Input for ta_ra_addr entry should be group_id, ta, ta_mask, ra and ra_mask. Group_id could be any value, in between 0 to 15. ta-> Tx address, should be the client MAC address, which will send the pkt to the AP. The status of the client could be directed or non-directed. ra-> Rx address, Could be the MAC address of AP or the user can provide 00:00:00:00:00:00, based on below statement. Evaluation of the address and is as follows. Note that a mask (ta_mask / ra_mask) of 0 will match any address, either TA or RA: Match = ((addr & expect_mask) == expect_addr) The example provided here, specifies that the configuration is being done for 0th group, with ta being the MAC address of client and ta_mask is ff:ff:ff:ff:ff:ff. Whereas ra and ra_mask being 0; which means ra could be any address. User could have also provided ra as Root AP's MAC address and ra_mask as ff:ff:ff:ff:ff:ff.

Table 5-2 TA_RA configuration commands

Parameter	Command	Description
bw_nss	wlanconfig athx cfr bw_nss <group_id> <bw> <nss>	Each filter set also includes bandwidth and NSS match condition. Group_id could be any value, in between 0 to 15. 4 bits are allotted for bandwidth. Bit 0 : if set, bw = 20 MHz Bit 1 : if set, bw = 40 MHz Bit 2 : if set, bw = 80 MHz Bit 3 : if set, bw = 160 MHz 8 bits are allotted for NSS mask: Specifies which numbers of spatial streams (MIMO factor) are permitted. Bit 0 : if set, NSS = 1 (non-MIMO) is permitted Bit 1 : if set, NSS = 2 (2x2 MIMO) is permitted Bit 2 : if set, NSS = 3 (3x3 MIMO) is permitted Bit 3 : if set, NSS = 4 (4x4 MIMO) is permitted Bit 4 : if set, NSS = 5 (5x5 MIMO) is permitted Bit 5 : if set, NSS = 6 (6x6 MIMO) is permitted Bit 6 : if set, NSS = 7 (7x7 MIMO) is permitted Bit 7 : if set, NSS = 8 (8x8 MIMO) is permitted For IPQ60xx, the maximum supported bw and NSS should be 80Mhz and 2. For IPQ60xx, the maximum supported bw and NSS should be 160Mhz and 4. If all bandwidth and NSS are to be captured, then enable all. Ex: wlanconfig ath0 cfr bw_nss 0 0xf 0xff For IPQ50xx internal 2G radio, the maximum supported bandwidth and NSS must be 40 MHz and 2 respectively
subtype	wlanconfig athx cfr subtype <group_id> <mgmt> <ctrl> <data>	User should provide the desired management/control/data type/subtype combinations for each filter set to capture CFR data. Group_id could be any value, in between 0 to 15. mgmt, ctrl and data, all are bitmasks, in which each bit represents the corresponding subtype value as per IEEE80211. If all valid type and subtype combinations are to be captured, then enable all. Ex: wlanconfig ath0 cfr subtype 0 0xffff 0xffff 0xffff

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 5-2 TA_RA configuration commands

Parameter	Command	Description
en_cfg	wlanconfig athx cfr en_cfg <bitmask>	After providing all the group configurations, user should provide the information of which groups need to be enabled. Based on that FW will enable the configurations for CFR groups. Here en_cfg is a bitmap of 16 bits, which indicates 16 groups. Valid entry should be in between 0 to 0xFFFF. Turning on a bit here will enable MAC TA_RA filter for corresponding group. If user has to enable only 0th group, then input should be: wlanconfig ath0 cfr en_cfg 0x1
reset_cfg	wlanconfig athx cfr reset_cfg <bitmask>	Turning a bit here will reset all 9 params in the corresponding group to default values. Reset_cfg is a bitmap of 16 bits, which indicates to 16 groups. Valid entry should be in between 0 to 0xFFFF. If user wants to reset the configurations of 0th, 1st and 3rd group, then the input should be : wlanconfig ath0 cfr reset_cfg 0xb

5.15.0.3 Other CFR configuration commands

There are other configurations which are neither group specific nor mandatory. Users might use it based on the necessity.

Table 5-3 Other CFR config commands

Parameter	Command	Description
capture_dur (capture duration)	wlanconfig athx cfr capture_ dur <value>	FW may limit the interval and duration during which HW may attempt capture by programming the user provided values. These values range from 1 us to roughly 16.8 s in 1 us units. Max value is 0xFFFFFF, i.e., 16.777215 s. If capture_dur= 0, it means that only one valid CFR will be gathered. After that, the timer of capture_intval will kick off. Once the timer expires, MAC will start capturing the next CFR. If both capture_dur and capture_intval are set to ‘0’, it means that there is no defined window, and all packets that have the freeze TLV set to ‘1’ will be eligible for channel capture. Note: capture_dur shall be less than or equal to capture_intval. capture_dur and capture_intval apply to all capture modes. Each MAC has its own set of parameters, i.e., one pair of capture_dur and capture_intval for 2.4G MAC and one pair for 5G MAC.
capture_intval (capture interval)	wlanconfig athx cfr capture_ intval <value>	
ul_mu_user_mask	wlanconfig athx cfr ul_mu_ user_mask <mask_lower_ 32> <mask_upper_32>	Since IPQ60xx supports max bandwidth of 80Mhz, and QCN90xx/QCN602x supports max bandwidth of 160Mhz, maximum number of users in a UL MU-MIMO transmission would be 37. mask_lower_32 and mask_upper_32 are 32 bits each. mask_lower_32: Bits from 31 to 0 indicate user indices for 32 users in a UL MU-MIMO session. If bit 0, 1, 15 are set, then CFR capture will happen for user indices 0,1 and 15. Mask_upper_32: Bits from 0 to 4 indicate user indices from 33 to 37; whereas bits from 5 to 31 are reserved. If bit 0 is set, then CFR capture is enabled for user index 33 in a UL MU-MIMO transmission. NOTE: For UL MU, repetitive channel capture only supports one user due to MAC only reporting one user index in the freeze TLV. PHY HW already captures channel for all RUs, and uCode already uploads channel capture for all RUs. If MAC can report user mask instead of single user index in the freeze TLV, repetitive channel capture can support multiple users without PHY HW or uCode changes. FW will configure the UL MU user mask as per the user configuration, to indicate for which user(s) channel capture needs to be provided. If multiple users pass the mask and filter, MAC only sends the user index for the one user with the highest priority in the freeze TLV. The priority order is user0>user1>…>userN

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 5-3 Other CFR config commands

Parameter	Command	Description
freeze_tlv_delay_cnt	wlanconfig ath0 cfr freeze_ tlv_delay_cnt <freeze_tlv_ delay_ cnt_en> <threshold>	It indicates the number of consecutive RX frames to be skipped before CFR capture is enabled again. Freeze_tlv_delay_cnt_en : only 1 bit is alloated for this. Either 0 or 1 for disabling or enabling. Threshold: Could be from 1 to 65535 .
Commit	wlanconfig athx cfr commit	After all the required configurations are prompted, it is expected from the user to issue a 'commit' command, which will trigger to send the configurations to FW.
GET Command:		
rcc_config_details	wlanconfig athx rcc_config_ details	With the initiation of "commit" command an array of TLV processing will be started and sent to FW. If that succeeds, then the committed configuration will be saved. rcc_config_details will prove the last successfully committed configuration.
CFR DEBUG LOG:		
To enable CFR debug		iwpriv ath0 qdf_cv_lvl 0x006b0008
To disable CFR debug		iwpriv ath0 qdf_cv_lvl 0x006b0000

5.15.0.4 Sample configuration commands for CFR repetitive capture

The following are examples of different mode configurations:

Table 5-4 Sample commands

Filter Mode	Configuration example (Enable :: value = 1 Disable :: value = 0)
M_DIRECTED_FTM	wlanconfig ath0 cfr m_directed_ftm < value > wlanconfig ath0 cfr commit
M_DIRECTED_NDPA_NDP	wlanconfig ath0 cfr m_directed_ndpa_ndp <value> wlanconfig ath0 cfr commit
M_TA_RA_Filter	Enable: wlanconfig athx cfr m_ta_ra_filter 1 wlanconfig ath0 cfr ta_ra_addr 0 00:03:7f:12:42:4b ff:ff:ff:ff:ff:ff 00:00:00:00:00:00 00:00:00:00:00:00 wlanconfig ath0 cfr bw_nss 0 0xf 0xff wlanconfig ath0 cfr subtype 0 0xffff 0xffff 0xffff wlanconfig ath0 cfr en_cfg 0x1 wlanconfig ath0 cfr commit Disable: wlanconfig athx cfr m_ta_ra_filter 0 wlanconfig ath0 cfr commit

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 5-4 Sample commands

M_ALL_FTM_ACK	wlanconfig ath0 cfr m_all_ftm_ack <value> wlanconfig ath0 cfr commit
M_NDPA_NDP_ALL	wlanconfig ath0 cfr m_ndpa_ndp_all <value> wlanconfig ath0 cfr commit
M_ALL_PKT	Enable: wlanconfig ath0 cfr m_all_pkt 1 wlanconfig ath0 cfr capture_dur 0x4ff wlanconfig ath0 cfr capture_intval 0xffffff wlanconfig ath0 cfr commit Disable: wlanconfig ath0 cfr m_all_pkt 0 wlanconfig ath0 cfr commit
Note: At a time more than one filter mode can be enabled.	

For example, if only M-All-Packet is enabled, and all other capture modes are disabled, then capture_reason will always be ALL_PACKET for any captured frame. That means, even if channel capture is done on an NDP packet, capture_reason will be ALL_PACKET.

Another example, if both M-All-Packet and M-NDPA-NDP are enabled, and all other capture modes are disabled, then capture_reason will be NDPA_NDP if channel capture is done on an NDP packet, and ALL_PACKET if channel capture is done on a non-NDP packet. That means, even if channel capture is done on an FTM packet, capture_reason will be ALL_PACKET.

5.15.1 Additional CFR capture capabilities for QCN90xx/QCN602x

5.15.1.1 Capture Interval mode selection

- QCN90xx/QCN602x support a new capture interval mode: capture_count, where after capture_count+1 number of CFR frames, MAC stops channel capture and waits for capture_interval duration before enabling again. MAC provides a nob register setting to choose either capture_duration mode or capture_count mode.
- The default mode will be the older capture_duration where MAC stops channel capture after capture_duration instead of capture_count and re-enables after capture_interval-capture_duration.

5.15.1.2 Command to enable capture_count mode

The following commands are added to support the capture_interval_mode selection.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Parameter	Command	Description
capture_count	wlanconfig athx cfr capture_count <value>	FW may limit the number of captures that HW should attempt by configuring the capture_count parameter. The max allowed value is 0xFFFF. The value of this parameter is considered only when parameter "**capture_intervalmode_sel**" is set to 1
capture_intervalmode_sel	wlanconfig athx cfr capture_ intervalmode_sel <value>	Controls whether **capture_ duration** mode to be enabled or **capture_count** mode be enabled. set 0: to enable capture duration mode set 1: to enable capture count mode

Qualcomm Confidential – May Contain Trade Secrets 2025-05-13 03:57:44 PDT pan.yaxing@zte.com.cn

5.16 Configure access control of clients based on a list of stations

Starting with the QCA_Networking_2020.SPF.11.2 release, this feature is supported on IPQ807x/IPQ807xA/IPQ817x, IPQ60xx, IPQ50xx, and QCN90xx/QCN602x platforms. Starting with the IPQ5018.ILQ.11.4 release, this feature is extended to QCN61xx chipsets attached to IPQ50xx platforms. This capability is introduced to configure a control mechanism for traffic that is transmitted from or received by identified LAN devices. A mechanism by which the host provides a list of STA (by MAC or IP) such that traffic from/to devices on the list is forwarded to the host (Linux bridge) for forwarding decision. For an example, if the list included STA A, B and C, traffic originated from A, B or C would get forwarded to the host for forwarding decision. Similarly, traffic originated from any device within the same AP to A, B, or C is forwarded to the host for forwarding decision.

For the stations on the list configured, forwarding decisions to and from the stations are ignored by the Wi-Fi driver and the decision is made by the Linux bridge or other proprietary modules.

Multicast traffic is forwarded to the Linux bridge or other proprietary modules for forwarding decisions.

Multicast enhancement must be enabled with this feature as this will allow converted unicast packet to reach clients in isolation list.

Multicast packets from the source in the peer isolation list is not forwarded and pushed to bridge stack.

5.16.1 Configure access control of clients based on a list

This feature is disabled using the **CONFIG MACRO**, and it is enabled by default using this compilation flag: **QCA_SUPPORT_PEER_ISOLATION.** List of stations under peer isolation list needs to be configured by user on all BSSs in the network.

To add or delete a station from the peer isolation list, use the following wlanconfig commands:

```
wlanconfig athx peer_isolation add <STA_MAC>
wlanconfig athx peer_isolation del <STA_MAC>
```

To display the list of stations in the peer isolation list, use the following command:

```
wlanconfig athx peer_isolation list
```

To flush all the stations that are being used from the peer isolation list, use the following command:

```
wlanconfig athx peer_isolation flush
```

5.17 Wireless network management commands

This section describes the wireless network management (WNM) settings that can be configured using the wlanconfig tool.

Table 5-5 Configure/show ATF parameters

Parameter	Format	Description
setbssmax <idle period in seconds> [<idle option>]	wlanconfig ath0 wnm setbssmax <idle period in seconds> [<idle option>]	Configures the max BSS Idle Timer value:
getbssmax	wlanconfig ath0 getbssmax	Retrieves and displays the configured the max BSS Idle Timer value:
tfsreq <filename>	wlanconfig ath0 wnm tfsreq <filename>	Configures WNM TFS request.
deltfs	wlanconfig ath0 wnm deltfs	Retrieves and displays the configured WNM traffic indication map interval.
timintvl <Interval>	wlanconfig ath0 wnm timintvl <Interval>	Configures WNM traffic indication map interval.
gettimparams	wlanconfig ath0 wnm gettimparams	Retrieves and displays the configured WNM traffic indication map interval.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 5-5 Configure/show ATF parameters (cont.)

timrate <highrateEnable> <lowRateEnable>	wlanconfig ath0 wnm timrate <highrateEnable> <lowRateEnable>	Configures WNM traffic indication map high and low rates.
bssterm <delay in TBTT> [<duration in minutes>]	wlanconfig ath0 wnm bssterm <delay in TBTT> [<duration in minutes>]	Configures the BSS termination duration and target beacon transmission time (TBTT) offset for control frames.

5.18 Smart mesh wlanconfig commands

The wireless subsystem needs to support API to add, remove, and update Vendor Information Element which used in 802.11 Beacon, ProbeReq, ProbeResp, AssocReq, and AssocResp frames. Wireless driver must be capable of providing API or CLI to support on adding, removing, and update Vendor Information Element which used in 802.11 Beacon, ProbeReq, ProbeResp, AssocReq, and AssocResp frames.

The following is the data struture of vendor IE with n currently defined as 4 bytes:

```
Struct ntgr_ie {
        Unsigned int8 id  ;   //221
            Unsigned int8 len ;   // non fixed length
        Unsigned int8 oui[3]  ;   // 00:09:5B Netgear OUI
            Unsigned int8 private [n]  ; // private info
      }
```

The wlanconfig application provides CLI interface for adding, removing, and updating vendor information element to the driver and from the driver as follows:

- wlanconfig athX vendorie add len <oui+pcap_data in bytes> oui <eg:xxxxxx> pcap_data <eg:xxxxxxxx> ftype_map <eg:xx>
- wlanconfig athX vendorie update len <oui+pcap_data in bytes> oui <eg:xxxxxx> pcap_data <eg:xxxxxxxx> ftype_map <eg:xx>
- wlanconfig athX vendorie remove
- wlanconfig athX vendorie list
- wlanconfig ath0 vendorie add len 7 oui 112233 pcap_data 11223344 ftype_map 12 /* beacon & assoc res */
- wlanconfig ath0 vendorie add update 9 oui 112233 pcap_data 112233445566 ftype_map 05 /* probe request & assoc res */

Keep the following points in mind while using the smart mesh feature:

- Under IEEE80211_IOCTL_CONFIG_GENERIC ioctl support , New sub ioctl commands are added under IEEE80211_IOCTL_CONFIG_GENERIC defined in driver and used by any

application like wlanconfig for adding /remvoing vendor IE elements to driver. This is something similar to existing IEEE80211_IOCTL_SET_APPIEBUF but will have more flexibility on how the IE are added to the different frame types

- Defined new application Information Element handle in ieee80211vap structure (vap to iv_smartmesh_ie_handle of type struct wlan_mlme_app_ie) which maintains separate list for storing vendor IE across each frames types like Beacon , Probe Request /Response , Assoc Request /Response.
- When wlanconfig application add/update/del/list smart mesh vendor IE and in turn calls IEEE80211_WLANCONFIG_VENDOR_IE_ADD /DEL/UPDATE/LIST ioctl handler. Based on the command ,it populate/update/delete/fetch smart mesh vendor IE in vap to iv_smartmesh_ie_handle inside vap structure and also to update correspondingly on the global vap ie list (vap to iv_app_ie_list). Wlanconfig vendor ie command got options to specify which mgmt frame like Beacon , Probe Request /Response , Assoc Request/Response are transmitted should contain the vendor ie element. These smart mesh IE also will be added as part of global application IE list inclusion apart from adding other application information elements.
- wlan_set_vendorie() and wlan_get_vendorie() as a driver API to add , remove and update& list vendor information elements.

Variable vendor ie length can be added. the max size limit set is 128 bytes.

5.19 Support for RTT responder-only without LOWI tool

Starting with QCA_Networking_2021.SPF.11.4 release, this feature is implemented on IPQ40xx, IPQ806x, IPQ807x/IPQ807xA/IPQ817x, IPQ50xx, IPQ60xx, and QCN90xx/QCN602x/QCN61xx platforms.

LOWI is required in responder mode to configure or trigger LCI, LCR, and FTMRR. The LOWI application is not really used for any other purpose in such cases. To enable configuration of LOWI settings using generic NL commands so that LOWI tool can be removed if responder only mode is required, a cfg80211tool and wlanconfig command for configuration of LOWI properties in responder mode is introduced. This enhancement enables removal of LOWI application if only responder mode is required and optimizes memory consumption with such a removal.

The following are the wlanconfig commands to configure LCI/LCR/FTMRR responder mode:

```
wlanconfig athX rtt ftmrr <ftmrr input values>

wlanconfig athX rtt lci <lci input values>

wlanconfig athX rtt lcr <lcr input values>
```

Until QCA_Networking_2020.SPF.11.3 release, LOWI test application reads the input LCI/LCR/FTMRR responder xml file from the command line and passes it to LOWI server. The purpose of the LOWI server for these commands is to prepare only WMI command and pass it to FW through host driver.

Enter the xml file values as an individual input values to wlanconfig command.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
wlanconfig athx rtt ftmrr 00:03:7F:12:DC:DC 20 1 00:03:7F:12:DC:DC
c91c0000 149 155 0 2 128 9
```

wlan_cfg80211_rtt_params() reads the input values and calls ieee80211_lowi_send_ftmrr_frame() to send FTMRR action frame to STA.

Qualcomm
Confidential – May Contain Trade Secrets
2021-05-13 03:57:44 PDT
pan.yafeng@zte.com.cn

6 Miscellaneous commands

This chapter describes the commands and parameters other than the iwconfig, iwpriv, wifitool utility, and wlanconfig commands.

6.1 Athssd parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Spectral and Classifier Demo Application: athssd* section of the *Spectral scan and analysis* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 6-1 Athssd parameters

Configuration	Format	Gen I	Gen II	Gen III	Description
Standalone Scan	athssd -i *wifiN* -j *athN* -s [-b *band*]				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. -b specifies the band in which standalone scan is to be done (1 - 2.4 GHz, 2 - 5 GHz, 3 - 6 GHz). This is optional and defaults to 2.4 GHz or 5 GHz depending on the channel in which standalone scan is requested. -b is applicable only when standalone scan is requested in a channel other than operating channel.
		Y	Y	Y	s=0
		Y	Y	Y	s>0
Agile Scan	athssd -i wifiN -j athN -a -f freq 1 [-g freq2]	N	N	Y	Start athssd, configuring it to carry out an agile scan on frequency span specified by *freq1, fre2*.

Table 6-2 Athssd debug parameters

Parameter	Gen I	Gen II	Gen II	Description
-c	Y	Y	Y	Capture logs (0: None 1:MWO 2:CW 3:WiFi 4:FHSS 5:All)
-x	N	N	Y	Enable(1)/disable(0) generation III linear bin format scaling (default: enabled, applicable for generation III). Will be ignored for other generations.
-u	Y	Y	Y	Uses UDP socket
- i <ra-dio_name>	Y	Y	Y	Indicates radio interface

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

Table 6-2 Athssd debug parameters (cont.)

Parameter	Gen I	Gen II	Gen II	Description
-j <de-vice_name>	Y	Y	Y	Indicates device interface
-d	Y	Y	Y	Enable debug prints
-h	Y	Y	Y	Print help message
-a	N	N	Y	Enable Agile mode (if available on chipset). Available only in cfg80211 mode.
-b	N	N	Y	Specifies the band in which standalone scan is to be done (1 - 2.4 GHz, 2 - 5 GHz, 3 - 6 GHz). This is optional and defaults to 2.4 GHz or 5 GHz depending on the channel in which standalone scan is requested. This field is applicable only when standalone scan is requested in a channel other than operating channel.
-f	N	N	Y	Spectral frequency - Currently applicable only for Agile mode. Center frequency (in MHz) of the span of interest, or for convenience, center frequency (in MHz) of any channel in the span of interest. Indicates primary 80 MHz span for 80p80 Agile Spectral scan. The value configured currently serves as the initial value with which to start operation. Available only in cfg80211 mode.
-g	N	N	Y	Spectral frequency - Currently applicable only for 80p80 Agile mode. Center frequency (in MHz) of the secondary 80 MHz span of interest, or for convenience, center frequency (in MHz) of any channel in the secondary 80 MHz span of interest. The value configured currently serves as the initial value with which to start operation. Available only in cfg80211 mode.
-s	Y	Y	Y	Stand alone Spectral scan on operational channel <channel> In normal mode: <channel> can be 0 in which case the current operating channel is used. In Agile mode: The -s option need not be provided since Agile mode forces stand alone Spectral scan on Spectral frequency configured using '-f' instead of operational channel. However if this option is specified, <channel> must be 0 and this indicates that the operational channel is inapplicable. Multiple instances of athssd can be started with stand alone Spectral operation. This is intended to allow parallel classification on different radios, or different modes (Normal vs. Agile) on the same radio. (However currently parallel classification on the same radio with the same mode is not prohibited if the user chooses to invoke athssd in this manner so that some potential debug use cases are enabled, but there might be limitations in this scenario such as higher than necessary CPU utilization, restoration of old configs to those being used by older instance of athssd rather than other applications, timeout experienced by other instances on a radio with a given mode if one of the instances on that radio with that mode is stopped, etc. Hence this should be used only for debug). Only a single instance of athssd can be started without stand alone Spectral operation, though this does not prevent other instances with stand alone Spectral operation from being started. Also note that athssd without stand alone Spectral operation is not supported in production currently and is present only for some limited debug purposes requiring additional changes. Invoking athssd without standalone Spectral operation may currently interfere with other parallel instances on the same radio.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

6.2 DFS parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Dynamic frequency selection (DFS)* section of the *Advanced channel selection capabilities* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

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

1. Use UCI command Wi-Fi detect to get the default parameters of the radio.
2. Set up the following parameters:
   a. Set up country code
   b. Select the proper RADIO
   c. Select the proper mode. Possible modes are
      i. 11A
      ii. 11NAHT20
      iii. 11NAHT40PLUS
      iv. 11NAHT40MINUS
      v. 11ACVHT20
      vi. 11ACVHT40PLUS
      vii. 11ACVHT40MINUS
      viii. 11ACVHT80
      ix. 11ACVHT160
      x. 11ACVHT80_80
   d. Select the appropriate channel
   e. For FCC testing following extra set up is necessary:
      i. Set rate control to manual mode.
      ii. Set manual rate to 9 Mbps (Use uci set wireless.@wifi-iface[0].set11NRates=0x80808080)
      iii. Iwpriv command can also be used for (i) and (ii)
      iv. Commit the configuration using uci commit

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

6.3 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

6.4 Radartool parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Dynamic frequency selection (DFS)* and *Zero wait for DFS channels* sections of the *Advanced channel selection capabilities* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Table 6-3 Radartool parameters

Parameter	Format	DA	OL	Description
usenol	radartool -i wifiX [*0*|*1*] usenol *[0/1/2]*	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. **NOTE** usenol 2—Make CSA with next channel, switch to a new channel on radar detect and add the radar hit channels to NOL. For full offload chipsets, NOL resides in FW as well and the NOL timeout of the FW cannot be modified. With usenol 2 option (used only for internal testing) the NOL timeout of the host can be configured and the channels are not added to FW NOL.
dfsdebug	radartool -i wifiX [*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 wifiX [*0*|*1*] shownol *debug_level*	Y	Y	Displays the NOL list. Set dfsdebug level to 0x2000 before using command
enable	radartool –i wifiX enable	Y	Y	Enables dfs on a particular channel
disable	radartool –i wifiX disable	Y	Y	Disables dfs on a particular channel

Qualcomm Confidential - May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.xfeng@zte.com.cn

Table 6-3 Radartool parameters (cont.)

Parameter	Format	DA	OL	Description
ignorecac	radartool –i wifiX ignorecac 0|1	Y	Y	Set ignore cac value; if this value is set to 1, waits for cactimeout value, and if this value is set as 0, this ignore-CAC setting is disabled.
shownolhistory	radartool -i wifiX [0|1] shownolhistory	Y	Y	Displays the NOL History. The NOL History is meaningful when the Wi-Fi device supports STA mode DFS. STA mode DFS can be enabled or disabled by using the command: iwpriv wifiX [0/1] `staDFSEnable` 1/0. NOL history bit is set for a channel if radar is seen in the channel at least once. The NOL history persists until the wireless driver is removed from the Operating System.

Table 6-4 Zero CAC DFS Radartool parameters

Parameter	Format	D	O	Description
showPreCACLists	radartool -i wifiX showPreCACLists	N	Y	It shows the contents of pre-CAC-required, pre-cac-done and pre-cac-NOL lists. Usage: `radartool -i wifiX showPreCACLists`

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

Table 6-4 Zero CAC DFS Radartool parameters

bangradar	radartool -i wifiX bangradar	N	Y	Type 1: Syntax: radartool -i wifi0 bangradar Description: Adds all subchannels (of all segments) of the current channel to NOL. Type 2: Syntax: radartool -i wifi0 bangradar X Description: Adds all subchannels of current channel of segment ID "X" to NOL. The current implementation only supports 0/1 as plausible Segment IDs (X). Also, in 160 mode of operation, issuing this type adds all channel, regardless of segment ID. Type 3: Syntax: radartool -i wifi0 bangradar X Y Z Description: Adds radar affected subchannels computed by segment ID "X", is_chirp "Y" and frequency offset "Z" to NOL. The first argument is the segment ID (0 - primary segment, 1 - secondary segment). The second argument is to indicate if radar to be simulated is chirp or non chirp (0 - non chirp, 1 - chirp, chirp radar will have a bandwidth of 20MHz). The third argument being the frequency offset value, the offset to be added to the center frequency of the segment (X) to simulate radar at required frequency. Example: To simulate non chirp radar injection at frequency 5540 during 100HT80 operation, issue command: radartool -i wifi0 bangradar 0 0 10. Since center frequency of segment 0 in our current mode of operation is 5530, offset being 10, radar will be injected at center frequency 5540, and 5540 will be added to NOL. To add only radar affected subchannels, enable subchannel marking using "cfg80211tool wifi0 mark_subchan 1", else all subchannels will be added regardless of the frequency offset provided. Issuing radartool -i wifi0 bangradar X Y (or with more than three arguments) will result in an error. Starting with QCA_Networking_2019.SPF.10.0, the `bangradar X Y Z` parameter (Type 3) replaces the `bangradar_enh` parameter, which has been removed,
bangradar 1	radartool -i wifi0 bangradar 1	N	Y	It simulates RADAR in the secondary VHT80. The value of 1 denotes the second segment. If secondary is not doing any pre-CAC, then this command has no effect. Usage: `radartool -i wifiX bangradar 1` Starting with QCA_Networking_2019.SPF.10.0, the `bangradar 1` parameter replaces the `secondSegmentBangradar` parameter, which has been removed,

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

6.5 Spectraltool parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Generation I spectral scans, Generation II spectral scans, Generation III spectral scans,* and *Spectraltool: Command-line utility* sections of the *Spectral scan and analysis* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

6.5.1 Spectraltool options

- -i wifiN—Specifies the radio on which Spectral scan is to be done
- [-a]—-a option indicates agile mode.

Table 6-5 Spectral tool parameters

Parameter	Format	Description
fft_period	spectraltool -i wifiN [-a] fft_period val	Set skip interval for FFT reports. (Applicable only for generation I)
scan_period	spectraltool -i wifiN [-a] scan_period val	Set spectral scan period. Till generation II: Period increment resolution is 256*Tclk, where Tclk = 1/44 MHz (Gmode), 1/40 MHz (Amode) Generation III: Period in microseconds
scan_count	spectraltool -i wifiN [-a] scan_count val	Set number of reports to return
short_report	spectraltool -i wifiN [-a] short_report {1|0}	Set to 1 to report only one set of FFT results per spectral_scan_period. (Applicable only for generation I)
priority	spectraltool -i wifiN [-a] priority {1|0}	Set priority.
fft_size	spectraltool -i wifiN [-a] fft_size val	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). (Applicable only for generation II onwards) For generation III chipsets, the minimum value is currently 5 (num_fft_pts=32). Note: The number of FFT data points finally delivered will also depend on rpt_mode.
gc_ena	spectraltool -i wifiN [-a] gc_ena {1|0}	Set to enable targeted gain change before starting the spectral scan FFT. (Applicable only for generation II. Not applicable for generation I and III).
noise_floor_ref	spectraltool -i wifiN [-a] noise_floor_ref val	Set noise floor reference number (signed) for the calculation of bin power (dBm). (Applicable only for generation II onwards).
init_delay	spectraltool -i wifiN [-a] init_delay val	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. (Applicable only for generation II onwards).
nb_tone_thr	spectraltool -i wifiN [-a] nb_tone_thr val	Set number of strong bins (inclusive) per sub-channel, below which a signal is declared a narrow band tone. (Applicable only for generation II onwards)
str_bin_thr	spectraltool -i wifiN [-a] str_bin_thr val	Set bin/max_bin ratio threshold over which a bin is declared strong, for spectral scan bandwidth analysis. (Applicable only for generation II onwards)

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 6-5 Spectral tool parameters (cont.)

Parameter	Format	Description
wb_rpt_mode	spectraltool -i wifiN [-a] wb_rpt_mode {1/0}	(Applicable only for generation II onwards) Generation II: Set this to 1 to report spectral scans as EXT_BLOCKER, if none of the sub-channels are deemed narrow band. Generation III: Set this bit to filter the scan based on whether the signal meets the narrowband threshold (nb_tone_thr). The HW will set the false_sscan bit in the report if the check is not satisfied.
rssi_thr	spectraltool -i wifiN [-a] rssi_thr val	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). (Applicable only for generation II onwards) Width is 8 and default value is 0xf0.
pwr_format	spectraltool -i wifiN [-a] pwr_format {0/1}	Format of frequency bin magnitude for spectral scan triggered FFTs. (Applicable only for generation II onwards) ■ 0—linear magnitude ■ 1—log magnitude (20*log10(lin_mag), 1/2 dB step size)
rpt_mode	spectraltool -i wifiN [-a] rpt_mode val	Format of per-FFT reports to software for spectral scan triggered FFTs. (Applicable only for generation II onwards) ■ 0—No FFT report (only pulse end summary) ■ 1—2-dword summary of metrics for each completed FFT ■ 2: 2-dword summary of metrics for each completed FFT + in-band bins per FFT (half of the number of FFT bins), where the FFT input is sampled at two times the channel bandwidth + spectral scan summary report. In the case of QCA9984/QCA9888, 8 additional bins are reported, 4 bins to the left and right of the band-edge ■ 3: 2-dword summary of metrics for each completed FFT + all bins per FFT, where the FFT input is sampled at two times the channel bandwidth + spectral scan summary report
bin_scale	spectraltool -i wifiN [-a] bin_scale val	Number of LSBs to shift out to scale the FFT bins for spectral scan triggered FFTs. (Applicable only for generation II onwards) It is recommended to set this to 0 for dBm format reporting, for generation III chipsets - note that the default is 1, and this would have to explicitly be set to 0. It is recommended to set this to 1 for linear format reporting, for generation III chipsets. Width is 2 and default value is 1.
dBm_adj	spectraltool -i wifiN [-a] dBm_adj {1/0}	Set to 1 (with pwr_format=1), to report bin magnitudes converted to dBm power using the noisefloor calibration results. (Applicable only for generation II onwards)

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 6-5 Spectral tool parameters (cont.)

Parameter	Format	Description
chn_mask	spectraltool -i wifiN [-a] chn_mask val	Set per chain enable mask to select input ADC for search FFT. (Applicable only for generation II. Currently ignored for generation III – this might or might not change for future releases) Width is 4 and default value is 0x1.
frequency	spectraltool -i wifiN [-a] frequency val1 [val2]	This parameter is applicable only for agile mode. val1 indicates the Center frequency (in MHz) of the span of interest or for convenience, center frequency (in MHz) of any channel in the span of interest. For 80p80 Agile spectral scan val1 describes the primary80 MHz span. val2 is applicable only for 80p80 Agile mode. val2 represents the center frequency (in MHz) of the secondary 80 MHz span of interest, or for convenience, center frequency (in MHz) of any channel in the secondary 80 MHz span of interest.

NOTE The 'debug' parameter (not included in preceding table) is currently intended for Qualcomm internal developer debugging uses under certain system limitations and is not currently supported for use by customers.

6.5.2 spectraltool commands

- spectraltool -i wifiX [-a]—To get the list of parameters supported and to see the current configured values
- spectraltool -i wifiX [-a] <parameter> <value>—To change the spectraltool parameters such as fftsize, scan_periodity, scan_period and so on.
- spectraltool -i wifiX [-a] -startscan—To enable spectral scan
- spectraltool -i wifiX [-a] stopscan—To disable the spectral scan

Spectral FFT data can be dumped to a file by spectral tool using the following commands:

```
# spectraltool -i wifiX [-a] startscan
# spectraltool -i wifiX [-a] raw_data
# spectraltool -i wifiX [-a] stopscan
```

This captures 1000 samples of raw data and writes them into a file called ‘outFile’ in the current working directory. Each sample is presented in one line of output. In the case of 160/80+80 MHz, there are two lines of output, one for each 80 MHz segment sample.In the case of 165/restricted 80+80 MHz there are three lines of output, one for each 80 MHz segment and one for the additional 5 MHz span in between. The format of each line is as follows:

1st value: Sample sequence number in file; will be the same for both 80 MHz segments, in the case of 160/80+80 MHz.

2nd value: Number of FFT bins (say ‘n’); will be the same for both 80 MHz segments, in the case of 160/80+80 MHz. For 165/restricted 80+80 mode number of FFT bins may be different for primary and secondary segments for report mode 3.

Next n bytes are the FFT bin values.

The next value is the 64 bit timestamp at which the sample was captured. Will be the same for both 80 MHz segments, in the case of 160/80+80 MHz. This is followed by RSSI, noise floor, AGC

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

gain, gain change bit, pri80 indication, raw timestamp, timestamp WAR offset, raw timestamp of previous report, reset delay and reset counter value.

The line representing the additional 5 MHz FFT bins have values only upto the n FFT bins.

NOTE The -x argument for scaling is available only with the get_samples command documented in *Section 6.5.3, Spectral debug enhancements*. If it is passed to raw_data, it will be ignored.

6.5.3 Spectral debug enhancements

The following functional and debug enhancements are made to Spectral Analysis feature to enhance the usability and troubleshooting capability.

- Configurable number FFT reports logging using spectraltool—Number of FFT reports to logged is made configurable (from fixed 1000 samples) between 1 to Max, where Max is decided by user, based on the available free memory..

  Usage: `#spectraltool -i wifiX [-a] get_samples 1024`

  Optional arguments after number of samples:

  -l <separation_character>

  Where separation_character is a single character to be used to separate values in output (e.g. ',' for comma). By default, space is used.

  -x <0/1>

  To disable (0) or enable (1) generation III linear bin format scaling (default: disabled). This is not applicable for other generations. This should be used only for linear format.

  Example, to enable scaling

  Usage: `#spectraltool -i wifi0 [-a] get_samples 1024 -x 1`

  NOTE Because spectraltool is a raw tool, it will currently not reject an enable if dBm format is used on generation III (to enable debugging of scenarios such as incorrect programmatic application of scaling in incorrect formats). But the results with dBm are undefined and not applicable.

  Note that this command explicitly starts and stops Spectral scan.

- Enhance spectraltool to support low configuration system—Spectraltool used to fail on low configuration system, owing to the limitation on buffers for FFT reports. This limitation is now resolved, which causes the spectraltool to function in low configuration platforms.
- Enhance the logging formats for spectraltool—Spectraltool dumps the FFT reports in plain text format. Now option is added to dump into CSV format

  Usage: `#spectraltool -i wifiX get_samples 1024 -l ','`
- Implement timeout feature for athssd and spectraltool—Applications such as athssd and spectraltool are in closed loop while receiving the FFT reports from HW. If HW fails to generate FFT reports, then these applications are blocked and do not respond. To recover from

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

this deadlock, timeout feature is implemented in athssd and spectraltool. If the FFT reports fail for any reason to arrive at application layer, these will timeout and give control to the user.

- Support to enable/disable the spectral direct DMA buffer debug at runtime (Disabled by default).

  Usage: #spectraltool -i wifiX dma_buff_debug <1/0>

- Support to enable/disable the spectral direct DMA ring debug at runtime (Disabled by default).

  Usage: #spectraltool -i wifiX dma_ring_debug <1/0>

- Support to enable/disable the spectral direct DMA buffer debug from the buffers creation time (Disabled by default).

  Usage: #uci set wireless.qcawifi=qca-wifi

  #uci set wireless.qcawifi.poison_spectral_bufs=<1/0>

  # uci commit wireless

- Spectraltool -i wifiX dma_ring_debug <1/0>

  Enables/Disables Spectral DMA ring debug at run time (Disabled by default)

- Spectraltool -i wifiX dma_buff_debug <1/0>

  Enables/Disables Spectral DMA buffer debug at run time (Disabled by default)

NOTE Runtime configuration *Spectraltool -i wifiX dma_buff_debug* poisons only those buffers which are replenished after the command is issued whereas the INI *poison_spectral_bufs* poisons all the buffers from their creation time.

6.6 Intelligent channel manager parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Intelligent Channel Manager* section of the *Advanced channel selection capabilities* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

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 standalone version here because since the latter (as a server) functionality is currently for Qualcomm internal use and is currently not supported.

Since ICM has functionality similar to ACS, the configuration settings for channel, hwmode and htmode are similar to those for ACS. The hwmode and htmode allow an initial desired PHY mode to be provided by the driver to ICM, and this mode is necessary for ICM to function. ‘auto’ value for hwmode, htmode (and desired PHY mode) are currently not supported by ICM.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Starting with QCA_Networking_2020.SPF.11.3 release, the ext_ifu_acs option is removed and is not available for use. Usage of this option does not take effect. Instead, the following global options, which have previously existed, can be used to enable ICM globally across all radios. ICM can be enabled and disabled using these UCI options that internally use INI settings.

```
uci set wireless.icm=icm
uci set wireless.icm.enable=1
```

Per-VAP configurability is being removed to prevent race conditions between ICM and ACS due to misconfiguration where some VAPs on a given radio have ext_ifu_acs=1, and some VAPs on same radio have ext_ifu_acs=0.

Whenever a transition is made between ICM enabled state to disabled state or vice-versa, enter the `wifi load` command for the settings to take effect correctly, which is required for all INI-based settings.

ICM should be enabled at wireless level, i.e.

```
uci set wireless.icm=icm
uci set wireless.icm.enable=1
uci set wireless.icm.mode=standalone
```

Since ICM also interacts with DCS, the DCS settings apply as-is.

- Until QCA_Networking_2020.SPF.11.2 release, ICM can be enabled per VAP using the cfg80211tool athX ext_ifu_acs 1 option and the status can be viewed using the get_ext_ifu_acs option. For such a configuration, the option *ext_ifu_acs* should be set to 1 for each interface (i.e. VAP), e.g.

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

6.6.1 Enabling selection debug information

ICM provides selection debug information by default in its logs. These logs can be accessed using 'logread | grep icm'. If it is desired to increase the log size, then option log_size in system configuration can be set to a higher value, for example:

uci set system.@system[0].log_size='1024'

Apply due caution when increasing the log size, depending upon space available on board.

In case it is desired to view additional debug information in a CSV file pertaining to the selection process (e.g. number of APs on every channel, Noise floor, Noise floor threshold, presence of various interferer's, etc.), then the following configuration can be used:

uci set wireless.icm.seldebug=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.

Here too, apply due caution when enabling, depending upon space available on the board.

6.6.2 ICM command line parameters (debugging only)

ICM is intended to be invoked from a bring-up script such as UCI based scripts. The command line parameters should not be used directly. However these may be used for QCA internal debugging purposes.

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

Table 6-6 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 nominal noise floor. The default is disable.
-h	icm –h	Display help
-n	icm -n	Enable usage of noise floor in channel selection
	Icm -r *val*	Policy regarding usage of representative Tx power in channel selection (if available from driver) 0: Do not use 1: Use representative Tx power, optimize for throughput 2: Use representative Tx power, optimize for range The default value is currently 0.
-b	icm -b *val*	Enable(1)/disable(0) preference for U-NII-3 band for 802.11ax (if 802.11ax support is available on the system) The default value is currently 0. Should be set to 1 only if Representative Tx power usage is set to non-zero value, and is available on driver.
-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 other purposes. It is mentioned here for completeness. It is disabled by default.
-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

Table 6-6 ICM command line parameters

Parameter	Format	Description	
-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
-c	icm -c	Enter icm -c cfg for cfg80211tool mode. Enter icm -c ioctl for WEXT mode. The default is cfg80211 mode. WEXT mode is now deprecated.	

6.7 Thermal mitigation

For more information on usage guidelines and the functionality regarding these parameters, see the *Thermal mitigation for offload chipsets* section of the *Power management techniques* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Thermal mitigation supports two ways for changing the thermal configuration.

6.7.1 Thermal tool

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

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

-set - Specifies set operation

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

-i - Interface name wifi0 or wifi1

-e - 1: enable, 0: disable

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

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

-dc - Duty cycle in milliseconds

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

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

-loN - Low threshold for level N

-hiN - High threshold for level N

-offN - Tx Off percentage for level N

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

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

Uses

Set operation:

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

Get operation:

```
#thermaltool –i wifiN -get
```

6.7.2 SYSFS entries

WLAN driver provides the following SYSFS entries:

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

mode

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

temp

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

Qualcomm Confidential – May Contain Trade Secrets 2022-05-13 03:57:44 PDT pan.yanfang@zte.com.cn

thlvl/off

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

dc

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

Default values

Execute the following command for reading the default values:

```
#thermaltool –i wifiN –get
```

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

6.8 Using miscellaneous commands for cfg80211 mode: examples

For cfg80211 based drivers, no generic applications such as iwpriv or iwlist are present to configure private commands. Therefore Qualcomm Technologies developed an nl80211-based cfg80211tool to configure private commands. Input to the cfg80211tool is aligned with existing iwpriv commands; for example, command form and the number of arguments are as same as existing iwpriv commands.

Miscellaneous commands, such as radartool, spectraltool, apstats, and athstats commands, can be configured for cfg80211 mode by adding the **-n** option. thermaltool can be configured for cfg80211 mode by adding the **-g** option. However, the addition of the **-n** option or **-g** option to configure these commands in cfg80211 mode is optional. This optional behavior is represented by enclosing the **-n** option and **-g** option in square brackets in the following tables.

NOTE If a certain proprietary wireless tool command, such as `radartool`, `spectraltool` or `thermaltool`, is entered, the driver checks the mode that is being currently used—WEXT or cfg80211—on the system, and processes the command accordingly.

radartool

Add the new option **–n** to the regular arguments to support cfg80211-based drivers.

WEXT command	cfg80211 command
radartool wifiN <command> < arguments>	radartool wifiN <command> < arguments > [-n]

spectraltool

Add the new option **–n** to the regular arguments to support cfg80211-based drivers.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

WEXT command	cfg80211 command
spectraltool wifiN <command> < arguments>	spectraltool wifiN <command> < arguments > [-n]

athssd

Add the new option **–n** to the regular arguments to support cfg80211-based drivers.

WEXT command	cfg80211 command
athssd -i wifiN <command> < arguments>	athssd -i wifiN <command> < arguments > [-n]

thermaltool

Add the new option **–g** to the regular arguments to support cfg80211-based drivers.

WEXT command	cfg80211 command
thermaltool wifiN <command> < arguments>	thermaltool wifiN <command> < arguments > [-g]

apstats/athstats

Add the new option **–n** to the regular arguments to support cfg80211-based drivers.

WEXT command	cfg80211 command
`apstats [Level] [[-i interface_name] | [-m STA_MAC_Address]] [-R]`	`apstats [Level] [[-i interface_name] | [-m STA_MAC_Address]] [-R] [-n]`

VAP creation and deletion

Use the generic **iw** tool to create or delete VAPs:

Create a VAP	`iw phy phyN interface add <athN>  type <__ap/managed/…>`
Delete a VAP	`iw dev <athN>  del`

6.9 Packet log analysis for IPQ807x/IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx/QCN90xx or QCN602x + IPQ807xA/IPQ60xx/IPQ50xx devices

For more information on usage guidelines and the functionality regarding these parameters, see the *Packet log analysis for IPQ807x/IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx/QCN90xx or QCN602x + IPQ807xA/IPQ60xx/IPQ50xx devices* section of the *Using logs and diagnostic tools*

for debugging chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

The packet logging utility enables monitoring and troubleshooting operations.

Run the decoder script from:

```
<workspace>wlan_proc/wlan/fw/target/mac_core/src/pktlog
```

6.9.1 Prerequisites

bash_prompt> cpan App::cpanminus

bash_prompt> ~/perl5/bin/cpanm Convert::Binary::C

bash_prompt> ~/perl5/bin/cpanm Data::Dumper

bash_prompt> ~/perl5/bin/cpanm Data::Hexdumper

bash_prompt> ~/perl5/bin/cpanm --force Tie::Hash::Indexed (use --force only for this module)

Detailed instruction will be present in wlan_proc/wlan/fw/target/mac_core/src/pktlog/README.txt

For each new session, export the PATH to the libraries to run the decoder script or export them to .bashrc

export PERL5LIB=~/perl5/lib/perl5

NOTE If an error related to sys/cdefs.h missing is seen when running the decoder the very first time, install the following library: sudo apt-get install g++-multilib

6.9.2 pktlog-lite

The pktlog-lite application is a lighter version of pktlog. This can be used to capture PPDU statistics. Use the following command to enable pktlog.

```
pktlogconf -a <radio interface> -elite -s <size of the pktlog buffer in
bytes>
Example: pktlogconf -a wifi0 -elite -s 10000000
```

After pktlog is enabled, it starts capturing Tx and Rx PPDU stats. Data is captured at the following location:

```
/proc/ath_pktlog/<radio interface>
```

Use the following command to disable pktlog:

```
pktlogconf -d <radio interface>
```

Copy packets into .dat file to be used by decoder script.

```
cp /proc/ath_pktlog/wifi0  <file name>.dat
```

To decode pktlog data, perform the following:

1. Set up the libraries and tools for decoder as in README.txt (inside <fw_ws>/wlan_proc/wlan/fw/target/mac_core/src/pktlog)
2. Decode and store raw data, which is used for debugging (run from within firmware workspace):

   ```
   perl pktlogdecoder_lithium.pl -i <input dat file> -o <output txt file>
   ```

3. Decode and save data in a readable format:

   ```
   perl pktlogdecoder_lithium.pl -i <input dat file> -o <output txt file> -f
   ```

6.9.3 pktlog-full

The pktlog-full application is used to capture all PPDU and MPDU TLVs. Use the following command to enable pktlog for Tx entries:

```
pktlogconf -a <radio interface> -etx
```

Use the following command to enable pktlog for Rx entries:

```
pktlogconf -a <radio interface> -erx
```

After pktlog is enabled, data is captured at the following location:

```
/proc/ath_pktlog/<radio interface>
```

Use the following command to disable pktlog:

```
pktlogconf -d <radio interface>
```

Copy packets into .dat file to be used by parser script.

```
cp /proc/ath_pktlog/wifi0  <file name>.dat
```

To decode pktlog data, do the following:

1. Set up the libraries and tools for decoder as in README.txt (inside <fw_ws>/wlan_proc/wlan/fw/target/mac_core/src/pktlog)
2. Decode and store raw data, which used for debugging (run from within firmware workspace):

   ```
   perl pktlogdecoder_lithium.pl -i <input dat file> -o <output txt file>
   ```

3. Decode and save data in a readable format:

   ```
   perl pktlogdecoder_lithium.pl -i <input dat file> -o <output txt file> -f
   ```

NOTE Currently, pktlog-lite utility is functional.

6.10 Channel change and CSA support for cfg80211

For more information on usage guidelines and the functionality regarding these parameters, see the *Channel change and CSA support for cfg80211* section of the *Core channel selection capabilities* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Channel-switch callback and channel switch announcements (CSA) are implemented, and this mechanism is used to change channels in cfg80211 mode. Using hostap_cli, set channel and CSA

so that hostapd identifies the channel that is used in the WLAN host driver. Alternatively, use the vendor command for channel-change only and not for CSA.

6.10.1 Method 1: Using hostapd_cli

Send CSA command on VAP interface to change in cfg80211 using hostpad_cli.

Command format:

```
hostapd_cli -i <athX> -p /var/run/hostapd-wifi0/ chan_switch  <cs_count>
<freq>  [bandwidth=20/40/80/160 ]  [ht/vht] [center_freq1=] [center_
freq2=]
```

The parameters are described as follows:

- cs_count—N; TBTT value.
  - N=0; change channel immediately (such as iwconfig ath0 <channel>)
  - N > 0; change channel with CSA
- freq—New frequency must be configured.
- bandwidth—This value can be 20/40/80/160.

  Whenever it is necessary to configure freq/channel only, specify the bandwidth (bw) based on current PHY mode (that is, ACVH80, specify bw as 80, and for NA20, specify bw as 20).
- ht/vht—Specify HT or VHT based on current PHY mode.
- center_freq1—Center frequency secondary channel.
- center_freq2—For 80+80 only and for other modes, it must be zero.

NOTE Bandwidth and center_freq1 are interlinked; based on bandwidth, configure center_freq1 and center_freq2.

Use the **wlanconfig athX list channel -cfg80211** command to configure center_freq1 values.

An example is as follows:

```
root@OpenWrt:/# wlanconfig ath0 list chan
Channel  36 : 5180    Mhz 11na C CU V VU V80- 42
Channel  40 : 5200    Mhz 11na C CL V VL V80- 42
Channel  44 : 5220    Mhz 11na C CU V VU V80- 42
Channel  48 : 5240    Mhz 11na C CL V VL V80- 42
Channel  52 : 5260 *^ Mhz 11na C CU V VU V80- 58
Channel  56 : 5280 *^ Mhz 11na C CL V VL V80- 58
Channel  60 : 5300 *^ Mhz 11na C CU V VU V80- 58
Channel  64 : 5320 *^ Mhz 11na C CL V VL V80- 58
Channel 100 : 5500 *^ Mhz 11na C CU V VU V80-106
Channel 104 : 5520 *^ Mhz 11na C CL V VL V80-106
Channel 108 : 5540 *^ Mhz 11na C CU V VU V80-106
Channel 112 : 5560 *^ Mhz 11na C CL V VL V80-106
Channel 116 : 5580 *^ Mhz 11na C CU V VU V80-122
Channel 120 : 5600 *^ Mhz 11na C CL V VL V80-122
Channel 124 : 5620 *^ Mhz 11na C CU V VU V80-122
Channel 128 : 5640 *^ Mhz 11na C CL V VL V80-122
Channel 132 : 5660 *^ Mhz 11na C CU V VU V80-138
Channel 136 : 5680 *^ Mhz 11na C CL V VL V80-138
Channel 140 : 5700 *^ Mhz 11na C CU V VU V80-138
Channel 144 : 5720 *^ Mhz 11na C CL V VL V80-138
Channel 149 : 5745    Mhz 11na C CU V VU V80-155
Channel 153 : 5765    Mhz 11na C CL V VL V80-155
Channel 157 : 5785    Mhz 11na C CU V VU V80-155
Channel 161 : 5805    Mhz 11na C CL V VL V80-155
Channel 165 : 5825    Mhz 11na C V
root@OpenWrt:/#
```

NOTE Starting with QCA_Networking_2020.SPF.11.3 release, due to a hostpad upgrade, the "sec_channel_offset" option for "hostapd_cli chan_switch" command is mandatory for BW greater than 20 MHz.

Example of command syntax until SPF.11.2:

```
hostapd_cli -i ath0 -p /var/run/hostapd-wifi0/ chan_switch 10 5320
bandwidth=40 vht center_freq1=5310 center_freq2=0
```

Example of command syntax starting with SPF.11.3:

```
hostapd_cli -i ath0 -p /var/run/hostapd-wifi0/ chan_switch 10 5320
bandwidth=40 vht center_freq1=5310 center_freq2=0 sec_channel_offset=-1
```

For 20 MHz, "sec_channel_offset" is not expected; +1/-1 provided for 20MHz will fail.

For 40 MHz, HT40+ requires sec_channel_offset=1, for 40- it is -1.

For 80 MHz and 160 MHz, sec_channel_offset should be 1.

- To configure channel 36 in VHT80 mode:

  hostapd_cli -i ath0 -p /var/run/hostapd-wifi0/ chan_switch 0 **5180** bandwidth=**80** vht center_freq1=**5210 sec_channel_offset=1 (Channel 42 is center freq; channels 36, 40, 44, and 48 in VHT80 mode)**

- To configure channel 48 in VHT80 mode:

  hostapd_cli -i ath0 -p /var/run/hostapd-wifi0/ chan_switch 0 **5240** bandwidth=**80** vht center_freq1=**5210 sec_channel_offset=1**

- To configure channel 36 in HT/VHT40 mode:

  hostapd_cli -i ath0 -p /var/run/hostapd-wifi0/ chan_switch 0 **5180** bandwidth=**40** vht center_freq1=**5190 sec_channel_offset=1 (Based on wlanconfig athX list chan – it can be +10/-10)**

- To configure channel 48 in VHT80 mode:

  hostapd_cli -i ath0 -p /var/run/hostapd-wifi0/ chan_switch 0 **5240** bandwidth=**40** vht center_freq1=**5230 sec_channel_offset=1 (Based on wlanconfig athX list chan – it can be +10/-10)**

- To configure channel 48 in VHT20 mode:

  hostapd_cli -i ath0 -p /var/run/hostapd-wifi0/ chan_switch 0 **5240** bandwidth=**40** vht center_freq1=**5240 (freq and center_freq1 must be the same)**

- For legacy and HT20 modes, center_freq1 can be ignored.

6.10.2 Method 2: Using vendor command (for channel change only and without CSA)

1. Use `cfg8021tool athX channel <channel number>` command (same as iwconfig) to configure channel.
2. Enter the `hostapd_cli -i ath0 -p /var/run/hostapd-wifi0/ update_beacon` command to update channel information in beacon.

6.11 Support for beacon count and beacon outage statistics

For more information on usage guidelines and the functionality regarding these parameters, see the *Support for beacon count and beacon outage statistics* section of the *Using statistics for debugging* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Successful beacon transmission and beacon outage statistics are received through the WMI command, WMI_HOST_REQUEST_BCN_STAT. The ol_ath_get_vdev_bcn_stats() module is used to receive VAP statistics, which is defined in the VAP function pointer.

vap->get_vdev_bcn_stats = ol_ath_get_vdev_bcn_stats;

The following two beacon statistical details are received from firmware:

u_int32_t tx_bcn_succ_cnt;

u_int32_t tx_bcn_outage_cnt

Use the following apstats command to receive and display the beacon success count and beacon outage statistics.

```
# apstats -v -i <interface_name>
```

6.12 exttool parameters

For more information on usage guidelines and the functionality regarding these parameters, see the *Cloud-based channel change* section of the *Advanced channel selection capabilities* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

- **Command usage**: exttool --scan --interface wifi1 --mindwell <milliseconds> --maxdwell <milliseconds> --resttime <milliseconds> --scanmode <1: ACTIVE, 2: PASSIVE> --chcount < Channel count and chan number > --maxscantime <milliseconds> --idletime <milliseconds>

  Configure channel scanning, and specify settings, such as dwell time, scanning time, rest time, and idle time.

  - mindwell—Minimum time, in milliseconds, to spend on each channel even if channel is idle. The allowed range is 50 ms to less than 100 ms.
  - maxdwell—Maximum time, in milliseconds, that can be spent on a channel. Default value is 300 msec. The value to be set must be greater than or equal to mindwell. Therefore, check the value of mindwell and choose a value accordingly; else, the command returns error. The allowed range is between any value higher than min dwell and 10000ms.
  - resttime—Amount of time, in milliseconds, for which FW waits or rests before triggering the scan.
  - scanmode—Scan mode. Enter 1 for active scan, or 2 for passive scan.
  - chcount—Channel count and channel number. The range is from 1 to 32.
  - maxscantime—Total amount of time, in milliseconds, allotted for scanning all the channels entered using the chcount option in the exttool command. After this maximum scanning duration is exceeded, the scan is canceled. For example, if six channels are

specified using the chcount option to scan and maxscantime is entered as 2000 ms, and if FW is able to scan only four channels in this time period based on other parameters, then the scan is canceled after the maximum scanning time is exceeded,. Therefore, the maxscantime value must be sufficiently hight to enable scanning to be completed on all channels.

- idletime—Amount of time, in milliseconds, for which FW checks whether the radio is idle for this amount of time before triggering the scan. The scanning moves to off-channel for scanning operations, only if there is no traffic on the radio for the idle time; otherwise, scanning is not triggered. The idletime value must be greater than 50 ms.

NOTE Scan will be started after either “idletime” or “resttime”, depending on whichever value is less. If idletime is less than resttime, the scan is started after the idletime if the interface has no activity (so that traffic on radio is not affected). Otherwise, scan is forcibly started after resttime.

- **Command usage**: exttool -i wifiX -p -a <Chan> -q <SSID>

  In repeater AP, run this command for STA VAP interface. Channel switch announcement will take place and repeater AP will move to a new channel.

- **Command usage**: exttool --chanswitch --interface wifiX --chan --chwidth --numcsa --secoffset <1: PLUS, 3: MINUS> --cfreq2

  Configure channel switching, and specify settings such as channel width, number of channel switch announcements permitted, secondary channel offset, and center channel frequency.

  - chan—Number of channels to scan. User needs to set the number of channel he wants to scan at one request. It is expected that the value always greater than 0 and less than maximum channels defined for IEEE channel. Starting the channel scan will return failure for invalid number of channels.
  - chanswitch—Trigger 802.11h channel switch mechanism. Forces the AP to perform a channel change, and forces a channel change announcement message.
  - chwidth—Channel width. Destination channel width: 0: 20 MHz, 1: 40 MHz, 2: 80 MHz, 3: 160 MHz, 4: 80 + 80 MHz.
  - numcsa—Number of channel switch announcements to send before channel change. If num_csa is specified as 0, default count will be used and as of now its 5. It can be changed any time by changing macro IEEE80211_DEFAULT_CHANSWITCH_COUNT.
  - secoffset—Secondary channel offset, either plus (+) or minus (-). Secondary 20 channel is above or below primary 20 channel.
  - cfreq2—Center channel frequency for secondary 80 channel. Used only for 80 + 80 MHz. Otherwise, it must be set as 0.

  Using the command, *exttool --chanswitch --interface wifiX --chan <dest primary channel> --chwidth <channelwidth> --numcsa <channel switch announcement count> --secoffset <1: PLUS, 3: MINUS> --cfreq2 <secondary 80 centre channel freq, applicable only for 80+80 MHz>--cfg80211*, initiate a CSA and move the AP to another channel bandwidth. After the AP channel width configuration, this node structure is accessed to reveal the capabilities of the peer in an iterative manner (for all connected peers) so as to check which of the peers are capable of switching to the new bandwidth.

Starting with QCA_Networking_2021.SPF.11.4 release, an option [--force, -j] is introduced in exttool command to force CSA if destination channel parameters are same as current channel parameters.

Example: exttool --chanswitch --interface wifi2 --chan 100 --chwidth 1 --band 2 --numcsa 5 --secoffset 1 --cfg80211 --force;

- **Command usage**: exttool --rptrmove --interface wifiX --chan --ssid --bssid --numcsa --cfreq2

  Configure the movement of repeater or range extender, and specify settings such as SSID, BSSID, number of channel switch announcements permitted, secondary channel offset, and center channel frequency.

  - chan—Number of channels to scan. User needs to set the number of channel he wants to scan at one request. It is expected that the value always greater than 0 and less than maximum channels defined for IEEE channel. Starting the channel scan will return failure for invalid number of channels.
  - rptrmove—Trigger movement of repeater AP or range extender.
  - ssid—Service set ID (SSID) of the AP specified in the xx:xx:xx:xx:xx:xx format.
  - bssid—Basic service set identifier of AP. BSSID specified in the xx:xx:xx:xx:xx:xx format in the beacon report request. Instead of a wildcard BSSID, a specific BSSID is defined.
  - numcsa—Number of channel switch announcements to send before channel change. If num_csa is specified as 0, default count will be used and as of now its 5. It can be changed any time by changing macro IEEE80211_DEFAULT_CHANSWITCH_COUNT.
  - cfreq2—Center channel frequency for secondary 80 channel. Used only for 80 + 80 MHz. Otherwise, it must be set as 0.

- **Command usage**: exttool --list_chan_state --interface wifiX

  Starting with QCA_Networking_2021.SPF.11.4 release, this command is introduced to list the current DFS state of all the channels in the channel list of the current country.

- **Command usage**: exttool --list_chan_info --interface wifiX

  Starting with QCA_Networking_2021.SPF.11.4 release, this command is introduced to list all the channel information of all the channels of the cuurent country.

6.12.1 Changes to existing exttool commands for 6 GHz

- exttool --chanswitch --interface wifi0 [--band 2] --chan 100 --chwidth 3 --numcsa 7
- exttool --scan --interface wifi0 --mindwell 50 --maxdwell 200 [--band 2] --maxscantime 200 --resttime 100 --scanmode 2 --idletime 50 --chcount 3 44 48 52
- exttool --rptrmove --interface wifi0 [--band 2] --chan 36 --ssid OPTDBG --numcsa 8

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

6.13 Manual VAP creation and deletion in cfg80211

For more information on usage guidelines and the functionality regarding these parameters, see the *Manual VAP creation and deletion in cfg80211* section of the *WLAN AP mode*s chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Make sure that all driver modules are loaded.

```
wifi detect > /etc/config/wireless
```

6.13.1 Create interface

1. Create interface.

```
wlanconfig athN create wlandev wifiN wlanmode ap -cfg80211
iw dev wifiN interface add athN type __ap
ifconfig wifiN up
```

2. Configure interface for channel , mode, and other necessary proprietary settings.

```
cfg80211tool athN channel 6
cfg80211tool athN mode 11NGHT40
cfg80211tool athN disablecoext 1
cfg80211tool athN wds 1
```

3. Prepare hostapd configuration and https://w1.fi/cgit/hostap/plain/hostapd/hostapd.conf.

```
cat hostapd-ath1.conf
      driver=nl80211
      interface=ath1
      logger_syslog=127
      logger_syslog_level=2
      logger_stdout=127
      logger_stdout_level=2
      channel=6
      ieee80211n=1
      hw_mode=g
      wmm_enabled=1
      dtim_period=1
      ignore_broadcast_ssid=0
      ctrl_interface=/var/run/hostapd-wifi1
      wpa=0
      ssid=cfg80211
      bridge=br-lan
```

4. Start Hostapd to bringup VAP.

```
      wpa_cli -g /var/run/hostapd/global raw ADD bss_
config=athN:/root/hostapd-athN.conf
```

6.13.2 Delete interface

1. Stop hostapd.

```
wpa_cli -g /var/run/hostapd/global raw REMOVE athN
```

2. Delete interface.

```
iw athN del
```

Qualcomm Confidential – May Contain Trade Secrets 2023-12-13 03:57:44 PDT pan.yifeng@zte.com.cn

6.14 Monitor mode setup

For more information on usage guidelines and the functionality regarding these parameters, see the *Set up monitor mode on IPQ807x/IPQ807xA/IPQ817x/IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx/QCN90xx or QCN602x + IPQ807xA/IPQ60xx/IPQ50xx platforms* section of the *Using logs and diagnostic tools for debugging* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

The radiotag sniffer packet is forwarded to interface when VAP is created as monitor mode VAP. The monitor mode can be created alone or created with other VAP in multi VAP configuration.

The command to create and bring up a monitor VAP is:

```
wlanconfig athx create wlandev wifiX wlanmode monitor ifconfig athx up
```

The tcpdump is used to collect packet on monitor mode interface. The command to collect the radiotag sniffer packet is as follows:

```
tcpdump -i athx -s 0 -w <filename>.pcap
```

How to set up IPQ807x AP as 802.11ax sniffer

On AP:

Set up Monitor mode

edit AP config file (vi /etc/config/wireless)

add the monitor vap, disable all other AP vaps

run "wifi" to apply change

Example for 5G monitor

```
config wifi-iface
option device   wifi0
option network  lan
option mode     monitor
option disabled 0

config wifi-iface
option device   wifi0
option network  lan
option mode     ap
option ssid     OpenWrt_11AXA
option encryption none
option disabled 1
```

On Linux PC (attached to AP via LAN Cable)

Terminal #1

$mkfifo ~/pipe

$ssh root@192.168.1.1 "tcpdump -i ath0 -s 0 -U -w -" > ~/pipe (192.168.1.1 is the AP's IP, ath0 is the monitor VAP)

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

Terminal #2

$sudo wireshark -k -i ~/pipe

6.15 Global and target-specific INI files

Certain initial driver configuration can be performed through the INI files available with the Wi-Fi driver.

Types of INI files

1. **Global INI file—**Contains configurations which applies to all the target (IPQ807x/QCA9984) in the system.

   File path: /ini

2. **Target specific INI files—**Contains target specific overrides for provided configurations.

   File path: /ini/

NOTE: For a given configuration item, defined in both the global and the target specific INI, the target-specific INI value overrides the value written in global INI file.

Configure INI values

1. UCI—Few of the INI items in the global INI file can be configured through UCI by providing the values in /etc/config/wireless.

   The INI configurations through UCI applies only to the global.ini file, which is applicable for all target types in the system.

2. INI file—Any given configuration in an INI file, can be edited manually, before the Wi-Fi drivers are loaded. After the configuration items in INI files are updated and the drivers are loaded, the driver are loaded with the new configuration values.

NOTE: If a given configuration in global INI is also configured through UCI, the UCI scripts updates the INI file, as part of wifi load scripts and the manual updates will be overwritten.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

6.15.1 Sample INI file

```
# This file allows user to override the factory
# defaults for the WLAN Driver
sap_restart_on_ch_avoid=0
gEnableForceTargetAssert=1
g_bug_on_reinit_failure=0
# Enable IMPS or not
gEnableImps=1

# Enable BMPS or not
gEnableBmps=1

# Phy Mode (auto, b, g, n, etc)
# Valid values are 0-9, with 0 = Auto, 4 = 11n, 9 = 11ac
# 1 = 11abg, 2 = 11b, 3 = 11g, 5 = 11g only, 6 = 11n only
# 7 = 11b only 8 = 11ac only.
gDot11Mode=0

# Assigned MAC Addresses - This will be used until NV items are in place
# Each byte of MAC address is represented in Hex format as XX
Intf0MacAddress=000AF58989FF
Intf1MacAddress=000AF58989FE
Intf2MacAddress=000AF58989FD
Intf3MacAddress=000AF58989FC

# UAPSD service interval for VO,VI, BE, BK traffic
InfraUapsdVoSrvIntv=0
InfraUapsdViSrvIntv=0
InfraUapsdBeSrvIntv=0
InfraUapsdBkSrvIntv=0

# Flag to allow STA send AddTspec even when ACM is Off
gAddTSWhenACMIsOff=1

# Flags to filter Mcast and Bcast RX packets.
# Value 0: No filtering, 1: Filter all Multicast.
# 2: Filter all Broadcast. 3: Filter all Mcast abd Bcast
McastBcastFilter=3

#Flag to enable HostARPOffload feature or not
hostArpOffload=1

#Flag to enable HostNSOffload feature or not
hostNSOffload=1

# 802.11n Protection flag
gEnableApProt=1

#Enable OBSS protection
gEnableApOBSSProt=1

#Enable/Disable UAPSD for SoftAP
gEnableApUapsd=1

# Fixed Rate
gFixedRate=0
```

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

The following are the UCI parameters that are added to INI (global.ini file) and exposed to external users for configuration:

INI item	Description
enable_11ax_stub	Enable 802.11ax stubbing support for testing. Valid only for QCA9984
enable_tx_tcp_cksum	Enable Tx TCP Checksum
vow_config	VoW Configuration
qwrap_enable	Enable QWRAP target config
cce_disable	Disable Hardware CCE Component
low_mem_system	Low Memory System
beacon_offload_disable	Beacon offload disable
enableuartprint	Enable uart/serial prints from target

INI item	Description
eapol_minrate_set	Enable/Disable EAPOL Minrate
eapol_minrate_ac_set	Set AC for the EAPOL minrate set
mesh_support	Configure Mesh support
cfg80211_config	cfg80211 config(enable/disable)
iphdr_pad	Disable IP header padding to manage IP header unalignment
lteu_support	LTEU support
bmi	BMI Handling: 0 - Driver, 1 - User agent
max_descs	Override default max descriptors
max_peers	Override default max peers
max_vdevs	Override default max vdevs
OL_ACBKMinfree	Min Free buffers reserved for AC-BK
OL_ACBEMinfree	Min Free buffers reserved for AC-BE
OL_ACVIMinfree	Min Free buffers reserved for AC-VI
OL_ACVOMinfree	Min Free buffers reserved for AC-VO
otp_mod_param	OTP
emu_type	Emulation Type : 0-->ASIC, 1-->M2M, 2-->BB
max_active_peers	Override max active peers in peer qcache
tgt_sched_params	Target Scheduler Parameters
fw_code_sign	FW Code Sign
allocram_track_max	Enable target allocram tracking
max_vaps	Max vap nodes for which mempool is statically allocated
max_clients	Max client nodes for which mempoolis statically allocated
fw_dump_options	Firmware dump options
enable_smart_antenna	Enable Smart Antenna
enable_smart_antenna_da	Enable Smart Antenna for Direct Attach
sa_validate_sw	Validate Smart Antenna Software
nss_wifi_nxthop_cfg	NSS Wi-Fi next hop configuration
nss_wifi_olcfg	NSS Wi-Fi offload configuration
nss_wifili_olcfg	NSS Wi-Fi offload configuration
spectral_disable	Spectral disable
twt_enable	TWT Enable/Disable
mbss_ie_enable	Enable MBSS IE
enable_rdk_stats	Enable RDK stats
gEnableSNRMonitoring	Enable/Disable SNR Monitoring
g_peer_del_wait_time	Timeout to print peer refs
self_obss_pd_tx_enable	SRP Non SRG OBSS PD MAX Offset Enable

INI item	Description
spatial_reuse_enable	Enable Spatial Reuse IE
allow_mon_vaps_in_sr	Allow monitor vaps to be present while using Spatial Reuse

6.16 athdiag over QMI for QCN90xx or QCN602x platforms

This section describes the use of the athdiag tool to read/write the memory addresses in IPQ60xx and IPQ807x/IPQ807xA/IPQ817x platforms that are attached with QCN90xx or QCN602x radios. athdiag is applicable only to IPQ8074.ILQ.11.2 releases that contain the support for QCN90xx or QCN602x platforms. This tool is only applicable for QCN90xx or QCN602x whereas IPQ807x/IPQ807xA/IPQ817x continue to support devmem.

The memory regions that can be read/written are as follows:

- reserved memory from DDR
- CE Registers
- System IRAM memory reserved for FW.
- All WCSS/PHY/UMAC registers etc which FW has access.

The athdiag tool is used to read/write the memory addresses and below options are primarily used for this purpose.

--get - read the memory word

--address - address of the memory word to be read/write.

--set - Write the memory word

--value - value to write in the memory address.

--wifi X - corresponding wifi interface to read/write

--count - number of words to be read.

--quiet - disable the verbose logging.

Note that -wifi is not mandatory if only one wifi interface is up.

6.16.1 Word read

athdiag --get --address <address>

<address> - address of the memory to be read

6.16.2 Word write

athdiag --set --address <address> --value <value>

<address> - address of the memory to write

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

<value> - value to write in the memory address

6.16.3 Block read

athdiag --get --address <address> --count <count>

<address> - start address of the memory to be read

<count> - number of the word to read

6.16.4 Block read and save into file

athdiag --read --address <address> --length <length> --quiet --file <filename>

<address> - start address to read

<length> - length of memory region to read in bytes

<filename> - file name to be saved.

Note that all the options [address/length/filename] are mandatory for -read option.

6.16.5 Examples of athdiag over QMI commands

Read a memory word

```
OpenWrt:/# athdiag --get --address 0xcb91028
Detected prop
Autodetected:  main Diag file
(/sys/devices/platform/soc/10000000.pci/pci0001:00/0001:00:00.0/0001:01:
00.0/athdiag)
 main Diag file
(/sys/devices/platform/soc/10000000.pci/pci0001:00/0001:00:00.0/0001:01:
00.0/athdiag) dev: 3
DIAG Read Word (address: 0xcb91028)
```

Read a memory word with quiet option

```
# athdiag --get --address 0xcb91028 --quiet
0x0249f000
```

Read a memory word from wifi 1 with quiet option

```
athdiag --get --address 0xcb91028 --wifi 1 -quiet
0x0249f000
```

Write a memory word

```
athdiag --set --address 0xcb91028 --value 0x249f001 --wifi 0
/sys/devices/platform/soc/10000000.pci/pci0001:00/0001:00:00.0/0001:01:0
0.0/net/wifi0
Autodetected:  main Diag file
(/sys/devices/platform/soc/10000000.pci/pci0001:00/0001:00:00.0/0001:01:
00.0/net/wifi0)
```

```
Reading from device
/sys/devices/platform/soc/10000000.pci/pci0001:00/0001:00:00.0/0001:01:0
0.0/
main Diag file
(/sys/devices/platform/soc/10000000.pci/pci0001:00/0001:00:00.0/0001:01:
00.0/athdiag) dev: 3
DIAG Write Word (address: 0xcb91028, param: 0x249f001)
```

Write a memory word with quiet option

```
root@OpenWrt:/#  athdiag --set --address 0xcb91028 --value 2 -quiet
root@OpenWrt:/#
```

Block read a memory region

```
OpenWrt:/# athdiag --get --address 0xcb91028 --count 4
Detected prop
Autodetected:  main Diag file
(/sys/devices/platform/soc/10000000.pci/pci0001:00/0001:00:00.0/0001:01:
00.0/athdiag)
 main Diag file
(/sys/devices/platform/soc/10000000.pci/pci0001:00/0001:00:00.0/0001:01:
00.0/athdiag) dev: 3
DIAG Read Word (address: 0xcb91028)
Value in target at 0xcb91028: 0x249f000 (38400000)
DIAG Read Word (address: 0xcb9102c)
Value in target at 0xcb9102c: 0x9600 (38400)
DIAG Read Word (address: 0xcb91030)
Value in target at 0xcb91030: 0x0 (0)
DIAG Read Word (address: 0xcb91034)
Value in target at 0xcb91034: 0x10000000 (268435456)
```

Block read with quiet option

```
/# athdiag --get --address 0x0164BB18 --count 40 --quiet
0x77cac001      0x00000000      0x00000000      0x00000000
0x00000000      0x00000000      0x00000000      0x00000000
0x00000000      0x00000000      0x00000000      0x00000000
0x00000000      0x00000000      0x00000000      0x00000000
0x00000000      0x00000000      0x00000000      0x00000000
0x00000000      0x00000000      0x00000000      0x00000000
0x00000000      0x00000000      0x00000000      0x00000000
0x00000000      0x00000000      0x00000000      0x00000000
0x00000000      0x00000000      0x00000000      0x00000000
0x00000000      0x00000000      0x00000000      0x00000000
```

Block read and save to file

```
athdiag --read --address 0x0164BB18 --length 16 --quiet --file t
ext.txt
Detected prop
root@OpenWrt:/# hexdump text.txt
0000000 c001 77ca 0000 0000 0000 0000 0000 0000
0000010
```

7 LOWI commands

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

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

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

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Location wireless interface (LOWI)* section of the *WLAN AP modes* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Qualcomm Confidential – May Contain Trade Secrets 2021-05-12 08:44 PDT pan.yafeng@zte.com.cn

7.1 Ranging measurements

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

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

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

As a minimum the following command can be issued:

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

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

7.1.1 Command line options

Following table list the command line options:

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

7.1.2 User input

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

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

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

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

<!-- Burst Duration.   Default 15.
Applicable only for double-sided RTT.
```

```
Refer to Specification 802.11 mc (D5.2)for values-->
            <burstduration>xx</burstduration>

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

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

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

            <!-- Channel number.                           -->
            <ch>xxx</ch>
<!-- MacId of the target.                        -->
            <mac>xx:xx:xx:xx:xx:xx</mac>
<!-- Interface on which RTT scan to be issued -- >
           <interface>wifiX</interface>
          </ap>
   </ranging>
</body>
```

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

7.1.3 Output files

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

Default raw result and summary files are as follows:

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

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

Console output

```
root@OpenWrt:/# lowi-test -r ap_list.xml -n 1
LOWI Test Version: LTST_1.4.18
[86693.95][LTST_1.4.18-LOWIClient] createInstance ()
[86693.95][LTST_1.4.18-LOWIUtils] inPostcardToResponse - FROM: LOWI-
SERVER, TO:   7faf6b1948-LC, RESP:  LOWI_CAPABILITY
[86693.98][LTST_1.4.18-LOWIUtils] inPostcardToResponse - FROM: LOWI-
SERVER, TO:   7faf6b1948-LC, RESP:  LOWI_RANGING_SCAN
Issued Request Type: RANGING
Summary stats: Scan Type: RANGING
Avg Response Time: Ranging: 33 ms
        AP       Chan   Detection rate   RSSI(0.5 dBm)
RTT(psec)                   RTT Meas Cnt
                                  Min  Max Avg(dBm) Avg(W)   Min
Max     Avg   #>5M   #4M   #3M   #2M   #1M   #0M
00:03:7f:75:59:17 149    1/1   (100%)  -26   -17   -24   -23   157812
498176   311805    1     0     0     0     0     0
```

```
==========Peer BSSID: 00:03:7f:75:59:17==========
Total number of measurements: 24, true distance: (0 in cm) requested BW
(2)
-----------------------------------------------
RTT stats ...(ps) from total measurements (22)
 median (310156) yield(100.0 %)
-----------------------------------------------
Distance stats...(cm) from total measurements (22)
 median(4652.3)
-----------------------------------------------
CEP stats in cm
 cep90(  0.00)
-----------------------------------------------
P2P stats in cm
 p2p90(  9.05)
=================================================
```

7.1.4 Example and test result

1. Bring up AP and WDS STA
2. Enable lowi using following command -

```
uci set wireless.lowi=lowi
uci set wireless.lowi.enable=1
uci commit
```

3. Run lowi-test with -r option to trigger RTT measurement on WDS-STA.
4. Output will be displayed on console and csv files will be generated in /usr/share/location/lowi or as requested in -o OR -s option.

7.1.5 Single sided RTT

Command line command

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

Contents of ap_list.xml file

```
<body>
    <ranging>
        <ap>
            <band>1</band>
            <rttType>2</rttType>
            <numFrames>5</numFrames>
            <burstduration>15</burstduration>
            <center_freq1>5500</center_freq1>
            <center_freq2>0</center_freq2>
            <ch>100</ch>
```

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.xifeng@zte.com.cn

```
            <mac>8c:fd:f0:01:e6:bd</mac>
            <interface>wifi0</interface>
        </ap>
    </ranging>
</body>
```

Command line output

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

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

7.1.6 Double sided RTT

Command line command

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

Contents of ap_list.xml file

```
<body>
    <ranging>
        <ap>
            <band>1</band>
            <rttType>3</rttType>
            <numFrames>5</numFrames>
            <burstduration>15</burstduration>
            <center_freq1>5500</center_freq1>
            <center_freq2>0</center_freq2>
            <ch>100</ch>
            <mac>8c:fd:f0:01:e6:bd</mac>
            <interface>wifi0</interface>
        </ap>
    </ranging>
</body>
```

Command line output

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

7.2 LCR configuration

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

To issue a LCR configuration request use following command format:

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

7.2.1 Command line options

Following table list the command line options:

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

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

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

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

7.2.2 Output files

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

7.2.3 Example and test result

Create an input file lci_lcr.xml on AP (e.g. /tmp) with following content:

```
<body>
<interface>wifi0</interface>
<lcr_info>
<country_code>US</country_code>
<civic_address>020E0102030405060708090A0B0C0D0E</civic_address>
</lcr_info>
<lci_info>
<latitude>37</latitude>
<longitude>12</longitude>
<altitude>7</altitude>
<latitude_unc>61</latitude_unc>
<longitude_unc>6</longitude_unc>
<altitude_unc>59</altitude_unc>
<motion_pattern>0</motion_pattern>
<floor>3</floor>
<height_above_floor>2</height_above_floor>
<height_unc>77</height_unc>
</lci_info>
</body>
```

Command line command

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

Contents of lcr.xml file

```
<lcr_info>
<country_code>US</country_code>        <civic_
address>0102030405060708090a0b0c0d0e0f101112131415161718191a1b1c1d1e1f20
212223242526272829 2a2b2c2d2e2f303132333435363738393a3b3c3d3e3f4041424344
45464748494a4b4c4d4e4f505152535455565758595a5b5c5d5e5f606162636465666768
696a6b6c6d6e6f707172737475767778797a7b7c7d7e7f808182838485868788898a8b8c
8d8e8f909192939495969798999a9b9c9d9e9f000102030405060708 09aaabacadae</ci
vic_address>
</lcr_info>
```

Command line output

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

Qualcomm Confidential – May Contain Trade Secrets 2022-05-13 03:57:41 PDT pan.yafeng@zte.com.cn

7.3 LCI configuration

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

To issue a LCI configuration request use following command format:

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

7.3.1 Command line options

Following table list the command line options:

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

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

When LCI input is in hex bytes format:

```
<lci_info>
    <!-- Latitude in the units of 1/(2^25) degrees  -->
    <latitude>xxxx</latitude>

    <!-- Longitude in the units of 1/(2^25) degrees-->
    <longitude>xxxxx</longitude>

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

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

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

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

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

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

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

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

```
</lci_info>
```

7.3.2 Output files

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

7.3.3 Example and test result

Create a sample input file lci_lcr.xml on AP (e.g. /tmp) with following content:

```
<body>
<interface>wifi0</interface>
<lcr_info>
<country_code>US</country_code>
<civic_address>020E0102030405060708090A0B0C0D0E</civic_address>
</lcr_info>
<lci_info>
<latitude>37</latitude>
<longitude>12</longitude>
<altitude>7</altitude>
<latitude_unc>61</latitude_unc>
<longitude_unc>6</longitude_unc>
<altitude_unc>59</altitude_unc>
<motion_pattern>0</motion_pattern>
<floor>3</floor>
<height_above_floor>2</height_above_floor>
<height_unc>77</height_unc>
</lci_info>
</body>
```

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

Command line command

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

Contents of lci.xml file

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

Command line output

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

Issue Ranging request with <lci> 1 </lci> option on wds-sta.

Verify that LCI information is properly displayed in sniffer.

7.4 Where are you (LCI request)

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

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

7.4.1 Command line options

Following table list the command line options:

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

7.4.2 Output files

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

7.4.3 Example and test results

Associate WDS STA with an AP.

Command line command

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

Command line output:

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

Where Are You (LCI Request) frame in sniffer is as follows:

```
▸ Frame 701: 57 bytes on wire (456 bits), 57 bytes captured (456 bits) on interface 0
▸ Radiotap Header v0, Length 18
▸ 802.11 radio information
▸ IEEE 802.11 Action, Flags: ........C
▾ IEEE 802.11 wireless LAN management frame
  ▾ Fixed parameters
      Category code: Radio Measurement (5)
      Action code: Radio Measurement Request (0)
      Dialog token: 1
      Repetitions: 0
  ▾ Tagged parameters (6 bytes)
    ▾ Tag: Measurement Request
        Tag Number: Measurement Request (38)
        Tag length: 4
        0000 0001 = Measurement Token: 0x01
      ▸ 0000 0000 = Measurement Request Mode: 0x00
      ▾ Measurement Request Type: Location Configuration Indication (LCI) Request (0x08)
        ▾ Unknown Data: 01
```

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

7.5 Fine Timing Measurement Range Request (FTMRR)

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

To issue a FTM range request use following command format:

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

7.5.1 Command line options

Following table list the command line options:

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

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

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

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

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

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

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

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

        <!-- Operating Class indicates the channel set of the AP
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
            indicated by this BSSID. Valid Operating classes are listed in
            Specification 802.11 mc (D5.2) Annex E -->
           <op_class>xx</op_class>

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

7.5.2 Output files

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

7.5.3 Example and test result

Command line command

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

Contents of ftmrr.xml file

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

Command line output

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

The FTMRR action frame is captured on sniffer as shown here.

```
▸ Frame 506: 103 bytes on wire (824 bits), 103 bytes captured (824 bits) on interface 0
▸ Radiotap Header v0, Length 18
▸ 802.11 radio information
▸ IEEE 802.11 Action, Flags: ........C
▾ IEEE 802.11 wireless LAN management frame
  ▾ Fixed parameters
      Category code: Radio Measurement (5)
      Action code: Radio Measurement Request (0)
      Dialog token: 2
      Repetitions: 0
  ▾ Tagged parameters (52 bytes)
    ▾ Tag: Measurement Request
        Tag Number: Measurement Request (38)
        Tag length: 50
        0000 0010 = Measurement Token: 0x02
      ▸ 0000 0000 = Measurement Request Mode: 0x00
      ▾ Measurement Request Type: Fine Timing Measurement Range (0x10)
          Randomization Interval: 0x0014
          Minimum AP Count: 1
          Neighbor report: 52
          Length: 18
          BSSID: 8c:fd:f0:07:1b:dd (8c:fd:f0:07:1b:dd)
        ▸ BSSID Information: 0x00001cc9
          Operating Class: 128
          Channel Number: 149 (iterative measurements on that Channel Number)
```

7.5.4 References

For details, refer the IEEE 802.11 mc specifications.

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

8 UCI commands

For details of unified configuration interface (UCI), visit http://wiki.openwrt.org/doc/uci. QSDK supports qca-wifi driver natively. The UCI database section to configure it is called ‘wireless’. It can be accessed using the following command:

```
uci show wireless
```

This command shows the whole wireless section and it is organized in the following subsections:

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

The driver will create the wifiN interface at initialization 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 are called: ath0, ath01, ath02, ath03...
- For radio 1, the vap network devices are 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.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

8.1 Per-VAP configuration parameters

These parameters must be set in the wireless.wifi-iface[...] section corresponding to the VAP that must be configured. The configured parameters are applied to this specific VAP.

Table 8-1 Per VAP configuration parameters

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

8.2 Example UCI configuration for WLAN AP modes

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *WLAN AP modes* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

802.11ac open mode

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

802.11ac open mode 11ACVHT80+80 operating mode

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

802.11ac WPA2-PSK

```
uci set wireless.wifi0=wifi-device
 uci set wireless.wifi0.type=qcawifi
 uci set wireless.wifi0.macaddr=00:60:02:00:c9:c9
 uci set wireless.wifi0.hwmode=11ac
 uci set wireless.wifi0.disabled=0
 uci set wireless.wifi0.htmode=HT80
 uci set wireless.wifi0.channel=100
```

```
uci set wireless.wifi0.txchainmask=15
uci set wireless.wifi0.rxchainmask=15
uci set wireless.wifi0.mode=ap
uci set wireless.@wifi-iface[0]=wifi-iface
uci set wireless.@wifi-iface[0].device=wifi0
uci set wireless.@wifi-iface[0].network=lan
uci set wireless.@wifi-iface[0].mode=ap
uci set wireless.@wifi-iface[0].ssid=5g_wpa2
uci set wireless.@wifi-iface[0].encryption=psk2+ccmp
uci set wireless.@wifi-iface[0].key=12345678
uci commit wireless
```

For other WPA2 security modes replace encryption type

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

For WPA-PSK security mode replace encryption type

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

For PMF (protected management frames) enabled AP

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

For enabling BIP CMAC/GMAC

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

WEP security configuration

```
uci set wireless.wifi0=wifi-device
uci set wireless.wifi0.type=qcawifi
uci set wireless.wifi0.macaddr=00:60:02:00:c9:c9
uci set wireless.wifi0.hwmode=11ac
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
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
```

5 GHz 802.11ax 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=11axa
uci set wireless.wifi0.disabled=0
uci set wireless.wifi0.htmode=HT80
uci set wireless.wifi0.channel=36
uci set wireless.wifi0.txchainmask=15
uci set wireless.wifi0.rxchainmask=15
uci set wireless.wifi0.mode=ap
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
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
```

2.4 GHz 802.11ax open mode

```
uci set wireless.wifi1=wifi-device
uci set wireless.wifi1.type=qcawifi
uci set wireless.wifi1.macaddr=00:60:02:00:c9:c8
uci set wireless.wifi1.hwmode=11axg
uci set wireless.wifi1.disabled=0
uci set wireless.wifi1.htmode=HT20
uci set wireless.wifi1.channel=1
uci set wireless.wifi1.txchainmask=15
uci set wireless.wifi1.rxchainmask=15
uci set wireless.wifi1.mode=ap
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=ap
uci set wireless.@wifi-iface[1].ssid=2g_open
uci set wireless.@wifi-iface[1].encryption=none
uci commit wireless
```

8.3 QWRAP configuration (basic)

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Band steering in QWRAP mode* section of the *Repeater access point functions* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

8.3.1 QWRAP per radio configuration

```
uci set wireless.wifi0.channel=36
   uci set wireless.wifi0.qwrap_enable=1
   uci set wireless.wifi0.disabled=0
```

8.3.2 QWRAP 'wrap' interface

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

```
uci set wireless.@wifi-iface[0].extender_device=wifi0
```

8.3.3 QWRAP 'sta' interface

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

8.3.4 Automatic addition of proxystas

UCI configurations

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

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

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

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

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

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

To change the wired station limit (see the following table for limits on maximum number of wired and wireless stations):

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

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

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

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

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

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

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

Table 8-2 Maximum number of wired and wireless clients

Chipset model	Maximum number of clients (wireless + wired)	Total VAPs on QWRAP (PSTAs + MPSTA + AP VAP)
QCA9980	22	24
QCA9984	28	30
IPQ401x	28	30

Table 8-2 Maximum number of wired and wireless clients

QCA9886	22	24
QCA9889	14	16
Non-offload chipsets such as QCA9531	30	32

8.3.5 wrapd_vma configuration

UCI configuration is as follows:

uci set wireless.wifiX.wrapd_vma=1

This config will enable wrapd to create PSTAs with mac addresses present on file path /etc/ath/wrap-vma-wifiX.conf.

When wrapd_vma flag is enabled, the user must make sure that different set of VMA addresses are configured on each QWRAP AP present in that network.

8.4 QWRAP configuration (DBDC)

8.4.1 QWRAP DBDC configuration 1

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

```
rm -rf /etc/config/wireless
 wifi detect > /etc/config/wireless
uci set wireless.wifi1=wifi-device
 uci set wireless.wifi0.type=qcawifi
 uci set wireless.wifi0.channel=36
 uci set wireless.wifi0.macaddr=8c:fd:f0:24:fa:d7
 uci set wireless.wifi0.hwmode=11ac
 uci set wireless.@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].extender_device=wifi1
 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.hwmode=11ng
 uci set wireless.wifi1.qwrap_enable=1
 uci set wireless.@wifi-iface[1]=wifi-iface
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
 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
```

8.4.2 QWRAP DBDC configuration 2

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

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

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

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

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

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

uci set wireless.@wifi-iface[2]=wifi-iface
 uci set wireless.@wifi-iface[2].device=wifi1
 uci set wireless.@wifi-iface[2].network=lan
 uci set wireless.@wifi-iface[2].mode=ap
 uci set wireless.@wifi-iface[2].ssid=dbdc-ap2
 uci set wireless.@wifi-iface[2].encryption=psk2+ccmp
 uci set wireless.@wifi-iface[2].key=12345678
uci set wireless.@wifi-iface[2].extender_device=wifi0
```

8.4.3 QWRAP DBDC configuration 3

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

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

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

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

8.4.4 QWRAP TBTC configuration 1

Two radios have AP VAP and STA VAP and one radio has AP VAP.

```
wifi detect > /etc/config/wireless
uci commit wireless

uci show wireless | grep =wifi-iface
uci delete wireless.@wifi-iface[1]
uci delete wireless.@wifi-iface[0]
uci show wireless | grep =wifi-iface
uci add wireless wifi-iface
uci set wireless.@wifi-iface[0].network=lan
uci set wireless.@wifi-iface[0].mode=ap
uci set wireless.@wifi-iface[0].ssid=OpenWrt
uci set wireless.@wifi-iface[0].encryption=none
uci add wireless wifi-iface
uci set wireless.@wifi-iface[1].network=lan
uci set wireless.@wifi-iface[1].mode=ap
uci set wireless.@wifi-iface[1].ssid=OpenWrt
uci set wireless.@wifi-iface[1].encryption=none
uci set wireless.wifi0=wifi-device
uci set wireless.wifi0.type='qcawifi'
uci set wireless.wifi0.channel='11'
uci set wireless.wifi0.macaddr='8c:fd:f0:06:bb:d4'
uci set wireless.wifi0.hwmode='11ng'
uci set wireless.wifi0.disabled='0'
uci set wireless.wifi0.qwrap_enable='1'
uci set wireless.@wifi-iface[0]=wifi-iface
uci set wireless.@wifi-iface[0].network='lan'
uci set wireless.@wifi-iface[0].encryption='none'
uci set wireless.@wifi-iface[0].device='wifi0'
uci set wireless.@wifi-iface[0].mode='wrap'
uci set wireless.@wifi-iface[0].ssid='QWRAP_2G'
uci set wireless.@wifi-iface[0].extender_device='wifi0'
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].encryption='none'
uci set wireless.@wifi-iface[1].mode='sta'
uci set wireless.@wifi-iface[1].ssid='Dakota_2G'
uci add wireless wifi-iface
uci set wireless.@wifi-iface[2].network=lan
uci set wireless.@wifi-iface[2].mode=ap
uci set wireless.@wifi-iface[2].ssid=OpenWrt
uci set wireless.@wifi-iface[2].encryption=none
uci add wireless wifi-iface
uci set wireless.@wifi-iface[3].network=lan
uci set wireless.@wifi-iface[3].mode=ap
uci set wireless.@wifi-iface[3].ssid=OpenWrt
uci set wireless.@wifi-iface[3].encryption=none
uci set wireless.wifi1=wifi-device
uci set wireless.wifi1.type='qcawifi'
uci set wireless.wifi1.hwmode='11ac'
uci set wireless.wifi1.macaddr='8c:fd:f0:06:bb:d3'
uci set wireless.wifi1.channel='36'
```

```
uci set wireless.wifi1.disabled='0'
uci set wireless.wifi1.qwrap_enable='1'
uci set wireless.wifi1.fast_lane='1'
uci set wireless.@wifi-iface[2]=wifi-iface
uci set wireless.@wifi-iface[2].network='lan'
uci set wireless.@wifi-iface[2].encryption='none'
uci set wireless.@wifi-iface[2].device='wifi1'
uci set wireless.@wifi-iface[2].mode='wrap'
uci set wireless.@wifi-iface[2].ssid='QWRAP_5G'
uci set wireless.@wifi-iface[2].extender_device='wifi1'
uci set wireless.@wifi-iface[3]=wifi-iface
uci set wireless.@wifi-iface[3].device='wifi1'
uci set wireless.@wifi-iface[3].network='lan'
uci set wireless.@wifi-iface[3].encryption='none'
uci set wireless.@wifi-iface[3].mode='sta'
uci set wireless.@wifi-iface[3].ssid='Dakota_5GL'
uci add wireless wifi-iface
uci set wireless.@wifi-iface[4].network=lan
uci set wireless.@wifi-iface[4].mode=ap
uci set wireless.@wifi-iface[4].ssid=OpenWrt
uci set wireless.@wifi-iface[4].encryption=none
uci set wireless.wifi2=wifi-device
uci set wireless.wifi2.type='qcawifi'
uci set wireless.wifi2.channel='149'
uci set wireless.wifi2.hwmode='11ac'
uci set wireless.wifi2.macaddr='8c:fd:f0:02:0c:3b'
uci set wireless.wifi2.disabled='0'
uci set wireless.wifi2.qwrap_enable='1'
uci set wireless.wifi2.fast_lane='1'
uci set wireless.@wifi-iface[4]=wifi-iface
uci set wireless.@wifi-iface[4].device='wifi2'
uci set wireless.@wifi-iface[4].network='lan'
uci set wireless.@wifi-iface[4].mode='ap'
uci set wireless.@wifi-iface[4].ssid='QWRAP_5G'
uci set wireless.@wifi-iface[4].encryption='none'
uci set wireless.@wifi-iface[4].extender_device='wifi1'

uci commit wireless
```

8.4.5 QWRAP TBTC configuration 2

All three radios have AP VAP and STA VAP.

```
wifi detect > /etc/config/wireless
uci commit wireless

uci show wireless | grep =wifi-iface
uci delete wireless.@wifi-iface[1]
uci delete wireless.@wifi-iface[0]
uci show wireless | grep =wifi-iface
uci add wireless wifi-iface
uci set wireless.@wifi-iface[0].network=lan
uci set wireless.@wifi-iface[0].mode=ap
uci set wireless.@wifi-iface[0].ssid=OpenWrt
uci set wireless.@wifi-iface[0].encryption=none
```

```
uci add wireless wifi-iface
uci set wireless.@wifi-iface[1].network=lan
uci set wireless.@wifi-iface[1].mode=ap
uci set wireless.@wifi-iface[1].ssid=OpenWrt
uci set wireless.@wifi-iface[1].encryption=none
uci set wireless.wifi0=wifi-device
uci set wireless.wifi0.type='qcawifi'
uci set wireless.wifi0.channel='11'
uci set wireless.wifi0.macaddr='8c:fd:f0:06:bb:d4'
uci set wireless.wifi0.hwmode='11ng'
uci set wireless.wifi0.disabled='0'
uci set wireless.wifi0.qwrap_enable='1'
uci set wireless.@wifi-iface[0]=wifi-iface
uci set wireless.@wifi-iface[0].network='lan'
uci set wireless.@wifi-iface[0].encryption='none'
uci set wireless.@wifi-iface[0].device='wifi0'
uci set wireless.@wifi-iface[0].mode='wrap'
uci set wireless.@wifi-iface[0].ssid='QWRAP_2G'
uci set wireless.@wifi-iface[0].extender_device='wifi0'
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].encryption='none'
uci set wireless.@wifi-iface[1].mode='sta'
uci set wireless.@wifi-iface[1].ssid='Dakota_2G'
uci add wireless wifi-iface
uci set wireless.@wifi-iface[2].network=lan
uci set wireless.@wifi-iface[2].mode=ap
uci set wireless.@wifi-iface[2].ssid=OpenWrt
uci set wireless.@wifi-iface[2].encryption=none
uci add wireless wifi-iface
uci set wireless.@wifi-iface[3].network=lan
uci set wireless.@wifi-iface[3].mode=ap
uci set wireless.@wifi-iface[3].ssid=OpenWrt
uci set wireless.@wifi-iface[3].encryption=none
uci set wireless.wifi1=wifi-device
uci set wireless.wifi1.type='qcawifi'
uci set wireless.wifi1.hwmode='11ac'
uci set wireless.wifi1.macaddr='8c:fd:f0:06:bb:d3'
uci set wireless.wifi1.channel='36'
uci set wireless.wifi1.disabled='0'
uci set wireless.wifi1.qwrap_enable='1'
uci set wireless.wifi1.fast_lane='1'
uci set wireless.@wifi-iface[2]=wifi-iface
uci set wireless.@wifi-iface[2].network='lan'
uci set wireless.@wifi-iface[2].encryption='none'
uci set wireless.@wifi-iface[2].device='wifi1'
uci set wireless.@wifi-iface[2].mode='wrap'
uci set wireless.@wifi-iface[2].ssid='QWRAP_5G'
uci set wireless.@wifi-iface[2].extender_device='wifi2'
uci set wireless.@wifi-iface[3]=wifi-iface
uci set wireless.@wifi-iface[3].device='wifi1'
uci set wireless.@wifi-iface[3].network='lan'
uci set wireless.@wifi-iface[3].encryption='none'
uci set wireless.@wifi-iface[3].mode='sta'
uci set wireless.@wifi-iface[3].ssid='Dakota_5GL'
```

```
uci add wireless wifi-iface
uci set wireless.@wifi-iface[4].network=lan
uci set wireless.@wifi-iface[4].mode=ap
uci set wireless.@wifi-iface[4].ssid=OpenWrt
uci set wireless.@wifi-iface[4].encryption=none
uci add wireless wifi-iface
uci set wireless.@wifi-iface[5].network=lan
uci set wireless.@wifi-iface[5].mode=ap
uci set wireless.@wifi-iface[5].ssid=OpenWrt
uci set wireless.@wifi-iface[5].encryption=none
uci set wireless.wifi2=wifi-device
uci set wireless.wifi2.type='qcawifi'
uci set wireless.wifi2.channel='149'
uci set wireless.wifi2.hwmode='11ac'
uci set wireless.wifi2.macaddr='8c:fd:f0:02:0c:3b'
uci set wireless.wifi2.disabled='0'
uci set wireless.wifi2.qwrap_enable='1'
uci set wireless.wifi2.fast_lane='1'
uci set wireless.wifi2.pref_uplink='1'
uci set wireless.@wifi-iface[4]=wifi-iface
uci set wireless.@wifi-iface[4].device='wifi2'
uci set wireless.@wifi-iface[4].network='lan'
uci set wireless.@wifi-iface[4].mode='wrap'
uci set wireless.@wifi-iface[4].ssid='QWRAP_5G'
uci set wireless.@wifi-iface[4].encryption='none'
uci set wireless.@wifi-iface[4].extender_device='wifi1'
uci set wireless.@wifi-iface[5]=wifi-iface
uci set wireless.@wifi-iface[5].device='wifi2'
uci set wireless.@wifi-iface[5].network='lan'
uci set wireless.@wifi-iface[5].encryption='none'
uci set wireless.@wifi-iface[5].mode='sta'
uci set wireless.@wifi-iface[5].ssid='Dakota_5GH'
uci commit wireless
```

8.5 DBDC repeater mode

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *DBDC repeater functionality* section of the *Repeater access point functions* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

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

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

uci set wireless.wifi0=wifi-device
uci set wireless.wifi0.type=qcawifi
uci set wireless.wifi0.channel=1
uci set wireless.wifi0.macaddr=00:03:7f:12:34:56
uci set wireless.wifi0.hwmode=11ng
uci set wireless.wifi0.htmode=HT40
```

```
uci set wireless.wifi0.disabled=0
uci set wireless.@wifi-iface[0]=wifi-iface
uci set wireless.@wifi-iface[0].device=wifi0
uci set wireless.@wifi-iface[0].network=lan
uci set wireless.@wifi-iface[0].mode=ap
uci set wireless.@wifi-iface[0].ssid=hb_extap
uci set wireless.@wifi-iface[0].encryption=psk2+ccmp
uci set wireless.@wifi-iface[0].key=12345678
uci add wireless wifi-iface
uci set wireless.@wifi-iface[1]=wifi-iface
uci set wireless.@wifi-iface[1].device=wifi0
uci set wireless.@wifi-iface[1].network=lan
uci set wireless.@wifi-iface[1].mode=sta
uci set wireless.@wifi-iface[1].extap=1
uci set wireless.@wifi-iface[1].ssid=hb_rootap
uci set wireless.@wifi-iface[1].encryption=psk2+ccmp
uci set wireless.@wifi-iface[1].key=12345678

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

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

8.6 Enable NSS Wi-Fi offload

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *NSS Wi-Fi offload* section of the *Networking offload and acceleration of link and multilink features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

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

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

These are per device level parameter.

UCI Config value specification

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

Bit 0 = 1 Radio 0 enabled for nss wifi offload

Bit 1 = 1 Radio 1 enabled for nss wifi offload

Bit 2 = 1 Radio 2 enabled for nss wifi offload

E.g.

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

8.7 Enable WPS enhancement for repeaters with a single push button

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *WPS enhancement for range extenders and repeater WPS configuration using a single push button* section of the *Repeater access point functions* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

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

Setting this to 1 enables this feature.

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

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

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

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

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


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

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

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

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

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

If set to 1, the WPS button push event is passed to this VAP (according to the 'start_time' and 'duration') settings

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

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

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

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

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

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

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

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

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

8.8 Enable Wi-Fi memory pre-allocation

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

This pre-allocation is disabled by default and to enable it, prealloc_disabled=0 must be passed as a module parameter. The module param, max_vaps can be used to specify the maximum number of VAPs that needs to be pre-allocated.

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

UCI config values:

For example:

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

8.9 Enable ATF using UCI commands

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Airtime fairness* section of the *Memory and bandwidth*

optimization features chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

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

NOTE These commands are available only in OpenWrt build environment.

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

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

8.9.1 UCI command to view airtime statistics and logs

To retrieve the airtime used as a percentage of total airtime allotted across all clients and the total airtime used since the client got connected.

```
wlanconfig athX atfstat <mac address>
For example, wlanconfig athX atfstat 001122334455
```

8.10 Single AP Band Steering Daemon (lbd) parameters

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Single AP band steering* section of the *Wi-Fi SON: Band steering and APS* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

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

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

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

Table 8-3 shows the parameters that can be updated in the **lbd** UCI configuration file.

Note the following about these parameters:

- Those parameters in a section with an *_Adv* suffix have been selected for functional and performance reasons. It is not recommended that they are modified without careful consideration. Those for which OEM changes are specifically not recommended have a WARNING in their description box.

- Some of these parameters are only relevant when operating in multiple AP mode and are denoted with **Multi-AP mode only** in the description. See Section 8.11 for details on starting the daemon that provides the multi-AP steering functionality.

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

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

Table 8-3 Band and AP Steering Configurable Parameters

Configuration Type	Section	Option	Description	Default
config	config	Enable	Whether the load balancing logic is enabled or not.	0
config	config	MatchingSSID	The SSID to match when limiting band steering to only a single SSID. Normally band steering will manage all SSIDs within the LAN network. This allows restricting it to a single SSID.	-
config	config	PHYBasedPrioritization	Boolean flag indicating whether preference should be given to putting or keeping 802.11ac clients on 5 GHz or not. **NOTE** This feature is currently not supported on platforms with three radios.	0
config	config_Adv	AgeLimit	The maximum age (in seconds) for measured values before they are considered too out of date from which to make a steering decision.	5
config	config	BlacklistOtherESS	To ensure bandwidth fairness or guaranteed throughput, when a client is being steered from one VAP to another VAP on the same ESS, all VAPs on other ESSes must be blacklisted. A mechanism to disable band steering on specific SSIDs is implemented. The feature is disabled by default. This functionality operates with multi-band steering and not with multi-AP steering. The user specifies the SSIDs to be part of band steering. All other SSIDs are excluded from band steering and do not participate in band steering. 'BlacklistOtherESS' is the option added in the lbd config file to enable/disable this feature. The default value is BlacklistOtherESS '0', which signifies that the band steering	0

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 8-3 Band and AP Steering Configurable Parameters

Configuration Type	Section	Option	Description	Default
IdleSteer	IdleSteer	NormalInactTimeout	Number of seconds for the inactivity value under no overload conditions on both bands. **NOTE** This value is set to ensure the ability to detect certain clients that periodically send higher layer keep alive messages (such as an ARP request to the gateway).	10
IdleSteer	IdleSteer	OverloadInactTimeout	Number of seconds for the inactivity value when the serving band is overloaded	10
IdleSteer	IdleSteer	InactCheckInterval	How frequently (in seconds) to check for inactive associated STAs on both bands	1
IdleSteer	IdleSteer	RSSISteeringPoint_DG	The point at which the measured or estimated RSSI on 2.4 GHz dictates a node associated on 5 GHz should be steered to 2.4 GHz. This default value effectively disables downgrade steering due to its limited benefit with modern clients.	5
IdleSteer	IdleSteer	RSSISteeringPoint_UG	The point at which the measured or estimated RSSI on 5 GHz dictates a node associated on 2.4 GHz should be steered to 5 GHz.	20
IdleSteer	IdleSteer	AuthAllow	Enable/disable the Authentication allow feature.	0
ActiveSteer	ActiveSteer	TxRateXingThreshold_UG	The rate (in Kbps) at which a rate crossing event should be generated for a potential active client upgrade to 5 GHz.	50000
ActiveSteer	ActiveSteer	RateRSSIXingThreshold_UG	The value (in dB) the uplink RSSI on 2.4 GHz must be above to be considered for active steering to 5 GHz. This threshold is in conjunction with the `TxRateXingThreshold_UG`.	30
ActiveSteer	ActiveSteer	TxRateXingThreshold_DG	The rate (in Kbps) at which a rate crossing event should be generated for a potential active client downgrade to 2.4 GHz. This threshold is an additional trigger (an OR condition) for downgrade (with the other trigger being the `RawRSSIXingThreshold_DG`. This value effectively disables downgrade active steering due to its limited usefulness with modem clients.	6000

Table 8-3 Band and AP Steering Configurable Parameters

Configuration Type	Section	Option	Description	Default
ActiveSteer	ActiveSteer	RateRSSIXingThreshold_DG	The value (in dB) the uplink RSSI on 5 GHz may be below to be considered for active steering to 2.4 GHz. This threshold is an additional trigger (an OR condition) for downgrade (with the other trigger being the `TxRateXingThreshold_DG`). This value effectively disables downgrade active steering due to its limited usefulness with modem clients.	0
IAS	IAS	Enable_W2	Whether to enable Interfernce Avoidance Steering on 2.4 GHz. If this is set to 0, no attempt will be made to detect interference for a STA associated on 2.4 GHz.	1
IAS	IAS	Enable_W5	Whether to enable Interference Avoidance Steering on 5 GHz. If this is set to 0, no attempt will be made to detect interference for a STA associated on 5 GHz.	1
IAS	IAS	MaxPollutionTime	The number of seconds after which a BSS that was previously marked as polluted is considered no longer polluted. **NOTE** BSS may also have its pollution cleared by output from the accumulator.	1200 (20 minutes)
IAS	IAS	UseBestEffort	Whether best effort steering should be used when the reason for the steering is IAS. When best effort steering is used, any failure will not update the client steering classification state machines.	0
Offload	Offload	MUAvgPeriod	Number of seconds to average before generating a new utilization report on both bands	60
Offload	Offload	MUOverloadThreshold_W2	Medium utilization threshold (in percentage) for an overload condition on 2.4 GHz	70
Offload	Offload	MUSOverloadThreshold_W5	Medium utilization threshold (in percentage) for an overload condition on 5 GHz	70
Offload	Offload	MUSafetyThreshold_W2	The percentage of medium utilization that the measured plus projected utilization is allowed to reach before all further upgrade steering is disallowed until a new utilization measurement is done.	50
Offload	Offload	MUSafetyThreshold_W5	The percentage of medium utilization that the measured plus projected utilization is allowed to reach before all further upgrade steering is disallowed until a new utilization measurement is done.	60
Offload	Offload	OffloadingMinRSSI	Uplink RSSI (in dB) above which pre-association steering and post-association offloading is allowed.	20

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 09:57:41 pan.yafeng@zte.com.cn

Table 8-3 Band and AP Steering Configurable Parameters

Configuration Type	Section	Option	Description	Default
StaDB	StaDB	IncludeOutOfNetwork	Whether out of network devices should be included in the database or not.	1
StaDB	StaDB_Adv	AgingSizeThershold	The number of entries allowed in the station database before periodic aging is triggered.	100
StaDB	StaDB_Adv	AgingFrequency	Once aging is triggered, how frequently (in seconds) to perform aging of the station database.	60
StaDB	StaDB_Adv	OutOfNetworkMaxAge	The number of seconds that must elapse since the last update for an out-of-network entry before it is considered too old and is removed from the database.	300
StaDB	StaDB_Adv	InNetworkMaxAge	The number of seconds that must elapse since the last update for an in-network entry before it is considered too old and is removed from the database. Only unassociated entries will be considered for removal.	2592000 (30 days)
StaDB	StaDB_Adv	NumRemoteBSSes	Multi-AP mode only. The maximum number of statistics to store for BSSes other than those provided by the serving AP.	4
StaDB	StaDB	NoSteer	Controls whether steering is allowed or disallowed for a STA. The usage of this option is as follows: `nosteer <mac_addr> <1|0>` ■ 1—Allow steering for the specified MAC address ■ 0—Disallow steering for the specified MAC address This functionality is used to mark clients as permanently steering-unfriendly or steering-friendly. based on MAC address. After clients are marked as permanently steering unfriendly, they are no longer steered. The default behavior is steering-friendly and not permanently steering-unfriendly.	1

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 8-3 Band and AP Steering Configurable Parameters

Configuration Type	Section	Option	Description	Default

StaDB	StaDB	s	Checks the default settings of disabling steering flag. The output of the sta db out command signifies whether the STAs are marked as “Steer Allowed” or “Steer Disallowed”.
			The following is an output of this command:
			`@stadb s out`
			`Num entries = 15`
			
			`MAC Address Age Bands`
			`E0:9D:31:09:C2:88 52 5 Steer Allowed`
			`02:03:7F:87:88:59 6 5 Steer Allowed`
			`8C:70:5A:C3:EE:A0 32 5 Steer Allowed`
			`08:11:96:3F:16:E4 6 5 Steer Allowed`
			`3C:A9:F4:1D:FC:D0 83 5 Steer Allowed`
			`AC:BC:32:94:20:CF 1 25 Steer Allowed`
			`00:03:7F:40:11:4F 1 25 Steer Allowed`
			`A0:88:B4:C7:0B:2C 26 5 Steer Allowed`
			`92:FD:F0:02:06:11 4 2 Steer Allowed`
			`92:FD:F0:02:0A:25 4 5 Steer Allowed`
			`48:51:B7:16:B5:B6 35 5 Steer Allowed`
			`BC:4C:C4:8B:3D:4A 3 Steer Disallowed`
			`00:24:D7:DF:54:AC 27 25 Steer Allowed`
			`A0:88:B4:D1:7A:D0 20 5 Steer Allowed`
			`CA:FF:28:CD:C9:E5 0 5 Steer Allowed`	

StaMonitor	StaMonitor_Adv	RSSIMeasureSamples_W2	Number of RSSI measurements to average using QoS Null Data Packets before generating a RSSI report on 2.4 GHz	5
StaMonitor	StaMonitor_Adv	RSSIMeasureSamples_W5	Number of RSSI measurements to average using QoS Null Data Packets before generating a RSSI report on 5 GHz	5

Qualcomm Confidential - May Contain Trade Secrets 2021-05-13 09:57:44 PDT pan.yafengzhe.com.cn

Table 8-3 Band and AP Steering Configurable Parameters

Configuration Type	Section	Option	Description	Default
BandMonitor	BandMonitor_Adv	MUCheckInterval_W2	How frequently (in seconds) to check the medium utilization on 2.4 GHz	10
BandMonitor	BandMonitor_Adv	MUCheckInterval_W5	How frequently (in seconds) to check the medium utilization on 5 GHz	10
BandMonitor	BandMonitor_Adv	ProbeCountThreshold	The number of consecutive probe request RSSI values that must be available to consider using the average RSSI when making pre-association steering decisions. **WARNING** Changing this value is not recommended as larger values are likely to reduce the chance of pre-association steering.	1
BandMonitor	BandMonitor_Adv	MUReportPeriod	Multi-AP mode only. How often (in seconds) the medium utilization information should be collected from all nodes in the network.	30
BandMonitor	BandMonitor_Adv	LoadBalancingAllowedMaxPeriod	Multi-AP mode only. The amount of time that must be remaining in the `MUReportPeriod` for a load balancing slot to be assigned. This allows a second device to attempt load balancing if the first device assigned had nothing to do.	15
BandMonitor	BandMonitor_Adv	NumRemoteChannels	Multi-AP mode only. The maximum number of channels that may be in use in the network on nodes other than the current one. This should be generally set based on the maximum number of radios in the devices being deployed in the network.	3
Estimator_Adv	Estimator_Adv	RSSIDiff_EstW5FromW2	Difference when estimating 5 GHz RSSI value from the one measured on 2.4 GHz.	-15
Estimator_Adv	Estimator_Adv	RSSIDiff_EstW2FromW5	Difference when estimating 2.4 GHz RSSI value from the one measured on 5 GHz.	5
Estimator_Adv	Estimator_Adv	ProbeCountThreshold	The number of consecutive probe request RSSI values that must be available to consider using the average RSSI on the unassociated band when making steering decisions. **WARNING** Reducing this value may lead to unnecessary steering in cases where there is higher variability in probe request RSSI.	3

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 8-3 Band and AP Steering Configurable Parameters

Configuration Type	Section	Option	Description	Default
Estimator_Adv	Estimator_Adv	StatsSampleInterval	The amount of time (in seconds) between consecutive samples of the byte count statistics for a STA when estimating its data rate.	1
Estimator_Adv	Estimator_Adv	11kProhibitTime	The minimum amount of time (in seconds) to enforce between consecutive 802.11k Beacon Requests. **WARNING** Reducing this value could increase the likelihood of clients rejecting requests that occur too frequently.	30
Estimator_Adv	Estimator_Adv	PhyRateScalingForAirtime	The factor by which to scale the estimate PHY rate to arrive at an approximate effective MAC rate.	50%
Estimator_Adv	Estimator_Adv	EnableContinuousThroughput	Run with throughput sampling always enabled (for demo or debugging purposes only). With this option enabled, the current throughput for each associated STA will be logged every second.	0
Estimator_Adv	Estimator_Adv	BcnrptActiveDuration	Duration (in milliseconds) for an active mode 802.11k Beacon Request. This is used on non-DFS channels. **WARNING** Reducing this value could lead to a lower success rate for measurements.	50
Estimator_Adv	Estimator_Adv	BcnrptPassiveDuration	Duration (in milliseconds) for a passive mode 802.11k Beacon Request. This is used on DFS channels. **WARNING** Reducing this value could lead to a lower success rate for measurements.	200
Estimator_Adv	Estimator_Adv	FastPollutionDetectBufSize	The size of the accumulator buffer when looking for interference starting from the no interference state. **WARNING** This value was carefully selected based on experimentation. Modification is not recommended.	5
Estimator_Ad	Estimator_Adv	NormalPollutionDetectBufSize	The size of the accumulator buffer after interference has been detected on the BSS. **WARNING** This value was carefully selected based on experimentation. Modification is not recommended.	5

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 8-3 Band and AP Steering Configurable Parameters

Configuration Type	Section	Option	Description	Default
Estimator_Adv	Estimator_Adv	PollutionDetectThreshold	Minimum percentage of detected samples in the accumulator buffer for the BSS to be declared polluted. **WARNING** This value was carefully selected based on experimentation. Modification is not recommended.	60
Estimator_Adv	Estimator_Adv	PollutionClearThreshold	Maximum percentage of detected samples in the accumulator buffer (after it is full) for the BSS to be declared no longer polluted. **WARNING** This value was carefully selected based on experimentation. Modification is not recommended.	40
Estimator_Adv	Estimator_Adv	InterferenceAgeLimit	The base time used to determine when detector samples should be aged out of the accumulator buffer. Samples that are older than InterferenceAgeLimit multiplied by the buffer size will be removed.	15
Estimator_Adv	Estimator_Adv	IASLowRSSIThreshold	Minimum RSSI (in dB) required for a sample to be fed to the interference detector. Below this value, interference will be assumed not to be present as any low downlink rate (MCS) value is likely due to the path loss. **WARNING** This value was carefully selected based on experimentation. Modification is not recommended.	12
SteerExec	SteerExec	SteeringProhibitTime	Number of seconds to wait prior to steering the client again after a steering when either the legacy steering mechanism is used or the 802.11v BSS Transition Management mechanism is used but the client still attempts to authenticate on a BSS other than the target one. **WARNING** Reducing this value could lead to a client blacklisting the AP due to too frequent steering.	300
SteerExec	SteerExec	BTMSteeringProhibitShortTime	The time period to wait prior to steering an 11v-capable client again after a successful steering within BTMAssociationTime. **WARNING** Reducing this value is not recommended.	30

Table 8-3 Band and AP Steering Configurable Parameters

Configuration Type	Section	Option	Description	Default
SteerExec	SteerExec_Adv	TSteering	Number of seconds allowed for the client to reconnect before AP aborts steering when performing legacy steering. **WARNING** Reducing this value is not recommended.	15
SteerExec	SteerExec_Adv	InitialAuthRejCoalesceTime	Number of seconds to coalesce multiple authentication rejects down to a single one when counting consecutive auth rejects. This parameter is used in conjunction with `AuthRejMax` below to abort steering when a client is not moving to the desired BSS. **WARNING** This value was carefully selected based on client testing. Modification is not recommended.	2
SteerExec	SteerExec_Adv	AuthRejMax	The number of consecutive authentication rejects that cause steering to be aborted and the device to be marked as steering unfriendly **WARNING** This value was carefully selected based on client testing. Modification is not recommended.	3
SteerExec	SteerExec_Adv	SteeringUnfriendlyTime	The amount of time a device is considered steering unfriendly before another attempt. This is used as the base for an exponential back-off scheme when a STA repeatedly fails legacy steering. **WARNING** This value was carefully selected based on client testing. Modification is not recommended.	600
SteerExec	SteerExec_Adv	MaxSteeringUnfriendlyTime	Maximum time (in seconds) for the legacy steering unfriendly timer. This is used in conjunction with `SteeringUnfriendlyTime`.	604800 (1 week)
SteerExec	SteerExec_Adv	TargetLowRSSIThreshold_ W2	RSSI threshold (in dB) indicating 2.4 GHz band is not strong enough for association. When steering to 2.4 GHz, if the uplink RSSI (as measured by probe requests) falls below this value, steering will be aborted.	5
SteerExec	SteerExec_Adv	TargetLowRSSIThreshold_ W5	RSSI threshold (in dB) indicating 5 GHz band is not strong enough for association. When steering to 5 GHz, if the uplink RSSI (as measured by probe requests) falls below this value, steering will be aborted.	15

Table 8-3 Band and AP Steering Configurable Parameters

Configuration Type	Section	Option	Description	Default
SteerExec	SteerExec_Adv	BlacklistTime	The amount of time (in seconds) before automatically removing the blacklist (independent of RSSI conditions, but still subject to overload checks).	900 (15 minutes)
SteerExec	SteerExec_Adv	BTMResponseTime	Maximum response delay for 802.11v BSS Transition Management Request.	10
SteerExec	SteerExec_Adv	BTMAssociationTime	The maximum time allowed for an 11v-capable client to reconnect before AP aborts steering the client, releases the blacklist (if in use) for the client, and marks the BTM steering attempt as having failed.	6
SteerExec	SteerExec_Adv	BTMUnfriendlyTime	The time period to wait prior to steering an 11v-capable client again upon BTM steering failures (subject to exponential back-off). **WARNING** Reducing this value is not recommended as it may contribute to a higher steering failure rate.	600
SteerExec	SteerExec_Adv	MaxBTMUnfriendly	Maximum time (in seconds) for the BTM steering unfriendly timer for idle steering.	86400 (1 day)
SteerExec	SteerExec_Adv	MaxBTMActiveUnfriendly	Maximum time (in seconds) for the BTM steering unfriendly timer for active steering.	604800 (1 week)
SteerExec	SteerExec_Adv	MinRSSIBestEffort	The RSSI (in dB) below which BTM-based steering will operate in best effort mode (where no blacklists are installed).	12
SteerExec	SteerExec_Adv	LowRSSIXingThreshold	RSSI threshold (in dB) to generate an indication when a client crosses it (in dB)	10
SteerAlg_ Adv	SteerAlg_Adv	MinTxRateIncreaseThreshold	Minimum amount the 5 GHz PHY rate (in Kbps) must be above the 2.4 GHz PHY rate when determining if the channel is good enough. This is only used in overload scenarios.	53
SteerAlg_Avd	SteerAlg_Adv	MaxSteeringTargetCount	The maximum number of candidates to include in an 802.11v BSS Transition Management Request or legacy steering operation. The default value reflects the fact that devices currently do not make good use of the preference value included in BTM request.	1
APSteer	APSteer	LowRSSIAPSteeringThreshol d_CAP	Multi-AP mode only RSSI value (in dB) below which the uplink RSSI of a STA associated to the Central AP (CAP) must fall for it to be considered as a candidate for AP steering.	20

Table 8-3 Band and AP Steering Configurable Parameters

Configuration Type	Section	Option	Description	Default
APSteer	APSteer	LowRSSIAPSteerThreshold_RE	Multi-AP mode only RSSI value (in dB) below which the uplink RSSI of a STA associated to a Range Extender (CAP) must fall for it to be considered as a candidate for AP steering.	45
APSteer	APSteer	APSteerToRootMinRSSIIncThreshold	Multi-AP mode only The amount (in dB) the RSSI of the CAP must be better than that of the serving RE, as measured by an 802.11k Beacon Measurement, for the STA to be steered to the CAP.	5
APSteer	APSteer	APSteerToLeafMinRSSIincThreshold	Multi-AP mode only The amount (in dB) the RSSI of an RE must be better than that of the CAP, as measured by an 802.11k Beacon Measurement, for the STA to be steered to the RE.	10
APSteer	APSteer	APSteerToPeerMinRSSIIncThreshold	Multi-AP mode only The amount (in dB) the RSSI of an RE must be better than that of the serving RE, as measured by an 802.11k Beacon Measurement, for the STA to be steered to the RE.	10
APSteer	APSteer	DownlinkRSSIThreshold_W5	Multi-AP mode only The value (in dBm) the downlink RSSI, as measured using 802.11k Beacon Measurement, must be above for a 5 GHz channel to be preferred over 2.4 GHz when AP steering is used. If the downlink RSSI is not above this value, the 2.4 GHz channel will be selected for AP steering so long as its Tx power is at least as good as the 5 GHz channel that was measured using 802.11k.	-65
APSteer	APSteer	LowRSSIAPSteering Threshol d_CAP_W2	In 2.4 GHz band, multi-AP mode only RSSI value (in dB) below which the uplink RSSI of a STA associated to the Central AP (CAP) must fall for it to be considered as a candidate for AP steering. When band-specific AP Steering threshold is present, it will be used, else non-band specific AP Steering parameter value will be used.	25
APSteer	APSteer	LowRSSIAPSteerThreshold _RE_W2	In 2.4 GHz band, multi-AP mode only RSSI value (in dB) below which the uplink RSSI of a STA associated to a Range Extender (RE) must fall for it to be considered as a candidate for AP steering. When band-specific AP Steering threshold is present, it will be used, else non-band specific AP Steering parameter value will be used.	25
APSteer	APSteer	LowRSSIAPSteering Threshol d_CAP_W5	In 5 GHz band, multi-AP mode only RSSI value (in dB) below which the uplink RSSI of a STA associated to the Central AP (CAP) must fall for it to be considered as a candidate for AP steering. When band-specific AP Steering threshold is present, it will be used, else non-band specific AP Steering parameter value will be used.	55

Table 8-3 Band and AP Steering Configurable Parameters

Configuration Type	Section	Option	Description	Default
APSteer	APSteer	LowRSSIAPSteerThreshold _RE_W5	In 5 GHz band, multi-AP mode only RSSI value (in dB) below which the uplink RSSI of a STA associated to a Range Extender (RE) must fall for it to be considered as a candidate for AP steering. When band-specific AP Steering threshold is present, it will be used, else non-band specific AP Steering parameter value will be used.	55
SteerExec	SteerExec_Adv	Delay24GProbeRSSIThreshold	The minimum RSSI threshold to delay probe responses in 2.4G band (in dB).	35
SteerExec	SteerExec_Adv	Delay24GProbeTimeWindow	The time window within which probe responses will not be sent for configured count(s).	0
SteerExec	SteerExec_Adv	Delay24GProbeMinReqCount	The probe request count above which probe responses will be sent for all received within time window.	0
Max11kUnfriendly	Estimator_Adv	Estimator_Adv	Enable the capability of AP steering for legacy clients to be extended to clients that fail for consecutive attempts of 802.11k beacon reports. For such 11k-unfriendly clients, edit /etc/config/lbd for the Max11kUnfriendly option under the Estimator_Adv 'Estimator_Adv' config section and restart hyd. The Max11kUnfriendly option must be configured to ‘n’ for clients to get initiated with Smart Monitoring if 11k beacon report fails ‘n’ consecutive attempts.	10
ApplyEstimatedAireTimeOnSteering	config	config	Enables improvement of the steering decision-making process when airtime is concerned. In the design without this lbd config parameter, when the same channel steering was intended, it was not assumed that the client airtime was already included into the current channel total airtime use. As a result, the client’s estimated airtime was counted double. To be conservative, the configuration flag ApplyEstimatedAireTimeOnSteering is added in file **lbd.config**. In running environment, the file is located at **/etc/config/lbd**. This flag is set to be 1 by default. If the value of this flag is changed as 0, this enhancement to improve airtime calculation is triggered, which denotes that the client’s estimated airtime is not added to the projected channel utilization when making same channel steering decision.	1
PersistPeriod	Persist	Persist	Indicates the periodicity of checking if certain entries to be persisted have been updated – if found to be updated, they are written to the (non-volatile) file specified. Value specified in seconds.	3600

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 8-3 Band and AP Steering Configurable Parameters

Configuration Type	Section	Option	Description	Default
StartInBTMAc-tiveState	SteerExec	SteerEx-ec_Adv	If this parameter is set to 1, it is not necessary to perform idle BTM steering before active BTM steering can be performed. To set this property, enter the following commands: uci set lbd.SteerExec_Adv.StartInBTMActiveS-tate=1 uci commit lbd	0
TrackRemoteAssoc	StaDB	StaDB	Whether the station database should track only local associations or both local and remote associations. The expectation is that this value is set to `1` on all nodes to enable the appropriate bridging table cleanup to occur when handling the list of associated STAs in Topology Response.	1
LowRSSIAPSteerThreshold_SIG	APSteer	APSteer	The threshold to use as the AP steering trigger point when operating in Multi-AP SIG mode.	17
APSteerMaxRetryCount			The number of times to attempt to AP steer a STA before giving up. The counter is reset whenever the STA associates on a new BSS.	2
ActDetectMinInterval	Estimator_Adv	Estimator_Adv	The interval (in seconds) over which to determine whether a STA is inactive based on packet counters.	30
ActDetectMinPktPerSec			The number of packets per second (both TX and RX) below which a client is considered inactive.	2
MinAssocAgeForStatsAssocUpdate	StaDB	StaDB_Adv	The number of seconds a STA must have been associated on one BSS for a stats update to force it to be marked as associated on a different BSS.	150
UsePathCapacityToSelectBSS	SteerAlg_Adv	SteerAlg_Adv	Whether to use the end-to-end path capacity scheme to select the target BSS. If this is false, the Wi-Fi SON BSS selection schemes are used. This is only relevant when operating in Multi-AP SIG mode.	1

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 8-3 Band and AP Steering Configurable Parameters

Configuration Type	Section	Option	Description	Default
config	config	ClientClassificationEnable	In a Wi-Fi SON environment, mesh node client handling based on classification is introduced. The supported platform is ARM based IPQ807x/IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx/QCN90xx or QCN602x + IPQ807xA/IPQ60xx/IPQ50xx 2-radio and 3-radio SBS. The feature provides a mechanism to classify clients into different groups and define specific behavior for each of the group. Also, the feature uses the category mechanism to differentiate the parameters and threshold to be used for Band Steering and AP Steering. The user space SON component uses band steering protocol which allows STAs to switch between 2G and 5G bands automatically when band steering is enabled and AP steering protocol which allows STAs to steer between APs in a Multi Node (Multiple APs connected to each other in the same network – Mesh networking) environment. The following are the supported client classification categories: Category 0 - iOS (No BTM and No RRM) and non-iOS clients Category 1 - iOS (Includes BTM and RRM) This feature is disabled by default and can be enabled only when ClientClassificationEnable is set in lbd config option. Otherwise client classification group will be zero by default for all the clients which falls into default behavior. This feature is added to all the included VAP interfaces only that acts in AP mode allowing clients to be connected. When this feature is enabled and if for not all parameters are provided by user, default parameter is copied to the missed-out client class group parameters. Configure the relevant parameters, such as `RSSISteeringPoint_UG`, `NormalInactTimeout`, `TxRateXingThreshold_UG`, `RateRSSIXingThreshold_UG`, `LegacyUpgradeMonitorDur`, `SteeringProhibitTime`, `BTMSteeringProhibitShortTime` for the two categories – iOS (BTM and RRM) and non-iOS clients – if ClientClassificationEnable is set as 1.	0

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 8-3 Band and AP Steering Configurable Parameters

Configuration Type	Section	Option	Description	Default
config	config	EnableAckRSSI	In a Wi-Fi SON environment, the capability to use RSSI from ACK packets for triggering steering instead of RSSI (Received Signal Strength Indicator) from data packets is implemented. This functionality is supported on IPQ807x/IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx/QCN90xx or QCN602x + IPQ807xA/IPQ60xx/IPQ50xx (2-radio and 3-radio SBS) and legacy IPQ4019 platforms. The user space SON component uses band steering protocol, which allows STAs to switch between 2G and 5G bands automatically when band steering is enabled. One of the parameters used by the band steering protocol is RSSI, which is currently retrieved from data packets. This new capability uses the RSSI value from ACK packets from STA device, which is expected to be more accurate instead of the RSSI from data packets. The Wi-Fi firmware component provides this data to the Wi-Fi host component, which, in turn, passes this value to SON component of driver. The user component of SON uses this value in band steering protocol. The “enable_ol_stats” parameter is used from user space, which, when enabled, the firmware component processes the ACK packets and sends the value of RSSI from ACK packets to host driver and the Wi-Fi SON component retrieves it from Wi-Fi host driver through an already existing data structure. This feature is added to all the included VAP interfaces only. By default, this feature is disabled.	0
config	config	EnableMulti11kRequest	This flag needs to be enabled for interoperation of dual-band tri-band mesh modes in Wi-Fi SON environment because i is disabled for homogeneous and MAP. This flag allows to determine if we have multiple 5 GHz fronthaul channels.	1
Estimator_Adv	Estimator_Adv	Delayed11kRequesttimer	Most of the 11k clients will reject back to back 11k request s. It is necessary to wait half a second delay between two 11k requests. Currently, this delay is configured as 1 second. However, in a noisy environment, this delay can be increased to 2. Only 1 and 2 can be used; any value other than 1 & 2 will not help in real time because a client would have roamed on its own.	1

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

8.11 Multi-AP Coordinated Steering and Adaptive Path Selection parameters

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Multi-AP coordinated steering* section of the *Wi-Fi SON: Band steering and APS* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

The multi-AP coordinated steering and Adaptive Path Selection features are implemented within the daemon named **hyd**. To make use of these features, some additional Wi-Fi settings must be enabled and then the daemon must be started. Typically this is done through the RE Placement and Auto-Configuration Daemon (**repacd**). Steps for using this daemon are in Section 8.12

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

1. Configure the SSID and pass-phrase on the Wi-Fi interfaces as desired, enabling an AP interface on each band.
2. Enable the RRM and WDS features on each AP interface using the following commands (which assume a 2 radio device with one wireless AP interface configured on each radio):
```
uci set wireless.wifi-iface[0].wds=1
uci set wireless.wifi-iface[0].rrm=1
uci set wireless.wifi-iface[1].wds=1
uci set wireless.wifi-iface[1].rrm=1
uci commit wireless
```
3. Bring up the wireless interfaces using the `wifi` command.
4. Disable **mcsd** (as it cannot run at the same time as **hyd**):
```
uci set mcsd.config.Enable=0
uci commit mcsd
/etc/init.d/mcsd stop
```
5. Enable and start **hyd**:
```
uci set hyd.@config[0].Enable=1
uci commit hyd
/etc/init.d/hyd start
```

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

1. Configure the device to not respond to DHCP requests:
```
uci set dhcp.lan.ignore=1
uci commit dhcp
/etc/init.d/dnsmasq restart
```
2. Configure the device as a pure bridge in one of two modes:
   a. Dynamic address
```
uci set network.lan.ifname='eth0 eth1'
uci set network.lan.proto=dhcp
uci delete network.wan
uci commit network
/etc/init.d/network restart
```
   b. Static IP address

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

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

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

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

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

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

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

Beyond this basic configuration, the parameters for configuring Adaptive Path Selection (APS) and coordinated steering are described in the following table. Note that although this daemon has its own configuration file, it also uses the **lbd** configuration file. For those parameters specific to AP steering, see Section 8.10.

Table 8-4 Multi-AP Coordinated Steering and Adaptive Path Selection Parameters

Option	CFG Type	Section	Description	Default
DisableSteering	config	config	Whether the steering feature should be disabled. This is primarily intended for use when testing APS where no steering of clients is desired.	0
Mode	config	config	AP HYROUTER mode for CAP or HYCLIENT for RE. HYROUTER is the master for coordinated steering. Repacd automatically configures this based on network settings (that is, if WAN port is defined or a connection to the gateway is detected and repacd.repacd.GatewayConnected is set to CAP).	

Table 8-4 Multi-AP Coordinated Steering and Adaptive Path Selection Parameters (cont.)

Option	CFG Type	Section	Description	Default
ConstrainTCPMedium	hy	hy	Whether the less dominant direction of a TCP connection should be forced onto the same interface as the dominant direction of the connection. Generally allowing for each direction to use a different interface will result in better performance, so this feature is defaulted to off.	0
HActiveMaxAge	hy	hy	Time (in msec) after which if a flow is idle and the entry is removed from the H-Active table	120000
ScalingFactorHighRate_W5	PathChWlan	PathChWlan	Rate (in Mbps) above which the scaling factor for high rate links on 5 GHz should be applied. See ScalingFactorHigh.	750
ScalingFactorHighRate_W2	PathChWlan	PathChWlan	Rate (in Mbps) above which the scaling factor for high rate links on 2.4 GHz should be applied. See ScalingFactorHigh.	200
ScalingFactorLow	PathChWlan	PathChWlan	Conversion factor (in %) when deriving a UDP capacity value from a PHY rate falls below the low rate threshold (as determined by LinkCapacityThreshold). The PHY rate is multiplied by this value to estimate the full UDP capacity.	60%
ScalingFactorMedium	PathChWlan	PathChWlan	Conversion factor (in %) when deriving a UDP capacity value from a PHY rate that falls between the low rate (as determined by LinkCapacityThreshold) and high rate thresholds. The PHY rate is multiplied by this value to estimate the full UDP capacity.	85%
ScalingFactorHigh	PathChWlan	PathChWlan	Conversion factor (in %) when deriving a UDP capacity value from a PHY rate that falls above the high rate threshold. The PHY rate is multiplied by this value to estimate the full UDP capacity.	60%
ScalingFactorTCP	PathChWlan	PathChWlan	Conversion factor (in %) when deriving a TCP capacity value from a UDP capacity value.	90%
UseWHCAlgorithm	PathChWlan	PathChWlan	Boolean flag to control whether this scheme is used to compute the capacity or the old scheme (that relies on questionable firmware stats) is used. Enable the WHC algorithm for all Qualcomm® HY-FI™ testing.	1
MaxMediumUtilization_W2	PathChWlan	PathChWlan	The overload threshold for 2.4 GHz backhaul link	70
MaxMediumUtilizaation_W5	PathChWlan	PathChWlan	The overload threshold for 5 GHz backhaul link	70
MaxMediumUtilizationForLC_W2	PathChWlan	PathChWlan	Medium utilization threshold for Link Capacity calculation; used to determine whether flow should be moved to other interfaces. In other words, the available capacity estimated is based on the available medium utilization being max (MaxMediumUtilization_W2 – MeasuredMediumUtilization_W2, 0).	70
MaxMediumUtilizationForLC_W5	PathChWlan	PathChWlan	Medium utilization threshold for Link Capacity calculation; used to determine whether flow should be moved to another interface.	70

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 8-4 Multi-AP Coordinated Steering and Adaptive Path Selection Parameters (cont.)

Option	CFG Type	Section	Description	Default
CPULimitedTCPThroughput_W2	PathChWlan	PathChWlan	Determines whether an underpowered CPU poses a limit on TCP throughput. The default value 4294967295 means no limit is placed for an underpowered CPU case. Set a value (in Mbps) as a limit depending on CPU characteristics. This parameter limits the available capacity to the minimum of this value and value computed using measured medium utilization and PHY rate. This parameter is for 2.4 GHz.	4294967295
CPULimitedUDPThroughput_W2	PathChWlan	PathChWlan	Determines whether an underpowered CPU poses a limit on UDP throughput. The default value 4294967295 means no limit is placed for an underpowered CPU case. Set a value (in Mbps) as a limit depending on CPU characteristics. This parameter limits the available capacity to the minimum of this value and value computed using measured medium utilization and PHY rate. This parameter is for 2.4 GHz.	4294967295
CPULimitedTCPThroughput_W5	PathChWlan	PathChWlan	Determines whether an underpowered CPU poses a limit on TCP throughput. The default value 4294967295 means no limit is placed for an underpowered CPU case. Set a value (in Mbps) as a limit depending on CPU characteristics. This parameter limits the available capacity to the minimum of this value and value computed using measured medium utilization and PHY rate. This parameter is for 5 GHz.	4294967295
CPULimitedUDPThroughput_W5	PathChWlan	PathChWlan	Determines whether an underpowered CPU poses a limit on UDP throughput. The default value 4294967295 means no limit is placed for an underpowered CPU case. Set a value (in Mbps) as a limit depending on CPU characteristics. This parameter limits the available capacity to the minimum of this value and value computed using measured medium utilization and PHY rate. This parameter is for 5 GHz.	4294967295
LinkCapacityThreshold	PathSelect	PathSelect	Threshold value (in Mbps) used for the low rate in when determining the scaling factor to use for the UDP and TCP capacity estimates. During path selection, value (expressed in megabits per second) used as a ceiling of bit rates available on each interface. This helps to ensure that the load is balanced across all available interfaces.	20
AvgUtilReqTimeout	SteerMsg	SteerMsg	Number of seconds to wait for the average utilization report to be sent back to the CAP after sending the average utilization request before timing out.	1
LoadBalancingCompleteTimeout	SteerMsg	SteerMsg	Number of seconds to allow for an RE assigned a load balancing slot to send back the complete message before the CAP assumes it was lost over the air and moves on to the next device.	90

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 8-4 Multi-AP Coordinated Steering and Adaptive Path Selection Parameters (cont.)

Option	CFG Type	Section	Description	Default
RspTimeout	SteerMsg	SteerMsg	Number of seconds to allow for a response message to come back from a node that was sent a Prepare for Steering Request, Abort Request, or STA Info Request.	2
DisableMonitoringLegacyClients	config	Monitor	Configures AP steering for legacy clients through RSSI monitoring using Smart Monitor mode. Multi AP with one 2.4 GHz and/or one or two 5 GHz radios. The radios operate concurrently on different channels. The steering improves clients Wi-Fi network performance by steering them to a better-serving AP through a better RSSI. Clients for steering include: A legacy client A client that does not support 802.11k beacon measurement A client certified by the upcoming Wi-Fi Alliance Multiband Operation (MBO) certification A client that fails for consecutive attempts of 802.11k beacon report Support is implemented for APs to steer legacy clients, which do not support 802.11k beacon measurement and MBO compliance. These legacy clients are clients that support 802.11v basic service set (BSS) transition management (BTM) and do not support radio resource management (802.11k). Only idle steering and not active steering is supported. Configure DisableMonitoringLegacyClients as 0 for smart monitor to be functional.	1
DisableSteeringInActiveLegacyClients	config	Monitor	Configures AP steering of inactive legacy clients. Configure DisableSteeringInActiveLegacyClients as 0 for monitoring/steering of inactive legacy clients.	1
DisableSteeringActiveLegacyClients	config	Monitor	Configures AP steering of active legacy clients. Configure DisableSteeringActiveLegacyClients as 0 for monitoring/steering of active legacy clients.	1
MonitorTimer	config	Monitor	Configure to 60 for periodic monitoring. The serving-AP (node initiating monitor RSSI request) contains timer control, within which it must make a steering decision. Until this timer expires, the serving-AP continues to receive Monitor RSSI Reports from other APs. Either before expiry or upon expiry of this timer, the serving AP can make STA steering decision based on received RSSIs. The serving AP gets the current STA RSSI using the LBD (QoS Null-ACK method). Monitor mode converts RSSI in same format as the format that the LBD maintains. If no decision is reached upon expiry of the timer or other RSSIs are not better for steering, serving-AP sends a Stop Monitor Event to all nodes to stop RSSI monitoring on other nodes.	60
MonitorResponseTimeout	config	Monitor	Configure to 5 for better monitoring of multiple clients. The MonitorResponse component causes non-serving AP to start monitoring, start timer and initiate host driver IOCTLs/calls for monitoring. Non-serving AP responds with an acknowledgment to the serving AP.	5

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 8-4 Multi-AP Coordinated Steering and Adaptive Path Selection Parameters (cont.)

Option	CFG Type	Section	Description	Default
PersistFile	Persist	Persist	This parameter is NOT present by default in the configuration file. To enable the STADB persistence feature, provide a file name where STADB data should be persisted. Care should be taken to ensure that the file is in non-volatile memory, for example, **/etc/hyd/persist.dat**.	
WlanCheckFreqInterval	PathChWlan	PathChWlan	Determines how frequently to check wlan for any change, this is used for collecting stats which are used in Path switch. (APS related)	
AssocDetectionDelay_W2	PathChWlan	PathChWlan	Time after which Topology database is updated after association. (Delay to ensure stability).	
ForwardingMode	hyd	ForwardingMode	This field takes one of three values that controls how the `hyfi-netfilter` bridge hooks behave: `APS` - Full support for multiple backhauls and Adaptive Path Selection `SINGLE` - Apply the loop prevention and 1905 multicast rules but disable the H-Default/H-Active tables. `MCAST` - Only apply the 1905 multicast rules. This mode must only be used if another mechanism such as RSTP is being used to prevent loops.	APS
MaxMultiAPSIGMode	hyd	hyd	The maximum messaging mode this node is allowed to operate in. The following modes are supported: 0 - Strict: No Q2Q messaging `1` - Auto homogeneous: Q2Q messaging allowed when all nodes in the network are QCA `2` - Auto heterogeneous: Q2Q messaging allowed even with a mixture of QCA and non-QCA devices (where the Q2Q messaging is only used with the QCA devices) Note that with the current systems HLD strategy of only supporting Multi-AP SIG messaging initially, these modes may not be relevant.	2
EnableController	IEEE1905Settings	IEEE1905Settings	Whether this device must operate in controller mode.	0
EnableAgent	MultiAP	MultiAP	Whether this device must operate in agent mode.	0
EnableSigmaDUT	MultiAP	MultiAP	Whether this device must enable the Sigma DUT functionality. This is only used for plug fests and is not relevant for normal testing.	0
MetricsReportingInterval	MultiAP	MultiAP	The rate (in seconds) at which the controller requests the agents to report AP and STA metrics.	5
RSSIHysteresis_W2	MultiAP	MultiAP	The amount by which the RSSI on 2.4 GHz has to change (in dBm) after crossing the threshold for another event to be generated.	5
RSSIHysteresis_W5	MultiAP	MultiAP	The amount by which the RSSI on 5 GHz has to change (in dBm) after crossing the threshold for another event to be generated.	5

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 8-4 Multi-AP Coordinated Steering and Adaptive Path Selection Parameters (cont.)

Option	CFG Type	Section	Description	Default
LoadBalancingInterval	MultiAP	MultiAP	The time period (in seconds) the controller averages utilization information over prior to performing load balancing steering.	30
EnableChannelSelection	MultiAP	MultiAP	Enable this option when the channel is set as auto in wireless configuration section to synchronize the channel between controller and agents.	
AvoidDupRenew	IEEE1905Settings	IEEE1905Settings	Avoids the AP Auto-Configuration Renew messages received on a STA interface to be relayed on to another STA interface.	0
AvoidDupTopologyNotification	IEEE1905Settings	IEEE1905Settings	Avoids the Topology Notification messages received on a STA interface to be relayed on to another STA interface.	0
EnableBadLinkStatsSwitchFlow	PathSelect	PathSelect	When set, the bad Wi-Fi link event detection is enabled for flow control.	1
HybridBridgeInterfaceName	Interface	Interface	This parameter configures the Interface name of the hybrid bridge. This is the IEEE1905.1 managed bridge interface. This Interface name is read from Network configuration file that has "ieee1905managed" option enabled and "type" option set to "bridge". If no entry exists, then default managed bridge name is set as "lan" interface	br-lan
ManagedInterfacesList	Interface	Interface	This parameter contains the list of Interfaces bound to the bridge. It includes WLAN/Ethernet/PLC Interfaces. Also, it includes VLAN configured WLAN Interface if any. The format used to represent the interface details is represented as follows, (1) WLAN: ifname@radioname:WLAN, (2) ETHERNET: ifname:ETHER (3) PLC : ifname:PLC (4) VLAN enabled WLAN: ifname.vlanid@radioname:WLAN_VLAN. Note: This parameter is automatically configured by the hyd init script based on the information present in Network configuration and Wireless configuration (/etc/config/network and /etc/config/wireless)	
WlanInterfaces	Interface	Interface	This parameter contains the List of all WLAN devices/radios bound to the bridge. This parameter is the sub-set of “ManagedInterfacesList” parameter	
NoRelayGroupList	Interface	Interface	This Parameter specifies the list of Non-Relay Group Interfaces. This parameter is configured for STA mode of operation i.e. HYCLIENT. This interface list includes all the WLAN STA Interfaces bound to managed bridge, all the WLAN VLAN STA Interfaces bound to managed bridge and non-relay ethernet interface i.e. Upstream Ethernet interface). Note: This parameter is automatically configured by the hyd init script based on the information present in Network configuration	
ForceGroupRelaying	Interface	Interface	This parameter specifies the Hybrid bridge configuration as Relay override or group-based relay. This parameter is configured for STA mode of operation i.e. HYCLIENT Mode. Note: This parameter is automatically configured by HYD init script	

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 8-4 Multi-AP Coordinated Steering and Adaptive Path Selection Parameters (cont.)

Option	CFG Type	Section	Description	Default
SwitchLanVid	Interface	Interface	This parameter specifies the vid information to be used in switch driver. This information is required to manage the FDB table created by hyfi-bridge module	1
SwitchCpuPort	Interface	Interface	This parameter specifies the Port information to filter the packets. All the packet in this port will pass to the CPU only. This parameter is derived from "SwitchLanVid" in HYD config	
VlanIds	Interface	Interface	This parameter has mapping of vlanid to Ethernet interface. This information is used in managing forwarding database information. Note: This parameter is derived from "ifname" and "vid" parameter in HYD config using HYD init script	
LoadBalancingSeamless	HY	HY	This parameter enables seamless load balancing. Enabling this feature will configure path switching parameter and this information is used while updating HA table entries. (Switch Start of flow and End of flow)	1
MaxLBReordTimeout	HY	HY	This parameter specifies the maximum reordering timeout value used to setup Load-balancing seamless path switching	1500
UpdatedStatsInterval_W2	PathChWlan	PathChWlan	This parameter specifies the time interval to request statistics information of 2G interface from WLAN driver. This parameter is used by pcwService module to configure the periodic timer and upon timer expiry, wlanManager module will trigger the request for stats update to the WLAN driver	1
UpdatedStatsInterval_W5	PathChWlan	PathChWlan	This parameter specifies the time interval to request statistics information of 5G interface from WLAN driver. This parameter is used by pcwService module to configure the periodic timer and upon timer expiry, wlanManager module will trigger the request for stats update to the WLAN driver	1
StatsAgedOutInterval_W2	PathChWlan	PathChWlan	Used to validate the stats message received from WLAN driver in 2G bands. If stats message received from driver is more recent than Age out interval, then generate the event to respective module, otherwise ignore this stats update	30
StatsAgedOutInterval_W5	PathChWlan	PathChWlan	Used to validate the stats message received from WLAN driver in 5G bands. If stats message received from driver is more recent than Age out interval, then generate the event to respective module, otherwise ignore this stats update	30
MediumChangeThreshold_W2	PathChWlan	PathChWlan	This parameter specifies the threshold limit for medium change event to notify the path selection service in 2G bands. Path Selection algorithm will populate all entries in HDefaultTable with default interface choices	10
MediumChangeThreshold_W5	PathChWlan	PathChWlan	This parameter specifies the threshold limit for medium change event to notify the path selection service in 5G bands. Path Selection algorithm will populate all entries in HDefaultTable with default interface choices	10

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 8-4 Multi-AP Coordinated Steering and Adaptive Path Selection Parameters (cont.)

Option	CFG Type	Section	Description	Default
LinkChangeThreshold_W2	PathChWlan	PathChWlan	This parameter specifies the threshold limit for link change event to notify the path selection service in 2G bands. Path Selection algorithm will populate all entries in HDefaultTable with default interface choices	10
LinkChangeThreshold_W5	PathChWlan	PathChWlan	This parameter specifies the threshold limit for link change event to notify the path selection service in 5G bands. Path Selection algorithm will populate all entries in HDefaultTable with default interface choices	10
PHYRateThresholdForMU_W2	PathChWlan	PathChWlan	This parameter is used in calculating the medium utilization based on raw link capacity in 2G bands.If raw link capacity is greater than PHYRateThresholdForMU_W2 value then actual medium utilization value is divided by two	2000
PHYRateThresholdForMU_W5	PathChWlan	PathChWlan	This parameter is used in calculating the medium utilization based on raw link capacity in 5G bands. If raw link capacity is greater than PHYRateThresholdForMU_W5 value then actual medium utilization value is divided by two	2000
EnableProbe_W2	PathChWlan	PathChWlan	This parameter is used to enable/disable probe packets between AP and STA in 2G bands. AP sends packet to each Station in TD database, but whereas Station sends the packet to AP only.	1
EnableProbe_W5	PathChWlan	PathChWlan	This parameter is used to enable/disable probe packets between AP and STA in 5G bands. AP sends packet to each Station in TD database, but whereas Station sends the packet to AP only.	1
ProbePacketInterval_W2	PathChWlan	PathChWlan	This parameter configures the time interval for consecutive probe packets in 2G bands.	1
ProbePacketInterval_W5	PathChWlan	PathChWlan	This parameter configures the time interval for consecutive probe packets in 5G bands.	1
ProbePacketSize_W2	PathChWlan	PathChWlan	This parameter configures the packet size of each probe packets in 2G bands.	64
ProbePacketSize_W5	PathChWlan	PathChWlan	This parameter configures the packet size of each probe packets in 5G bands.	64
AssocDetectionDelay_W5	PathChWlan	PathChWlan	Time after which Topology database is updated after association. (Delay to ensure stability)	5
EntryExpirationInterval	PathChPlc	PathChPlc	Time after which if no activity(tx/rx) has happened from the PLC device, then entry is removed from the DB	120
LCThresholdForUnreachable	PathChPlc	PathChPlc	Threshold to mark the device unreachable	5
LCThresholdForReachable	PathChPlc	PathChPlc	Threshold to mark the device reachable	10
HostPLCInterfaceSpeed	PathChPlc	PathChPlc	This parameter specifies the medium oversubscription threshold for plc interface. If link is oversubscribed then event is sent to trigger the path selection module to switch the flows accordingly	0

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 8-4 Multi-AP Coordinated Steering and Adaptive Path Selection Parameters (cont.)

Option	CFG Type	Section	Description	Default
MaxMediumUtilization	PathChPlc	PathChPlc	In a daisy chain topology implemented for interoperation of Wi-Fi SON with PLC, which allows an RE to connect to another RE and to extend the range further in Wi-Fi links in conjunction with PLC interfaces, Aggregate link capacity rate and medium utilization for PLC interface are supported. Dynamic Link Metrics is supported for PLC devices. To enable Dynamic Link Metrics, set the ■ In /etc/config/hyd , the MaxMediumUtilization should be set to □ 40 for AR742x Ethernet speed 100Mbps (uci set hyd.PathChPlc.MaxMediumUtilization='40') □ 80 for AR742x Ethernet Host Speed 1Gbps (uci set hyd.PathChPlc.MaxMediumUtilization='80') □ 80 for QCA75xx Ethernet Host Speed 1Gbps (uci set hyd.PathChPlc.MaxMediumUtilization='80')	80
MediumChangeThreshold	PathChPlc	PathChPlc	If the medium utilization changes by this value between 2 sampling then an event is triggered.	10
LinkChangeThreshold	PathChPlc	PathChPlc	If any link capacity changes by this value between 2 sampling then an event is triggered.	10
StatsAgedOutInterval	PathChPlc	PathChPlc	Indicates the length of time to go without obtaining fresh statistics before the medium is considered uncharacterized.	60
UpdateStatsInterval	PathChPlc	PathChPlc	Controls the frequency at which the driver is sampled to update the medium statistics.	1
MaxMediumUtilizationForLC	PathChPlc	PathChPlc	Typically should be set equal to MaxMediumUtilization configuration parameter. Used for calculation of LC: prorates the full link capacity to provide a remainder link capacity. Range is 0-100.	80
ENABLE_NOTIFICATION_UNICAST	Topology	Topology	Enable this option to Send 1905 multicast packets as unicast packets for better reliability. This helps to increase chances to reach the destination	15
ND_UPDATE_INTERVAL	Topology	Topology	This parameter specifies the Neighbor Discovery update interval (TD1)	3
BD_UPDATE_INTERVAL	Topology	Topology	This parameter specifies the Bridged Discovery update interval (TD2)	190
HOLDING_TIME	Topology	Topology	This parameter specifies the maximum holding time. This parameter is used as boundary check in calculating the Neighbor discovery update interval and Bridged discovery update interval, where the calculated value cannot exceed the holding time. This holding time is also used to remove the aged H-Active entries	7
TIMER_LOW_BOUND	Topology	Topology	This parameter specifies the Lower bound value used in Random timer value generation. This Random value is used in configuring the HA table aging update interval, Neighbor Discover update interval and Bridged Discovery update interval	11

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 8-4 Multi-AP Coordinated Steering and Adaptive Path Selection Parameters (cont.)

Option	CFG Type	Section	Description	Default
TIMER_UPPER_BOUND	Topology	Topology	This parameter specifies the Upper bound value used in Random timer value generation. This Random value is used in configuring the HA table aging update interval, Neighbor Discover update interval and Bridged Discovery update interval	64
MSGID_DELTA	Topology	Topology	This parameter specifies the message ID delta. This parameter is used to validate the received 1905 packets and discard the older packets if an	120
HA_AGING_INTERVAL	Topology	Topology	Time interval to trigger the aging check for H-Active table entries.	1
ENABLE_TD3	Topology	Topology	Used by topology service to enable loop avoidance logic block for any topology changes in the network	1
ENABLE_BD_SPOOFING	Topology	Topology	This parameter is used to enable Bridged Discovery packets spoofing	1
NOTIFICATION_THROTTLING_WINDOW	Topology	Topology	This parameter specifies the 1905 notification message throttling window duration. Messages are accumulated during time period and sent after the timeout	15
PERIODIC_QUERY_INTERVAL	Topology	Topology	This parameter specifies the periodic query interval. This parameter is used to send the Topology Query message periodically to all devices to ensure it has up-to-date information	1
UpdateHDInterval	PathSelect	PathSelect	This parameter specifies the time interval to update the H-Default table. HDefaultTable is programmed with the interface to use for both UDP and "other" traffic for all entries in the topology. This helps to Populates all entries in HDefaultTable with default interface choices	10
UDPInterfaceOrder	PathSelect	PathSelect	This parameter specifies the Default order of interface selection for UDP flows	EP5 2
NonUDPInterfaceOrder	PathSelect	PathSelect	This parameter specifies the Default order of interface selection for Non-UDP flows	EP5 2
SerialflowIterations	PathSelect	PathSelect	This parameter specifies the Number of attempts to find a flow to switch	10
DeltaLCThreshold	PathSelect	PathSelect	This parameter specifies the threshold limit to select the capable interface for flow switching	10
UpdateHSPECInterval	HspecEst	HspecEst	This parameter specifies the HSPEC Estimation service update interval (in seconds). Also updated for each active flow is generated	1
NotificationThresholdLimit	HspecEst	HspecEst	This parameter has no usage	10
NotificationThresholdPercentage	HspecEst	HspecEst	This parameter has no usage	20
AlphaNumerator	HspecEst	HspecEst	This parameter is used to Calculate the weighted average flow rate. Weighted average value is calculated from newValue and oldValue. Older value is scaled by AlphaNumerator / AlphaDenominator	3

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 8-4 Multi-AP Coordinated Steering and Adaptive Path Selection Parameters (cont.)

Option	CFG Type	Section	Description	Default
AlphaDenominator	HspecEst	HspecEst	This parameter is used to Calculate the weighted average flow rate. Weighted average value is calculated from newValue and oldValue. Older value is scaled by AlphaNumerator / AlphaDenominator	8
LocalFlowRateThreshold	HspecEst	HspecEst	This parameter specifies the Maximum rate for a TCP flow to remain "local"	2000000
LocalFlowRatioThreshold	HspecEst	HspecEst	Used in QCA-Local Flow information messages only. This parameter is used to set/clear the bridge local flow	5
MaxHActiveEntries	HspecEst	HspecEst	Maximum number of H-Active entries maintained in user-space. Note: that since the number of entries in the kernel can grow without bound (until limited by the amount of memory on the device), care should be taken to size this appropriately	8192
EnableLog	LogSettings	LogSettings	Enable Hybrid daemon "log" service	0
LogRestartIntervalSec	LogSettings	LogSettings	This parameter specifies Log service restart interval (in Seconds)	10
LogServerIP	LogSettings	LogSettings	This parameter specifies Log service Server IP address	192.168.1.10
LogServerPort	LogSettings	LogSettings	This parameter specifies Log service Server Port Information	5555
EnableLogPCW2	LogSettings	LogSettings	Enable/Disable Logging of Path Characterization in WLAN 2.4G Interface	1
EnableLogPCW5	LogSettings	LogSettings	Enable/Disable Logging of Path Characterization in WLAN 5G Interface	1
EnableLogPCP	LogSettings	LogSettings	Enable/Disable Logging of Path Characterization in PLC Interface	1
EnableLogTD	LogSettings	LogSettings	Enable/Disable Logging of Topology Discovery service	1
EnableLogHE	LogSettings	LogSettings	Enable/Disable Logging of HE Service	1
EnableLogPS	LogSettings	LogSettings	Enable/Disable Logging of Path Selection Service	1
LogHEThreshold1	LogSettings	LogSettings	Threshold to use at the time of logging HE table H-Active entries	200000
LogHEThreshold2	LogSettings	LogSettings	Threshold to use at the time of logging HE table H-Active entries	10000000
StrictIEEE1905Mode	IEEE1905	IEEE1905	This parameter is used to enable/disable IEEE1905 Strict mode. If strict mode is enabled, then QCA extensions are not used.	0
GenerateLLDP	IEEE1905	IEEE1905	Enable/Disable LLDP message generation	1
V1Compat	HCP	HCP	This parameter specifies the Hi-Fi Control Packet Compatibility mode. Configurable values are 0,1,2 (0-OFF, 1-ALLOW, 2-ON). This parameter is enabled, only if legacy mode is enabled and a legacy device was detected. Used in processing BD and TD messages	

Table 8-4 Multi-AP Coordinated Steering and Adaptive Path Selection Parameters (cont.)

Option	CFG Type	Section	Description	Default
WlanALDNLNumOverride	WLAN	WLAN	Override the Netlink number of hyd core with the specified number	0
WlanALDMulticast	WLAN	WLAN	Enable/Disable sending Netlink messages to Multicast group instead of unicast	0
DisableSteeringMax11kUnfriendlyClients	config	Monitor	Enable this parameter to treat clients that advertise support for 802.11k but do not return a beacon report when requested as legacy clients. After a client fails to send back report after 10 consecutive requests, it is treated as a legacy client	1
ChanUtilSafetyThreshold_W2	Offload	Offload	Percentage of medium utilization that the measured plus projected utilization can reach before all further upgrade steering is disallowed until a new utilization measurement is done.	50
ChanUtilSafetyThreshold_W5	Offload	Offload	Percentage of medium utilization that the measured plus projected utilization can reach before all further upgrade steering is disallowed until a new utilization measurement is done.	60
MultiAP.EnableIEEE1905Assist,	MultiAP	MultiAP	The Map Service Module of HYD has been extended to include MapPublicService to handle exposure of IEEE1905 messaging in Mesh Layer. This provide logic to create buffers creating the data as per TLVs defined for EasyMesh (MAP) spec. These buffers can be sent to applications running on top of hyd such as Mesh Application using socket defined in dispatch manager. The module is disabled by default. To enable it, a new config parameter, hyd.MultiAP.EnableIEEE1905Assist, has been added.	

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 8-4 Multi-AP Coordinated Steering and Adaptive Path Selection Parameters (cont.)

Option	CFG Type	Section	Description	Default
EnableZeroBss	MultiAP	MultiAP	The AP radio BSS basic capability is needed by the controller to determine the agent and its details such as the number of radios supported by agent, Maximum transmit power by radio, maximum number of BSS supported by each radio, operating class of each radio and non-operable channels list, which comes as part of M1 from agent. On extender node, the hyd and wsplcd configuration files contain ap radio basic capability for the HK3R SBS (RDP385) platforms as default values since this feature has been primarily developed for HK 3R. Also, ensure that the AP basic radio capability has been updated in UCI config files with new values if this feature needs to be enabled in platforms other than HK3R platform. However, this feature is not supported in other platforms except IPQ807xA/IPQ817x three-radio SBS platforms. An important point to note is that, the feature is spread in both components hyd and wsplcd and feature can be enabled by setting global UCI option EnableZeroBss in hyd config. Both hyd and wsplcd daemons should be restarted if zero BSS feature is enabled or disabled at runtime dynamically.	
AlwaysFloodUpdate	config	mc	Starting with QCA_Networking_2020.SPF.11.3 release, this feature is implemented on IPQ401x, IPQ806x, IPQ807x/IPQ817x/IPQ807xA, IPQ50xx, IPQ60xx, and QCN90xx/QCN60xx attached to IPQ807xA, IPQ60xx platforms. Starting with the IPQ5018.ILQ.11.4 release, this feature is extended to QCN61xx chipsets attached to IPQ50xx platforms. New configuration "AlwaysFloodUpdate" will be added to hyd configuration, when enabled will ensure that flood table is always updated. This is disabled by default and enabled on need basis on customer request. uci set hyd.mc.AlwaysFloodUpdate=1 uci commit hyd /etc/init.d/repacd restart When "AlwaysFloodUpdate" is enabled and the device keeps receiving IGMP join and leave messages continuously, there is scope for flood table getting filled up completely which will result in discarding new entries addition and may halt traffic for some time until the ageing timer completes.	0

8.12 Range Extender Placement and Auto-configuration Daemon

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Range extender placement and auto-configuration* section of the *Wi-Fi SON: Additional features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

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

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

1. Configure the SSID and pass-phrase on the Wi-Fi interfaces as desired. It is not necessary to enable the Wi-Fi interfaces as repacd will do this itself.
2. Enable and start **repacd**:

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

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

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

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

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

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

   b. Static IP address

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

3. Enable and start **repacd**:

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

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

Beyond this basic configuration, the parameters in the Table 8-5 can further control the behavior of **repacd**. The parameters in Table 8-6 can also per set on the per-Wi-Fi device or per-Wi-Fi interface level to further control the behavior (as described in the table).

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:14 PDT zhou.yafeng@zte.com.cn

Table 8-5 Replacement and Auto-Configuration Daemon Parameters

Configuration Type	Section	Option	Description	Default
config	repacd	Enable	Whether the RE placement and auto-configuration logic is enabled or not	0
config	repacd	ManagedNetwork	The name of the network where the Wi-Fi interfaces being managed will reside.	lan
config	repacd	DeviceType	The primary role of the device. Must be one of `RE` or `Client`. In `Client` mode, the device will only operate as a range extender if its connection to the CAP falls into the desired range.	RE
config	repacd	Role	The current role `CAP` or `NonCAP` for this device. This should generally not be changed directly, as the value is set by the init script and read by the daemon.	NonCAP
config	repacd	ConfigREMode	The mechanism to use for range extension. Supported values are: `auto`, `son`, `wds`, `qwrap`, and `extap`. In `auto` mode, the RE will configure itself based on the detected configuration of the root AP. The exact behavior is further controlled by the `DefaultREMode` parameter below. Note that QWrap and ExtAP mode do not currently support the full credential cloning logic.	wds
config	repacd	DefaultREMode	The fallback mode to use when the central AP is not detected to be running in full Wi-Fi SON or WDS mode. This can be one of `qwrap` or `extap`.	qwrap
config	repacd	BlockDFSChannels	Whether to disable DFS channels when creating AP interfaces. This may be removed in the future now that the existing VAP configuration can be reused when starting repacd.	0
config	repacd	EnableSteering	When operating in WDS mode, whether single AP band steering should be enabled.	1
config	repacd	EnableSON	When operating in full Wi-Fi SON mode, whether to enable Multi-AP Coordinated Steering. This is generally only intended for use in debugging or testing where steering is not desired. For best performance in production, leaving this feature enabled is recommended.	1

Table 8-5 Replacement and Auto-Configuration Daemon Parameters (cont.)

Configuration Type	Section	Option	Description	Default
config	repacd	LinkCheckDelay	The amount of time (in seconds) to wait between successive link checks. Note that the actual amount of time between two link checks may be 1 second larger than this (due to implementation considerations).	2
WiFiLink	WiFiLink	MinAssocCheckPostWPS	The number of times the association must be deemed up after a WPS button press before it is considered stable enough before an RSSI measurement can begin.	5
config	repacd	TrafficSeparationEnabled	When operating in full Wi-Fi SON mode, whether to enable multi SSID and traffic separation support.	0
config	repacd	NetworkGuest	When multi SSID and traffic separation is enabled this parameter is used to hold the guest network's bridge name. If multi SSID and traffic separation is disabled then this parameter has no effect.	guest
config	repacd	NetworkGuestBackhaulInterface	When multi SSID and traffic separation is enabled this parameter is used to hold the guest network's backhaul interface.	2.4G
config	repacd	EnableEthernetMonitoring	Indicates whether the ethernet monitoring logic is enabled. When EnableEthernetMonitoring is enabled, the device monitors link state and prevents bridge loops. The device will also detect port direction(upstream or downstream).	0
WiFiLink	WiFiLink	WPSTimeout	The amount of time (in seconds) to wait for an association to take place after the WPS button is pressed. If this amount of time elapses without the STA interface associating, the device will be assumed to be too far from the CAP.	180
WiFiLink	WiFiLink	AssociationTimeout	The amount of time (in seconds) to wait for the STA interface to associate before considering the device as too far from any uplink node. Note that a WPS push button cancels this timer and runs the WPS timeout instead.	300
WiFiLink	WiFiLink	RSSINumMeasurements	The number of measurements to take to arrive at an average RSSI to compare against the near/far thresholds.	5
WiFiLink	WiFiLink	RSSIThresholdFar	The signal level (in dBm) below which the RE is considered too far from the CAP and should be moved closer.	-75

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 8-5 Replacement and Auto-Configuration Daemon Parameters (cont.)

Configuration Type	Section	Option	Description	Default
WiFiLink	WiFiLink	RSSIThresholdNear	The signal level (in dBm) above which the RE is considered too close to the CAP and should be moved farther.	-60
WiFiLink	WiFiLink	RSSIThresholdMin	The signal level (in dBm) above which a device whose primary role is as a client is eligible to become a range extender (so long as it does not exceed RSSIThresholdNear).	-75
LEDState	Varies	Name_1 Name_2	The name of the LED configuration section (in `/etc/config/system`) to use to resolve this to a SysFS name.	
LEDState	Varies	Trigger_1 Trigger_2	The mode in which the LED should operate. `none` - Solid on or off `timer` - Blinking	
LEDState	Varies	Brightness_1	The value to set for the LED brightness. At least on AP148, the brightness does not seem to matter, so a value of `1` should be used for on and a value of 0 for off.	
LEDState	Varies	DelayOn_1 DelayOn_2	The amount of time (in milliseconds) the LED should stay on. This is only relevant if the corresponding trigger is set to `timer`.	
LEDState	Varies	DelayOff_1	The amount of time (in milliseconds) the LED should stay off. This is only relevant if the corresponding trigger is set to `timer`.	
WiFiLink	WiFiLink	RSSIThresholdPrefer2GBackhaul	The signal level (in dBm) on 5 GHz STA VAP, below which the 5 GHz STA VAP is brought down.	-100
WiFiLink	WiFiLink	2GBackhaulSwitchDownTime	The duration of time during which the 2.4 GHz backhaul interface must be down before the 5 GHz backhaul is brought back up.	10
WiFiLink	WiFiLink	MaxMeasuringStateAttempts	Number of consecutive times 5 GHz interface can go down while doing an RSSI measurement before it is considered unstable and is forcefully disabled.	3
repacd	repacd	EnableBest5gBackhaulSelection	By default, this parameter is set to 0 in case if a tri-radio repeater having three STA VAPs. A customer can enable this flag to run best uplink on two 5 GHz STA VAPs' to determine the best node. Enabling this flag will delay the first boot stabilization time.	0

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 8-5 Replacement and Auto-Configuration Daemon Parameters (cont.)

Configuration Type	Section	Option	Description	Default
WiFiLink	WiFiLink	.MinBestUplinkRetry	By default, this parameter is set to 5. For DFS channels, increase the retry count to 10. This will increase the retry timer and delay the stabilization process.	5
config	repacd	AdditionalFHCount	Wi-Fi SON config for repacd to auto create additional VAPs with default ssid_x and key. Default Value: 0. For example: uci set repacd.repacd.AdditionalFHCount=3 For Wi-Fi SON with 2R repacd auto-configuration supported devices, VAP deletion should be done manually using UCI command and repacd config "AdditionalFHCount" count should be decremented.	0

Table 8-6 Wireless configuration file parameters

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

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

- `NotAssociated` - STA interface is still trying to associate.
- `WPSInProgress` - WPS button was pressed and the timeout has not yet occurred.
- `Measuring` - STA is associated and an average downlink RSSI value is being computed.
- `WPSTimeout` - Failed to establish an association within WPSTimeout seconds.
- `AssociationTimeout` - Failed to establish an association within `AssociationTimeout` seconds.
- `RE_MoveCloser` - RSSI is too weak or the STA was unable to associate.
- `RE_MoveFarther` - RSSI is too strong (duplicating coverage).
- `RE_LocationSuitable` - RSSI is sufficient for the backhaul without too much coverage overlap.
- `CL_LinkSufficient` - RSSI is sufficient for the device to act as a client device but is not sufficient for it to become a range extender.
- `CL_LinkInadequate` - RSSI is too weak or the device cannot even associate.
- `CL_ActingAsRE` - RSSI is in the sweet spot to allow the device to act as an RE while continuing to meet the client requirements.

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

8.13 Wi-Fi SON with PLC: UCI commands to enable Dynamic Link Metrics

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Wi-Fi SON and PLC* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Support for daisy chain topology is implemented for interoperation of Wi-Fi SON with PLC, which allows an RE to connect to another RE and to extend the range further in Wi-Fi links in conjunction with PLC interfaces. Also, support for star topology is introduced, which enables a CAP to be connected to all repeaters or REs using Wi-Fi and PLC interfaces.

Aggregate link capacity rate and medium utilization for PLC interface are supported. Dynamic Link Metrics is supported for PLC devices.

The following usage guidelines apply for full support of Dynamic Link Metrics for PLC devices:

1. PLC interface MAC address must be the same as PLC device MAC address.

   a. For an external PLC device, the MAC address of PLC device learned at initialization of PLCHOSTSRV is updated as local MAC address. (The OUI of PLC device is addressed.)

   b. For REH172, the host's br-lan MAC address + 2 is applied to PLC chipset.

2. Chipset ID is obtained from the PLC device through management message entry (MME) and the threshold value is set for Medium Utilization.

3. Cumulative Phy Rate Tx/Rx is required for more than 2 PLC nodes in the AVLN.

4. Dynamic Link metrics is expected to work for PLC device of same chipset ID.

5. A combination of AR742x and QCA75xx chipsets in the same AVLN causes data imbalance, while calculating the Average Link Capacity.

6. Perform the following workaround to test the Dynamic Link Metrics in PLC

   a. Set the PLC interface MAC address to PLC device using EDM

   b. Set the fixed MAC address in IPQ4019 either using UCI commands or at the uboot prompt.

To enable Dynamic Link Metrics, set the following parameters before starting hyd and PLC daemon:

- In /etc/config/hyd, the HostPLCInterfaceSpeed should be set to
  - 100 for AR742x Ethernet speed 100Mbps (uci set hyd.PathChPlc.HostPLCInterfaceSpeed='100')

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

  - 500 for AR742x Ethernet Host Speed 1Gbps (uci set hyd.PathChPlc.HostPLCInterfaceSpeed='500')
  - 1000 for QCA75xx Ethernet Host Speed 1Gbps (uci set hyd.PathChPlc.HostPLCInterfaceSpeed='1000')
- In /etc/config/hyd , the MaxMediumUtilization should be set to
  - 40 for AR742x Ethernet speed 100Mbps (uci set hyd.PathChPlc.MaxMediumUtilization='40')
  - 80 for AR742x Ethernet Host Speed 1Gbps (uci set hyd.PathChPlc.MaxMediumUtilization='80')
  - 80 for QCA75xx Ethernet Host Speed 1Gbps (uci set hyd.PathChPlc.MaxMediumUtilization='80')
- In /etc/config/plc AggrLinkRate should be set to zero.
  - uci set plc.config.AggrLinkRate='0'

8.14 Co-ordinated ATF between root AP and repeater

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Co-ordinated ATF between root AP and repeater* section of the *Memory and bandwidth optimization features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

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

8.14.1 Enable coordinated ATF feature in wsplcd config file

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

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

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

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

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

8.14.2 Config section 'atf-config'

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

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

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

8.14.3 Config section 'atf-re-config'

This section need to be created per Repeater & per ATF rule. Each section can have one ATF configuration say, either 1 SSID based or 1 Peer based. Multiple rules should not be clubbed together under a single section. If there are multiple ATF rules to be configured for a repeater, multiple such sections should be created with Rename pointing to the same Repeater Identifier.

In other words, the number of 'atf-re-config' section depends on the number of ATF rules that is intended to be configured. 'REname' is used to map 'atf-re-config' section with 'atf-config' section & thereby to the Repeater for which this configuration is intended for.

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

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

SSID Based ATF configuration

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

Peer Based ATF configuration

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

Group Based ATF configuration

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

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

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

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

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

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

Sample configuration file

The following sample configuration file represents ATF configuration for 3 Repeaters

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

Repeater 2: 3 SSID based configurations

Repeater 3: 2 Peer based configurations

config atf-config 'Repeater1'

```
option REmac '00:01:02:03:04:05'
option REname 'RE0'
```

config atf-config 'Repeater2'

```
option REmac '00:01:02:03:04:06'
option REname 'RE1'
```

config atf-config 'Repeater3'

```
option REmac '00:01:02:03:04:07'
option REname 'RE2'
```

config atf-re-config

```
option REname 'RE0'
option ssid 'rootap_AP'
```

```
    option ssid_wifidev 'ath01'
    option ssid_val '60'
    option ssid_cmd 'addssid'
```

config atf-re-config

```
    option REname 'RE0'
    option sta '00:33:33:33:33:33'
    option sta_wifidev 'ath01'
    option sta_val '10'
    option sta_cmd 'addsta'
```

config atf-re-config

```
    option REname 'RE0'
    option group 'group1'
    option group_wifidev 'ath01'
    option group_val '20'
    list group_ssid 'ssid1'
    list group_ssid 'ssid2'
    list group_ssid 'ssid3'
    option group_cmd 'addgroup'
    option group_enable '1'
```

config atf-re-config

```
    option REname 'RE1'
    option ssid 'ssid_RE1'
    option ssid_cmd 'addssid'
    option ssid_wifidev 'ath0'
    option ssid_val '10'
```

config atf-re-config

```
    option REname 'RE1'
    option ssid 'ssid_RE1_2'
    option ssid_cmd 'addssid'
    option ssid_wifidev 'ath0'
    option ssid_val '20'
```

config atf-re-config

```
    option REname 'RE1'
    option ssid 'ssid_RE1_3'
    option ssid_cmd 'addssid'
    option ssid_wifidev 'ath0'
    option ssid_val '20'
```

config atf-re-config

```
    option REname 'RE2'
    option sta '00:22:22:22:22:22'
    option sta_wifidev 'ath01'
    option sta_val '10'
    option sta_cmd 'addsta'
```

config atf-re-config

```
    option REname 'RE2'
    option sta '00:44:44:44:44:44'
```

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

```
option sta_wifidev 'ath01'
option sta_val '10'
option sta_cmd 'addsta'
```

8.15 Enable LOWI (Locationing Service)

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Location wireless interface (LOWI)* section of the *WLAN AP modes* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

LOWI is default locationing application which provides command line option to perform ranging (RTT: round trip time) and triggering IEEE 802.11mc frames. By default, LOWI service is disabled.

LOWI service are enabled by either adding following as new sub-section in /etc/config/wireless file

```
config lowi 'lowi'
option enable '1'
```

OR use following UCI commands

```
uci set wireless.lowi=lowi
uci set wireless.lowi.enable=1
uci commit
```

Execute ‘wifi’ command in order to restart wireless interfaces and apply new LOWI service. When interface comes up, LOWI service running in background can be checked via ‘ps’ command (one should LOWI process /usr/sbin/lowi-server). Note that LOWI service is independent of VAP or radio. It automatically picks radio/vap to use based on content and type of each LOWI message.

8.16 Avoid module reload during wifi up/down

Instead of reloading all the wifi modules during wifi up/down, just a firmware reload could help in reducing memory free/alloc cycles. During the last VAP removal, the firmware is unloaded and during the first VAP creation, the firmware is re-initialized.

With the new implementation, ‘wifi’ will just unload and reload the firmware. For normal configuration changes, ‘wifi’ is sufficient.

If any module parameters are changed or if the user specifically wants to reload the modules, a new command ‘wifi load’ has been added to reload the modules.

List of module params for which wifi load needs to be passed so that the modules are reloaded:

```
testmode
vow_config
ol_bk_min_free
ol_be_min_free
ol_vi_min_free
ol_vo_min_free
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
ar900b_emu
frac
intval
atf_mode
fw_dump_options
enableuartprint
ar900b_20_targ_clk
qca9888_20_targ_clk
max_descs
max_peers
qwrap_enable
otp_mod_param
max_active_peers
enable_smart_antenna
nss_wifi_olcfg
max_clients
max_vaps
enable_smart_antenna_da
prealloc_disabled
lteu_support
enable_mesh_support
enable_mesh_peer_cap_update
```

For other uci changes, normal wifi command is sufficient.

8.17 Content aware routing

To enable this feature, set 1 and to disable set 0.

```
uci set wireless.wifiX.tid_override_queue_map=<0,1>
uci commit
```

8.18 Support for SFE on IPQ8064 platforms-Coexistence between offload engines

Three types of coexistence with acceleration engines are present. These three types of coexistences are as follows:

- Coexistence of offload engines
- Coexistence of connection manager
- Coexistence of front-end in ECM

8.18.1 Select acceleration engine using UCI command

Select the acceleration engine using the UCI configuration file "/etc/config/ecm".

Table 8-7 Acceleration engine selection parameters

Configuration Type	Section	Option	Description
ecm	ecm	"auto" | "nss" | "sfe"	Select an acceleration engine for ECM. "auto" means automatic selection according to board type. For example, IPQ40xx board will select SFE, and IPQ8064 board will select NSS.

The shell script "/etc/init.d/qca-nss-ecm" reads the ecm UCI configuration, and converts the "acceleration_engine" UCI option to the "front_end_selection" kernel module parameter.

Users can change the preceding UCI configuration and then run the "/etc/init.d/qca-nss-ecm restart" command to select different acceleration engines for ECM.

8.18.2 Select acceleration engine automatically

To enable ECM to select acceleration engine, based on the board type, do the following:

1. Configure DUT with following UCI configuration:

   ```
   ecm.global=ecm
   ecm.global.acceleration_engine='auto'
   ```

2. Restart ECM by entering the following command:

   ```
   "/etc/init.d/qca-nss-ecm restart"
   ```

3. Check directory "/sys/kernel/debug/ecm":

   There must be two subdirectories "ecm_nss_ipv4" and "ecm_nss_ipv6" in IPQ8064 platforms such as "AP148", "AP161". This means NSS acceleration engine is selected.

   There must be two subdirectories "ecm_sfe_ipv4" and "ecm_sfe_ipv6" in IPQ40xx platforms such as "DK01", "DK04". This means SFE acceleration engine is selected.

4. Run TCP or UDP traffic from LAN to WAN, and ensure that traffic is accelerated by acceleration engine. For SFE, use the "sfe_dump" command to confirm traffic is accelerated by SFE. For NSS, check the CPU load to confirm traffic is accelerated by NSS.

   In IPQ8064 platforms such as "AP148", "AP161", NSS must be selected. Hence traffic must be accelerated by NSS. In IPQ40xx platforms such as "DK01", "DK04", SFE must be selected. Therefore, traffic must be accelerated by SFE.

8.18.3 Select NSS acceleration engine explicitly

To select NSS acceleration engine using the ecm UCI configuration, do the following:

1. Configure DUT with following UCI configuration:

   ```
   ecm.global=ecm
   ```

```
ecm.global.acceleration_engine='nss'
```

2. Restart ECM by entering the following command:

   ```
   /etc/init.d/qca-nss-ecm restart
   ```

3. Check the directory "/sys/kernel/debug/ecm".

   There must be two subdirectories "ecm_nss_ipv4" and "ecm_nss_ipv6" in IPQ8064 platforms such as "AP148", "AP161". This means NSS acceleration engine is selected.

   There must not be directory "/sys/kernel/debug/ecm" in IPQ40xx platforms such as "DK01", "DK04". This means NSS acceleration engine is not supported in IPQ40xx platform.

4. Run TCP or UDP traffic from LAN to WAN, make sure traffic is accelerated by NSS acceleration engine. Check the CPU load to confirm traffic is accelerated by NSS.

   In IPQ8064 platforms such as "AP148", "AP161", NSS must be selected. Hence traffic must be accelerated by NSS.

   In IPQ40xx platforms such as "DK01", "DK04", ECM must fail to load. Hence traffic must not be accelerated.

8.18.4 Select SFE acceleration engine explicitly

To select SFE acceleration engine using the ecm UCI configuration, do the following:

1. Configure DUT with following UCI configuration:

   ```
   ecm.global=ecm
   ecm.global.acceleration_engine='sfe'
   ```

2. Restart ECM by running the following command:

   ```
   "/etc/init.d/qca-nss-ecm restart"
   ```

3. Check the "/sys/kernel/debug/ecm" directory.

   There must always be two subdirectories "ecm_sfe_ipv4" and "ecm_sfe_ipv6". This means SFE acceleration engine is selected.

   SFE is a pure software acceleration engine. It is supported in both IPQ8064 and IPQ40xx platform. This means SFE acceleration engine is always able to run.

4. Run TCP or UDP traffic from LAN to WAN, make sure traffic is accelerated by SFE acceleration engine. Use command "sfe_dump" to confirm traffic is accelerated by SFE.

   In both IPQ8064 and IPQ40xx platforms, SFE must be selected. Hence traffic must be accelerated by SFE.

8.18.5 Select NSS acceleration engine in IPQ40xx platform

To select NSS acceleration engine in IPQ40xx platform to ensure that ECM fails to load, do the following:

1. Configure DUT with the following UCI configuration:

   ```
   ecm.global=ecm
   ecm.global.acceleration_engine='nss'
   ```

2. Restart ECM by running the following command:

   ```
   /etc/init.d/qca-nss-ecm restart"
   ```

3. Check the"/sys/kernel/debug/ecm" directory

   The "/sys/kernel/debug/ecm" directory must not be present in IPQ40xx platforms such as "DK01", "DK04". This means NSS acceleration engine is not supported in IPQ40xx platform.

4. Run TCP or UDP traffic from LAN to WAN, and ensure traffic is not accelerated. Check CPU load to confirm traffic is not accelerated.

   In IPQ40xx platforms such as "DK01", "DK04", ECM must fail to load. Therefore, traffic must not be accelerated.

8.18.6 Select unknown acceleration engine

To select an unknown acceleration engine to ensure ECM fails to load, do the following:

1. Configure DUT with the following ecm UCI configuration:

   ```
   ecm.global=ecm
   ecm.global.acceleration_engine='xxx'
   ```

2. Restart ECM by entering the following command:

   ```
   "/etc/init.d/qca-nss-ecm restart"
   ```

3. Check the "/sys/kernel/debug/ecm" directory.

   The "/sys/kernel/debug/ecm" directory must not be present because the acceleration engine is unknown.

4. Run TCP or UDP traffic from LAN to WAN, make sure traffic is not accelerated. Examine the CPU load to confirm traffic is not accelerated.

   ECM must fail to load. Therefore, traffic must not be accelerated.

8.19 Disable Wi-Fi SON blacklist functionality in band steering

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Disable Wi-Fi SON blacklist functionality in band steering* section of the *Wi-Fi SON: Additional features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

The blacklist functionality from band steering is made available as a configurable option. To enable or disable this functionality, a configuration parameter called AUTH_ALLOW is added to the lbd config file. When this feature is enabled, a new IOCTL, "WLANIFBSTEERCONTROL_AUTH_ALLOW", is sent to the driver which will set a new flag across the MAC address present in the ACL list in the driver.

This IOCTL is sent along with the other IOCTL that sets the probe response for withholding flag. The IOCTL to clear the Auth Allow flag is also sent when the probe response withholding flag is cleared. When the event ATH_EVENT_BSTEERING_DBG_TX_AUTH_ALLOW is received

from the driver then the print indicating that the Authentication is allowed due to the setting of this flag.

In the driver, a new flag "IEEE80211_ACL_FLAG_AUTH_ALLOW" is added in the ACL entry. When this flag is set, Authentication response is allowed. This flag is set or cleared whenever the lbd sends an ioctl to the driver for the particular MAC address.

A wifitool command is also present to set/clear the IEEE80211_ACL_FLAG_AUTH_ALLOW flag for the MAC address present in the ACL list. When there is Authentication request for the particular MAC for which the AUTH ALLOW flag is set then Authentication is allowed. When authentication is allowed due to the setting of the flag then an event ATH_EVENT_BSTEERING_DBG_TX_AUTH_ALLOW is sent to the lbd which will print the corresponding message.

The configuration parameter to enable the blacklist functionality in band steering is present is the config file. Under the IdleSteer configuration section of the config file, set the AUTH_ALLOW option as 1 (enable) or 0 (disable).

Enter the UCI command as follows:

```
uci set lbd.IdleSteer.AUTH_ALLOW=1
uci commit lbd
/etc/init.d/lbd start
```

Enabling this feature might bring down the steering success rate. Steering might fail for those clients that would send an AUTH directly on previously serving band when being steered. The steering success rate characterization and tuning would be ultimately left to the customer.

When this feature is enabled using the configuration parameter, then it means it is applicable for both the bands. In other words, this is a global feature that is applicable across all Radios & cannot be enabled/disabled on individual radios.

The T_Steering timer logic is not affected. The timer is started whenever the First Authentication rejection happens. This feature can be enabled or disabled only through lbd. This feature cannot be configured using command line such as iwpriv or wifitool. The lbd daemon is restarted, whenever this configuration parameter is changed so that the latest changes becomes effective.

8.20 UCI commands to specify separate Wi-FI SON AP steering threshold values for 2.4 GHz and 5 GHz bands

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Separate Wi-FI SON AP steering threshold values for 2.4 GHz and 5 GHz bands* section of the *Wi-Fi SON: Additional features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

The functionality to configure different AP steering threshold values for 2.4 GHz and 5 GHz bands using UCI commands and the lbd configuration file are introduced. UCI commands and additional parameters per band are implemented to allow configuration of the LowRSSIAPSteerThreshold_CAP and LowRSSIAPSteerThreshold_RE HYD parameters separately for 2.4 GHz and 5 GHz.

LowRSSIAPSteerThreshold_CAP, LowRSSIAPSteerThreshold_RE are existing non-band specific lbd configuration parameters. These are used **only in multi-AP setup**. On AP, they are

present in default lbd.config file i.e. /etc/config/lbd. They can be set via UCI command or directly in /etc/config/lbd. When hyd service starts, **lbd-config.sh** script to parse these multi-AP specific parameters and finally stores it in **/tmp/hyd.conf** file on AP. Hyd service takes values from /tmp/hyd.conf.

Additional band-specific parameters are introduced with _W2 and _W5 suffixes, which represent 2.4 GHz and 5 GHz band respectively. The lbd-config.sh file is modified for parsing these parameters. Rule for parsing is that if band-specific AP Steering parameter is present, it will be used, else it will fall back on non-band generic AP Steering parameter value.

NOTE These new options are not present in default lbd config file (/etc/config/lbd). User needs to configure them explicitly.

The following UCI options are added to configure AP Steering thresholds for 2.4 GHz and 5 GHz bands. The _W2 and _W5 suffixes in these UCI options represent 2.4 GHz and 5 GHz bands respectively.

```
LowRSSIAPSteerThreshold_CAP_W2
LowRSSIAPSteerThreshold_RE_W2
LowRSSIAPSteerThreshold_CAP_W5
LowRSSIAPSteerThreshold_RE_W5
```

Original non-band specific options LowRSSIAPSteerThreshold_CAP , LowRSSIAPSteerThreshold_RE still exists and will remain in default lbd config file.

When band-specific AP Steering threshold is present, it will be used, else non-band specific AP Steering parameter value will be used.

Band specific AP Steering values can be either set via UCI commands or directly in /etc/config/lbd file.

To configure the band-specific AP steering values using CLI commands:

```
uci set lbd.APSteer.LowRSSIAPSteerThreshold_CAP_W2=25
   uci set lbd.APSteer.LowRSSIAPSteerThreshold_CAP_W5=55
   uci set lbd.APSteer. LowRSSIAPSteerThreshold_RE_W2=25
   uci set lbd.APSteer. LowRSSIAPSteerThreshold_RE_W5=55
```

To configure the band-specific AP steering values using the **/etc/config/lbd** file:

```
config APSteer 'APSteer'
option LowRSSIAPSteerThreshold_CAP_W2 '25'
option LowRSSIAPSteerThreshold_CAP_W5 '55'
option LowRSSIAPSteerThreshold_RE_W2 '25'
option LowRSSIAPSteerThreshold_RE_W5 '55'
```

To set up a multi-AP and range extender and enter the band-specific AP steering parameters using the UCI commands, do the following:

On CAP, enable 2 GHz and 5 GHz VAPs, enable radio resource measurement (RRM), and wireless distribution system (WDS).

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

Bring up the wireless interfaces using the wifi command.

Disable mcsd (as it cannot run at the same time as hyd):

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

Enable and start **hyd**:

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

Configure the range extender (RE) in a similar manner.

To validate the UCI options for low RSSI AP steering threshold for central AP and RE, enter the following UCI commands or add new band-specific parameters directly in /etc/config/lbd file.

```
uci set lbd.APSteer.LowRSSIAPSteerThreshold_CAP_W2=25
   uci set lbd.APSteer.LowRSSIAPSteerThreshold_CAP_W5=55
   uci set lbd.APSteer. LowRSSIAPSteerThreshold_RE_W2=25
   uci set lbd.APSteer. LowRSSIAPSteerThreshold_RE_W5=55
   uci commit
```

Restart **hyd** service by entering the following command:

```
root@OpenWrt:~# /etc/init.d/hyd start
net.bridge.bridge-nf-call-custom = 1
hyd: restarting daemon
```

New values for 2 GHz and 5 GHz are reflected in /tmp/hyd.conf in [WLANIF2G] and [WLANIF5G] respectively as shown in the following portion of the output:

NOTE

- hyd service uses values from /tmp/hyd.conf, so final values are verified from this file.
- **LowRSSIAPSteeringThreshold** is the name used by hyd for AP Steering threshold

8.21 UCI command to specify backhaul for root AP and RE for enhanced WPS in Wi-Fi SON

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Enhanced WPS for RE to establish PBC connection in Wi-Fi SON* section of the *Wi-Fi SON: Additional features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

In the Wi-Fi SON network with multiple nodes, an enhancement to the WPS mechanism is implemented for the push-button event from Root AP to be propagated to one of the VAP on the node in the SON domain so RE is able to establish the PBC connection.

This functionality requires WPS PBC to be always performed in Root AP WPS and propagated to other node. It is applicable only for routers in backhaul VAP. When WPS PBC is pushed in

ROOTAP, Action frame is transmitted to the Range Extender (RE) AP's to listen for probe Request with WPS IE with PBC enabled from connecting RE. WPS is enabled in the non-backhaul AP's until WPS is started in backhaul VAP.

After WPS with PBC in probe request is received in RE from the new RE, action frame from RE to Root AP is transmitted to notify the presence of WPS client. The root AP cancels the WPS in all radios and transmits action frames to the first RE from which the action frame was received to start WPS. Immediately after the Action frame to start WPS is received in RE, WPS PBC event is triggered based on the action frame received

The following five types of action Frames are used for the implementation:

1. Action frame from Root AP to RE's to listen for WPS IE in probe request. (WPS_LISTEN)
2. Action frame from RE's to RootAP to inform the WPS client. (WPS_INFORM_ROOT)
3. Action frame from Root AP to RE to start the WPS.(START_WPS)
4. Action frame from RootAP to all REs to disable wps once wps is already started at Root AP(DISABLE_WPS)
5. Action frame from Root AP to particular RE(when RE informs of wps ie in probe request) to disable WPS after WPS is already started at one of the REs(STOP_WPS)

To set the backhaul for Root AP and RE (both AP and STA VAPs), enter the following UCI command:

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

8.22 Disable satellite fronthaul during backhaul disconnection

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Interface manager application* section of the *Repeater access point functions* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

An application that functions as an interface manager to bring down and bring up AP VAPs on all radios based on STA VAP connection is implemented. This application is started as part of post qcawifi script. The application is restarted every time during Wi-Fi down and Wi-Fi up events. The post qcawifi script creates a configuration file based on the specified UCI options at "/var/run/iface_mgr.conf" path. The configuration file is created in the following format:

```
mode=1                  //1-global_wds
timeout=50           // timeout to bring down AP VAPs, when STA VAP goes
down.
group=0 ap_vap=ath0
group=0 sta_vap=ath01
group=0 ap_vap=ath1
```

By default, all AP VAPs and STA VAPs are added on group 0. If user wants to change the group for any VAP, enter the following UCI commands for corresponding VAP.

```
uci set wireless.@wifi-iface[0].group=2
uci commit
```

When application starts, it reads inputs from this configuration file.

Two timeouts are provided by user. Stavap_disconnection_timeout is the time that can be used by STA VAP to establish reconnection with RootAP. RootAP_reconfiguration_timeout is the time that can be used to reconfigure RootAP.

A maximum of two timeouts are added on configuration file. The application derives timeout value from configuration file. The default Stavap_disconnection_timeout is 10 seconds and the default RootAP_reconfiguration_timeout is 60 seconds. When the user changes timeout value through iwpriv cmd, the application obtains updated timeout value from driver. UCI and iwpriv commands also are provided to update timeout value on driver and configuration file.

```
uci set wireless.qcawifi.discon_time='20'
uci set wireless.qcawifi.reconfig_time='45'

iwpriv wifix discon_time 20
iwpriv wifix gdiscon_time
iwpriv wifix reconfig_time 45
iwpriv wifix greconfig_time
```

8.22.1 Network socket connection with wpa_supplicant

This application establishes network socket connection with wpa_supplicant. When STA VAP connection goes up or down, application will get CONNECT/DISCONNECT message from wpa_supplicant accordingly. When last sta vap connection goes down on any group, bring down all AP VAPs belong to that group after specified timeout. When first sta vap connection comes up on any group, bring up all AP VAPs belong to that group.

Because the application is taking care of bringing down/up AP VAPs based on STA VAP connection. Changes need to be made on driver for not to bring up and bring down AP VAPs when STA VAP connects or disconnects with RootAP.

This application runs only when global_wds flag is set through UCI.

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

The WDS repeater operates in Enhanced Independent repeater (EIR) mode when this feature is enabled.

8.23 UCI command to configure fast_lane mode in tri-radio boards

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Interface manager application* section of the *Repeater access point functions* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Fast lane is created between one STA VAP of one 5 GHZ radio and one AP VAP of another 5 GHZ radio in a tri-radio board.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

To enable this feature, user has to enable fast_lane option on both the radios that are intended to create a fast lane with by entering the following UCI command:

```
uci set wireless.wifix.fast_lane='1'
```

Also, user has to set pref_uplink option only on one radio that is preferred to be an uplink by entering the following command:

```
uci set wireless.wifix.pref_uplink='1'
```

The radios having fast_lane option enabled will act as an enhanced independent repeater. In iface_mgr_setup (), it will be checked that in which radio fast_lane option has been enabled and accordingly set the operating mode (iface_mgr_op_mode) to 2. If the operating mode is 2, then as per the pref_uplink option script will classify the sub-interfaces (VAPs) into different groups.

There will be two groups (group0 and group1). If operating mode is 2 and pref_uplink is enabled, then AP VAP of that radio will be in group0 and STA VAP of that radio will be in group1.

Similarly if operating mode is 2 but pref_uplink option is not set, then AP VAP of the corresponding radio will be in group1 and STA VAP will be in group0.

In this manner, the groups for fast lane are created at: /var/run/iface_mgr.conf.

The configuration file is created in the following format:

```
mode=2 // 2 indicates fast lane mode
timeout=60// Timeout for the primary group STA to get connected to the
ROOT
group=1 ap_vap=ath1 group=0 sta_vap=ath11 group=0 ap_vap=ath2 group=1
sta_vap=ath21
```

Out of two groups, the group in which the STA VAP of the radio having pref_uplink option set; will be considered as Primary group and the other one will be considered as Secondary group.

- Initially, both the STA VAPs are scanning and the AP VAPs of both the groups are beaconing.
- Assume that one STA VAP is connected, then the application will check whether that STA VAP belongs to Primary group or Secondary group.
  - If STA VAP is from primary group, then bring up the AP VAPs that belong to Primary group if in any case they are down and bring down all the STA and AP VAPs belonging to the Secondary group if they are up.
  - If STA VAP is from secondary group; first bring up the AP VAPs of Secondary group and then register a timeout; so that in that time the STA belongs to Primary group can scan and search for ROOT AP. If STA from Primary group finds the ROOT AP and gets connected then the above step will be followed else at the end of the timer, the application will bring down all the AP and STA VAPs that belongs to the Primary group.
- In case of DISCONNECT message from wpa_supplicant, application will bring up only the STA VAP from other group:
  - When STA VAP from Primary group is connected that means all AP and STA VAPs from Secondary group will be in down state. Suppose STA VAP from Primary group got disconnected, then the application will bring up the STA VAP from Secondary group. No need to bring up the AP VAP of Secondary radio, since AP VAPS on both the radios will be having same SSID.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Make command

```
make package/feeds/qca_11_0/qca-iface-mgr-11.0/{clean,compile} V=s
```

8.24 Configure memory footprint reduction on QCA9880, QCA9886, and QCA9889

Perform the following to configure memory footprint reduction on QCA988x/QCA989x, QCA9880/QCA9881, and QCA9887/QCA9889 platforms:

1. Flashing instruction for Entry Level platform (Low memory QCA988x/QCA989x, QCA9880/QCA9881, and QCA9887/QCA9889 platforms with 8M flash + 64M DDR support) Image:

```
   "setenv bc 'ap147-8M'
   "setenv lou 'tftp 0x80060000 openwrt-ar71xx-${bc}-qca-legacy-
uboot.bin && erase 0x9f000000 +0x30000 && cp.b $fileaddr 0x9f000000
$filesize'
   "setenv lok 'tftp 0x80060000 openwrt-ar71xx-generic-${bc}-kernel.bin
&& erase 0x9f680000 +${filesize} && cp.b $fileaddr 0x9f680000 0x160000'
   "setenv lof 'tftp 0x80060000 openwrt-ar71xx-generic-${bc}-rootfs-
squashfs.bin && erase 0x9f050000 +${filesize} && cp.b $fileaddr
0x9f050000 $filesize'
   "setenv bootargs 'console=ttyS0,115200 root=31:02 rootfstype=jffs2
init=/sbin/init mtdparts=ath-nor0:256k(u-boot),64k(u-boot-
env),6336k(rootfs),1472k(uImage),64k(ART)'
   "setenv bootcmd 'bootm 0x9f680000'
   "run lou;re
   "run lok lof;re
```

2. Debug build (Crash scope enabled) image for entry level platform need to be flashed and tested on the 16MB flash + 128 MB DDR board, using 8MB+64MB DDR flash instruction mentioned above, then configure 16MB flash + 128 MB platform to up with 76MB DDR support using the following command:

```
   "setenv bootargs 'mem=76M console=ttyS0,115200 root=31:02
rootfstype=jffs2 init=/sbin/init mtdparts=ath-nor0:256k(u-boot),64k(u-
boot-env),6336k(rootfs),1472k(uImage),64k(ART)'
```

3. Install the Target FW binary packages (ipk) manually after the image is being flashed based on the Target Chip Set (WiFi Radio) connected to the Platform.
4. The following is the Target FW binary packages (ipk) list for different offload chipsets:
   - QCA988x/QCA989x: qca-wifi-fw-hw10-10.4-asic_WLAN.BL.3.4-00xxx-S-1-1_all.ipk
   - QCA9880/QCA9881: qca-wifi-fw-hw6-10.4-asic_CNSS.PS.2.0-00xxx-S-1-1_all.ipk
   - QCA9887/QCA9889: qca-wifi-fw-hw3-10.4-asic_CNSS.PS.2.0-00xxx-S-1-1_all.ipk
5. For Board calibration, utility packages related to calibration has to be installed manually after the image is being flashed.
6. Because the debug Image generally took more Flash space than normal Image, there will not sufficient flash memory available after flashing the Debug Image, to install Target FW binary packages and other utility packages, in which case Engineer need to install those packages on USB Stick to continue the test.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

a. Perform the following steps to install a packages (ipk) on USB Stick

   i. Format the USB Stick with NTFS File system.

   ii. Copy ntfs-3g_2011.4.12-1-fuseint_ar71xx.ipk or any compatible version of ntfs-3g ipk to board's /tmp location and install using "opkg install ntfs-3g_2011.4.12-1-fuseint_ar71xx.ipk"

   iii. Issue `mkdir -p mnt/usb` command and add "dest usbmount /mnt/usb" in etc/opkg.conf

   iv. Copy the package (ipk) to board's /tmp location and install using "opkg install -d usbmount tmp/<package name>.ipk

   v. In case of Target FW binary package issue the below commands and reboot the platform.

```
`ln -fs /mnt/usb/lib/firmware/QCA9888 lib/firmware/QCA9888`
`ln -fs /mnt/usb/lib/firmware/AR9888 lib/firmware/AR9888`
`ln -fs /mnt/usb/lib/firmware/AR9887 lib/firmware/AR9887
`touch /etc/rc.d/S10lowmempltform`
`chmod 777 /etc/rc.d/S10lowmempltform`
```

b. Copy the following lines to /etc/rc.d/S10lowmempltform file

```
#!/bin/sh
mount /dev/sda1 /mnt/usb
/sbin/wifi detect > /tmp/wireless.tmp
[ -s /tmp/wireless.tmp ] && {
  cat /tmp/wireless.tmp >> /etc/config/wireless
}
rm -f /tmp/wireless.tmp
wifi load
```

c. In case of some utility package create appropriate soft links. Foe example, in case of art utility create soft link to /mnt/usb/etc/init.d/

```
ln -s /mnt/usb/etc/init.d/art /etc/init.d/art
```

d. Export library & executable path using following commands after installing the package or after every reboot

```
mount /dev/sda1 /mnt/usb >/dev/null 2>&1
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/mnt/usb/usr/lib
export PATH=$PATH:/mnt/usb/usr/sbin/
```

7. Because packet log feature is disabled by default for entry level platforms, the user can enable the packet log feature by issuing following UCI commands and then reloading the WiFi Driver:

```
uci set wireless.qcawifi=qcawifi
uci set wireless.qcawifi.enable_pktlog_support='1'
uci commit wireless
```

Limitations

- This feature allow only 2 VAP with 16 Client support per Radio.
- The following Wi-Fi features are not supported for entry level platform:
- Support for legacy chipset (AR5416), Spectral, WIFIPOS, HYFI Enhancements, QWRAP, Smart antenna, Airtime fairness, Band steering and LOWI support, WMI event logging.

- SKB Recycler is disabled for this platform because of which a limitation exists with throughput and CPU usage.
- Install the Target FW binary packages manually after the image is being flashed as per the attached Radio.
- Because entry level platform support 8M flash and out of which ~6M is reserved for Root FS, only ~512K flash memory can be available for utility packages.

8.25 Suspension of probes in 2.4 GHz band

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Suspension of probes in 2.4 GHz band* section of the *Wi-Fi SON: Band steering and APS* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

On top of single AP band steering, an add-on functionality to enable suspension of probes in 2.4 GHz band is introduced. This feature allows user to configure the AP such that probe responses are suppressed on 2.4 GHz band, which effectively forces a dual band client to select 5 GHz band at association time.

Supported SPs: QCA9531.ILQ.4.0, QCA9563.ILQ.4.0, and QCA9558.ILQ.4.0

Supported chipsets: QCA9880, QCA9889, QCA9531, and QCA9886

By default, when a dual band client tries to associate with an AP, client scans on both the bands 2.4 GHz and 5 GHz. Therefore, it is possible for a client to connect initially with 2.4 GHz band and then be pushed to 5 GHz band based on band-steering decisions.

To prevent such a scenario, the user can configure the Load Balancing Daemon (lbd) for suppressing the 2.4 GHz probe responses from APs for minimal time and probe request count. This feature is enabled only if the received probe request RSSI of the client is greater than configured threshold. These parameters are configured in lbd configuration file under the "steerexec_adv" section.

By default, this feature is disabled and it can be enabled using configuring the aforementioned parameters with values other than zero. The probe responses are suppressed for Delay24GProbeMinReqCount times in Delay24GProbeTimeWindow time.

See Table 8-3 for more information.

8.26 Configure round-trip time on AP using UCI commands

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *location wireless interface (LOWI)* section of the *WLAN AP modes* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Wi-fi Location certification program is based on IEEE802.11mc where clients and AP use Fine Timing Measurements (FTMs) to do ranging (also known as RTT - Round trip time). To enable ranging or RTT capability on interface X on AP, following commands can be used:

```
uci set wireless.@wifi-iface[X].enable_rtt='1'
uci set wireless.@wifi-iface[X].enable_lci='1'
uci set wireless.@wifi-iface[X].enable_lcr='1'
uci set wireless.@wifi-iface[X].rrm='1'
uci commit
```

8.27 Enable or disable cfg80211 for UCI configuration

After boot-up, the **wifi** script finds available configurations provided by the driver. See **/etc/config/wireless** for the default configuration.

To enable cfg80211 for UCI:

```
uci set qcacfg80211.config.enable='1'
uci commit.
wifi detect > /etc/config/wireless

config wifi-device wifi0
option type qcawificfg80211
option channel auto
option macaddr 8c:fd:f0:00:e4:d7
option hwmode 11ac
# REMOVE THIS LINE TO ENABLE WIFI:
option disabled 1
config wifi-iface
option device wifi0
option network lan
option mode ap
option ssid OpenWrt
option encryption none
```

The Qualcomm Technologies proprietary driver has different configuration types, such as qcawifi or qcawificfg80211. The default mode is qcawifi (WEXT Mode). All parameters/configuration items are the same for WEXT and UCI except that UCI changes the line:

```
option type qcawificfg80211
```

To disable cfg80211 for UCI (WEXT Mode):

```
uci set qcacfg80211.config.enable='0'
uci commit.
wifi detect > /etc/config/wireless
```

NOTE Starting with WLAN driver, version 11.0, both WEXT mode and the Linux 802.11 configuration API, cfg80211tool mode, are supported. If a certain proprietary wireless tool command, such as `wlanconfig`, `wifitool`, or `apstats`, is entered, the driver checks the mode that is being currently used—WEXT or cfg80211—on the system, and processes the command accordingly.

Qualcomm Confidential - May Contain Trade Secrets 2021-05-11 03:57:44 PDT pan.yafeng@zte.com.cn

8.28 UCI configuration of the multiwan

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Multiple WLAN and mwan3 overview* sections of the *Networking offload and acceleration of link and multilink features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Option	Default	Value range	Description
Section : config			
Enable	0	0 | 1	Enable multiwan or not
default_route	balancer	balancer/fastbalancer/<interface>	Select the default route for all the unspecified traffic
`debug`	0	0 | 1	0 : disable 1:enable
`health_monitor`	parallel	parallel/serial	Serial can save some system resource
Section : interface			
`weight`	10	*disable/1-10*	The weight in the load balance calculation
`health_interval`	10	*disable/5/10/20/30/60/120*	Health interval in seconds
`icmp_hosts`	dns	*disable/dns/gateway/<host>*	The Host to be pinged
`timeout`	3	*disable/1-5/10*	Time out in the icmp
`health_fail_retries`	3	*1/3/5/10/15/20*	Attempt before the fail over
`health_recovery_retries`	5	*1/3/5/10/15/20*	Attempt before the recovery
`failover_to`	fastbalancer	disable/balancer/fastbalancer/<interface>	Traffic failover destination

Qualcomm Confidential – May Contain Trade Secrets 2021-02-13 03:57:44 PDT pan.yafeng@zte.com.cn

dns	auto	*auto/<dns>*	The dns server's ip address, auto will use the dns get from the wan conneciton
Section : mwanfw			
src	All	*all/<IP>/<hostname>*	Src ip address
dst	all	*all/<IP>/<hostname>*	Destination ip address
port_type	dports	*dports/source-ports*	Dst port or src port
proto	all	*all/tcp/udp/icmp/<custom>*	Mapped port number in the switch
ports	all	*all/<port,port:range>*	Ports or port range
wanrule		*all/tcp/udp/icmp/<custom>*	Wan uplink
failover_to		*balancer/fastbalancer/<interface>*	Destionation when fail over

8.28.1 Sample multiwan connection

Consider a sample scenario in which two PPPoE WAN connections on the PPP server side are created. These connections are used to create two accounts for users, namely isp1 and isp2. Also , the multiwan configuration is created in this example.

To create the WAN connection, enter the following commands:

```
uci set network.wan=interface
uci set network.wan.proto=pppoe
uci set network.wan.ifname=eth0
uci set network.wan.ac=isp1
uci set network.wan.username=bob
uci set network.wan.password=test
uci set network.wan2=interface
uci set network.wan2.proto=pppoe
uci set network.wan2.ifname=eth0
uci set network.wan2.ac=isp2
uci set network.wan2.username=mary
uci set network.wan2.password=test
uci commit
/etc/init.d/network restart
```

To configure firewall, enter the following commands:

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
uci set firewall.@zone[1].network=
uci add_list firewall.@zone[1].network=wan
uci add_list firewall.@zone[1].network=wan2
uci add_list firewall.@zone[1].network=wan6
uci commit
/etc/init.d/firewall restart
```

To configure multiwan, enter the following commands:

```
uci set multiwan.wan=interface
uci set multiwan.wan.weight=3
uci set multiwan.wan.health_interval=4
uci set multiwan.wan.icmp_hosts=gateway
uci set multiwan.wan.timeout=3
uci set multiwan.wan.health_fail_retries=3
uci set multiwan.wan.health_recovery_retries=5
uci set multiwan.wan.failover_to=balancer
uci set multiwan.wan.dns=auto

uci set multiwan.wan2=interface
uci set multiwan.wan2.weight=3
uci set multiwan.wan2.health_interval=4
uci set multiwan.wan2.icmp_hosts=gateway
uci set multiwan.wan2.timeout=3
uci set multiwan.wan2.health_fail_retries=3
uci set multiwan.wan2.health_recovery_retries=5
uci set multiwan.wan2.failover_to=balancer
uci set multiwan.wan2.dns=auto
uci commit multiwan
/etc/init.d/multiwan restart
```

To set up customized rules, enter the following commands:

```
uci add multiwan mwanfw
uci set multiwan.@mwanfw[-1].src=192.168.1.0/24
uci set multiwan.@mwanfw[-1].dst=www.googole.com
uci set multiwan.@mwanfw[-1].wanrule=balancer
uci commit multiwan
/etc/init.d/multiwan restart
```

8.28.2 User interface for configuring mwan3

Two ways are provided to manage the mwan3: Command-Line or LuCI Web UI.

8.28.2.1 Command line interface

```
root@OpenWrt:~# mwan3
Syntax: mwan3 [command]
Available commands:
      start           Load iptables rules, ip rules and ip routes
      stop           Unload iptables rules, ip rules and ip routes
      restart        Reload iptables rules, ip rules and ip routes
      ifup <iface>    Load rules and routes for specific interface
```

```
            ifdown <iface>  Unload rules and routes for specific
interface
            interfaces      Show interfaces status
            policies        Show currently active policy
            connected    Show directly connected networks
            rules           Show active rules
            status          Show all status
```

8.28.2.2 LuCI Web interface

Log in the OpenWrt web interface, navigate mwan3 settings by clicking “Load Balancing” under “Network” tab in the head bar. All settings are under that subpage.

8.28.2.3 Prerequisites for configuring mwan3

The package can either be installed on OpenWrt web interface or be built as a part of the root file system while compiling QSDK APSS image. For the latter, two options need to be selected in QSDK menuconfig page:

- PACKAGE_luci-app-mwan3 (optional but advised)
- PACKAGE_mwan3

Ensure no other multiple WAN package is installed such as multiwan – having multiwan installed at the same time as mwan3 is known not to work.

8.28.2.4 Multiple WAN interfaces

At least two WAN interfaces are needed. Before using mwan3, ensure they’re working and the default routing table is correctly configured for each WAN connection.

A simple way to create multiple WAN interface is to create multiple VLANs on WAN interface if a router is VLAN-capable.

It’s important to configure a different routing metric for each WAN interface. This is compulsory. The metric will only have an effect on the default routing table, not on mwan3 routing table. The default (primary) WAN interface should have the lowest metric (e.g. 10) and each additional WAN interface should have a higher metric (e.g. 20, 30, etc). Values are not important, but should always be unique.

Each WAN interface should have a default gateway configured.

8.28.3 Configure mwan3

mwan3 configuration is in file /etc/config/mwan3. It consists of four section elements: interface, member, policy and rule.

8.28.3.1 Configure interface

For each WAN, it has an interface section. The section defines how each WAN interface is tested for up/down status.

Name	Type	Required	Default	Description
interface *name*	string	yes	(none)	Interface name shown in network configuration, e.g. wan/wan2
enabled	boolean	yes	(none)	Should mwan3 run on this interface
track_ip	list of ip addresses	no	(none)	The hosts to test if interface is still alive. If this value is missing the interface is always considered up
reliability	number	no	1	Number of track_ip hosts that must reply for the test to be considered as successful. Ensure there are at least this many track_ip hosts defined or the interface will always be considered down
count	number	no	1	Number of pings to send to each host with each test
timeout	seconds	no	4	Number of seconds to wait for an echo-reply after an echo-request
interval	seconds	no	10	Number of seconds between each test
up	number	no	5	Number of successful tests to considered link as alive
down	number	no	5	Number of failed tests to considered link as dead

8.28.3.2 Configure member

Each member represents an interface with a metric and weight value. Members are referenced in policies to define a pool of interfaces with corresponding metric and load-balancing weight. Members cannot be used for rules directly. The default settings are described below:

Name	Type	Required	Default	Description
interface	string	yes	(none)	Member applies to this interface (use the same interface name as used in the mwan3 interface section, above)
metric	number	no	1	Members within one policy with a lower metric have precedence over higher metric members
weight	number	no	1	Members with same metric will distribute load based on this weight value

A typical member section looks like this:

```
config 'member' 'wan1_m1_w3'
        option 'interface' 'wan1'
        option 'metric' '1'
        option 'weight' '3'
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

A working mwan3 config has at least 2 members configured.

8.28.4 Configure policy

Policies define how traffic is routed through the different WAN interface(s). Every policy has to have one or more members assigned to it, which defines the policy's traffic behavior. If a policy has a single member, traffic will only go out that member. If it has more than one member, it will either load-balancing among members or use one member but fail-over to another, depending on how members are configured.

If there is more than one member assigned to a policy, members within the policy with a lower metric have precedence over higher metric members. Members with the same metric will load-balance. Load-balancing members (with same metric) will distribute load based on assigned weights values.

A typical policy section appears as follows:

```
config policy 'balanced'
        list use_member 'wan_m1_w3'
        list use_member 'wan2_m1_w2'
```

If a policy is not referenced by a specific traffic rule, the policy will not do anything, so it is fine to leave unused policies in place in case they are desired in the future.

If you have a traffic rule that matches a policy, but all the members (interfaces) for that policy are down, the exit strategy for that policy defaults to "unreachable". This is configurable with the last_resort option. Valid values are: blackhole, unreachable or default.

A working mwan3 config has at least 1 policy configured.

Ensure no policy name is longer than 15 characters

8.28.4.1 Configure rule

A rule describes what traffic to match and what policy to assign for that traffic. These are the available options:

Name	Type	Required	Default	Description
use_policy	string	yes	(none)	Use this policy for traffic that matches or set to `default` to use the default routing table to lookup
src_ip	ip address	no	any	Match traffic from the specified source ip address
src_port	port or range	no	any	Match traffic from the specified source port or port range, if relevant `proto` is specified
proto	protocol	no	all	Match traffic using the given protocol. Can be one of `tcp`, `udp`, `icmp` or `all` or it can be a numeric value, representing one of these protocols or a different one
dest_ip	ip address	no	any	Match traffic directed to the specified destination ip address

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

dest_port	port or range	no	any	Match traffic directed at the given destination port or port range, if relevant `proto` is specified
ipset	string	no	(none)	Match traffic directed at the given destination ip address to an ipset set
sticky	boolean	no	0	Allow traffic from the same source ip address within the timeout limit to use same wan interface as prior session
timeout	number	no	600	Stickiness timeout value in seconds

8.28.5 Sample mwan3 settings on IPQ40x

8.28.5.1 Common settings for both NAT and route mode

DUT 1

```
/* WAN Interfaces */
uci set network.wan=interface
uci set network.wan.proto=static
uci set network.wan.ifname=eth0.10
uci set network.wan.ipaddr=172.16.10.10
uci set network.wan.netmask=255.255.255.0
uci set network.wan.gateway=172.16.10.20
uci set network.wan.metric=10

uci set network.wan2=interface
uci set network.wan2.proto=static
uci set network.wan2.ifname=eth0.20
uci set network.wan2.ipaddr=172.16.20.10
uci set network.wan2.netmask=255.255.255.0
uci set network.wan2.gateway=172.16.20.20
uci set network.wan2.metric=20

/* ADD VLANs */
uci add network switch_vlan
uci set network.@switch_vlan[-1].device=switch0
uci set network.@switch_vlan[-1].vlan=3
uci set network.@switch_vlan[-1].ports='0t 5t'
uci set network.@switch_vlan[-1].vid='10'

uci add network switch_vlan
uci set network.@switch_vlan[-1].device=switch0
uci set network.@switch_vlan[-1].vlan=4
uci set network.@switch_vlan[-1].ports='0t 5t'
uci set network.@switch_vlan[-1].vid='20'
uci commit

/etc/init.d/network restart
```

```
/* MWAN3 Settings */
echo '' > /etc/config/mwan3
uci set mwan3.wan=interface
uci set mwan3.wan.enabled='1'
uci add_list mwan3.wan.track_ip='172.16.10.20'
uci set mwan3.wan.reliability='1'
uci set mwan3.wan.count='1'
uci set mwan3.wan.timeout='2'
uci set mwan3.wan.interval='5'
uci set mwan3.wan.down='3'
uci set mwan3.wan.up='8'
uci set mwan3.wan2=interface
uci add_list mwan3.wan2.track_ip='172.16.20.20'
uci set mwan3.wan2.reliability='1'
uci set mwan3.wan2.count='1'
uci set mwan3.wan2.timeout='2'
uci set mwan3.wan2.interval='5'
uci set mwan3.wan2.down='3'
uci set mwan3.wan2.up='8'
uci set mwan3.wan2.enabled='1'
uci set mwan3.wan_m1_w10=member
uci set mwan3.wan_m1_w10.metric='1'
uci set mwan3.wan_m1_w10.weight='10'
uci set mwan3.wan_m1_w10.interface='wan'
uci set mwan3.wan2_m1_w10=member
uci set mwan3.wan2_m1_w10.metric='1'
uci set mwan3.wan2_m1_w10.weight='10'
uci set mwan3.wan2_m1_w10.interface='wan2'
uci set mwan3.balanced=policy
uci add_list mwan3.balanced.use_member='wan_m1_w10'
uci add_list mwan3.balanced.use_member='wan2_m1_w10'
uci set mwan3.default_rule=rule
uci set mwan3.default_rule.sticky='0'
uci set mwan3.default_rule.use_policy='balanced'
uci set mwan3.default_rule.timeout='10'
uci set mwan3.default_rule.proto='all'
uci set mwan3.default_rule.src_ip='192.168.1.0/24'
uci set mwan3.default_rule.dest_ip='192.168.3.0/24'
uci commit
mwan3 restart
```

DUT 2

```
/* LAN Interface */
uci set network.lan.ipaddr=192.168.3.1

/* WAN Interfaces */
uci set network.wan=interface
```

```
uci set network.wan.proto=static
uci set network.wan.ifname=eth0.10
uci set network.wan.ipaddr=172.16.10.20
uci set network.wan.netmask=255.255.255.0
uci set network.wan.gateway=172.16.10.10
uci set network.wan.metric=10

uci set network.wan2=interface
uci set network.wan2.proto=static
uci set network.wan2.ifname=eth0.20
uci set network.wan2.ipaddr=172.16.20.20
uci set network.wan2.netmask=255.255.255.0
uci set network.wan2.gateway=172.16.20.10
uci set network.wan2.metric=20

/* Add VLANs */
uci add network switch_vlan
uci set network.@switch_vlan[-1].device=switch0
uci set network.@switch_vlan[-1].vlan=3
uci set network.@switch_vlan[-1].ports='0t 5t'
uci set network.@switch_vlan[-1].vid='10'

uci add network switch_vlan
uci set network.@switch_vlan[-1].device=switch0
uci set network.@switch_vlan[-1].vlan=4
uci set network.@switch_vlan[-1].ports='0t 5t'
uci set network.@switch_vlan[-1].vid='20'
uci commit

/etc/init.d/network restart

/* MWAN3 Settings */
echo '' > /etc/config/mwan3
uci set mwan3.wan=interface
uci set mwan3.wan.enabled='1'
uci add_list mwan3.wan.track_ip='172.16.10.10'
uci set mwan3.wan.reliability='1'
uci set mwan3.wan.count='1'
uci set mwan3.wan.timeout='2'
uci set mwan3.wan.interval='5'
uci set mwan3.wan.down='3'
uci set mwan3.wan.up='8'
uci set mwan3.wan2=interface
uci add_list mwan3.wan2.track_ip ='172.16.20.10'
uci set mwan3.wan2.reliability='1'
uci set mwan3.wan2.count='1'
uci set mwan3.wan2.timeout='2'
uci set mwan3.wan2.interval='5'
```

```
uci set mwan3.wan2.down='3'
uci set mwan3.wan2.up='8'
uci set mwan3.wan2.enabled='1'
uci set mwan3.wan_m1_w10=member
uci set mwan3.wan_m1_w10.metric='1'
uci set mwan3.wan_m1_w10.weight='10'
uci set mwan3.wan_m1_w10.interface='wan'
uci set mwan3.wan2_m1_w10=member
uci set mwan3.wan2_m1_w10.metric='1'
uci set mwan3.wan2_m1_w10.weight='10'
uci set mwan3.wan2_m1_w10.interface='wan2'
uci set mwan3.balanced=policy
uci add_list mwan3.balanced.use_member='wan_m1_w10'
uci add_list mwan3.balanced.use_member='wan2_m1_w10'
uci set mwan3.default_rule=rule
uci set mwan3.default_rule.sticky='0'
uci set mwan3.default_rule.use_policy='balanced'
uci set mwan3.default_rule.timeout='10'
uci set mwan3.default_rule.proto='all'
uci set mwan3.default_rule.src_ip='192.168.3.0/24'
uci set mwan3.default_rule.dest_ip='192.168.1.0/24'
uci commit
mwan3 restart
```

PC1

```
/* Add route */
sudo ip route add 172.16.0.0/16 via 192.168.1.1
sudo ip route add 192.168.0.0/16 via 192.168.1.1
```

PC2

```
/* Add route */
sudo ip route add 172.16.0.0/16 via 192.168.3.1
sudo ip route add 192.168.0.0/16 via 192.168.3.1
```

8.28.5.2 Settings specific to NAT

```
uci set firewall.@defaults[0].forward=ACCEPT
uci set firewall.@defaults[0].drop_invalid='1'
uci add_list firewall.@zone[1].network=wan2
uci set firewall.@zone[1].input=ACCEPT
uci set firewall.@zone[1].forward=ACCEPT
uci commit
/etc/init.d/firewall restart
```

8.28.5.3 Settings specific to routing

```
uci set firewall.@defaults[0].forward=ACCEPT
```

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT panxiaoqing@zte.com.cn

```
uci set firewall.@defaults[0].drop_invalid='1'
uci add_list firewall.@zone[1].network=wan2
uci set firewall.@zone[1].input=ACCEPT
uci set firewall.@zone[1].forward=ACCEPT
uci set firewall.@zone[1].masq=0
uci commit
/etc/init.d/firewall restart
```

Qualcomm
Confidential – May Contain Trade Secrets
2021-05-13 03:57:44 PDT
pan.yafeng@zte.com.cn

8.29 User interface of link aggregation

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Link aggregation group* section of the *Networking offload and acceleration of link and multilink features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

8.29.0.1 Start/stop link aggregation

To start link aggregation, enter the /etc/init.d/lacpd start command.

To stop link aggregation, enter the /etc/init.d/lacpd stop command.

8.29.0.2 UCI configuration interface

Switch link aggregation can be configured by UCI configuration file "**/etc/config/lacpd**"

option	Value range	Description
Section : switch		
ifname	Name of Linux network device	Linux network device which represent the switch
Section : lacp		
enable	0 | 1	Enable lacp instance, user can define multiple instance
name	string	Name of lacp instance
txHashPolicy	"L2" | "L3" | "L2_L3"	Hash policy to select output port when transmit packet
lacpRate	0 | 1	Lacp packet sending rate, 0 : slow, 1 per 30 seconds; 1 : fast, 1 per 1 second.
adSelect	"all" | "stable" | "bandwidth" | "count"	All: all aggregator are active Stable: select only one aggregator, keep current active aggregator if it is still useable Bandwidth: select only one aggregator which has biggest bandwidth Count: select only one aggregator which has the most members
Section : port		
enable	0 | 1	Enable or disable lacp on this port
lacp	Name of lacp section	Specify a lacp instance which this port belongs to
switch	Name of switch section	Specify a switch which this port belongs to
switchPortId	1-4	Mapped port number in the switch
userKey	0-65535	User defined operation key, default is 0

The following is an example for file "/etc/config/lacpd"

```
config switch S17
        option ifname 'eth1'
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
config lacp lan
        option enable '1'
        option name 'lan'
        option txHashPolicy 'L2_L3'
        option lacpRate '0'
        option adSelect 'all'
config port port1
        option enable '1'
        option lacp 'lan'
        option switch 'S17'
        option switchPortId '1'
        option userKey '0'
config port port2
        option enable '1'
        option lacp 'lan'
        option switch 'S17'
        option switchPortId '2'
        option userKey '0'
config port port3
        option enable '1'
        option lacp 'lan'
        option switch 'S17'
        option switchPortId '3'
        option userKey '0'
config port port4
        option enable '1'
        option lacp 'lan'
        option switch 'S17'
        option switchPortId '4'
        option userKey '0'
```

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yaheng@ze.com.cn

8.30 UCI configuration for ACL

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *ACL-based steering* section of the *Networking offload and acceleration of tunnel and encryption features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

All switch core configurations can be implemented by UCI.

8.30.0.1 ACL related UCI configuration

There are two parts for ACL module:

- UDF profile part
- ACL rule part

UDF profile part

The following table shows the configuration format.

Table 8-8 UDF profile configuration format

UCI format	Description
config switch_ext option device 'switch0' option name 'AclUdfprofile' option port '2' option user_defined_type 'l2' option user_defined_offset '1' option user_defined_length '4'	Set user defined field offset and length on a particular port. § port = 0-6 § user_defined_type = l2/l2snap/l3/l3plus/l4 § user_defined_offset = offset value § user_defined_length = 0-15

In this case, the UCI profile configuration is as follows:

```
config switch_ext
   option device 'switch0'
   option name 'AclUdfprofile'
   option port '1'
   option user_defined_type 'l3'
   option user_defined_offset '22'
   option user_defined_length '2'

config switch_ext
   option device 'switch0'
   option name 'AclUdfprofile'
   option port '1'
   option user_defined_type 'l3plus'
   option user_defined_offset '31'
```

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT daiyafeng@zte.com.cn

```
        option user_defined_length '13'
```

ACL rule part

Table 8-9 ACL rule confirmation format

Qualcomm
Confidential – May Contain Trade Secrets
2021-05-13 03:57:44 PDT
pan.yafeng@zte.com.cn

UCI format	Description
config switch_ext /*MUST have field for one rule*/ option device 'switch0' option name 'AclRule' option rule_id '1' option priority '1' option rule_type 'ip4' option port_bitmap '0x1e' /*Basic L2 checking field*/ option dst_mac_address '00-00-00-00-00-00' option dst_mac_address_mask 'ff-ff-ff-ff-ff-ff' option src_mac_address '00-00-00-00-00-00' option src_mac_address_mask 'ff-ff-ff-ff-ff-ff' option ethernet_type '3' option ethernet_type_mask '0xff' option vlan_id '3' option vlan_id_mask '0xff' option vlan_priority '1' option vlan_priority_mask '1' option tagged '1' option tagged_mask '1' option cfi '1' /*STAG mode, enhanced L2 checking field*/ option ctagged '1' option ctag_vlan_id '3' option ctag_vlan_id_mask '0xff' option ctag_vlan_priority '1' option ctag_vlan_priority_mask '1' option ctag_cfi '1' option stagged '1' option stag_vlan_id '3' option stag_vlan_id_mask '0xff' option stag_vlan_priority '1' option stag_vlan_priority_mask '1' option stag_dei '1'	§ rule_id = 1-95 § priority = 1-95 § rule_type = MAC | IPv4 | IPv6 | UDF § dst_mac_address = destination MAC § dst_mac_address_mask = destination MAC mask § src_mac_address = source MAC § src_mac_address_mask = source MAC mask § ethernet_type = 0-0xffff § vlan_id = 0-4095 § vlan_priority = 0-7 § tagged = 0 | 1 § cfi = 0|1 § ctag_vlan_id = 0-4095 § ctag_vlan_priority = 0-7 § ctagged = 0 | 1 § ctag_cfi = 0 | 1 § stag_vlan_id = 0-4095 § stag_vlan_priority = 0-7 § staged = 0 | 1 § stag_cfi = 0 | 1 § ipv4_src_address = IPv4 source IP address § ipv4_dst_address = IPv4 destination IP § ipv6_src_address = IPv4 source IP address § ipv6_dst_address = IPv4 destination IP § ipv6_flow_label = IPv6 flow label § ip_protocol = 0-0xff § ip_dscp = 0-63 § ip_dst_port = IP destination port § ip_src_port = IP source port § icmp_type = ICMP type § icmp_code = ICMP code § tcp_flag = TCP flag § ripv1 = RIP packet § dhcpv4 = DHCPv4 packet § dhcpv6 = DHCPv6 packet § inverse_check_fields = 0 | 1 ú 0 = Do not inverse any field. ú 1 = Inverse all fields in this rule.

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:54:41 PDT pan.yafeng@qualcomm.com

/*IPv4 L3 checking field*/ option ipv4_src_address '1.1.1.1' option ipv4_src_address_mask '1.1.1.1' option ipv4_dst_address '1.1.1.1' option ipv4_dst_address_mask '1.1.1.1' /*IPv6 L3 checking field*/ option ipv6_src_address 'ff::00' option ipv6_src_address_mask 'ff::00' option ipv6_dst_address 'ff::00' option ipv6_dst_address_mask 'ff::00' option ipv6_flow_label '0x12345' option ipv6_flow_label_mask '0xfffff' /*IP L4 checking field*/ option ip_protocol '1' option ip_protocol_mask '1' option ip_dscp '1' option ip_dscp_mask '1' option ip_dst_port '3' option ip_dst_port_mask '0xff' option ip_src_port '3' option ip_src_port_mask '0xff' /*ICMP tcp rip dhcp flag checking field*/ option icmp_type '100' option icmp_type_mask '0xff' option icmp_code '100' option icmp_code_mask '0xff' option tcp_flag '100' option tcp_flag_mask '0xff' /*RIP, DHCP checking field*/ option ripv1 '1' option dhcpv4 '1' option dhcpv6 '1'	§ packet_drop = yes | no § redirect_to_cpu = yes | no § copy_to_cpu = yes | no § redirect_to_ports = destination ports bitmap § mirror = yes | no § dscp_of_remark = 0-63 § queue_of_remark = 0-7 § port_bitmap = define into which ports this rule binds. ú Bit[0] for port 0 ú Bit[1] for port 1 ú … ú Bit[6] for port 6 § remark_lookup_vid = yes | no § stag_vid_of_remark = 0-4095 § stag_priority_of_remark = 0-7 § stag_dei_of_remark = 0 | 1 § ctag_vid_of_remark = 0-4095 § ctag_priority_of_remark = 0-7 § ctag_dei_of_remark = 0 | 1 § action_policer_id = 0-31 § action_arp_ptr = 0-127 § action_wcmp_ptr = 0-3 § action_snat = yes | no § action_dnat = yes | no § bypass_egress_translation = yes | no § interrupt_trigger = yes | no

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 09:37:41 PDT pan.yafeng@qualcomm.com

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

/*rule forward action. If drop is yes, no action is valid. redirect_to_cpu and copy_to_cpu can't exist together.*/ option inverse_check_fields 'no' option packet_drop 'no' option redirect_to_cpu 'y' option copy_to_cpu 'y' option redirect_to_ports '0x1e' option mirror 'y' /*rule vlan change action*/ option remark_lookup_vid 'y' option stag_vid_of_remark '100' option stag_priority_of_remark '2' option stag_dei_of_remark '1' option ctag_vid_of_remark '100' option ctag_priority_of_remark '10' option ctag_dei_of_remark '4' /*rule dscp change action*/ option dscp_of_remark '1' /*rule queue change action*/ option queue_of_remark '1' /*rule rate limit action*/ option action_policer_id '1' /*rule L3 action. arp and wcmp cannot exist together; snat and dnat cannot exist together.*/ option action_arp_ptr '1' option action_wcmp_ptr '1' option action_snat 'y' option action_dnat 'y' /*rule egress qinq bypass action and interrupt trigger action*/ option bypass_egress_translation 'y' option interrupt_trigger 'y'	§

For ACL rule, only checked field must be considered, so the following ACL UCI can be used:

```
/*IPv4 rule*/
config switch_ext
   option device 'switch0'
   option name 'AclRule'
   option rule_id '10'
   option priority '10'
   option rule_type 'ip4'
   option ip_protocol '0x2f'
```

Qualcomm Confidential – May Contain Trade Secrets 2025-05-13 03:57:44 PDT pan.yaren@zte.com.cn

```
   option ip_protocol_mask '0xff'
   option user_defined_field_value '00-44'
   option user_defined_field_mask 'ff-ff'
   option action_arp_ptr '0x60'
   option port_bitmap '0x2'
/*IPv6 rule*/
config switch_ext
   option device 'switch0'
   option name 'AclRule'
   option rule_id '20'
   option priority '20'
   option rule_type 'ip6'
   option ip_protocol '0x2f'
   option ip_protocol_mask '0xff'
   option user_defined_field_value '00-00-00-00-00-00-00-00-00-00-00-00-
   00-00-00-44'
   option user_defined_field_mask '00-00-00-00-00-00-00-00-00-00-00-00-
   00-00-ff-ff'
   option action_arp_ptr '0x20'
   option port_bitmap '0x2'
```

8.30.0.2 ARP related UCI configuration

There are two main configurations:

- ARP
- Interface entry

ARP

The following table shows the ARP configuration format.

Table 8-10 ARP configuration format

UCI command	Description

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

config switch_ext option device 'switch0' option name 'IpHostentry' option entry_id '0' option entry_flags '1' option entry_status '0xf' option ip_addr '1.1.1.1' option mac_addr '00-00-00-00-00-11' option interface_id '0' opton load_balance_num ‘0’ option vrf_id ‘0’ option port_id '4' option action 'forward' option mirror 'no' option counter 'no'	Add a host entry with following parameter: § entryid = 0-1023 § entryflags ú #define FAL_IP_IP4_ADDR 0x1 ú #define FAL_IP_IP6_ADDR 0x2 ú #define FAL_IP_CPU_ADDR 0x4 § entry_status = 0-15 ú 0 = entry is empty. ú 1-7 = entry is dynamic and valid. ú 8-14 = entry is dynamic and valid, can be aged but cannot be changed by any other address. ú 15 = entry is static and cannot be aged or changed by hardware. § ip4 addr = IPv4 address § mac addr = MAC address § interface id = 0-4094 § load_balance_num = 0-3 (Only available in IPQ4018/IPQ4019/IPQ4028/IPQ4029 ESS) § vrf_id = 0-7 (Only available in IPQ4018/IPQ4019/IPQ4028/IPQ4029 ESS) § port id = 0-6 § action = forward § mirror = yes | no § counter = yes | no

IP4 ARP entry

```
config switch_ext
    option device 'switch0'
    option name 'IpHostentry'
    option entry_id '0'
    option entry_flags '1'
    option entry_status '7'
    option ip_addr '0.0.0.0'
    option mac_addr '00-00-00-00-00-00'
    option interface_id '1'
    option load_balance_num '4'
    option vrf_id '0'
    option port_id '0'
    option action 'rdtcpu'
    option mirror 'no'
    option counter 'no'
```

IPv6 ARP entry

```
config switch_ext
    option device 'switch0'
    option name 'IpHostentry'
    option entry_id '0'
```

Qualcomm Confidential – May Contain Trade Secrets 2022-05-13 03:27:47 PDT kanxiang@zte.com.cn

```
        option entry_flags '2'
        option entry_status '7'
        option ip_addr '1::9'
        option mac_addr '00-00-00-00-00-00'
        option interface_id '1'
        option load_balance_num '4'
        option vrf_id '0'
        option port_id '0'
        option action 'rdtcpu'
        option mirror 'no'
        option counter 'no'
```

8.30.0.3 Interface entry

The following table shows the interface entry configuration format.

Table 8-11 Interface entry configuration format

UCI command	Description
config switch_ext option device 'switch0' option name 'IpIntfentry' option entry_id '0' option vrf_id '0' option vlan_low '1' option vlan_high '1' option mac_addr '00-00-00-00-00-11' option ipv4_route 'yes' option ipv6_route 'yes'	Add interface entry. § entry_id = 0-7 § vrf_id = 0-7 (available in IPQ4018/IPQ4019/IPQ4028/IPQ4029 ESS only)

Interface entry UCI

```
    config switch_ext
        option device 'switch0'
        option name 'IpIntfentry'
        option entry_id '0'
        option vrf_id '0'
        option vlan_low '1'
        option vlan_high '1'
        option mac_addr 'ea-48-5b-d7-43-4c'
        option ipv4_route 'yes'
        option ipv6_route 'yes'
```

8.31 UCI configuration for multiple PPPoE sessions

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Multiple PPPoE sessions* section of the *Networking offload and acceleration of link and multilink features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

8.31.1 Create multiple PPPoE sessions

To create four PPPoE sessions, add the following lines in /etc/config/network file of the DUT:

```
config interface 'wan0'
        option ifname 'eth0'
        option proto 'pppoe'
        option username builder1
        option password 12341
config interface 'wan1'
        option ifname 'eth0'
        option proto 'pppoe'
        option username builder2
        option password 12342
config interface 'wan2'
        option ifname 'eth0'
        option proto 'pppoe'
        option username builder3
        option password 12343

config interface 'wan3'
        option ifname 'eth0'
        option proto 'pppoe'
        option username builder4
        option password 12344
```

Make sure to restart network service as follows:

```
# /etc/init.d/network restart
```

8.31.2 Create dual WAN port

To create dual WAN ports, all you need is to specify the second physical Ethernet port in the /etc/config/network file.

```
config interface 'wan0'
        option ifname 'eth0'
        option proto 'pppoe'
        option username builder
        option password 1234
```

Qualcomm Confidential – May Contain Trade Secrets 2021-05-19 03:57:44 PDT pan.yafeng@zte.com.cn

```
config interface 'wan1'
        option ifname 'eth2'
        option proto 'pppoe'
        option username builder
        option password 1234
```

Make sure to restart network service as follows:

```
# /etc/init.d/network restart
```

8.31.3 Display PPPoE statistics

Obtain the statistics by running following commands:

 # ifconfig pppo-wanX (X is the number specified in the /etc/config/network file)

Also, sar –n DEV 2 | egrep "IFACE|ppp" can be run to see the throughput and PPS on these PPPoE clients.

NSS host also provides these statistics however they are maintained by NSS FW:

 # cat /sys/kernel/debug/qca-nss-drv/stats/pppoe

8.31.4 UCI commands

uci set network.loopback=interface

uci set network.loopback.ifname=lo

uci set network.loopback.proto=static

uci set network.loopback.ipaddr=127.0.0.1

uci set network.loopback.netmask=255.0.0.0

uci set network.lan=interface

uci set network.lan.ifname='eth1'

uci set network.lan.type=bridge

uci set network.lan.proto=static

uci set network.lan.ipaddr=192.168.1.1

uci set network.lan.netmask=255.255.255.0

uci set network.lan.ip6addr=2aaa::1/64

uci set network.wan0=interface

uci set network.wan0.ifname=eth0

uci set network.wan0.proto=pppoe

uci set network.wan0.username='builder'

uci set network.wan0.password='builder'

```
uci set network.wan1=interface
uci set network.wan1.ifname=eth0
uci set network.wan1.proto=pppoe
uci set network.wan1.username='builder1'
uci set network.wan1.password='builder1'
uci set network.wan2=interface
uci set network.wan2.ifname=eth0
uci set network.wan2.proto=pppoe
uci set network.wan2.username='builder2'
uci set network.wan2.password='builder2'
uci set network.wan3=interface
uci set network.wan3.ifname=eth0
uci set network.wan3.proto=pppoe
uci set network.wan3.username='builder3'
uci set network.wan3.password='builder3'
uci set firewall.@defaults[0]=defaults
uci set firewall.@defaults[0].syn_flood=1
uci set firewall.@defaults[0].input=ACCEPT
uci set firewall.@defaults[0].output=ACCEPT
uci set firewall.@defaults[0].forward=ACCEPT
uci set firewall.@zone[0]=zone
uci set firewall.@zone[0].name=lan
uci set firewall.@zone[0].network=lan
uci set firewall.@zone[0].input=ACCEPT
uci set firewall.@zone[0].output=ACCEPT
uci set firewall.@zone[0].forward=ACCEPT
uci set firewall.@zone[1]=zone
uci set firewall.@zone[1].name=wan
uci set firewall.@zone[1].network=wan
uci set firewall.@zone[1].input=ACCEPT
uci set firewall.@zone[1].output=ACCEPT
uci set firewall.@zone[1].forward=ACCEPT
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
uci set firewall.@zone[1].masq=1
uci set firewall.@zone[1].mtu_fix=1
uci add firewall redirect
uci set firewall.@redirect[0]=redirect
uci set firewall.@redirect[0].target=DNAT
uci set firewall.@redirect[0].src=wan
uci set firewall.@redirect[0].dest=lan
uci set firewall.@redirect[0].proto='tcp udp'
uci set firewall.@redirect[0].name=DNAT
uci set firewall.@redirect[0].src_ip=192.168.2.2
uci set firewall.@redirect[0].src_dip=172.31.7.2
uci set firewall.@redirect[0].dest_ip=192.168.1.2
uci add firewall redirect
uci set firewall.@redirect[1]=redirect
uci set firewall.@redirect[1].target=DNAT
uci set firewall.@redirect[1].src=wan
uci set firewall.@redirect[1].dest=lan
uci set firewall.@redirect[1].proto='tcp udp'
uci set firewall.@redirect[1].name=DNAT
uci set firewall.@redirect[1].src_ip=192.168.2.3
uci set firewall.@redirect[1].src_dip=172.31.7.3
uci set firewall.@redirect[1].dest_ip=192.168.1.3
uci commit
uci add firewall redirect
uci set firewall.@redirect[2]=redirect
uci set firewall.@redirect[2].target=DNAT
uci set firewall.@redirect[2].src=wan
uci set firewall.@redirect[2].dest=lan
uci set firewall.@redirect[2].proto='tcp udp'
uci set firewall.@redirect[2].name=DNAT
uci set firewall.@redirect[2].src_ip=192.168.2.4
uci set firewall.@redirect[2].src_dip=172.31.7.4
```

uci set firewall.@redirect[2].dest_ip=192.168.1.4

uci commit

uci add firewall redirect

uci set firewall.@redirect[3]=redirect

uci set firewall.@redirect[3].target=DNAT

uci set firewall.@redirect[3].src=wan

uci set firewall.@redirect[3].dest=lan

uci set firewall.@redirect[3].proto='tcp udp'

uci set firewall.@redirect[3].name=DNAT

uci set firewall.@redirect[3].src_ip=192.168.2.5

uci set firewall.@redirect[3].src_dip=172.31.7.5

uci set firewall.@redirect[3].dest_ip=192.168.1.5

uci commit

Make sure to restart both firewall and network service:

/etc/init.d/firewall restart

/etc/init.d/network restart

If you need to enable debugging, then uncomment debug line in /etc/ppp/options file:

echo '#debug' > /etc/ppp/options

echo 'logfile /dev/null' >>/etc/ppp/options

echo 'noipdefault'>>/etc/ppp/options

echo 'noaccomp'>>/etc/ppp/options

echo 'nopcomp'>>/etc/ppp/options

echo 'nocrtscts'>>/etc/ppp/options

echo 'lock'>>/etc/ppp/options

echo 'maxfail 0'>>/etc/ppp/options

echo 'lcp-echo-failure 10000'>>/etc/ppp/options

echo 'lcp-echo-interval 1'>>/etc/ppp/

8.32 UCI commands to configure RADIUS parameters for retries

The following RADIUS parameters to configure the number of times for which an attempt is made to establish a connection with the RADIUS server, and the frequency or the interval at which these retries are made can be set using UCI commands:

- identity_request_retry_interval (Range 0 to 20)
- radius_server_retries (Range 0 to 10)

Enter the following command to specify a RADIUS server retry interval of 3 seconds:

```
uci set wireless.@wifi-iface[0]. identity_request_retry_interval =3
```

Enter the following command to specify the number of retries to be attempted with the RADIUS server as 5:

```
uci set wireless.@wifi-iface[0]. radius_server_retries =5
```

Commit the settings:

```
uci commit wireless
```

The following is an example of a UCI configuration file with the RADIUS server retry settings:

```
root@OpenWrt:/# vi /etc/config/wireless

config wifi-device 'wifi0'
        option type 'qcawifi'
        option macaddr '8c:fd:f0:01:42:c4'
        option hwmode '11ac'
        option disabled '0'
        option htmode 'HT80'
        option channel '36'
        option txchainmask '7'
        option rxchainmask '7'

config wifi-device 'wifi1'
        option type 'qcawifi'
        option channel 'auto'
        option macaddr '8c:fd:f0:00:bd:f3'
        option hwmode '11ng'
        option disabled '1'

config wifi-iface
        option network 'lan'
        option mode 'ap'
        option ssid 'SecurityMat_3'
        option encryption 'wpa2+ccmp'
        option device 'wifi0'
        option server '192.168.1.150'
        option port '1812'
        option key 'atheros123'
        option nss '3'
        option identity_request_retry_interval '3'
        option radius_server_retries '5'
```

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yifeng@ate.com.cn

Configure the AP in WPA2 EAP-TLS – AES CCMP using QSPR. Configure the identity_request_ retry_interval and radius_server_retries param values using UCI. Bring down the Ethernet interface so that the station cannot authenticate with the RADIUS server. Start the hostapd with debug enabled. Check for the maximum retry attempts in hostapd log . The default maximum retry attempts without the configuration is 10.

RADIUS: **Retry attempts :2 Maximum retry attempts :5**

ath0: RADIUS Next RADIUS client retransmit in 1 seconds

ath0: STA 00:03:7f:40:00:2e RADIUS: Resending RADIUS message (id=0)

RADIUS: **Retry attempts :4 Maximum retry attempts :5**

ath0: RADIUS Next RADIUS client retransmit in 5 seconds

ath0: STA 00:03:7f:40:00:2e RADIUS: Resending RADIUS message (id=1)

Check for identity_request_retry_interval . In default cases, time is calculated based on dynamic backoff and keeps increasing. The identity_request_retry_interval value is saved to the value specified using the parameter in UCI.

8.33 Disable hyd IGMP snooping and enable hardware snooping

The Hy-Fi daemon (hyd) does not support LAN-side IGMP snooping. Both port 0 and port 6 can be used as CPU port. However, if a customer has only one CPU port, it is recommended to use port 0 instead of port 6. S17 ASIC considers it as internal CPU port, and some default behavior to CPU port, for example, trap packets to CPU or forward packets to CPU, causing the packets to be forwarded to port 0 instead of port 6.

The software hyd IGMP snooping does not support LAN-side server that cannot forward IGMP join and leave messages to other S17c LAN port, if IGMP snooping is enabled. Therefore, if the device needs LAN side multicast IGMP snooping function, enter the following command to make eth1 as a common interface to disable hyd IGMP snooping, refer to the *QCA8337 Seven-Port Gigabit Ethernet Switch Data Sheet* (80-Y0619-3) to enable hardware IGMP snooping.

```
uci set hyd.config.SwitchInterface=eth1
uci commit hyd
/etc/init.d/hyd restart
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

8.34 Enable 802.11v DMS

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Support for 802.11v-2011 DMS* section of the *Power management techniques* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Directed Multicast Service (DMS) allows a client to request the AP to convert multicast frames that match a certain traffic classifier into unicast frames for the client. This request can be sent either as a DMS Request Information Element in the Association/Reassociation Request frame or explicitly using a DMS Request Action Frame after the client has completed the association. If the AP accepts the DMS Request, all multicast frames matching the traffic classifier specified in the

DMS Request is sent in unicast mode directly to the client as an AMSDU. The original multicast frame will still be transmitted for the benefit of those clients that do not support or request for DMS.

This requires the hardware to be able to encapsulate multicast frames in single-AMSDU format. In Lithium-based WLAN chipsets, there is a hardware issue wherein single-MSDU MPDU in AMSDU format is not supported. Due to this limitation, Host and FW need to implement a workaround (WAR) to ensure the packets are sent in unicast mode only in AMSDU format to the DMS-subscribed peer.

The following meta-header signature must be added by the customer stack in skb->head to notify radio driver of the unicast STA address for a given multicast frame. The customer stack shall just duplicate multicast frames for DMS supported clients and not be doing multicast-to-unicast (mc2uc) conversion. Also, for a valid meta-header signature, DMS ID is a positive integer.

The meta header structure, which is a part of skb head, is defined as follows:

```
struct{
UINT8 DMS_Id;   //Shall be non-zero
UINT8 unicast_addr[6];
}__attribute__((__packed__)) meta_hdr;
```

For any multicast packet received by the host, meta information, part of the skb->head, is processed to determine if the packet is intended to a DMS-subscribed peer. If the DMS ID is valid, the host performs the WAR required for catering to the DMS requirement.

The host converts the DMS multicast streams to Native Wi-Fi format and enqueue these frames to respective peers, similar to the Enhanced Multicast mode. Native Wi-Fi format enables hardware to override the A-MSDU format and also enables the other regular Ethernet unicast frames to continue to be send in the normal MPDU mode.

The packets converted into native Wi-Fi format are then sent with TO_FW descriptor flag set to enter ToFW exception path, where the FW performs DMS-specific operations.

Host sets the following in the 802.11 header:

- Frame Control Type – Data
- Frame Control Subtype – QOS

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

- Length of Ethernet subframe header – Length of original MSDU - Ethernet Header + LLC Header

To enable DMS using UCI, enter the following command:

```
uci set wireless.@wifi-iface[0].enable11v_dms=1
uci commit wireless
```

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

8.35 Configure best uplink hysteresis for Wi-Fi SON

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Configure best uplink hysteresis for Wi-Fi SON* section of the *Wi-Fi SON: Additional features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

To configure best uplink hysteresis using UCI commands for a Wi-Fi SON network, enter the following:

```
uci set wireless.@wifi-iface[x].ul_hyst=<val>
```

The `iwpriv athX ul_hyst` command configures the best uplink hysteresis.

In the componentized WLAN driver, version 11.0 driver, the best uplink hysteresis has been moved to the vdev object. As a result, the iwpriv command must be used in the VAP, instead of the radio interface. Also, the UCI setting must be configured on the wifi-iface section (of the backhaul STA VAP).

If a unified configuration is required for 10.4 and 11.0 drivers, set the value for both the radio and VAP through UCI, and the appropriate setting is applied.

The `iwpriv athX g_ul_hyst` command displays the configured best uplink hysteresis.

8.36 Configure FILS capability using UCI commands

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *FILS capability* and *FILS Discovery frames* sections of the *Performance enhancement capabilities* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Fast initial link setup (FILS) protocol defined in 802.11ai specification is helpful to reduce the connection delay. Initial WLAN connection delay is reduced without any compromise on security with the usage of FILS. FILS uses EAP-RP protocol for authentication and key derivation. Association to a FILS-capable AP always occurs through a full EAP exchange. The subsequent connections use the FILS authentication algorithm to minimize the delay in connection.

From the configuration perspective, a user must set a single UCI configuration option to turn on the FILS support in both driver and hostapd applications. To enable FILS capability in both driver and hostapd, enter the following command:

```
uci set wireless.@wifi-iface[1].ieee80211ai=1
```

To disable FILS capability in both driver and hostapd, enter the following command:

```
uci set wireless.@wifi-iface[1].ieee80211ai=0
```

When FILS is compiled and when the feature is enabled, the EXT CAP field indicates FILS capability as defined in IEEE 802.11ai draft specification.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

NOTE To enable FILS capability, enter the `iwpriv athX enable_fils 1` command. To disable FILS capability, enter the `iwpriv athX enable_fils 0` command. To retrieve the configured value, enter the `iwpriv athX g_enable_fils` command.

8.37 UCI commands to configure FILS Discovery Frames

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *FILS capability* and *FILS Discovery frames* sections of the *Performance enhancement capabilities* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

The functionality to enable FILS through UCI is extended to add support for FD frames. Setting the option, 'ieee80211ai', triggers the iwpriv command mentioned in the previous section. A new option, 'fils_fd_period', is added to allow a user to configure the FD frame period. If this option is not mentioned, a duration of 20 TUs is considered as the default period. A sample UCI configuration to enable FILS is as follows:

```
uci set wireless.@wifi-iface[0].ssid='FILS-5G'
uci set wireless.@wifi-iface[0].encryption='wpa2+ccmp'
uci set wireless.@wifi-iface[0].wpa_group_rekey='3600'
uci set wireless.@wifi-iface[0].key='12345678'
uci set wireless.@wifi-iface[0].ieee80211ai=1
uci set wireless.@wifi-iface[0].fils_realm=example.com
uci set wireless.@wifi-iface[0].fils_cache_id='1234'
uci set wireless.@wifi-iface[0].add_sha256='1'
uci set wireless.@wifi-iface[0].erp_send_reauth_start=1
uci set wireless.@wifi-iface[0].erp_domain=example.com
uci set wireless.@wifi-iface[0].auth_server_port=1812
uci set wireless.@wifi-iface[0].server=192.168.1.35
uci set wireless.wifi0.disabled='0'
uci set wireless.wifi0.channel=36
uci set wireless.@wifi-iface[0].own_ip_addr=192.168.1.1
uci set wireless.@wifi-iface[0].dhcp_server=192.168.1.55
uci set wireless.@wifi-iface[0].dhcp_rapid_commit_proxy=1
uci commit wireless
wifi load
```

Note that the default period is 20TUs. To explicitly set fd_period, the following command must be issued. This command sets fd_period to 100TUs.

```
uci set wireless.@wifi-iface[0].fils_fd_period=100
uci commit wireless
wifi load
```

8.38 UCI command to configure SA query timeout for PMF

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Using SA query timeout for PMF* section of the *IEEE 802.11 features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

To enable PMF using the timeout interval mechanism for Security Association (SA) query, enter the `iwpriv athX pmf_assoc 1` command. To check whether PMF using SA query timeout interval is configured, enter the `iwpriv athX g_pmf_assoc` command. Alternatively, enter the following UCI commands to enable this feature:

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

Both the AP and STA can be enabled with PMF by using `uci set wireless.@wifi-iface[X].ieee80211w=2` command.

For a non PMF client, or if "pmf_assoc" is not enabled the association response must not be dropped from hostapd and the driver must not send an association response frame.

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

8.39 Sample AP UCI configuration for 11ax

The commands for 11ax mode configuration are similar to the legacy 11n/11ac mode. Only difference is the hardware mode. The security and encryption configuration is the same as legacy mode. The following is the UCI configuration example for 5 GHz/2.4 GHz 11ax open mode:

5G 802.11ax open mode

```
uci set wireless.wifi0=wifi-device
uci set wireless.wifi0.type= qcawificfg80211
uci set wireless.wifi0.macaddr=00:60:02:00:c9:c9
uci set wireless.wifi0.hwmode=11axa
uci set wireless.wifi0.disabled=0
uci set wireless.wifi0.htmode=HT80
uci set wireless.wifi0.channel=36
uci set wireless.wifi0.txchainmask='0x0ff'
uci set wireless.wifi0.rxchainmask='0x0ff'
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
wifi down
wifi up
```

2.4 GHz 802.11ax open mode

```
uci set wireless.wifi1=wifi-device
uci set wireless.wifi1.type= qcawificfg80211
uci set wireless.wifi1.macaddr=00:60:02:00:c9:c8
uci set wireless.wifi1.hwmode=11axg
uci set wireless.wifi1.disabled=0
uci set wireless.wifi1.htmode=HT20
uci set wireless.wifi1.channel=1
uci set wireless.wifi1.txchainmask=15
uci set wireless.wifi1.rxchainmask=15
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=ap
uci set wireless.@wifi-iface[1].ssid=2g_open
uci set wireless.@wifi-iface[1].encryption=none
uci commit wireless
wifi down
wifi up
```

WPA2-PSK

```
uci set wireless.@wifi-iface[0].encryption=psk2+ccmp
uci set wireless.@wifi-iface[0].key=12345678
uci commit wireless
wifi down
wifi up
```

Qualcomm Confidential – May Contain Trade Secrets 2024-05-13 03:57:44 PDT liu.fangang@zte.com.cn

5 GHz Monitor Mode

```
uci set wireless.wifi0=wifi-device
uci set wireless.wifi0.type= qcawificfg80211
uci set wireless.wifi0.macaddr=00:60:02:00:c9:c9
uci set wireless.wifi0.hwmode=11axa
uci set wireless.wifi0.disabled=0
uci set wireless.wifi0.htmode=HT80
uci set wireless.wifi0.channel=36
uci set wireless.wifi0.txchainmask='0x0ff'
uci set wireless.wifi0.rxchainmask='0x0ff'
uci set wireless.wifi0.mode=monitor
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 commit wireless
wifi down
wifi up
```

2.4 GHz Monitor Mode

```
uci set wireless.wifi1=wifi-device
uci set wireless.wifi1.type= qcawificfg80211
uci set wireless.wifi1.macaddr=00:60:02:00:c9:c8
uci set wireless.wifi1.hwmode=11axg
uci set wireless.wifi1.disabled=0
uci set wireless.wifi1.htmode=HT20
uci set wireless.wifi1.channel=1
uci set wireless.wifi1.txchainmask=15
uci set wireless.wifi1.rxchainmask=15
uci set wireless.wifi1.mode=ap
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=monitor
uci set wireless.@wifi-iface[1].ssid=2g_open
uci set wireless.@wifi-iface[1].encryption=none
uci commit wireless
wifi down
wifi up
```

WDS STA Mode

```
uci set wireless.wifi0=wifi-device
uci set wireless.wifi0.type= qcawificfg80211
uci set wireless.wifi0.macaddr=00:60:02:00:c9:c9
uci set wireless.wifi0.hwmode=11axa
uci set wireless.wifi0.disabled=0
uci set wireless.wifi0.htmode=HT80
uci set wireless.wifi0.channel=auto
uci set wireless.wifi0.txchainmask='0x0ff'
uci set wireless.wifi0.rxchainmask='0x0ff'
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=sta
uci set wireless.@wifi-iface[0].ssid=5g_open
uci set wireless.@wifi-iface[0].wds='1'
uci commit wireless
wifi down
wifi up
```

Infra Mode

```
uci set wireless.@wifi-iface[0].extap='1'
uci set wireless.@wifi-iface[0].wds='0'
uci set wireless.@wifi-iface[1].extap='1'
uci set wireless.@wifi-iface[1].wds='0'
```

8.40 UCI commands to configure SAE

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Support for SAE* section of the *WLAN AP modes* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

A limitation of 802.1X is server reachability. If the authentication server is on the wired network, using 802.1X implies that both mesh stations have access to the wired network and the server. This limitation is the reason for another authentication mechanism to be built, which is called Simultaneous Authentication of Equals (SAE).

SAE is resistant to offline dictionary attacks. It relies on the fact that a password was defined on both neighbors (and that they have the same password): no central server is required. There are no defined roles in a mesh — no clients and servers, no initiators and responders. Anyone can initiate authentication. Refer to the IEEE 802.11-2012 spec for more information.

Configure SAE using the following UCI commands:

STA

```
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].wds='1'
uci set wireless.@wifi-iface[1].ssid='OpenWrtsae'
uci set wireless.@wifi-iface[1].encryption='ccmp'
uci set wireless.@wifi-iface[1].sae='1'
uci add_list wireless.@wifi-iface[1].sae_password='1234567890'
uci set wireless.@wifi-iface[1].ieee80211w='2'
/* We can also set key instead of sae_password.
If you provide both key and sae_password then it takes sae_password as
default. */
uci set wireless.@wifi-iface[0].force_invalid_group='1'
uci set wireless.@wifi-iface[0].sae_confirm_immediate='1'
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

AP

```
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='ap'
uci set wireless.@wifi-iface[1].wds='1'
uci set wireless.@wifi-iface[1].ssid='OpenWrtsae'
uci set wireless.@wifi-iface[1].encryption='ccmp'
uci set wireless.@wifi-iface[1].sae='1'
uci add_list wireless.@wifi-iface[1].sae_password='1234567890'
uci set wireless.@wifi-iface[1].ieee80211w='2'
/* We can also set key instead of sae_password.
If you provide both key and sae_password then it takes sae_password as
default. */

uci set wireless.@wifi-iface[0].force_invalid_group='1'
uci set wireless.@wifi-iface[0].sae_confirm_immediate='1'
```

- SAE password—This parameter can be used to set a password for SAE. By default, the wpa_passphrase value is used if this separate parameter is not used, but wpa_passphrase follows the WPA-PSK constraints (8-63 characters) even though SAE passwords do not have such constraints. If the BSS enabled both SAE and WPA-PSK and both values are set, SAE uses the sae_password value and WPA-PSK uses the wpa_passphrase value.

  Example: sae_password=secret

  WPA3 SAE adds a new feature called 'Password Identifier' which allows configuration of multiple passwords with identifier.

  For example:

  ```
  sae_password=secret
  sae_password=secret1|id=identifierA
  sae_password=secret2|id=identifierB
  sae_password=secret3|id=identifierC
  ```

  The corresponding UCI command is uci add_list wireless.@wifi-iface[N].sae_password='xxxxxxxx'

  NOTE: To enable WPS support for SAE in SON networks, the following must be noted:

  uci add_list wireless.@wifi-iface[0].sae_password=< length <=64 >

  This parameter is used to specify the sae_password length. After specifying this parameter in addition to other WPS SON parameters, the length should be <=64. WPS specifications support passphrase length of 64, according to the WPA2/1 requirements. Any enhancements/reuse of this feature must work with this rule. Therefore, application of WPS for WPA3 supports only 64-byte passphrase.

- SAE threshold for anti-clogging mechanism (dot11RSNASAEAntiCloggingThreshold)—This parameter defines how many open SAE instances can be in progress at the same time before the anti-clogging mechanism is taken into use.

  Example: sae_anti_clogging_threshold=5

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

- Maximum number of SAE synchronization errors (dot11RSNASAESync)—The offending SAE peer is disconnected if the specified maximum number of synchronization errors occur.

  Example: sae_sync=5

- Enabled SAE finite cyclic groups—SAE implementation are required to support group 19 (ECC group defined over a 256-bit prime order field). All groups that are supported by the implementation are enabled by default. This configuration parameter can be used to specify a limited set of allowed groups. The group values are listed in the IANA registry: http://www.iana.org/assignments/ipsec-registry/ipsec-registry.xml#ipsec-registry-9

  Example: sae_groups=19 20 21 25 26

- Require MFP for all associations using SAE—This parameter can be used to enforce negotiation of MFP for all associations that negotiate use of SAE. This is used in cases where SAE-capable devices are known to be MFP-capable and the BSS is configured with optional MFP (ieee80211w=1) for legacy support. The non-SAE stations can connect without MFP while SAE stations are required to negotiate MFP if sae_require_mfp=1.

  Example: sae_require_mfp=0

- Force enable invalid groups—AP will not be brought up with SAE groups that are not supported. Enabling this option will allow the AP to be configured with the given group, without any validity checks. STA will not be able to connect with those groups, if those groups are invalid. This can be used for testing purposes only.

  Example: force_invalid_group=0

- Send SAE Confirm immediately – Normally, AP will wait for the STA to send SAE-Confirm to send its SAE-Confirm. By enabling this option, AP will send SAE-Confirm immediately after SAE-Commit.

  Example: sae_confirm_immediate=0

- Enable WPA3 H2E—Use the `uci set wireless.@wifi-iface[0].sae_pwe='1'` command to set the SAE mechanism for PWE derivation.

  0 = hunting-and-pecking loop only (default without password identifier)

  1 = hash-to-element only (default with password identifier)

  2 = both hunting-and-pecking loop and hash-to-element enabled

  The default value is likely to change from 0 to 2 after the new hash-to-element mechanism has received more interoperability testing. When using SAE password identifier, the hash-to-element mechanism is used, regardless of the sae_pwe parameter value.

  Example: sae_pwe=0

Configure PSK2+SAE using UCI commands

STA

```
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'
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
uci set wireless.@wifi-iface[1].wds='1'
uci set wireless.@wifi-iface[1].ssid='OpenWrtsae'
uci set wireless.@wifi-iface[1].encryption='psk2+ccmp'
uci set wireless.@wifi-iface[1].key='1234567890'
uci set wireless.@wifi-iface[1].sae='1'
uci add_list wireless.@wifi-iface[1].sae_password='1234567890'
uci set wireless.@wifi-iface[1].ieee80211w='2'
/* If you provide both key and sae_password in the STA configuration  then
it uses sae_password to connect SAE AP and key for NON-SAE AP.

If you provide key alone then key will be using to connect both SAE and
NON-SAE AP's.

*/
```

AP

```
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='ap'
uci set wireless.@wifi-iface[1].wds='1'
uci set wireless.@wifi-iface[1].ssid='OpenWrtsae'
uci set wireless.@wifi-iface[1].encryption='psk2+ccmp'
uci set wireless.@wifi-iface[1].key='1234567890'
uci set wireless.@wifi-iface[1].sae='1'
uci add_list wireless.@wifi-iface[1].sae_password='1234567890'
uci set wireless.@wifi-iface[1].ieee80211w='2'
/* ieee80211w=’1’ if we wants to connect AP with SAE STA, PMF enabled STA
& Non-PMF enabled STA.
Ieee80211w=’2’ if we wants to connect AP with SAE STA, PMF enabled STA
*/
/* If you provide both key and sae_password in the STA configuration  then
it uses sae_password to connect SAE AP and key for NON-SAE AP.

If you provide key alone then key will be using to connect both SAE and
NON-SAE AP's.

*/
```

8.41 UCI commands for OWE

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Opportunistic Wireless Encryption (OWE) for Open SSID* section of the *WLAN AP modes* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Opportunistic Wireless Encryption (OWE) extracts an encryption key between an access point (AP) and a client, even if no previous communication between the AP and client has occurred to determine any shared secret. OWE implementations are required to support group 19 (NIST P-256). All groups that are supported by the implementation (e.g., groups 19, 20, and 21 while using OpenSSL) are enabled by default. This configuration parameter can be used to specify a limited set of allowed groups. The group values are listed in the IANA registry:

http://www.iana.org/assignments/ipsec-registry/ipsec-registry.xml#ipsec-registry-10. Supported owe_groups=19 20 21; it is necessary to enable owe=1 in AP to use this feature.

Configuration for OWE

STA

```
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].wds='1'
uci set wireless.@wifi-iface[1].ssid='OpenWrtOWE'
uci set wireless.@wifi-iface[1].encryption='ccmp'    /* We can use
encryption as 'gcmp' also instead of 'ccmp */
uci set wireless.@wifi-iface[1].owe='1'
uci set wireless.@wifi-iface[1].ieee80211w='2'
```

AP

```
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='ap'
uci set wireless.@wifi-iface[1].wds='1'
uci set wireless.@wifi-iface[1].ssid='OpenWrtOWE'
uci set wireless.@wifi-iface[1].encryption='ccmp'    /* We can use
encryption as 'gcmp' also instead of 'ccmp */
uci set wireless.@wifi-iface[1].owe='1'
uci set wireless.@wifi-iface[1].ieee80211w='2'
```

Configuration for OWE transition BSSID mode

STA

```
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].wds='1'
uci set wireless.@wifi-iface[1].ssid='non-OWE'
uci set wireless.@wifi-iface[1].encryption='ccmp'    /* We can use
encryption as 'gcmp' also instead of 'ccmp */
uci set wireless.@wifi-iface[1].owe='1'
uci set wireless.@wifi-iface[1].ieee80211w='2'
```

AP

```
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='ap'
uci set wireless.@wifi-iface[1].ssid=' OpenWrtOWE'
uci set wireless.@wifi-iface[1].owe='1'
uci set wireless.@wifi-iface[1].hidden='1'
```

```
uci set wireless.@wifi-iface[1].encryption='ccmp'
uci set wireless.@wifi-iface[1].ieee80211w='2'
uci set wireless.@wifi-iface[2].owe_transition_ifname='ath11'
/* transition interface name nothing but non-OWE enabled interface */

uci set wireless.@wifi-iface[2]=wifi-iface
uci set wireless.@wifi-iface[2].device='wifi1'
uci set wireless.@wifi-iface[2].network='lan'
uci set wireless.@wifi-iface[2].mode='ap'
uci set wireless.@wifi-iface[2].encryption='none'
uci set wireless.@wifi-iface[2].ssid='non-OWE'
uci set wireless.@wifi-iface[2].owe_transition_ifname='ath1'
/* transition interface name nothing but OWE enabled interface */
uci set wireless.@wifi-iface[2].owe_transition_ssid=' OpenWrtOWE '
/* transition ssid is nothing but OWE enabled ssid */
uci set wireless.@wifi-iface[2].owe_transition_bssid='8e:fd:f0:02:22:31'
/* bssid is nothing but OWE enabled ssid's(owe-265debce) bssid */
```

OWE transition mode configuration

The pointer to the matching open/OWE BSS is as follows:

#owe_transition_bssid=<bssid>

SSID is in the same format as ssid2 described previously.

#owe_transition_ssid=<SSID>

Alternatively, OWE transition mode BSSID/SSID can be configured with a reference to a BSS operated by this hostapd process as follows:

#owe_transition_ifname=<ifname>

In OWE transition BSSID mode, perform the following:

1. Bring up two interfaces and are independent of bands (Two 2G bands/ Two 5G bands/ one 2G band and one 5G band). One is with owe enabled and another is with non-owe enabled AP.
2. Configure STA with non-owe enabled ssid with OWE enabled.
3. Hide the ssid of OWE enabled ssid in one interface and needs to provide OWE enabled SSID, BSSID, and interface name using owe_transition_ssid, owe_transition_bssid, and owe_transition_ifname respectively in another interface.
4. From the scan entry table of STA, the OWE-enabled BSSID is determined from the non-OWE BSSID beacon frame and the probe request is sent to OWE-enabled AP after which further association occurs.

8.42 UCI commands for Suite B in WPA3

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Suite B in WPA3* section of the *WLAN AP modes* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Suite B cryptography has been selected from cryptography that has been approved by NIST for use by the U.S. Government and specified in NIST standards or recommendations. Configuration for Suite-B

```
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='ap'
uci set wireless.@wifi-iface[1].ssid='AP_SUITE-B'
uci set wireless.@wifi-iface[1].suite_b='192'
uci set wireless.@wifi-iface[1].auth_server='192.168.1.30'
uci set wireless.@wifi-iface[1].auth_
secret='12345678901234567890123456789012345678901234567890123456789012345678901234
'
uci set wireless.@wifi-iface[1].nasid='wifi-user@example.com'
uci set wireless.@wifi-iface[1].auth_port='1812'
```

To enable Suite B feature, enter the uci set wireless.@wifi-iface[1].suite_b='192' command.

8.43 Display the hyd and lbd versions

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Display the hyd and lbd versions* section of the *Wi-Fi SON: Additional features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

The Hy-Fi daemon (hyd) and load balancing daemon (lbd) version information are essential to identify the images of these processes or daemons that are used in the release of a software product. While using different software images, it is difficult to determine the version of the hyd that is being used in a particular software image release. The mechanism to display the hyd and lbd versions is implemented. The hyd and lbd versions are displayed in the output of explicit commands introduced for this purpose.

The hyd version is determined by user space command with the –v argument. The command retrieves and displays the version of the hyd and lbd. The lbd version is displayed only if the WLB module is enabled as part of hyd. The lbd version command retrieves only the version of the lbd module that is used. Enter the `hyd -v` command to view the hyd and lbd versions. Enter the `lbd -v` command to view the lbd version.

The following is a sample output of the hyd –v command:

```
root@OpenWrt:/# hyd -v
hyd v1.0
User space daemon for HYD
Copyright (c) 2011 Qualcomm Atheros, Inc.
All Rights Reserved.
Qualcomm Atheros Confidential and Proprietary.

lbd v1.0
User space daemon for LBD
Copyright (c) 2011 Qualcomm Atheros, Inc.
All Rights Reserved.
Qualcomm Atheros Confidential and Proprietary.
```

The following is a sample output of the lbd –v command:

```
root@OpenWrt:/# lbd -v
lbd v1.0
User space daemon for LBD
Copyright (c) 2011 Qualcomm Atheros, Inc.
All Rights Reserved.
Qualcomm Atheros Confidential and Proprietary.
```

8.44 UCI commands to configure AP steering for legacy clients

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *AP steering for legacy clients* section of the *Wi-Fi SON: Band steering and APS* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

This section describes the AP steering for legacy clients by use of RSSI Monitoring using Smart Monitor mode. Multi AP with one 2.4 GHz and/or one or two 5 GHz radio(s). The radios operate concurrently on different channels. This section describes the design rationale and the actual steering. The purpose of this steering is to position the clients with improved WiFi Network performance by steering the clients to a better-serving AP through a better RSSI. A client for steering will be a legacy client, a client that does not support 802.11k beacon measurement, a client that is certified by the upcoming Wi-Fi Alliance MultiBand Operation (MBO) certification, or a client that fails for consecutive attempts of 802.11k beacon report.

Support is implemented for APs to steer legacy clients, which do not support 802.11k beacon measurement and MBO compliance. These legacy clients are clients that support 802.11v basic service set (BSS) transition management (BTM) and do not support Radio Resource Management (802.11k). Only idle steering and not active steering is supported.

After the image is loaded into the flash memory and the device is configured (including Smart Monitor Configuration as listed later in this subsection as an example), reboot AP.

Edit `/etc/config/hyd` for the following and restart hyd.

The following is the default configuration:

```
config Monitor 'Monitor'
    option DisableMonitoringLegacyClients '1'
    option DisableSteeringInActiveLegacyClients '1'
    option DisableSteeringActiveLegacyClients '1'
    option MonitorTimer '60'
    option MonitorResponseTimeout '5'
```

1. Configure *DisableMonitoringLegacyClients* as 0 for smart monitor to be functional.
2. Configure *MonitorTimer as* 60 for periodic monitoring.
3. Configure *MonitorResponseTimeout* as 5 for better monitoring of multiple clients.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

4. Configure *DisableSteeringInActiveLegacyClients* as 0 for monitoring/steering of Inactive Legacy clients.
5. Configure *DisableSteeringActiveLegacyClients* as 0 for monitoring/steering of Active Legacy clients.
6. Enter the `$ /etc/init.d/hyd restart` command to restart hyd.

NOTE For this feature to work, it is necessary to create monitor VAPs on the two radios used for client connection, which are wifi0 and wifi1 in this case. If these VAPs are different on the board in a a particular network, the user must perform appropriate changes.

Smart Monitor configuration

```
uci set wireless.wifi0.repacd_auto_create_vaps=0
uci set wireless.wifi1.repacd_auto_create_vaps=0
uci add wireless wifi-iface
uci set wireless.@wifi-iface[4].device=wifi0
uci set wireless.@wifi-iface[4].network=lan
uci set wireless.@wifi-iface[4].mode=ap_smart_monitor
uci set wireless.@wifi-iface[4].ssid=ap_smart_monitor1
uci set wireless.@wifi-iface[4].encryption=none
uci set wireless.@wifi-iface[4].neighbourfilter=1
uci set wireless.@wifi-iface[4].set_monrxfilter=1
uci set wireless.@wifi-iface[4].disable=0
uci set wireless.@wifi-iface[4].wsplcd_unmanaged=1
uci set wireless.@wifi-iface[4].repacd_security_unmanaged=1

uci add wireless wifi-iface
uci set wireless.@wifi-iface[5].device=wifi1
uci set wireless.@wifi-iface[5].network=lan
uci set wireless.@wifi-iface[5].mode=ap_smart_monitor
uci set wireless.@wifi-iface[5].ssid=ap_smart_monitor2
uci set wireless.@wifi-iface[5].encryption=none
uci set wireless.@wifi-iface[5].neighbourfilter=1
uci set wireless.@wifi-iface[5].set_monrxfilter=1
uci set wireless.@wifi-iface[5].disable=0
uci set wireless.@wifi-iface[5].wsplcd_unmanaged=1
uci set wireless.@wifi-iface[5].repacd_security_unmanaged=1
uci commit
reboot -f
```

Steering for 802.11k unfriendly clients

Edit /etc/config/lbd for the following and restart hyd. The default configuration is as follows:

```
config Estimator_Adv 'Estimator_Adv'
    option Max11kUnfriendly '10'
```

Max11kUnfriendly must be configured to ‘n’ for clients to be initiated with Smart Monitoring if 11k beacon report fails ‘n’ consecutive attempts.

8.45 Best uplink algorithm for PLC interface metrics in a daisy chain Wi-Fi SON and PLC network

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Wi-Fi SON and PLC* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

The capability to apply the best uplink algorithm to use PLC interface link metrics in a Wi-Fi SON network integrated with PLC is introduced. With daisy chain topologies involving a CAP, RE1, and RE2 (where RE1 can either be an REH172 device or a DK01/DK07 device), RE2 can reach the CAP (gateway) through different upstream devices with different backhaul interfaces. When multiple paths are available to reach the CAP, a potential loop can occur in the Wi-Fi SON with PLC network. As a result, the following problems are noticed in the Wi-Fi SON network

- Loop observed on all nodes with broadcast packets
- Redundant MAC learning on multiple SON nodes
- Ping instability between repeaters and Ethernet clients
- Slowness in the network

To avoid the complexity in daisy chain topologies when different backhaul interfaces connect to different upstream devices, the recommendation is to select only one backhaul interface from second hop onwards. The selection of backhaul interface is performed using the best uplink selection algorithm applying the PLC interface link metrics.

Only one backhaul interface, when found in a daisy chain topology, is selected. In a Wi-Fi SON network integrated with PLC that contains daisy chain topology support, a possibility of the second-hop repeater reaching the CAP through two different backhaul interfaces (WLAN/PLC) exists. The capability to apply the best uplink selection algorithm is implemented in a daisy chain topology that contains the *CAP ---- RE1 ---- RE2* connection.

The feature is supports the first level of daisy chain. The following figures illustrate a simple first level of daisy-chain network, where RE1 is connected to the CAP and RE2 is connected to RE1 along with PLC interfaces. Both REs have HYD instances running, and AP and band steering occurs in the usual manner across them.

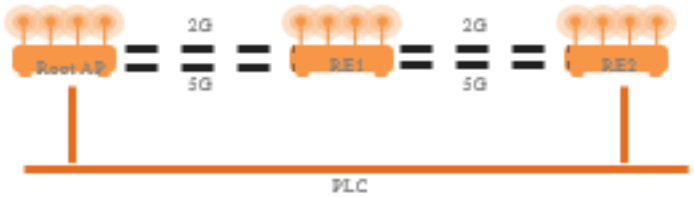


Figure 8-1 PLC daisy chain with RE1 as DK01/DK07

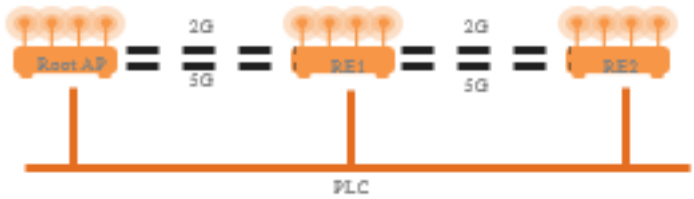


Figure 8-2 PLC daisy chain with RE1 as REH172

The RE2 can reach CAP through WLAN and PLC two different backhaul links. The repacd best-uplink algorithm running on RE2 must consider PLC Link Metrics also to select the best path to reach CAP. After selecting the best path, the preceding daisy chain topologies will become the following topologies:

With RE1 as DK01/DK07, if PLC interface is selected as backhaul, then the topology becomes a star topology as shown in the following figure. The daisy chain topology becomes star topology with one repeater connected with WLAN and another repeater connected with PLC.

Figure 8-3 PLC daisy chain becomes a star topology after selecting PLC as backhaul in Daisy Chain topology

With RE1 as REH172, if WLAN interface is selected as backhaul, then the topology becomes a pure WLAN daisy chain topology as shown in the following figure:

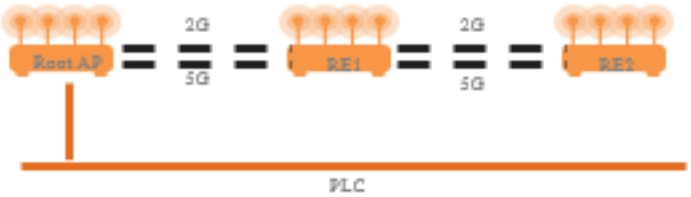


Figure 8-4 PLC daisy chain becomes pure WLAN daisy chain topology for CAP-RE1-RE2 after selecting WLAN interface as backhaul

With RE1 as REH172, if PLC interface is selected as backhaul, then the topology becomes a star topology as shown in the following figure. The daisy chain topology becomes star topology with one repeater connected with WLAN and PLC, and another repeater connected only with PLC.

Figure 8-5 PLC daisy chain becomes a star topology after selecting PLC as backhaul in Daisy Chain topology

With RE1 as REH172, if PLC interface is selected as backhaul, then the topology becomes a pure WLAN daisy chain topology as shown in the following figure:

Figure 8-6 PLC daisy chain becomes pure WLAN daisy chain topology for CAP-RE1-RE2 after selecting WLAN interface as backhaul

Set the following parameters in the repacd configuration file,:

```
uci set repacd.BackhaulMgr.SelectOneBackHaulInterfaceInDaisy='1'
uci set repacd.BackhaulMgr.BackHaulMgrRateNumMeasurements='10'
uci set repacd.BackhaulMgr.PLCLinkThresholdto2G='60'
uci set repacd.BackhaulMgr.SwitchInterfaceAfterCAPPingTimeouts='10'
uci commit repacd
uci commit
```

- `SelectOneBackHaulInterfaceInDaisy` is enable the feature to select only one backhaul interface when found WLAN and PLC.
- `BackHaulMgrRateNumMeasurements` is used to consider number of samples to measure before selecting a backhaul interface, either WLAN or PLC.
- `PLCLinkThresholdto2G` is used to select PLC interface as backhaul when WLAN 5G link rate is zero or 5G is not connected to its upstream and 2G is connected to upstream. In such situation if PLC link rate is less than configured threshold value then WLAN 2G interface is selected, otherwise if PLC link rate is greater than configured threshold value then PLC interface is selected as backhaul.
- `SwitchInterfaceAfterCAPPingTimeouts` is used to consider number of ping outs to switch interface from PLC to WLAN, when CAP is not reachable thru PLC interface.

In cases of failover scenarios such as the CAP not being reachable through PLC interface, then the expected switchover time for traffic depends on the following:

- Bring up of WLAN (2G and 5G) interfaces

- Association of interfaces

8.46 UCI commands for ETSI compliance for COT and CAT on QCA9880 and QCA9980 chipsets

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *ETSI compliance for COT and CAT on QCA9880 and QCA9980 chipsets* section of the *Power management techniques* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

QCA9980 family boards (QCA9980, QCA9984, IPQ4019, and QCA9886) and QCA9880 platforms had failed some of the European Telecommunications Standards Institute (ETSI) regulation test cases. These cases were related to Channel Occupancy Test (COT) and Channel Access Latency (CAL) tests.

An enhancement is introduced to make burst duration, and AIFSN configurable based on WMM Access Category (VO: 2ms, VI: 4ms and BE/BK: 6ms max). This enhancement addresses the ETSI Adaptivity (Channel Access) Testing for QCA9880 and QCA9980 chipsets. Also, modifications are made to the current SIFS burst algorithm for the mechanism to be duration-based and for addition of the beeliner_fw_test command support.

Consider the following sample configuration scenario in which an AP is brought up in ETSI domain and connect the client. Run the UDP traffic.

AP Configuration:

```
config wifi-device  wifi1
        option type    qcawifi
        option channel  36
        option macaddr  00:a0:c6:eb:67:92
        option hwmode   11ac
        option htmode   HT80
        # REMOVE THIS LINE TO ENABLE WIFI:
        option disabled 0
        option country  0x0D0

config wifi-iface
        option device   wifi1
        option network  lan
        option mode     ap
        option ssid     OpenWrt_ETSI
        option encryption none
```

Before testing for certification, disable Infinite sequencing feature using the below command.

```
wifitool athX beeliner_fw_test 227 0
```

The following are the commands available to configure the Burst duration and AIFSN. By default, BE/BK is configured with 6ms, VI with 4ms and VO with 2ms.

To configure AIFSN:

```
wifitool athX beeliner_fw_test 225 value (value = ac << 8 | AIFSN).
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

To configure Burst duration:

```
iwpriv ath0 aggr_burst <ac> <duration>
```

Verify burst not exceeded the configured value using the command

```
wifitool athX beeliner_fw_test 154 0
```

Observe the last dbglog print with first 2 parameters in the hexa 0x format. These values must not exceed the configured burst duration.

8.47 UCI commands for Green AP power-saving in connected-idle status

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Green AP power-saving in connected-idle status* section of the *Power management techniques* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Starting with QCA_Networking_2017.SPF.6.0, the Green AP power-save feature is enhanced to allow users to configure the AP such that when only 1x1 or no client is connected to the AP Green AP, power save is turned ON.

Connected idle is defined as stations connected without generating much traffic, IOE units, for example, which stay online 24/7 but rarely generate traffic and this connected idle consumption is becoming an important KPI in Europe, thereby, modifying clock frequency to enable routing the required traffic and enable improved power-conversation metrics.

This feature mainly focused on power saving by configuring target CPU Clock speed as required, and the same is applicable only for IPQ4019 chipsets. This radio specific feature helps configure the default CPU clock frequency (250 MHz) to 200 MHz through "ipq4019_20_targ_clk" module parameter.

By default, IPQ4019 CPU clock frequency is 250 MHz, which is configured through the AHB bus using the register FW_CPU_PLL_CONFIG. The host writes clk_freq onto the same register to enable the target radio. Then, FW reads the clock frequency from the same memory address provided in that register.

Here, to modify the clock frequency, a new module parameter is added in host, which is as follows, ipq4019_20_targ_clk.

Using UCI commands, Host will set a particular bit to change the clock frequency present in the register FW_CPU_PLL_CONFIG. Then, FW will mask this register value to retrieve the clock frequency.

```
uci set wireless.qcawifi=qcawifi
uci set wireless.qcawifi.ipq4019_20_targ_clk=200000000
uci commit
wifi load
```

Example:

Without changing the clock frequency:

```
root@OpenWrt:/# athdiag --get --address=0x4f014
Value in target at 0x4f014: 0x37d1300 (58528000) -? 250Mhz
```

With a change of the clock frequency with 200 MHz:

```
root@OpenWrt:/# athdiag --get --address=0x4f014
Value in target at 0x4f014: 0x37d1300 (58528512) -? 200Mhz
```

If the clock frequency is set to a value other than 200 MHz, then this feature resets the value back to 250 MHz, by default.

Example:

```
uci set wireless.qcawifi=qcawifi
uci set wireless.qcawifi.ipq4019_20_targ_clk=300000000
root@OpenWrt:/# athdiag --get --address=0x4f014
Value in target at 0x4f014: 0x37d1300 (58528000) -? 250Mhz
```

8.48 UCI command to configure idle client detection for Tx path

The client activity detection logic, used by band steering, is extended to be triggered upon successful Tx completions. Previously, this idle client detection capability was triggered only when a successful packet was received (Rx).

By default, the idle client detection feature is enabled. The Band Steering daemon (lbd) has to be restarted after a UCI config change.

To enable this feature, specify the following option in LBD UCI configuration file:

```
uci set lbd.config.InactDetectionFromTx='1'
```

8.49 UCI parameters for multi-AP

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Multi-AP architecture* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

This section describes the configuration parameters multiple AP (MAP) functionality in Wi-Fi SON networks. Multi-AP functionality is also called Easy Mesh. Easy mesh is supported, starting with QCA_Networking_2019.SPF.10.0.

NOTE The soft-blocking feature adds and clears ACL entries. Therefore, this feature must not be used in conjunction with the Multi-AP SIG feature (also called as Easy Mesh) or with the Wi-Fi SON capability. If a combination of soft-blocking feature and Easy Mesh feature, or a combination of soft-blocking feature and Wi-Fi SON capability are configured, an unexpected behavior might occur because soft-blocking and Wi-Fi SON (or Easy Mesh) have independent logic of adding and clearing the ACL entries.

8.49.1 Wi-Fi SON messages and multi AP equivalents

In a multi-AP environment, Qualcomm central APs (CAPs) are supported to function as controllers and repeaters are supported to function as agents. For Wi-Fi SON, steering is done in a distributed manner (where each AP makes the steering decision for the STAs that it is serving), whereas for MAP, a centralized mechanism is implemented in which the controller makes the steering decisions. Certain messages in Wi-Fi SON and their equivalent messages in MAP are mapped in the following table.

Message	TLV	Purpose	Multi-AP equivalent
Average Utilization Request	AVG_UTIL_REQ	Request that each AP report its average utilization information on its operating channels back to the sender (the CAP).	AP Metrics Query
Average Utilization Report	AVG_UTIL_REPORT	Two purposes: ■ Individual AP report of utilization back to the CAP. ■ Aggregated utilization report sent by CAP to all APs after collecting all metrics.	Individual reports back using AP Metrics Response message. Aggregate report using Combined Infrastructure Metrics message.
Load Balancing Allowed	LOAD_BALANCING_ALLOWED	Informs an AP that is can perform load balancing and active upgrade steering for a pre-configured period of time.	Client Steering Request with the Request Mode field set to Steering Opportunity and k=0 (no STAs specified). Note that steering opportunity steering is not currently used by the controller nor supported by the agent.
Load Balancing Complete	LOAD_BALANCING_COMPLETE	Indicate back to the CAP that the AP that was assigned a load balancing slot has completed its operations.	Steering Completed message.
Station Info Request	STA_INFO_REQ	Request the information about a specific station.	Client Capability Query message.
Station Info Response	STA_INFO_RSP + STA_CSBC_STATE	Provides back the station capabilities and CSBC state information. Capabilities include: ■ Supported bands ■ 11k/v support ■ Per band: □ Max bandwidth □ NSS □ Max MCS index □ Max Tx power	Client Capability Report message.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Message	TLV	Purpose	Multi-AP equivalent
Prepare For Steering Request	PREPARE_FOR_STEERING_REQ + STA_INFO_RSP + STA_CSBC_STATE + STA_POLLUTION_STATE	Provides the list of candidate BSSes for a given steering operation that is initiated by the serving AP. All other APs are to adjust their blacklists and probe response withholding accordingly.	Series of Client Association Control Request messages to update the blacklists followed by a Client Steering Request message.
Prepare For Steering Response	PREPARE_FOR_STEERING_RSP	Provides feedback on whether the preparation succeeded and if not, why it failed. Also contains the number of seconds since the STA was last steered from the sending node's perspective.	Client Steering BTM Report message for BTM-based steering. The standard does not have a way to decline the request.
Authentication Reject Sent	AUTH_REJ_SENT	Indication that the STA tried to auth on a BSS where it was blacklisted.	None
Steering Abort Request / Response	STEERING_ABORT_REQ / STEERING_ABORT_RSP	Request from any node to remove all blacklists and consider the steering aborted.	No direct equivalent, although the blacklists can be removed using Client Association Control Request message.
Prepare For Monitoring Request / Response	PREPARE_FOR_MONITORING_REQ / PREPARE_FOR_MONITORING_RSP	Request that a STA MAC address be sniffed in order to measure its RSSI. The request contains the STA MAC address, BSSID, and channel. The response also contains a status.	Unassociated STA Link Metrics Query / Response message.
Abort Monitoring Req / Response	ABORT_MONITORING_REQ / ABORT_MONITORING_RSP	Cancel a previously installed MAC address from being monitored.	None
Average RSSI Request / Report	AVG_RSSI_REQ / AVG_RSSI_ REPORT	Sent when a client is below the AP steering low RSSI crossing threshold (when the periodic timer expires).	None (report comes in response to the query two rows above)

8.49.2 Load balancing daemon (lbd) parameters for multi-AP

The following parameters are used by both **lbd** and **hyd** and control the steering behavior while operating in Multi-AP SIG mode.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Configuration Type	Section	Option	Description	Default
StaDB	StaDB	TrackRemoteAssoc	Whether the station database should track only local associations or both local and remote associations. The expectation is that this value is set to `1` on all nodes to enable the appropriate bridging table cleanup to occur when handling the list of associated STAs in Topology Response.	1
APSteer	APSteer	LowRSSIAPSteerThreshold_SIG	The threshold to use as the AP steering trigger point when operating in Multi-AP SIG mode.	17
		APSteerMaxRetryCount	The number of times to attempt to AP steer a STA before giving up. The counter is reset whenever the STA associates on a new BSS.	2
Estimator_Adv	Estimator_Adv	ActDetectMinInterval	The interval (in seconds) over which to determine whether a STA is inactive based on packet counters.	30
		ActDetectMinPktPerSec	The number of packets per second (both TX and RX) below which a client is considered inactive.	2
StaDB	StaDB_Adv	MinAssocAgeForStatsAssocUpdate	The number of seconds a STA must have been associated on one BSS for a stats update to force it to be marked as associated on a different BSS.	150
SteerAlg_Adv	SteerAlg_Adv	UsePathCapacityToSelectBSS	Whether to use the end-to-end path capacity scheme to select the target BSS. If this is false, the Wi-Fi SON BSS selection schemes are used. This is only relevant when operating in Multi-AP SIG mode.	1

Qualcomm Confidential May Contain Trade Secrets 2024-05-13 03:57:44 PDT pan.yaheng@zte.com.cn

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

8.49.3 Wi-Fi security and PLC daemon (wsplcd) parameters for multi-AP

The following parameters are used to control the behavior of **wsplcd** when operating in Multi-AP SIG mode:

Configuration Type	Section	Option	Description	Default
config	config	MapGenericPolicyFile	The name of the template file that contains the rules for the BSS instantiation scheme.	
		MapPolicyFile	The name of the file that contains the BSS instantiation rules for specific AL MAC addresses. This is only used if MapGenericPolicyFile is not set. It is only intended for use in plug fests.	
		MapEnable	Whether the daemon must operate in Multi-AP SIG mode.	0
		MapMaxBss	The number of BSSes that the agent allows the controller to instantiate on each of its radios.	
		MapPFCompliant	When enabled, apply the following special rules to ensure the WFA Easy Mesh test cases can be passed:	0

8.49.4 RE placement and auto-configuration daemon (repacd) parameters for multi-AP

The following parameters are used to control the behavior of the RE Placement and Auto-Config Daemon (**repacd**) when operating in Multi-AP SIG mode:

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Configuration Type	Section	Option	Description	Default
MAPConfig	MAPConfig	Enable	Whether the Multi-AP SIG mode of operation is enabled.	0
		FirstConfigeRequired	Whether this device is fresh from the factory and needs to go through the complete onboarding process (including WPS).	1
		FronthaulSSID	The value to configure for the fronthaul SSID during first boot configuration. If this parameter is not specified, a random number is generated.	
		FronthaulKey	The value to configure for the fronthaul passphrase during first boot configuration. If this parameter is not specified, a random number is generated.	
		BackhaulSSID	The value to configure for the fronthaul SSID during first boot configuration. If this parameter is not specified, a random number is generated.	
		BackhaulKey	The value to configure for the fronthaul passphrase during first boot configuration. If this parameter is not specified, a random number is generated.	
		BackhaulSuffix	When generating a random SSID, the suffix to add for the backhaul network. If this parameter is not specified, the backhaul and fronthaul are configured to operate on the same VAP.	
		StandaloneController	Whether to set the device up as a standalone controller (no AP interfaces).	0

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

Configuration Type	Section	Option	Description	Default
MAPWiFiLink	MAPWiFiLink	MinAssocCheckPostWPS	The number of times the association must be deemed up after a WPS button press before it is considered stable enough before an RSSI measurement can begin.	5
		WPSTimeout	The amount of time (in seconds) to wait for an association to take place after the WPS button is pressed. If this amount of time elapses without the STA interface associating, the device is assumed to be too far from the CAP.	180
		AssociationTimeout	The amount of time (in seconds) to wait for the STA interface to associate before considering the device as too far from the CAP. Note that a WPS push button cancels this timer and runs the WPS timeout instead.	300
		RSSINumMeasurements	The number of measurements to take to arrive at an average RSSI to compare against the near/far thresholds.	5
		BackhaulRSSIThreshold_2	The signal level (in dBm) above which the 2.4 GHz backhaul link is considered adequate.	-78
		BackhaulRSSIThreshold_5	The signal level (in dBm) above which the 5 GHz backhaul link is considered adequate.	-78
		BackhaulRSSIThreshold_offset	The offset to add to `BackhaulRSSIThreshold_2` to determine whether to re-evaluate the 5 GHz bSTA. If the 2.4 GHz bSTA RSSI is above this combined threshold, the re-evaluation takes place.	10
		Max5gAttempts	The number of times to attempt using the 5 GHz bSTA interface before deciding to stick with the 2.4 GHz bSTA.	3
		MaxMeasuringStateAttempts	The number of times an attempt to measure the RSSI on a given bSTA interface is allowed to fail before giving up and considering the bSTA interface bad.	3

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yifeng@zte.com.cn

Configuration Type	Section	Option	Description	Default
MAPConfig	repacd	MapVersionEnabled	Change configuration for version from MAP R2 to R3 by setting the following: uci set repacd.MAPConfig.MapVersionEnabled=3 uci commit	
MAPConfig	repacd	MAPConfig.AdditionalFH	MAP config for repacd to get additional VAPs ssid:key:authType. Default Value: NULL. If key is NULL AP will be configured in open mode. authType 0 = wpa2, 1 = mixed mode. For example: uci set repacd.MAPConfig.AdditionalFH="Adi_fbss1:Adifbsspri:0,Adi_fbss2:Adifbsspri2:0,Adi_fbss3:Adifbsspri3:0"	Null
MAPConfig	repacd	OnboardingType	Flag to enable or disable map config service in HYD. To enable DPP onboarding: WSPLCD will not be started. Onboarding will happen through HYD uci set repacd.MAPConfig.OnboardingType='dpp' To disable DPP and MAP Config Service: WSPLCD will start. Onboarding will happen through WSPLCD uci set repacd.MAPConfig.OnboardingType='none'	

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

Configuration Type	Section	Option	Description	Default
MAPConfig	repacd	EnableConfigServiceLogs	Flag to dump mapConfigService logs for debug. Logs will be dumped to /tmp/mapConfig.log uci set repacd.MAPConfig.EnableConfigServiceLogs =1	
MAPConfig	repacd	DPPDumpSecurityLog	Flag to dump Encryption and Integrity logs for every message. Logs will be dumped to /tmp/mapConfig.log . Needs to enabled with *EnableConfigServiceLogs* to debug encryption / decryption failures *uci set repacd.MAPConfig. DPPDumpSecurityLog =1*	

The following LED placement states are used when operating in Multi-AP mode:

- Reset - Used when **repacd** is terminated
- NotAssociated - STA interface is still trying to associate
- AutoConfigInProgress - WPS button was pressed and the timeout has not yet occurred or a range extender mode switch is occurring
- Measuring - STA is associated and an average downlink RSSI value is being computed
- WPSTimeout - Failed to establish an association within `WPSTimeout` seconds
- AssociationTimeout - Failed to establish an association within `AssociationTimeout` seconds
- RE_BackhaulPoor - RSSI is too weak on 2.4 GHz bSTA
- RE_BackhaulFair - RSSI is sufficient on 2.4 GHz bSTA
- RE_BackhaulGood - RSSI is sufficient on 5 GHz bSTA
- RE_SwitchingBSTA - Transient state used when the device has decided to try a different bSTA interface and is in the process of switching
- InCAPMode - Device is connected to the gateway using Ethernet and is acting only as an AP

8.49.5 Hy-Fi (hyd) parameters for multi-AP

The following parameters are used within the Hy-Fi daemon (hyd) that implements IEEE1905.1 and the Multi-AP SIG messaging and agent/controller functionality.

Configuration Type	Section	Option	Description	Default
hyd	hyd	ForwardingMode	This field takes one of three values that controls how the `hyfi-netfilter` bridge hooks behave: ■ `APS` - Full support for multiple backhauls and Adaptive Path Selection ■ `SINGLE` - Apply the loop prevention and 1905 multicast rules but disable the H-Default/H-Active tables. ■ `MCAST` - Only apply the 1905 multicast rules. □ This mode must only be used if another mechanism such as RSTP is being used to prevent loops.	APS
MultiAP	MultiAP	EnableController	Whether this device must operate in controller mode.	0
		EnableAgent	Whether this device must operate in agent mode.	0
		EnableSigmaDUT	Whether this device must enable the Sigma DUT functionality. This is only used for plug fests and is not relevant for normal testing.	0
		MetricsReportingInterval	The rate (in seconds) at which the controller requests the agents to report AP and STA metrics.	5
		RSSIHysteresis_W2	The amount by which the RSSI on 2.4 GHz has to change (in dBm) after crossing the threshold for another event to be generated.	5
		RSSIHysteresis_W5	The amount by which the RSSI on 5 GHz has to change (in dBm) after crossing the threshold for another event to be generated.	5
		LoadBalancingInterval	The time period (in seconds) the controller averages utilization information over prior to performing load balancing steering.	30

Qualcomm Confidential – May Contain Trade Secrets 2021-05-18 03:57:44 PDT jian.yafeng@zte.com.cn

MAPInventorySettings	hyd	SerialNumber	A string Identifying the particular device that is unique for the indicated model and manufacturer.	8074
		ExecutionEnvironment	A string identifying the execution environment (operating system) in the device.	Linux
		ChipsetVendor	A string identifying the Wi-Fi chip vendor of this radio.	QTI
		SoftwareVersion	A string identifying the software version currently installed in the device (i.e. version of the overall device firmware).	11.x
MAPSPSettings	hyd	MultiAPSPSettings	The section that defines the MAP service prioritization configuration settings	MultiAP SPSettings
		EnableSP	Enable (1) or disable (0) MAP service prioritization	0
		EnableEnhancedSP	Enable (1) or disable (0) enhanced MAP service prioritization	0
		MaxSPRules	A string identifying the maximum number of service prioritization rules	0

8.49.6 Wi-Fi parameters for multi-AP

Configuration Type	Option	Description	Default
wifi-device	repacd_map_bsta_selected	Used to mark the radio that is currently selected to act as the bSTA.	
	repacd_map_bsta_preference	The relative priority for the radio to act as the bSTA. This must be set to a value greater than 0 for a radio to be considered as a bSTA.	
wifi-iface	wps_pbc_skip	Do not perform WPS Push Button Configuration on a STA interface that is currently connected.	0

Qualcomm Confidential – May Contain Trade Secrets 2021-09-13 03:57:44 PDT yuan.yafengchen@qualcomm.com.cn

8.49.7 UCI parameters for legacy AP steering, backhaul steering, and channel selection in MAP

Table 8-12 UCI parameters for legacy AP steering, backhaul steering, and channel selection in MAP

Parameter	Default value	Description
sysMaxOffChanScanTimePerScan	40 ms	The maximum time an Agent spends on offchannel scan.
sysThresholdMeanRSSIChangeAsChannelSelectionTrigger	See description	The RSSI threshold for which a Controller should trigger the Channel Preference Query. By default, it is set to at least one neighboring AP's RSSI is changed up to 9 dB, or at least two neighboring AP's RSSIs are changed up to 6 dB.
LowRSSIAPSteeringThreshold	-78 dBm	A threshold for triggering AP steering evaluation.
APSteeringMaxRetryCount	12	The maximum number of retries when AP steering trigger condition persists.
SSID_backhaul	"default_ backhaul"	The SSID of all the backhaul BSSes.
SSID_fronthaul	"default_ fronthaul"	The SSID of all the fronthaul BSSes. Note it is allowed that this SSID is the same as SSID_ backhaul.
SSID_guest	"default_guest"	The SSID of the guest network (only applies to the root Agent).
BackhaulRSSIThreshold_2	-78 dBm	The minimum RSSI of a 2.4 GHz BSS that an Agent may connect to for its backhaul link.
BackhaulRSSIThreshold_5	-78 dBm	The minimum RSSI of a 5 GHz BSS that an Agent may connect to for its backhaul link.
num5Gattempts		The number of times an Agent (re)enables its bSTA on 5 GHz to attempt to connect to an upstream AP on 5 GHz.
BackhaulRSSIThreshold_offset		The rssi offset (relative to BackhaulRSSIThreshold_2) the 2.4 GHz RSSI strength needs to exceed for an Agent to enable its bSTA on 5 GHz to attempt to connect to an upstream AP on 5 GHz.
ScanPeriod	5 sec	The time to re-scan for a BSS with SSID_ backhaul if one is not found.
NoExtendtThreshold_2	-80 dBm	The minimum RSSI value for an Agent to be an extender. This value should be less than BackhaulRSSIThreshold_2.
TargetGainThreshold	40%	The expected capacity gain for an Agent when it associates to the target BSS commanded by the Controller using backhaul steering request.
NQCLegacySteering	0	A flag to allow legacy client steering by the Controller. If NQCLegacySteering is set to 0, legacy client steering will be only triggered on the Qualcomm Controller for Qualcomm Agents.

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 09:27:44 GMT pan.yaxing@zte.com.cn

Table 8-12 UCI parameters for legacy AP steering, backhaul steering, and channel selection in MAP

Parameter	Default value	Description
MetricReportIntvl	5 sec	The periodic metric reports sent by the Agent to the Controller.
APSteerToPeerMinRSSIIncThreshold	7 dBm	The minimum RSSI improvement of the target BSS.
APSteerUnassociatedSTARSSIWaitTime	3 sec	The time the Controller waits when collecting all unassociated STA RSSI measurements after sending the query.
APSteerUnassociatedSTARSSIValidTime	5 sec	The maximum age of the unassociated STA RSSI measurements that are deemed valid and being considered for AP steering.
UnassociatedRSSIProhibitTimeShort	TBD	The prohibit time for Unassociated STA RSSI measurements when no valid RSSI is received in time besides the serving RSSI.
UnassociatedRSSIProhibitTimeLong	TBD	The prohibit time for Unassociated STA RSSI measurements when any valid RSSI is received in time besides the serving RSSI.
BSSAPSteerTopCandidateCapacityThreshold	80%	The threshold to select top steering targets, whose ranking is determined based on capacity and hop count. Assume that the highest capacity offered by the eligible target BSSes is *max_capacity*. Our algorithm selects the top BSSes whose remaining capacity is greater than (max_capacity * BSSAPSteerTopCandidateCapacityThreshold).
BSSAPSteerUplinkDownlinkImbalance	0	The conservative offset that is used to convert an uplink RSSI to downlink RSSI.
reservedAirtime	30%	The airtime reserved on the channel of the target BSS for steering to ensure the channel will not be overloaded by the client that is steered to it.

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT panyafeng@zte.com.cn

8.50 UCI settings to preserve ATF configurations across a system reboot

For more information on usage guidelines and the functionality regarding these parameters, see the *Preserve ATF configurations across a system reboot* section of the *Memory and bandwidth optimization features* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

ATF configurations are preserved across a reboot of a system. To enable ATF settings to be retained after a system restart, UCI support is necessary for these settings. The following ATF configurations are maintained across a system restart:

1. SSID-based configuration
2. Group-based configuration
3. Peer-based configuration
4. Throughput-based configuration
5. AC-based configuration
6. Global configuration - VAP and radio parameters for ATF

8.50.1 UCI configuration sections for ATF

Different configuration sections are added in the UCI wireless configuration file to support ATF configurations.

8.50.1.1 SSID-based ATF configuration

For every SSID-based configuration, the user must create a new configuration section by the name, ‘atf-config-ssid’ and add appropriate options in this section.

The desired options for SSID-based configurations are as follows:

1. device—Wi-Fi device on which the rule has to be configured
2. command—addssid for adding an SSID-based configuration and delssid for deleting an SSID-based configuration
3. ssid—SSID for which the rule is configured
4. airtime—Airtime percentage to be configured for the SSID

UCI commands

UCI commands to create the section and update the preceding list of options are as follows:

```
uci add wireless atf-config-ssid
uci set wireless.@atf-config-ssid[0].device=wifi2
uci set wireless.@atf-config-ssid[0].command=addssid
uci set wireless.@atf-config-ssid[0].ssid=atf1
uci set wireless.@atf-config-ssid[0].airtime=80
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Wireless config file

After issuing the preceding set of UCI commands, the wireless config file is appended with the following section.

```
config atf-config-ssid 'ssid'
        option device wifi2/* wifi device on which the rule has to be
configured */
        option command 'addssid'/* command – addssid/delssid */
        option ssid 'atf1'/* SSID */
        option airtime '80'/* Airtime percentage */
```

On invoking the command ‘wifi’ or on board reboot, qcawifi.sh will parse this configuration section and invoke the following wlanconfig commands to configure SSID-based ATF

```
wlanconfig <vap> addssid <ssid> <val>
wlanconfig <vap> delssid <ssid>
```

8.50.1.2 STA-based ATF configuration

For every STA-based configuration, the user must create a new configuration section by the name, ‘atf-config-sta’ and add appropriate options in this section.

The desired options for STA-based configurations are:

1. device—Wi-Fi device on which the rule must be configured
2. command—addsta for adding a STA-based configuration and delsta for deleting a STA-based configuration
3. macaddr—Peer MAC address for which the rule is configured
4. airtime—Airtime percentage to be configured for the SSID
5. ssid—SSID for which this rule is applicable

UCI commands

UCI commands to create the section and update the preceding list of options are as follows:

```
uci add wireless atf-config-sta
uci set wireless.@atf-config-sta[0].device=wifi2
uci set wireless.@atf-config-sta[0].command=addsta
uci set wireless.@atf-config-sta[0].macaddr=02:03:04:05:06:07
uci set wireless.@atf-config-sta[0].airtime=15
uci set wireless.@atf-config-sta[0].ssid=atf1
```

Wireless config file

After issuing the preceding set of UCI commands, the wireless config file is appended with the following section.

```
config atf-config-sta 'sta'
        option device wifi2/* wifi device on which the rule has to be
configured */
        option command 'addsta'/* command – addsta/delsta */
        option macaddr '02:03:04:05:06:07'  /* MAC address of the STA */
        option airtime '15'/* Airtime percentage */
```

```
        option ssid 'atf1'/* SSID for which this configuration is
applicable */
```

qcawifi.sh will parse this configuration section and invoke the following wlanconfig command

```
wlanconfig <vap> addsta <mac> <val>
wlanconfig <vap> delsta <ssid>
```

8.50.1.3 Group-based ATF configuration

Group-based ATF configuration is applicable for non-offload chipsets and QCA9880 only. Other offload chipsets do not support this feature. For every group-based ATF configuration, the user must create a new configuration section by the name, 'atf-config-group' and add appropriate options in this section.

The desired options for group-based ATF configurations are:

1. device—Wi-Fi device on which the rule has to be configured
2. command—addgroup for adding and configuring a group, and delgroup for deleting a group
3. group—Name of the group to be added/deleted
4. ssid —List of SSIDs to be added in the group
5. airtime—Airtime to be configured for the group

UCI commands

UCI commands to create the section and update the preceding list of options are as follows:

```
uci set wireless.@wifi-iface[0].atfssidgroup='1'
uci add wireless atf-config-group
uci set wireless.@atf-config-group[0].device='wifi0'
uci set wireless.@atf-config-group[0].command='addgroup'
uci set wireless.@atf-config-group[0].group='GROUP1'
uci set wireless.@atf-config-group[0].airtime='80'
uci add_list wireless.@atf-config-group[0].ssid='atf1'
uci add_list wireless.@atf-config-group[0].ssid='atf2'
```

Wireless config file

After issuing the preceding set of UCI commands, the wireless config file is appended with the following section.

```
config atf-config-group 'GROUP'
        option device wifi0/* wifi device on which the rule has to be
configured */
        option command 'addgroup'/* command - addgroup/delgroup */
        option group 'group1'/* group name */
        list ssid 'atf1'/* List of SSID's  to be added to the group */
        list ssid 'atf2'/* List of SSID's  to be added to the group */
       option airtime '80'/* airtime percentage */
```

qcawifi.sh parses this configuration section and invokes the following wlanconfig commands:

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
wlanconfig ath0 addatfgroup <group> <ssid>
wlanconfig ath0 configatfgroup <group> <val>
wlanconfig ath0 delatfgroup <group>
```

Throughput-based ATF configuration

For every throughput-based ATF configuration, the user must create a new configuration section by the name, 'atf-config-tput' and add appropriate options in this section.

The desired options for throughput-based ATF configurations are:

1. device—Wi-Fi device on which the rule has to be configured
2. command—addtputsta for adding a throughput-based configuration and deltputsta for deleting a throughput-based configuration
3. throughput—Desired throughput in kbps
4. max_airtime—Maximum airtime percentage that can be used by this peer
5. macaddr—MAC address of the PEER

UCI commands

UCI commands to create the section and update the preceding list of options are as follows:

```
uci set wireless.@wifi-iface[3].atf_tput_at='1
uci add wireless atf-config-tput
uci set wireless.@atf-config-tput[0].device='wifi2'
uci set wireless.@atf-config-tput[0].command='addtputsta'
uci set wireless.@atf-config-tput[0].throughput='25000'
uci set wireless.@atf-config-tput[0].max_airtime='30'
uci set wireless.@atf-config-tput[0].macaddr='00:01:02:03:04:05'
```

Wireless config file

After issuing the preceding set of UCI commands, the wireless config file is appended with the following section.

```
config atf-config-group 'GROUP'
        option device wifi2/* wifi device on which the rule has to be
configured */
        option command 'addtputsta'/* command - addtputsta/deltputsta */
        option throughput '25000'/* Desired throughput in Kbps*/
        option max_airtime '30'/* Maximum airtime for this peer */
        option macaddr '00:01:02:03:04:05'/* Peer MAC*/
```

qcawifi.sh will parse this configuration section and invoke the following wlanconfig commands:

```
wlanconfig ath0 addtputsta 00:01:02:03:04:05 25000 30
wlanconfig ath0 deltputsta 00:01:02:03:04:05
```

8.50.1.4 AC-based ATF configuration

The support for AC-based ATF configuration is currently available only on QCA_Networking_ 2017_SPF.6.0 release and supported only for DA chipsets and QCA9880.

For every AC-based ATF configuration, the user must create a new configuration section by the name, ‘atf-config-ac’ and add appropriate options in this section.

The desired options for AC-based ATF configurations are:

1. device—Wi-Fi device on which the rule has to be configured
2. command—addac to add AC-based config; delac to delete AC-based config
3. ac—Access category name. BE/BK/VI/VO
4. airtime—Airtime percentage for this AC
5. ssid—SSID for which this rule applies.

UCI commands

UCI commands to create the section and update the preceding list of options are as follows:

```
uci add wireless atf-config-ac
uci set wireless.@atf-config-tput[0].device='wifi1'
uci set wireless.@atf-config-tput[0].command='addac'
uci set wireless.@atf-config-tput[0].ac=’BE’
uci set wireless.@atf-config-tput[0].airtime='20'
uci set wireless.@atf-config-tput[0].ssid= ‘atf1
```

Wireless config file

After issuing the preceding set of UCI commands, the wireless config file is appended with the following section.

```
config atf-config-ac 'AC'
        option device wifi1/* wifi device on which the rule has to be
configured */
        option command 'addac'/* command - addac/delac */
        option ac 'BE'/* AC name - BE/BK/VI/VO */
        option airtime '20'/* airtime percentage */
        option ssid 'atf1'/* SSID on which this rule is applicable */
```

qcawifi.sh parses this configuration section and invokes the following wlanconfig commands:

```
wlanconfig athX atfaddac <ssid/groupname> <ac_name>:<val>
wlanconfig athX atfdelac <ssid/groupname> <ac_name>
```

8.50.1.5 ATF radio parameters

Support for the following ATF radio (wifi) parameters is also added. These options can be defined under the wifi-device section:

1. uci set wireless.wifi2.atfstrictsched='0'—Set to 0 for Fair-queue scheduling. Set to 1 for Strict scheduling.
2. uci set wireless.wifi2.atfobsssched='0'—Set to 0 to disable OBSS in ATF. Set to 1 to enabled OBSS in ATF.
3. uci set wireless.wifi2.atfgrouppolicy='0'—Set to 0 for inter group policy as fair. Set to 1 for inter group policy as strict.

8.50.1.6 ATF VAP parameters

Support for the following ATF VAP parameters is also added. These options can be defined under the wifi-interface section:

1. uci set wireless.@wifi-iface[1].commitatf=1—Enable or disable ATF on runtime.
2. uci set wireless.@wifi-iface[1].atfssidsched=1—Set to 1 for Strict scheduling on a specific SSID. Set to 0 for Fair scheduling on a specific SSID.
3. uci set wireless.@wifi-iface[3].atf_tput_at='1'—Enable or disable throughput-based ATF.
4. uci set wireless.@wifi-iface[3].atfssidgroup='1'—Enable or disable ATF SSID grouping.
5. uci set wireless.@wifi-iface[3].atf_max_buf='512'—Set maximum buffers per peer.
6. uci set wireless.@wifi-iface[3].atf_min_buf='256'—Set minimum buffers per peer.
7. uci set wireless.@wifi-iface[3].atfmaxclient='1'—Enable or disable ATF maximum clients feature.

8.51 UCI parameters to configure AP TWT

For more information on usage guidelines and the functionality regarding these parameters, see the *Configure AP TWT* section of the *Beacons and frames transmission* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Negotiation of implicit Target Wake Time (TWT) sessions, including handling of TWT Information elements is supported. Host-side enhancements are implemented for AP TWT.

To reduce the consequences of channel contention, the TWT mechanism included in the IEEE 802.11ax amendment facilitates an easy and effective mechanism to schedule transmissions in time. Apart from reducing the contention between stations, the use of TWT also enables other mechanisms in the IEEE 802.11 universe, such as multiuser transmissions, multi-AP cooperation, spatial reuse and coexistence in high-density WLAN scenarios to be leveraged. The TWT mechanism in IEEE 802.11ax is based on the implementation in IEEE 802.11ah. In addition to the Individual TWT mode, it also includes a new Broadcast TWT mode. It is expected that this new mode will be able to improve the efficiency of the TWT operation and leverage the new multiuser capabilities of 802.11ax. With the individual TWT mechanism, stations can agree with the AP on a common wake scheduling, allowing them to wake up only when required, hence to minimize energy consumption and contention within the Basic Service Set (BSS).

To initiate a TWT session, a negotiation phase occurs at first in which the AP and the target station agree on a common set of parameters. The significant parameters are as follows:

- *Target Wake Time (TWT)*: The next point in time, in microseconds, at which the station participating in the TWT-based communication must wake up for the TWT session.
- *TWT Wake interval*: The time interval between subsequent TWT sessions for the station; the value is higher than 0 when the TWT is periodic.
- *Minimum TWT awake duration*: Minimum time duration a station must stay awake since the starting time of the TWT session so as to be able to receive frames from the other stations.
- *TWT Channel*: The channel a station can use temporarily as the primary one.

- *TWT Protection*: Mechanism employed to protect a TWT session from transmissions of external stations, such as the RTS/CTS.

The TWT timing can be explicit or implicit: while the former requires specification of the TWT parameters before each new session, the latter allows to set periodic sessions by relying on the first set of parameters until a new set is received. After the TWT parameters are agreed, the stations can go to sleep until the next TWT session is reached. In practice, each station can establish up to 8 TWT sessions with the AP.

To modify TWT-enabled configuration parameters, enter the following commands:

```
uci set wireless.qcawifi=qcawifi
uci set wireless.qcawifi.twt_add_sta_slot_interval=1000
uci set wireless.qcawifi.twt_mbss_support=0
uci set wireless.qcawifi.twt_congestion_thresh_critical=100
uci set wireless.qcawifi.twt_min_no_sta_setup=10
uci set wireless.qcawifi.twt_congestion_thresh_setup=50
uci set wireless.qcawifi.twt_min_no_sta_teardown=2
uci set wireless.qcawifi.twt_congestion_thresh_teardown=20
uci set wireless.qcawifi.twt_min_no_twt_slots=4
uci set wireless.qcawifi.twt_default_slot_size=10
uci set wireless.qcawifi.twt_mode_check_interval=10000
uci set wireless.qcawifi.twt_interference_thresh_setup=50
uci set wireless.qcawifi.twt_no_of_bcast_mcast_slots=2
uci set wireless.qcawifi.twt_interference_thresh_teardown=80
uci set wireless.qcawifi.twt_remove_sta_slot_interval=5000
uci set wireless.qcawifi.twt_max_no_sta_twt=500
uci set wireless.qcawifi.twt_sta_cong_timer_ms=5000
uci commit
wifi load

wifitool athX ap_twt_add_dialog <peer_mac> <dialog id> <wake intvl (us)>
<wake int mantis> <wake dur(us)> <sp offset(us)> <cmd (0-7)> <flags>
wifitool athX ap_twt_del_dialog <dialog id>
wifitool athX ap_twt_pause_dialog <dialog id>
wifitool athX ap_twt_resume_dialog <dialog id> <sp offset(us)>
```

8.52 UCI commands to configure the legacy upgrade-steering mechanism for clients

To enable the feature to control the legacy upgrade-steering mechanism for clients, enter the following UCI commands:

```
uci set lbd.StaDB_Adv.LegacyUpgradeAllowedCnt=3
uci set lbd.StaDB_Adv.LegacyUpgradeMonitorDur=2100
uci set lbd.SteerExec_Adv.LegacyUpgradeUnfriendlyTime=21600
uci commit lbd
```

Configuration Type	Section	Option	Description
StaDB	StaDB_Adv	LegacyUpgradeAllowedCnt	Number of times for which initiation of legacy upgrade-steering is allowed, for example, 3 times
StaDB	StaDB_Adv	LegacyUpgradeMonitorDur	Duration for which legacy upgrade-steering is allowed for a client, for example, 35 min
SteerExec	SteerExec_Adv	LegacyUpgradeUnfriendlyTime	Duration for which a client is considered to be ineligible for legacy upgrade-steering, for example, 6 hours

Enable the 'LegacyUpgradeAllowedCnt' lbd config option using the UCI command. Bring up repacd using the default two-radio configuration on CAP and RE. Initiate legacy-upgrade steering. When the number of legacy-upgrade steering attempts reaches the maximum limit specified in the LegacyUpgradeAllowedCnt parameter within the LegacyUpgradeMonitorDur interval, then the client is marked as legacy upgrade unfriendly.

8.53 UCI commands for EAPOL minimum rate WMI resource configuration

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Support for EAPOL minimum rate WMI resource configuration* section of the *Beacons and frames transmission* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Using the global.ini configuration parameter - eapol_minrate_set and eapol_minrate_ac_set, the user can set Extensible Authentication Protocol Over LAN (EAPOL) frames to be sent in minimum rate in tunnel mode. In addition to this setting, the user can also select between the 4 ACs (BE, BK, VI, VO) to send the EAPOL frames. Two UCI configuration commands are introduced for this purpose. To enable the feature (1 sends the EAPOL frames in minimum rate and 0 disables the feature), enter the following:

```
uci set wireless.qcawifi=qcawifi;
uci set wireless.qcawifi.enable_eapol_minrate=1;
uci commit;
wifi load
```

To select the access class (AC) based on the number provided (0 - Best Effort, 1 - Background, 2 - Video, 3 - Voice), enter the following:

```
uci set wireless.qcawifi=qcawifi;
uci set wireless.qcawifi.set_eapol_minrate_ac=2;
uci commit;
wifi load
```

By default, if the access class is not set, the EAPOL frames are sent with the Voice access class (3); otherwise, they are sent with the specified access class.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

8.54 Configure smart mesh enhancements based on monitor-direct feature for IPQ807xA and IPQ817x

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Smart mesh enhancements based on monitor-direct feature for IPQ807xA, IPQ817x, and IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx/QCN90xx or QCN602x + IPQ807xA/IPQ60xx/IPQ50xx* section of the *Core channel selection capabilities* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

The IPQ807xA and IPQ817x platforms, while functioning as the AP device under test (DUT), can capture packets from non-associated clients (NAC) using monitor direct feature. On programming the NAC from user, AP can sniff packets from this NAC. The hardware (IPQ807xA in this case) indicates that these packets are from NAC using monitor direct bit set in the TLV.

The NACs on IPQ807xA are programmed in HW Address Search Table (AST) with monitor direct bit set by the FW. The existing WMI for NAC is used here. BSSID programming is no longer required for IPQ807xA. Packets that are captured are forwarded to upper layer after radiotap header is appended.

Packets from NAC contain the Wi-Fi header only and data is not subscribed. Therefore, only the monitor status ring is used for this feature.

A maximum of 24 clients can be programmed currently.

Enter the following settings in `/etc/config/wireless` to configure monitor-direct feature for packet-capture from NACs:

```
config wifi-iface
        option device   wifi0
        option network  lan
        option mode     ap_smart_monitor
        option ssid     MON_AP
        option encryption none
```

To program NAC client and read RSSI, enter the following wlanconfig commands :

```
wlanconfig athX nac add/del client <ad1 eg: xx:xx:xx:xx:xx:xx> <ad2> …..
wlanconfig athX nac list client
```

8.55 UCI commands to configure DPP

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Configure DPP for WFA Easy Connect* section of the *WLAN AP modes* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

In Device Provisioning Protocol (DPP), public keys are used to identify and authenticate all devices. The private key associated with a public key should be generated within each device and protected from disclosure. Devices use public key cryptographic techniques to authenticate peer devices and establish shared keys for further secure communications. This architecture simplifies

the establishment of secure connectivity between devices and provides a foundation for improved usability in provisioning and connecting devices

Enter the following UCI commands to configure DPP:

```
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='ap'
uci set wireless.@wifi-iface[1].wds='1'
uci set wireless.@wifi-iface[1].ssid='DPP'
uci set wireless.@wifi-iface[1].encryption='none'  (or) encryption='dpp'
uci set wireless.@wifi-iface[1].dpp='1'
uci set wireless.@wifi-iface[1].dpp_type='qrcode'
uci set wireless.@wifi-iface[1].dpp_curve='P-256'
uci set wireless.@wifi-iface[1].dpp_
key='30770201010420e7a9e2a43291cb5889cc37b540434cfacf6762156d013d6e2e88b
d1ce0364562a00a06082a8648ce3d030107a144034200040dbb18afb95c60a634a8653f1
24e4863a041aea37a69f10391ac57496d70fa60e86707b8e0bde3981d8622fc3385aa1a3
a29e204ceedc6ce0b70b07fe0fc5112'
/* Following will be used if the dpp_type is pkex */
uci set wireless.@wifi-iface[1].pkex_identifier='dpp_device'
uci set wireless.@wifi-iface[1].pkex_code='password'
```

- When encryption is set as none, the AP comes up in open security mode and it can have support of DPP by entering "option dpp 1" (default value is 0 for open security mode). When encryption is set as dpp, the AP comes up in DPP mode and it can have support of DPP by entering "option dpp 1".
- dpp_type, dpp_curve, dpp_key, pkex_identifiers are optional to set
- default dpp_type is 'qrcode' if we don't configure dpp_type. Supported dpp_types are ‘qrcode’, ‘pkex’
- dpp_curve:
  - The DPP protocol design binds the keys used in bootstrapping, authentication, and data connection in a manner that mandates a single elliptic curve is selected for all devices in any given network.
  - Supported dpp_curves are P-256, P-384, P-521, BP-256, BP-384, BP-512.
- dpp_key—This is a base64 encoded string and the length depends on which dpp_curve is being used.

Elliptic curve	base64 encoded length
P-256	80 characters
P-384	96 characters
P-521	120 characters

- pkex_identifier—Non-secret identifier for the code can be transmitted to support the case where a PKEX implementation may be provisioned to connect to a plurality of devices and needs to know which code to use to process a received PKEX frame. If an optional code

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

identifier is used, it shall be a UTF-8 string not greater than eighty (80) octets that is provisioned at the same time as the shared code. This to be same in both configurator and enrollee.

- pkex_code—It is a password/key to unmask the bootstrapping key.

Qualcomm
Confidential – May Contain Trade Secrets
2021-05-13 03:57:44 PDT
pan.yafeng@zte.com.cn

8.56 UCI commands to configure Rx flow identification based on protocol type

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Rx flow identification based on protocol type* section of the *WLAN AP driver layer operations* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

The capability to identify Rx flows matching a specific criterion as programmed by the application is implemented. Support is implemented for tagging of Rx packets based on the following protocol types:

- Ethernet types: ARP (NS not required), EAP
- IP protocol (v4 and v6): TCP, UDP, ICMP
- UDP: DHCP, DNS

When a packet matches the preceding criteria, the application is to be notified with an appropriate meta header/meta data.

Tagging of packets should occur on both reorder (REO) ring and monitor rings.

8.56.0.1 UCI interface for defining protocol tags

The UCI interface can be used to add tags for select packet types. The protocol rules would then be triggered before a VAP is enabled.

```
uci set wireless.wifiN.rx_protocol_arp_type_tag=<16-bit  non-zero tag>
uci set wireless.wifiN.rx_protocol_dhcpv4_type_tag=<16-bit  non-zero tag>
uci set wireless.wifiN.rx_protocol_dhcpv6_type_tag=<16-bit  non-zero tag>
uci set wireless.wifiN.rx_protocol_dns_tcpv4_type_tag=<16-bit  non-zero
tag>
uci set wireless.wifiN.rx_protocol_dns_tcpv6_type_tag=<16-bit  non-zero
tag>
uci set wireless.wifiN.rx_protocol_dns_udpv4_type_tag=<16-bit  non-zero
tag>
uci set wireless.wifiN.rx_protocol_dns_udpv6_type_tag=<16-bit  non-zero
tag>
uci set wireless.wifiN.rx_protocol_icmpv4_type_tag=<16-bit  non-zero tag>
uci set wireless.wifiN.rx_protocol_icmpv6_type_tag=<16-bit  non-zero tag>
uci set wireless.wifiN.rx_protocol_tcpv4_type_tag=<16-bit  non-zero tag>
uci set wireless.wifiN.rx_protocol_tcpv6_type_tag=<16-bit  non-zero tag>
uci set wireless.wifiN.rx_protocol_udpv4_type_tag=<16-bit  non-zero tag>
uci set wireless.wifiN.rx_protocol_udpv6_type_tag=<16-bit  non-zero tag>
uci set wireless.wifiN.rx_protocol_ipv4_type_tag=<16-bit  non-zero tag>
uci set wireless.wifiN.rx_protocol_ipv6_type_tag=<16-bit  non-zero tag>
uci set wireless.wifiN.rx_protocol_eap_type_tag=<16-bit  non-zero tag>
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

8.56.0.2 cfg80211tool commands

Tagging packets based on protocol type

Usage: *cfg80211tool <radio_id> set_rxProtocolTag 0 <protocol_type_enum> <16-bit tag>*

NOTE All arguments are mandatory.

Example:

cfg80211tool wifi0 *set_rxProtocolTag* 0 0 XXXX //For Addition of RX protocol type 0 (ARP)

Untagging packets based on protocol type

Usage: *cfg80211tool <radio_id> set_rxProtocolTag 1 <protocol_type_enum>*

Example:

cfg80211_tool wifi0 *set_rxProtocolTag* 1 1 //For Deletion of RX protocol type 1

Obtaining statistics for protocol type tags

Usage: *cfg80211tool <radio_id> dump_rxProtocolTagStats <protocol_type>*

NOTE If protocol_type is 255, the counters for all protocol types will be provided. If a specific protocol_type is provided, then only its counter will be provided.

NOTE Per-protocol counters are not maintained for monitor ring. Counters are tracked only for REO ring.

Example:

cfg80211tool wifi0 *dump_rxProtocolTagStats* 0 //Dump counters for *protocol_type = 0 (ARP).*

cfg80211_tool wifi0 *dump_rxProtocolTagStats 255 //Dump counters for all protocol type tags*

8.57 UCI configuration of IPv4 over IPv6 (464) XLAT and CLAT

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *IPv4 over IPv6 (464) XLAT implementation in Linux* section of the

Networking offload and acceleration of link and multilink features chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Support for 464xlat is introduced in Linux.

CLAT is done by the combination of nat44 and nat46. UCI configuration is supported. An example of this configuration setting is as follows:

```
uci set network.wan6=interface
uci set network.wan6.proto=static
uci set network.wan6.ifname='eth0’
uci set network.wan6.ip6addr='3001:db8::10/64’
uci set network.wan6.ip6gw='3001:db8::1’
uci set network.xlat=interface
uci set network.xlat.ip6prefix='fd63:fab9:6ccf:64::/96’
uci set network.xlat.proto='464xlat’
uci set network.xlat.ip6addr=‘2001:db8::100’
uci commit
```

NAT44 setting is added at runtime.

Test 464xlat (between CLAT and PLAT)

PLAT configuration

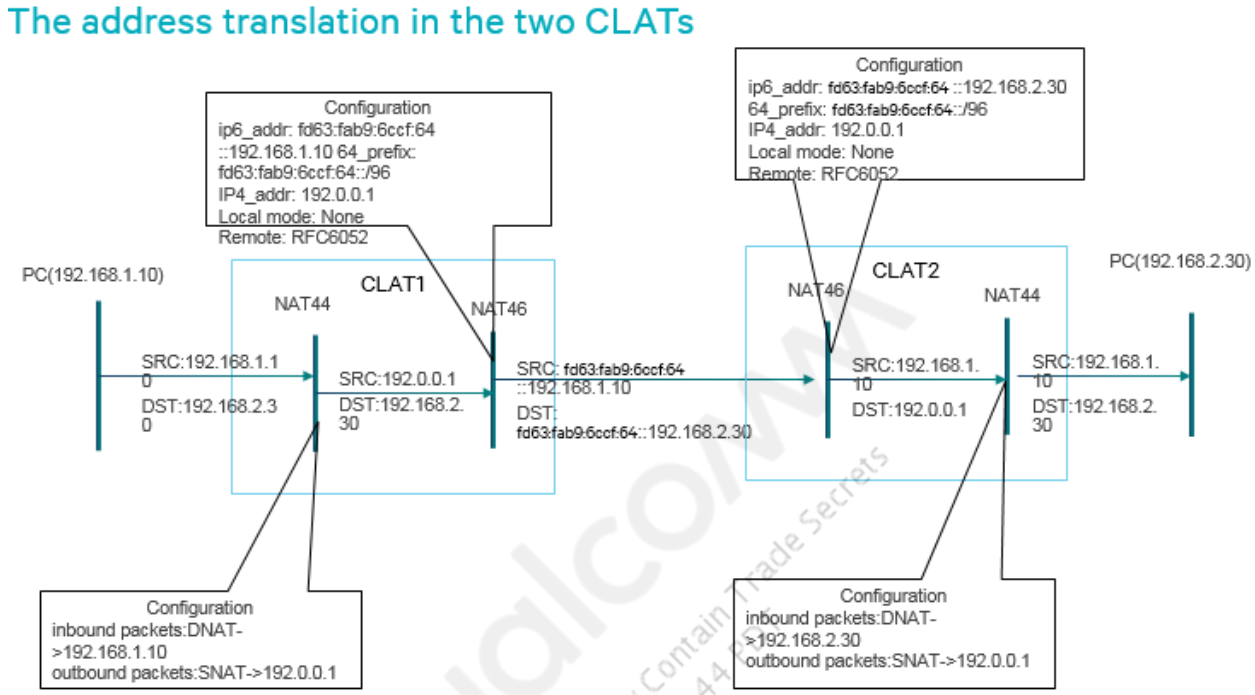
```
uci set network.nat64=interface
uci set network.nat64.proto='tayga’
uci set network.nat64.ifname='tayga-nat64’
uci set network.nat64.ipv4_addr='192.168.64.1’
uci set network.nat64.prefix='fd63:fab9:6ccf:64::/96’
uci set network.nat64.dynamic_pool='192.0.64.0/24’
uci set network.nat64.accept_ra='0’
uci set network.nat64.send_rs=‘0’
uci commit
```

CLAT configuration

```
uci set network.xlat=interface
uci set network.xlat.ip6prefix='fd63:fab9:6ccf:64::/96’
uci set network.xlat.proto='464xlat’
uci set network.xlat.ip6addr=‘2001:db8::100’
uci commit
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Test 464xlat (between CLAT and PLAT)



Test 464xlat (between two CLATs)

Use two CLATs with DNAT setting.

CLAT1

```
uci set network.wan6=interface
uci set network.wan6.proto=static
uci set network.wan6.ifname='eth0'
uci set network.wan6.ip6addr='3001:db8::10/64'
uci set network.wan6.ip6gw='3001:db8::1'
uci set network.xlat=interface
uci set network.xlat.ip6prefix='fd63:fab9:6ccf:64::/96'
uci set network.xlat.proto='464xlat'
uci set network.xlat.ip6addr='fd63:fab9:6ccf:64::192.168.1.10'
uci add firewall redirect
uci set firewall.@redirect[-1].target=DNAT
uci set firewall.@redirect[-1].src=wan
uci set firewall.@redirect[-1].dest=lan
uci set firewall.@redirect[-1].proto="tcp udp icmp"
uci set firewall.@redirect[-1].dest_ip=192.168.1.10
uci set firewall.@redirect[-1].name=DNAT
uci commit
```

CLAT2

```
uci set network.lan.ipaddr=192.168.2.1
   uci set network.wan6=interface
   uci set network.wan6.proto=static
   uci set network.wan6.ifname='eth0’
   uci set network.wan6.ip6addr='3001:db8::10/64’
   uci set network.wan6.ip6gw='3001:db8::1’
   uci set network.xlat=interface
   uci set network.xlat.ip6prefix='fd63:fab9:6ccf:64::/96’
   uci set network.xlat.proto='464xlat’
   uci set network.xlat.ip6addr='fd63:fab9:6ccf:64::192.168.2.30’
   uci add firewall redirect
   uci set firewall.@redirect[-1].target=DNAT
   uci set firewall.@redirect[-1].src=wan
   uci set firewall.@redirect[-1].dest=lan
   uci set firewall.@redirect[-1].proto="tcp udp icmp"
   uci set firewall.@redirect[-1].dest_ip=192.168.2.30
   uci set firewall.@redirect[-1].name=DNAT
   uci commit
```

Qualcomm
Confidential – May Contain Trade Secrets
2021-05-13 03:57:44 PDT
pan.yafeng@zte.com.cn

8.58 Configure the live dump collection utility

NOTE For more information on usage guidelines and the functionality regarding these parameters, see the *Qualcomm live dump collection utility to debug WLAN host driver* section of the *WLAN AP driver layer operations* chapter in the *Wireless LAN Access Point (Driver Version 11.0) Programming Guide* (80-YA728-6).

Starting with QCA_Networking_2019.SPF11.0 release, on IPQ401x, IPQ806x, and IPQ807x platforms, the Qualcomm live dump (QLD) utility is introduced to enable the collection of application dumps from the registered modules in the WLAN driver, whenever a target assert occurs or when a user needs to generate such a dump for troubleshooting and diagnosis. The firmware recovery functionality must be enabled by entering the `cfg80211tool wifiX set_fw_recovery 1` command before the live dump collection utility can be used.

On FW assert, the default behavior is to reboot the system, to collect the whole Host/FW memory for debugging later. This method is not useful in the field because collecting the whole memory requires extra configurations, such as additional setup with TFTP server. Although the FW memory can be collected separately without rebooting the system, there is no host state or status collected. Also, if FW recovery is enabled, the recovery routines will reinitialize the whole host, thereby losing critical host information. The requirement is to collect and store extensive host state or status of different modules across the WLAN driver to enable the FW issues to be debugged.

The mechanism called QCA Live Dump (QLD) proposes a framework within the WLAN host driver and application layer, which allows different modules of the driver to register any critical data structures, so that these data structures can be collected, in the event of a target assert or on user request, and then transferred to the application for further dumping in file system or network.

These application dumps will be in the format same as the Linux application core dump, so that they can be loaded in GDB for further analysis. In fact, the dump is created using one of SIGABRT or SIGQUIT signals so that its understood by GDB.

The dump also provides a master table indicating the 'address' and 'name' of the data structure, as registered by different driver modules. Optionally, the 'name' can match the 'type' of the data structure, so that the 'address' can be sent using typecast to the corresponding 'name' in GDB.

Any unregistered data structure can also be retrieved from a live system, by issuing a command with the 'address' and 'size' of the data structure. The memory will be collected in a similar fashion and the core dump will be made available for debug using GDB.

The user registers the structures of interest. Kernel makes a linked list in driver that continues adding into this linked list. At user space there is a daemon listening for events when the target asserts. On receiving target assert kernel module collects all the entries that have been registered with following data ("Kernel address","size of structure","name") .The entries are combined and send to the event listener registered from user space. At user space, the event listener starts parsing the events to extract the metadata and fill user-level structure with data ("User address ""Kernel address","size of structure","name") at this state; the user address is zeroed out at init.

After all the entries have been parsed at user space, user starts process of getting the memory of each entry by allocating user memory and doing Netlink call to copy kernel memory to user memory.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

After all the requested memories have been copied to user space, a child process is started and then forced to fault, resulting in coredump collection.

The coredump is collected and analyzed under gdb for debugging.

NOTE The firmware recovery functionality must be enabled by entering the `cfg80211tool wifiX set_fw_recovery 1` command before the live dump collection utility can be used.

Enter the following UCI configuration commands to set up the utility:

```
uci set wireless.qcawifi=qcawifi
uci set wireless.qcawifi.qld_max_list=13
uci commit
```

8.59 Globally disable same SSID feature using UCI settings

Only one AP, VAP, or radio is supported when the same SSID feature is enabled up to a maximum of three radios. It is necessary to manually disable the same SSID support in certain scenarios. Starting with QCA_Networking_2019.SPF.11.0, a UCI command to disable the same SSID support is implemented. To disable the same SSID feature, enter the following:

```
uci set wireless.qcawifi=qcawifi
uci set wireless.qcawifi.samessid_disable=1
uci commit wireless
```

By default, the same SSID disable parameter, `samessid_disable`, is 0. The same SSID disable functionality cannot be configured using the cfg80211tool tool. It is recommended to disable this same SSID support feature (option samessid_disable '1'), when Wi-Fi SON is enabled.

8.60 Collection of Wi-Fi SON statistics using nl80211 vendor command

Starting with QCA_Networking_2019.SPF.11.0 release, collection of Wi-Fi SON statistics using nl80211 vendor command is implemented for IPQ401x, IPQ806x, and IPQ807x/IPQ807xA/IPQ817x platforms.

QCA_NL80211_VENDOR_SUBCMD_SON_CLI_STATS_ALL command is added to collect SON details from umac/son. Also, ald event information is collected for SoN. By default, collection of Wi-Fi SON statistics is disabled. This is added as a module parameter, which can be enabled at runtime through UCI command. To enable the collection of Wi-Fi SON statistics, enter the `uci set wireless.qcawifi.enable_son_cli='1'` command.

The following is the list of sub commands that are introduced. These are SON specific and added in umac/son.

- QCA_NL80211_VENDOR_SON_CLI_GET_CHAN_UTIL =1,
- QCA_NL80211_VENDOR_SON_CLI_GET_RSSI = 2,
- QCA_NL80211_VENDOR_SON_CLI_GET_PEER_INFO = 3,

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

- QCA_NL80211_VENDOR_SON_CLI_GET_TX_AUTH_DENIED_INFO = 4,
- QCA_NL80211_VENDOR_SON_CLI_GET_ACTIVE_STATUS_ON_CHANGE = 5,
- QCA_NL80211_VENDOR_SON_CLI_GET_RSSI_CROSSED_THRESHOLD_INFO = 6,
- QCA_NL80211_VENDOR_SON_CLI_GET_RSSI_MEASUREMENT = 7,
- QCA_NL80211_VENDOR_SON_CLI_GET_BEACON_FRAME_INFO = 8,
- QCA_NL80211_VENDOR_SON_CLI_GET_WNM_EVENT_INFO = 9,
- QCA_NL80211_VENDOR_SON_CLI_GET_TX_RATE_CROSSED_THRESHOLD_INFO = 10,
- QCA_NL80211_VENDOR_SON_CLI_GET_TX_POWER_INFO = 11,
- QCA_NL80211_VENDOR_SON_CLI_GET_STA_STATS = 12,
- QCA_NL80211_VENDOR_SON_CLI_GET_SMPS_NODE_UPDATE = 13,
- QCA_NL80211_VENDOR_SON_CLI_GET_OPMODE_UPDATE = 14,
- QCA_NL80211_VENDOR_SON_CLI_GET_TX_RATE_MEASUREMENT_INFO = 15,
- QCA_NL80211_VENDOR_SON_CLI_GET_RSSI_THRESH_MAP_EVENT_INFO = 16,
- QCA_NL80211_VENDOR_SON_CLI_GET_ALD_STATS_INFO = 17,
- QCA_NL80211_VENDOR_SON_CLI_GET_ALD_ASSOC_INFO = 18,
- QCA_NL80211_VENDOR_SON_CLI_GET_ALD_TX_BUFFULL_INFO = 19

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:41 PDT pan.yafeng@zte.com.cn

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

8.61 UCI script changes for 6 GHz support

Radio level “band” is added to specify band to qualify “channel” field. Band value 1 for 2 GHz, 2 for 5 GHz and 3 for 6 GHz should be used. If band field is not specified, it is assumed that “channel” is from 2 GHz or 5 GHz band. Failure to set correct band will result in either auto channel being selected or previous valid channel being used.

8.61.1 Guidelines for 6 GHz out-of-band functionality

- In response to ANQP request with Neighbor report query, AP includes information of all co-located APs across all bands supported. This is because request cannot query for specific 6 GHz AP info
- Advertisement of ANQP support in beacon to be configurable and used for WFA test case only through sigma-dut/manual selection
- 20 TU Probe Response will be a full broadcast probe response (All IE included)
- All tests to be carried out in MBSSID mode enabled in 6 GHz
- TBTT offset in RNR to have accurate value only for same beacon interval case
- Few beacons in lower band may not contain RNR of co-located 6 GHz AP upon coming up
- OCT and related Out of band tunneling requirements not supported.

Command	Arguments	Description
cfg80211tool wifiX oob_enable *arg*	{1/0}	Enable/disable OOB feature for lower band APs to advertise collocated 6 GHz APs
cfg80211tool athX bcast_prbrsp_en *arg1 arg2*	arg1-{1/0} arg2 - {interval}	Enable/disable 20TU probe response with user defined period. Enabling 20TU PrbRsp disables FILS
cfg80211tool athX enable_fils *arg1 arg2*	arg1-{1/0} arg2 - {interval}	Switch back to FILS from 20TU probe response mode

To enable MBSSID, use the following commands:

```
uci set wireless.qcawifi=qcawifi
uci set wireless.qcawifi.mbss_ie_enable=1
uci commit
wifi unload
wifi load
```

Advertisement of 6 GHz AP in lower band AP is for collocated case only.

ACS support for 6 GHz

Existing algorithm is expanded to support 6G band. Existing logic uses NF, RSSI(SNR), Channel load, Channel efficiency, txpower, number of beacon in the channel to derive best channel to use. The same logic is extended to 6 GHz band.

6G is treated as separate band and whereever there is band check is expanded to 6G band

5G and 6G channels are given equal preference in case both are supported on same radio

Existing linear structures will be used to store stats. 6G channel stats will be stored with some offset which wont be colliding with 2G and 5G stats

UCI command to select only PSC channels in 6G:

uci set wireless.@wifi-iface[0].acs_6g_only_psc=1

UCI command to select only particular channels:

uci set wireless.@wifi-iface[0].acs_freq_list=freq1-freqn

8.62 Configure ACS optimization using target provided channel grading metrics

Starting with QCA_Networking_2020.SPF.11.1 release, the ACS optimization using target-provided channel grading metrics feature is implemented on IPQ401x, IPQ806x, IPQ807x/IPQ817x/IPQ807xA, and IPQ60xx/IPQ50xx and QCN61xx + IPQ50xx/QCN90xx or QCN602x + IPQ807xA/IPQ60xx/IPQ50xx platforms.

IPQ807x and later platform target SW might optionally provide channel characterization information via WMI to host at load time. This information will consist of a grade from 0-100 for each channel. The grading will be decided on a variety of PHY level factors applicable to the channel, and will give a relative indication to the host regarding expected performance degradation if the channel is used. Currently, this information will apply only if the channel is to be used as primary 20 MHz.

If this advertisement is sent by target, the host auto channel selections algorithms (ACS and ICM) will factor in this information during channel selection (in addition to other metrics already being used). This will currently apply only to selection of primary 20 MHz.

ACS algorithm has the following assumptions:

- All BSS in the channel will have fair share of the medium.
- Based on the distance from AP fair share is further effected by distance.
- Channel grade metric given by firmware is proportional to the throughput that can be achieved on the channel

Both UCI and runtime cfg80211tool commands are added to enable and disable this feature.

```
uci set wireless.wifiX. acs_chan_grade_algo =0/1 , Default value is 1
cfg80211tool wifiX acs_chan_grade_algo 0/1, Default value is 1
cfg80211tool wifiX get_acs_chan_grade_algo
```

8.62.1 ICM with target-provided channel grading

ICM algorithm has the following assumption, apart from its existing design capabilities:

Channel grade metric given by firmware is proportional to the throughput that can be achieved on the channel

The following UCI configuration is added to enable/disable use of channel grade information:

```
uci set wireless.icm.enablechangradeusage=0/1, default is 1
```

Qualcomm
Confidential – May Contain Trade Secrets
2021-05-13 03:57:44 PDT
pan.yafeng@zte.com.cn

8.63 Configure DMA buffer debug for spectral scan

Starting with QCA_Networking_2020.SPF.11.1 release, the direct DMA debug framework for spectral scan is introduced on IPQ401x, IPQ806x, IPQ807x, and IPQ60xx/IPQ50xx/QCN90xx or QCN602x + IPQ807xA/IPQ60xx/IPQ50xx platforms. Starting with the IPQ5018.ILQ.11.4 release, this feature is extended to QCN61xx chipsets attached to IPQ50xx platforms.

Compilation Feature flag:

DIRECT_BUF_RX_DEBUG

enabled by default in the unified profile, disabled by default in other profiles

INI:

poison_spectral_bufs

Enables/Disables Spectral DMA buffer debug from their creation time (Disabled by default)

A user can use the following command to enable the poisoning through INI

```
uci set wireless.qcawifi=qca-wifi
uci set wireless.qcawifi.poison_spectral_bufs=1
uci commit wireless
```

8.64 Support interoperation with dual-band and tri-band mesh nodes

Starting with QCA_Networking_2020.SPF.11.1 release, this feature is implemented on IPQ401x, IPQ806x, IPQ807x/IPQ817x/IPQ807xA, and IPQ60xx/IPQ50xx/QCN90xx or QCN602x + IPQ807xA/IPQ60xx/IPQ50xx platforms. Starting with the IPQ5018.ILQ.11.4 release, this feature is extended to QCN61xx chipsets attached to IPQ50xx platforms.

This section describes support of interoperation with dual-band and tri-band Mesh Nodes. The feature provides a mechanism support all SON features across dual-band and tri-band mesh nodes. The following core functionalities are supported:

- Support different VAPs for front-haul and back-haul on same radio.
- AP steering should be supported between mesh nodes on different channels.
- Fronthaul support across all radios.
- Credential cloning between dual band/Tri band mesh nodes.

This feature is used for the interoperation of Wi-Fi SON between Heterogeneous Mesh nodes but still provide the same level for user experience in a Homogeneous Mesh node. In a heterogeneous topology, all the following Wi-Fi SON functionalities are expected to provide the same user experience as homogeneous topology.

1. Guest network

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

2. Ethernet backhauls
3. Single/APS mode.
4. AP steering for 11k/11v.
5. Single AP band steering.

There are three main Wi-Fi SON functionalities which needs to be optimized as part to this feature to have the same user experience.

- Credential cloning across Heterogeneous Mesh nodes
- AP steering across different channels
- Best uplink selection

All existing debug options are appended with new design, and default logs will show the new additional information for wsplcd, repacd, and AP steering.

For debugging cloning related issues, enable the following logs:

```
uci set wsplcd.config.WriteDebugLogToFile=APPEND
uci set wsplcd.config.DebugLevel=DEBUG
uci commit
/etc/init.d/wsplcd restart
```

For debugging best-uplink selection related issues, enable the following logs:

Dump wireless/network/hyd config.

Dump iwconfig/ifconfig

"logread -f | grep repacd &" on repeater

For debugging steering related issues, enable the following logs:

'td s' log from all nodes

dbg level steerexec dump

dbg level steeralg dump

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:41 PDT pan.yafeng@zte.com.cn

8.65 Configure MAP 1905-only operation mode

HYD module—DispatchManager

New manager module, dispatchManager has been introduced in Hybrid daemon. This manager module implements the socket to communicate to applications running on top of HYD such as the Mesh Application Daemon. This module is enabled by default.

Logging for this module follows HYD logging conventions and the messages are logged under tag '**dispatcher'**, with DBGINFO being the default logging level.

MapPublicService

The Map Service Module of HYD has been extended to include MapPublicService to handle exposure of IEEE1905 messaging in Mesh Layer.

This provide logic to create buffers creating the data as per TLVs defined for EasyMesh (MAP) spec. These buffers can be sent to applications running on top of hyd such as Mesh Application using socket defined in dispatch manager.

The module is disabled by default. To enable it, a new config parameter, **hyd.MultiAP.EnableIEEE1905Assist**, has been added.

Logging for this module also follows HYD conventions and are logged under tag '**mesh1905'**, with DBGINFO being the default logging level.

Mesh Application Daemon

The mesh application is a new daemon that provides a test API to verify IEEE1905 TLV specific data received from HYD daemon via datagram socket communication. This daemon will be extended to handle DataElements functionality as well.

Running the Daemon

To run the mesh application daemon, the following UCI command must be given to enable the daemon in ieee1905 mode –

```
uci set mesh_app.config.EnableIeee1905=1
```

The following commands should be executed in console to start/stop/restart this daemon. If the process is killed, apart from stopping the application using below command, the mesh application daemon will get respawned automatically after 5 seconds.

- Start Application - /etc/init.d/mesh_app start
- Stop Application - /etc/init.d/mesh_app stop
- Restart Application - /etc/init.d/mesh_app restart

To run the daemon in foreground, following command should be used –

/usr/sbin/mesh_app [-m mode] – Mode for ieee1905 is '5'

8.66 UCI configuration for Wi-Fi SON static and dynamic memory computation

The UCI configuration parameters introduced for Wi-Fi SON dynamic memory computation are as follows:

Mandatory:

EnableMemDebug='1'

MemDbgReportInterval='3'

MemDbgWriteLogToFile='1'

Optional:

MemDbgFreedMemCount='1'

MemDbgAuditingOnly='0'

MemDbgDisableModule='0'

The following is the list of modules identified as memory usage area in SON Applications and tracked using this memory debug tool library:

```
/* WSPLCD */
   QCA_MOD_WSPLCD                = 0,
/* SON Libraries */
    QCA_MOD_LIBSTORAGE           = 1,
    QCA_MOD_LIBHYFICOMMON        = 2,
    QCA_MOD_LIBWPA2              = 3,
    QCA_MOD_LIBIEEE1905          = 4,
    QCA_MOD_LIBWIFISONCFG        = 5,
    QCA_MOD_LIBHYFI_BRIDGE       = 6,
/* HYD Modules */
    QCA_MOD_HYD_CORE             = 7,
    QCA_MOD_HYD_HYCTL            = 8,
    QCA_MOD_HYD_MANAGERS_MC      = 9,
    QCA_MOD_HYD_MANAGERS_PLC     = 10,
    QCA_MOD_HYD_MANAGERS_WLAN    = 11,
    QCA_MOD_HYD_MCFWDTBL_WLAN2G  = 12,
    QCA_MOD_HYD_MCFWDTBL_WLAN5G  = 13,
    QCA_MOD_HYD_PC_SERVICE       = 14,
    QCA_MOD_HYD_PS_SERVICE       = 15,
    QCA_MOD_HYD_PCW_SERVICE      = 16,
    QCA_MOD_HYD_PCE_SERVICE      = 17,
    QCA_MOD_HYD_PCP_SERVICE      = 18,
    QCA_MOD_HYD_HE_SERVICE       = 19,
    QCA_MOD_HYD_IEEE1905_SERVICE = 20,
    QCA_MOD_HYD_TDSERVICE        = 21,
    QCA_MOD_HYD_LOGSERVICE       = 22,
    QCA_MOD_HYD_WLB_BANDMONMBSA  = 23,
    QCA_MOD_HYD_WLB_MONITOR      = 24,
    QCA_MOD_HYD_WLB_STEERALGMBSA = 25,
    QCA_MOD_HYD_WLB_STEERMSG     = 26,
    QCA_MOD_HYD_WLB_WLANIF_MBSA  = 27,
```

Qualcomm Confidential - May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

```
/* LBD Modules */
    QCA_MOD_LBD_CORE                = 28,
    QCA_MOD_LBD_BANDMON             = 29,
    QCA_MOD_LBD_DIAGLOG             = 30,
    QCA_MOD_LBD_ESTIMATOR           = 31,
    QCA_MOD_LBD_PERSIST             = 32,
    QCA_MOD_LBD_STADB               = 33,
    QCA_MOD_LBD_STEERALG            = 34,
    QCA_MOD_LBD_STEEREXEC           = 35,
    QCA_MOD_LBD_WLANIF_ATH10K       = 36,
    QCA_MOD_LBD_WLANIF              = 37,
/* PLC SON */
    QCA_MOD_PLC_SON                 = 38,
/* MAP */
    QCA_MOD_MAP_SERVICE             = 39,
/* Non SON modules */
    QCA_NON_SON                     = 40,
```

EnableMemDebug option

“EnableMemDebug” option is a 4-bit Bitfield value. This is a mandatory option:

- 0—Disable the Memory debug feature
- 1—Enable the Memory debug feature and start tracking the SON application memory usage. This memory usage summary information is written to file only if the config "MemDbgWriteLogToFile" is set to 1 and report interval is configured with a non-zero value.

The following summary information is taken from HYD application at the time of boot up

```
 1 [PID:8298][Configuration Settings]:
 2 EnableMemoryDebug:[15]
 3 Only Auditing:[0]
 4 Disable Module:[0]
 5 Max Freed mem tracking:[1]
 6 WriteLogToFile:[3]
 7 Enable Filter:[0]
 8 Filter configuration memory usage: 0 total_count [0]
 9
10
11 Report Date and Time:[Wed Feb  5 23:34:13 2020
12 ]
13 Module_Name         Total_Mem  Peak       Tot_Alloc  Tot_Free   AllocEntry   AllocFail
14 LIBHYFICOMMON       17638      17638      110        1          110          0
15 LIBWIFISONCFG       208        208        4          0          4            0
16 LIBHYFI_BRIDGE      271236     271236     8          3          5            0
17 HYD_CORE            10292      14068      679        521        137          0
18 HYD_MANAGERS_WLAN   0          3072       6          7          0            0
19 HYD_PS_SERVICE      18         18         2          2          2            0
20 HYD_PCW_SERVICE     47488      47488      11         0          11           0
21 HYD_PCE_SERVICE     288        288        4          0          4            0
22 HYD_WLB_BANDMONMBSA 72         72         3          0          3            0
23 HYD_WLB_MONITOR     16         16         1          0          1            0
24 HYD_WLB_WLANIF_MBSA 16         16         1          0          1            0
25 LBD_BANDMON         168        168        1          0          1            0
26 LBD_PERSIST         4096       4096       1          0          1            0
27 LBD_STADB           14424      14424      16         0          16           0
28 LBD_STEEREXEC       672        672        1          0          1            0
29 LBD_WLANIF          3464       6536       6          6          2            0
30
31 VmPeak     VmSize     VmHWM      VmRSS      VmData     VmStk
32 20140 kB   20140 kB   17600 kB   17600 kB   14080 kB   136 kB
33
34 SON Application + SON Libraries Mem Usage = 370096 (in bytes), 361.421875 (in KB)
35 Memory Debug Tool Usage = 55637 (in bytes), 54.333008 (in KB)
36 Peak allocation: 371120 (in bytes), 362.421875 (in KB)
37
```

Advanced options: More advanced options are available to get in-depth data about the memory allocation. “EnableMemDebug” option is a 4-bit Bitfield value. Bit-0 is used to enable/disable this feature and remaining 3-bits (Bit-1 to Bit-3) are used to configure the advanced options explained below, These can be used for extensive debugging purpose.

- X X 1 1- Display content of “Allocated list”, which gives information about the memory allocation that are allocated and is in-use by the application, yet to be freed
- X 1 X 1 - Display content of “Freed List” that has last ‘N’ freed memory information. N is configured using another UCI option “MemDbgFreedMemCount”
- 1 X X 1 - Display “Filtered list” having the memory usage information.
- X – Do not care

Enable all the bits to enable this feature and display all the information (1 1 1 1) i.e. option EnableMemDebug = ‘15’

Allocated List

```
[LIBWPA2]: Displaying Memory allocation list : Last 15 Transactions
=====================================================
[1]: ptr[0x7f816feba0] size[48] alloc_func[eloop_register_timeout] alloc_line[234]
[2]: ptr[0x7f816fe840] size[48] alloc_func[eloop_register_timeout] alloc_line[234]
[3]: ptr[0x7f816fe800] size[48] alloc_func[eloop_register_timeout] alloc_line[234]
[4]: ptr[0x7f816fed00] size[48] alloc_func[eloop_register_timeout] alloc_line[234]
[5]: ptr[0x7f816fecc0] size[48] alloc_func[eloop_register_timeout] alloc_line[234]
[6]: ptr[0x7f8177e4c0] size[128] alloc_func[eloop_run] alloc_line[424]
[7]: ptr[0x7f8177e340] size[128] alloc_func[eloop_run] alloc_line[423]
[8]: ptr[0x7f8177e100] size[128] alloc_func[eloop_run] alloc_line[422]
[9]: ptr[0x7f8177ec20] size[48] alloc_func[eloop_register_timeout] alloc_line[234]
[10]: ptr[0x7f816fe3c0] size[160] alloc_func[eloop_sock_table_add_sock] alloc_line[93]
[11]: ptr[0x7f8148afa0] size[33] alloc_func[os_zalloc] alloc_line[220]
[12]: ptr[0x7f8148af60] size[33] alloc_func[os_zalloc] alloc_line[220]
[13]: ptr[0x7f814cdfa0] size[33] alloc_func[os_zalloc] alloc_line[220]
[14]: ptr[0x7f814cdf60] size[33] alloc_func[os_zalloc] alloc_line[220]
[15]: ptr[0x7f816fef80] size[65] alloc_func[os_zalloc] alloc_line[220]
=====================================================
```

Freed List

```
[LIBWPA2]: Displaying Free list : Last 1 Transactions
=====================================================
[1]: ptr[0x7f816fea80] size[48] alloc_func[eloop_register_timeout] alloc_line[234] free_func[eloop_run] free_line[468]
=====================================================
```

Filtered List

```
TOTAL FILTERED FUNCTION MEM USAGE = 208 (in bytes), 0.203125 (in KB)
Filtered Allocation Memory List : Total Entries [4]
==================================

ptr [0x7f8177e920] size [72]
ptr [0x7f816fe880] size [32]
ptr [0x55635f4f80] size [72]
ptr [0x7f815f9fc0] size [32]
```

Qualcomm Confidential – May Contain Trade Secrets 2021-05-12 09:57:44 PDT Jian. Yafeng@zte.com.cn

MemDbgReportInterval option

This option can be used to configure the required periodic interval (in seconds) to report the summary information and other required information

Example: option MemDbgReportInterval ‘10’

This configuration generates report every 10 seconds

Report has date and time information which is useful in analyzing the logs

MemDbgWriteLogToFile option

This option is used to enable/disable writing summary information log to the file. At present, logs are stored in data structure and written to the log file periodically.

- 0 - Disable writing summary information log to file
- 1 - Enable writing summary information log to file

Log file location :

/tmp/<log_file>.txt

Log file name semantics :

sonmeminfo-<app_name>-<app_config>-<process_id>.txt

app_name – wsplcd/hyd/lbd

app_config – lan/guest

process_id – Process ID of this application

Example: option MemDbgWriteLogToFile = ‘1’

```
-rw-r--r-- 1 root root 5.1M Feb  5 23:31 sonmeminfo-wsplcd-guest-3682.txt
-rw-r--r-- 1 root root 5.1M Feb  5 23:31 sonmeminfo-wsplcd-lan-3681.txt
-rw-r--r-- 1 root root 20M Feb  5 23:31 sonmeminfo-hyd-guest-4174.txt
-rw-r--r-- 1 root root 20M Feb  5 23:31 sonmeminfo-hyd-lan-4173.txt
```

Advanced options: More advanced options are available to get in-depth data about the memory allocation. “MemDbgWriteLogToFile” option is a 3-bit Bitfield value. Bit-0 is used to enable/disable this feature and remaining 2-bits (Bit-1 and Bit-2) are used to configure the advanced options explained below, These can be used for extensive debugging purpose.

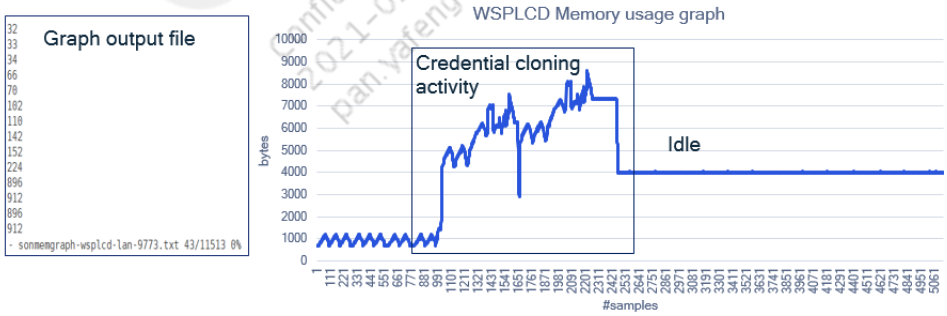
- X 1 X - Enable Graph output logging. A log file is created, and an entry is added whenever there is an increase/decrease in memory usage. This file can be used to plot the graph

Ex: option MemDbgWriteLogToFile = ‘2’

The following graph is plotted for WSPLCD application at the time of bootup in CAP and credential cloning completed one time with RE

Example :

```
-rw-r--r-- 1 root root 5.1M Feb 5 23:31 sonmeminfo-wsplcd-guest-3682.txt
-rw-r--r-- 1 root root 5.1M Feb 5 23:31 sonmeminfo-wsplcd-lan-3681.txt
-rw-r--r-- 1 root root 20M Feb 5 23:31 sonmeminfo-hyd-guest-4174.txt
-rw-r--r-- 1 root root 20M Feb 5 23:31 sonmeminfo-hyd-lan-4173.txt
```



MemDbgFreedMemCount option

MemDbgFreedMemCount- This option is used to configure the maximum number of freed information that can be stored by the memory debug tool

- 0 - Store all the freed information
- <N> - Store the last N freed memory information.

This information is displayed in output log file, only if Bit-2 is set in “EnableMemDebug” option i.e. “X1X1”

Note: By default, this option is set to '0', so none of the freed memory information is stored. Configure this parameter with non-zero value to store last N freed memory information.

The following configurations are the minimal mandatory configuration required to enable the basic operation i.e.

1. Enable memory debug feature

2. Configure periodic interval

3. Enable writing summary information to log file

Example:

```
EnableMemDebug='1'
MemDbgReportInterval='10'
MemDbgWriteLogToFile='1'
```

The following are the optional configuration used for extensive debugging,

```
MemDbgFreedMemCount='1'
MemDbgAuditingOnly='0'
MemDbgDisableModule='0'
```

MemDbgAuditingOnly option

This option is used to enable storing only the minimal information about the memory usage. Only memory allocation size is stored for all the modules. This gives us the total memory usage of the application as a whole.

This details about this memory allocation such as pointer and size is stored in “Filter list” and displayed only if EnableMemDebug option is configured to display the “Filter List”, that is, Bit-4 in EnableMemDebug option is set (1 X X 1).

Advanced options present in “EnableMemDebug” option to display “Allocated List/ Freed List” (Bit-1 and Bit-2) is Invalid with this option “MemDbgAuditionOnly”

MemDbgFreedMemCount option is also Invalid with this option. Meta data of Freed memory information is removed immediately.

MemDbgWriteLogToFile option can be used along with EnableMemDebug to get graph output and debug output

Configuration to Enable “Auditing only” feature,

Example:

```
EnableMemDebug='1'// Only Summary information is displayed (Filter list
can be enabled if needed)
EnableMemDebug = '1'
MemDbgReportInterval='10'// Report for every 10 seconds
MemDbgWriteLogToFile='1'// only summary is displayed (Graph/Dump can be
enabled if needed)
```

The following AuditingOnly information is taken from HYD application at the time of boot up and kept in idle state:

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
11874][Configuration Settings]:
eMemoryDebug:[1]
Auditing:[1]
le Module:[0]
reed mem tracking:[0]
LogToFile:[3]
e Filter:[0]
r configuration memory usage: 0 total_count [0]
y Usage Information (Audit Only Output: SON Application + Libraries Mem Usage)
==============================================================================
_count   Mem_Usage     MemToolUsage  Total         VmPeak   VmSize   VmHWM    VmRSS    VmData   VmStk
         376976        7485          384461        20096 kB   20096 kB   17628 kB   17628 kB   14036 kB   136
         404384        8277          412661        20128 kB   20128 kB   17628 kB   17628 kB   14068 kB   136
         414016        8613          422629        20136 kB   20136 kB   17628 kB   17628 kB   14076 kB   136
         418832        8781          427613        20144 kB   20144 kB   17628 kB   17628 kB   14084 kB   136
         428464        9117          437581        20152 kB   20152 kB   17628 kB   17628 kB   14092 kB   136
         433968        9309          443277        20160 kB   20160 kB   17628 kB   17628 kB   14100 kB   136
         442912        9621          452533        20168 kB   20168 kB   17628 kB   17628 kB   14108 kB   136
         453232        9981          463213        20176 kB   20176 kB   17628 kB   17628 kB   14116 kB   136
         458736        10173         468909        20192 kB   20192 kB   17628 kB   17628 kB   14132 kB   136
         462176        10293         472469        20192 kB   20192 kB   17772 kB   17772 kB   14132 kB   136
         469056        10533         479589        20200 kB   20200 kB   17772 kB   17772 kB   14140 kB   136
         474560        10725         485285        20208 kB   20208 kB   17772 kB   17772 kB   14148 kB   136
         478000        10845         488845        20208 kB   20208 kB   17772 kB   17772 kB   14148 kB   136
         484192        11061         495253        20216 kB   20216 kB   17772 kB   17772 kB   14156 kB   136
         489008        11229         500237        20224 kB   20224 kB   17772 kB   17772 kB   14164 kB   136
         495888        11469         507357        20232 kB   20232 kB   17772 kB   17772 kB   14172 kB   136
         500016        11613         511629        20232 kB   20232 kB   17772 kB   17772 kB   14172 kB   136
         506208        11829         518037        20240 kB   20240 kB   17772 kB   17772 kB   14180 kB   136
         511712        12021         523733        20248 kB   20248 kB   17772 kB   17772 kB   14188 kB   136
         517216        12213         529429        20248 kB   20248 kB   17828 kB   17828 kB   14188 kB   136
         517216        12213         529429        20256 kB   20256 kB   17828 kB   17828 kB   14196 kB   136
         517216        12213         529429        20256 kB   20256 kB   17828 kB   17828 kB   14196 kB   136
         517904        12237         530141        20256 kB   20256 kB   17828 kB   17828 kB   14196 kB   136
         517904        12237         530141        20256 kB   20256 kB   17828 kB   17828 kB   14196 kB   136
         518592        12261         530853        20256 kB   20256 kB   17828 kB   17828 kB   14196 kB   136
         519968        12309         532277        20256 kB   20256 kB   17828 kB   17828 kB   14196 kB   136
         522032        12381         534413        20256 kB   20256 kB   17828 kB   17828 kB   14196 kB   136
         524096        12453         536549        20256 kB   20256 kB   17828 kB   17828 kB   14196 kB   136
         524784        12477         537261        20264 kB   20264 kB   17828 kB   17828 kB   14204 kB   136
```

MemDbgDisableModule option

This option is used to disable selected modules from tracking. This help saving space and track only required modules.

Disabled modules are tracked with minimal information whereas other modules are tracked with detailed information.

This minimal information is stored in “Filter List”

This is a Bit-Field of 40 Bits. Each bit corresponds to one module. Module list is available in the *Libraries tracked using the memory debug tool* section.

Example: To disable LIBSTORAGE (module 1) and LIBWIFISONCFG (module 5)

option MemDbgDisableModule ‘34’// …0 0 1 0 0 0 1 0

Note: “MemDbgDisableModule” is Invalid, if “MemDbgAuditionOnly” is enabled

8.66.1 Enable and configure memory debug feature

The following are the UCI configuration commands for enabling and setting up memory debug feature for hyd, lbd, and wsplcd. The UCI configuration enables the basic memory tracking operation with periodic report interval set as 60 seconds. This writes memory summary information and Graph information to log files.

```
[hyd]
uci set hyd.Debug.EnableMemDebug='1'
uci set hyd.Debug.MemDbgReportInterval='60'
uci set hyd.Debug.MemDbgWriteLogToFile='3'
uci commit hyd

#[lbd]
uci set lbd.Debug.EnableMemDebug='1'
uci set lbd.Debug.MemDbgReportInterval='60'
uci set lbd.Debug.MemDbgWriteLogToFile='3'
uci commit lbd

#[wsplcd]
uci set wsplcd.config.EnableMemDebug='1'
uci set wsplcd.config.MemDbgReportInterval='60'
uci set wsplcd.config.MemDbgWriteLogToFile='3'
uci commit wsplcd

/etc/init.d/repacd restart
```

8.67 UCI configuration of Easy Mesh R2 commercialization algorithms

Starting with QCA_Networking_2020.SPF.11.1 release, Easy Mesh R2 commercialization algorithms are supported on IPQ401x, IPQ806x, IPQ807x/IPQ817x/IPQ807xA, and IPQ60xx/IPQ50xx platforms. These include traffic separation, channel scan, WPA3, and coordinated CAC.

Traffic separation

Traffic separation feature is about logically separating out traffic based on VLANs . As part of EasyMesh R2 specification, messages are defined to achieve this separation.

There is a limitation in specification to tell the controller about r1 and r2 Association DisAllowed BSS on an Agent. This is raised to task group and will be corrected in Easy Mesh r3. This might impact support of r1 and r2 devices in the same network.

Channel scan

Channel scan verifies if an Easy Mesh Agent responds to a channel scan request message from Controller with a channel scan report message containing a list of channel scan result TLVs that were scanned upon receiving the request. WFA certification for channel scan is supported.

WPA3 support

WPA3 protocol enables the high security between the AP interface and STA interface to protect the data communication. Multi-AP Specification defines the SAE authentication configuration for fronthaul and backhaul BSSes. Fronthaul BSSes will be always configured with mixed mode, that is, WPA2 and SAE. MAP-R1 agent can not onboard with MAP R2 agent when the backhaul has SAE only authentication.

Coordinated CAC support

Coordinated CAC support enables the controller to choose the DFS channel effectively based on its channel condition as part of channel selection. Qualcomm controller will send the Coordinated CAC Request to agents when the channel selection request selects the DFS channel as best to operate. Qualcomm controller sets the channel preference reason code based on CAC result that is received from the agent. Qualcomm controller always choose the leaf agent to perform the CAC request to avoid topology disturbance.

8.67.1 Traffic separation configuration

CONFIG	Description
repacd.MAPConfig.MapTrafficSeparationEnable	Flag to enable or disable traffic separation
repacd.MAPConfig.NumberOfVLANSupported	Number of VLAN supported by the device
repacd.MAPConfig.VlanNetworkPrimary repacd.MAPConfig.VlanNetworkOne repacd.MAPConfig.VlanNetworkTwo repacd.MAPConfig.VlanNetworkThree	Names for the different bridges that needs to be created. By default they are ‘lan’, ‘lan1’, ‘lan2’, ‘lan3’ respectively Ex: uci set repacd.MAPConfig.VlanNetworkPrimary=lan
repacd.MAPConfig.VlanNetworkBackHaul	Name of the backhaul network . Name ‘backhaul’ is default
repacd.MAPConfig.BridgeProtoNwOne repacd.MAPConfig.BridgeIPAddressNwOne repacd.MAPConfig.BridgeProtoNwTwo repacd.MAPConfig.BridgeIPAddressNwTwo repacd.MAPConfig.BridgeProtoNwThree repacd.MAPConfig.BridgeIPAddressNwThree	Speficy the bridge protocol as ‘static’ or ‘dhcp’ If ‘static’ specify the IP address . Ex. uci set repacd.MAPConfig.BridgeProtoNwOne='static' uci set repacd.MAPConfig.BridgeIPAddressNwOne='192.168.1.201'
repacd.MAPConfig.EnableMapTSLogs	Flag to enable traffic separation Logs

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 08:57:41 PDT pan.yafeng@qualcomm.com

repacd.MAPConfig.FronthaulSSID repacd.MAPConfig.FronthaulKey repacd.MAPConfig.FronthaulSSIDNwOne repacd.MAPConfig.FronthaulKeyNwOne repacd.MAPConfig.FronthaulSSIDNwTwo repacd.MAPConfig.FronthaulKeyNwTwo repacd.MAPConfig.FronthaulSSIDNwThree repacd.MAPConfig.FronthaulKeyNwThree	The SSID and key for each network. Will be used to fill Traffic Separation TLV Ex. uci set repacd.MAPConfig.FronthaulSSID='map-tst-fbss' uci set repacd.MAPConfig.FronthaulKey='map-fbss'
repacd.MAPConfig.VlanIDNwPrimary repacd.MAPConfig.VlanIDNwOne repacd.MAPConfig.VlanIDNwTwo repacd.MAPConfig.VlanIDNwThree	The VLAN that needs to be mapped to corresponding SSID. Will be used to fill Traffic Separation TLV Ex. uci set repacd.MAPConfig.VlanIDNwPrimary=10
repacd.MAPConfig.Map8021QPCP	PCP value for 802.1Q TLV
repacd.MAPConfig.R1AgentAssocDisAllowed	If set to 1 then Single Backhaul is created on each Radio with r1 Agent Assoc DisAllowed
repacd.MAPConfig.R2AgentAssocDisAllowed	If set to 1 then Single Backhaul is created on each Radio with r2 Agent Assoc DisAllowed
repacd.MAPConfig.CombinedR1R2Backhaul	Combined Backhaul is not supported . Is Set to 0. Always create 2 backhauls . 1 for r1 Agent Assoc and 1 for r2 Agent Assoc
repacd.MAPConfig.TSUseSameBridgeMAC	Use the same MAC for all the bridges that are created

8.67.2 Channel scan configuration

Config	Description
hyd.MultiAP.EnableBootOnlyScan	Config to enable Boot only scan on EasyMesh R2 Agent
hyd.MultiAP.ChanScanIntervalMin	The minimum time between 2 consequtive channel scans on Easy Mesh R2 Agent in minutes
hyd.MultiAP.EnableChannelScanRequest	If enabled Controller sends Channel Scan Request to all the Easy Mesh R2 Agents in the
hyd.MultiAP.EnableRptIndependentScan	If enabled Controller sends policy config to Easy Mesh R2 Agent to report Independent Channel Scan

8.67.3 WPA3 support configuration

Config	Description

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

uci set repacd.MAPConfig.FronthaulAuthType uci set repacd.MAPConfig.FronthaulAuthTypeNwOne uci set repacd.MAPConfig.FronthaulAuthTypeNwTwo uci set repacd.MAPConfig.FronthaulAuthTypeNwThree	Authentication Type for each network. NwOne,NwTwo,NwThree will be used for Traffic separation. Eg: uci set repacd.MAPConfig.FronthaulAuthType = 1 to enable SAE authentication for Fronthaul network.
uci set repacd.MAPConfig.BackhaulAuthType	Authentication Type used for backhaul link. 0 – WPA2 authentication 1- SAE Authentication

8.67.4 Coordinated CAC configuration

Config	Description
uci set hyd.MultiAP.UseCACResultForChanSelection	If set to 1 then controller will trigger the Coordincated CAC and includes the CAC results for channel selection before applying the channel to agents.

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

8.68 Configure single AP and multi-AP mode in data elements

Data elements supports the single AP mode and multi-AP modes of operations.

8.68.0.1 Configure single AP mode

Do the following to configure Single AP mode in Data Elements:

1. Ensure that Wifi is up and all the VAPs are beaconing
2. Configure mesh_app with below configuration,
   a. uci set mesh_app.config.EnableDataElements='1'
   b. uci set mesh_app.DataElements.DEReportingInteval=’600’

      The above parameter controls the periodicity of stats collection, by default the value is set as 600 seconds (10 minutes). It is recommended to use 10 minutes for long runs, in case of short runs we can use 300 seconds (5 minutes) as interval.
   c. uci commit mesh_app
3. Start mesh_app with the following command:
   a. /etc/init.d/mesh_app start
4. Start mesh_app terminal “mat” to enable the log with “dbg here”
5. After the set interval time is crossed, data Elements will start collecting logs. The logs will be generated after seeing the following print:

   “dEDumpJsonObject: json write successful”

8.68.0.2 Configure multi-AP mode

Do the following to configure multi-AP mode in Data Elements:

1. Ensure the setup is configured and the agent have onboarded before starting data elements.
2. Configure mesh_app with below configuration,
   a. uci set mesh_app.config.EnableDataElements='1'
   b. uci set mesh_app.DataElements.DEReportingInteval=’600’

      The preceding parameter controls the periodicity of stats collection, by default the value is set as 600 seconds (10 minutes). It is recommended to use 10 minutes for long runs, in case of short runs we can use 300 seconds (5 minutes) as interval.
   c. uci commit mesh_app
3. Start mesh_app with the following command:
      i. /etc/init.d/mesh_app start
4. Start mesh_app terminal “mat” to enable the log with “dbg here”

5. After the set interval time is crossed, data Elements will start collecting logs. The logs will be generated after seeing the following print:

   "dEDumpJsonObject: json write successful"

8.68.0.3 Query on-demand statistics

DE supports on demand stats request from external client. The DE listens to the localhost (127.0.0.8) on port 8090 for request. We have developed a de_nb_client that will connect to the above port to send the request to start collecting stats on demand.

Use the following command to start stats collection on demand:

de_nb_client getStats

An interval of 5 minutes is recommended between consecutive requests.

8.68.0.4 Configure Northbound events

Do the following to configure northbound events listed in the **North bound Event Notification Support** section in Data Elements in multi-AP mode:

1. Ensure the setup is configured and the agent have onboarded before starting data elements.
2. Configure mesh_app with below configuration,
   a. uci set mesh_app.config.EnableDataElements='1'
   b. uci set mesh_app.DataElements.DEReportingInteval='600'

      The preceding parameter controls the periodicity of stats collection, by default the value is set as 600 seconds (10 minutes). It is recommended to use 10 minutes for long runs, in case of short runs we can use 300 seconds (5 minutes) as interval.
   c. uci set mesh_app.DataElements.NBEventEnable='1'
   d. uci commit mesh_app
3. Start mesh_app with the below command,
   a. /etc/init.d/mesh_app start
4. Start mesh_app terminal "mat" to enable the log with "dbg here"
5. Prints will be generated in the "mat" terminal under "info" tag when events are generated in the hyd.

8.68.1 UCI configuration of data elements

The following UCI configurations are defined for Data Elements

- mesh_app.config.EnableDataElements='1': *Used to enable data element support*
- mesh_app.DataElements.DEReportingInteval='600': *Timer Interval used to fetch and update JSON file*

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

- mesh_app.DataElements.DEEnableBase64Encoding='1' : *To Enable Base 64 Encoding on Object Values (Required for certification)*

The remaining UCI are for including the Data Parameters in the JSON file

- mesh_app.DataElements.DERadioJsonEnable=1 : *Enables Radio Element*
- mesh_app.DataElements.DERadioCapsJsonEnable=1 : *Enables Radio Capability Element*
- mesh_app.DataElements.DERadioBssListJsonEnable=1 : *Enables Radio BSS Element*
- mesh_app.DataElements.DERadioBkStaJsonEnable=1 : *Enables Backhaul STA Element*
- mesh_app.DataElements.DERadioScanResultJsonEnable=1 : *Enables Scan Result Element*
- mesh_app.DataElements.DERadioUnAssocStaJsonEnable=1 : *Enables Unassoc STA Element*
- mesh_app.DataElements.DERadioCurOpClassJsonEnable=1 : *Enables Current opclass Element*
- mesh_app.DataElements.DEBssSTAListJsonEnable=1 : *Enables BSS STA Element*

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

8.69 Configure interoperation with dual-band and tri-band mesh nodes in Wi-Fi SON

repacd parameters

- *uci set repacd.*repacd.EnableBest5gBackhaulSelection='0'—By default, this parameter is set to 0 for a tri-radio repeater with three STA VAPs. A customer can enable this flag to run best uplink on two 5 GHz STA VAPs' to determine the best node. Enabling this flag will delay the stabilization time after first boot.
- uci set repacd.WiFiLink.MinBestUplinkRetry='5'—By default, this parameter is set to 5. For DFS channels, increase the retry count to 10. This will increase the retry timer and might delay the stabilization process. This retry count indicates the number of times connection over each 5G station interface should be attempted.

wsplcd parameters

- *uci set wsplcd.*config.DeepClone_SON='1'—This flag needs to be enabled always for SON mode, and, by default, it is enabled. This will pass the current operating channel as low or high to wsplcd.
- uci set wsplcd.config.Prefered5GLChannel='40' and
  uci set wsplcd.config.Prefered5GHChannel='149'—These two flags are the required mandatory flags to populate user configured channels whenever the DUT does not support both 5 GHz fronthaul bands simultaneously.

lbd parameters

- *uci set* lbd.config.EnableMulti11kRequest='1'—This flag needs to be enabled for interoperation of dual-band tri-band mesh modes in Wi-Fi SON environment. It is disabled for homogeneous MAP. This flag allows to determine if we have multiple 5 GHz fronthaul channels.
- uci set lbd.Estimator_Adv.Delayed11kRequesttimer='1'—Most of the 11k clients will reject back to back 11k request s. It is necessary to wait for a short period of time between two 11k requests. Currently, this delay is configured to be 1 second. However, in a noisy environment, this delay can be increased to 2. Only 1 and 2 are recommended for steering to work effectively.

8.70 Configure RCPI support for Wi-Fi SON networks

Starting with QCA_Networking_2020.SPF.11.2 release, a mechanism to encode client Received channel power indicator (RCPI) measurements, that is, save RCPI type per client for SON, is introduced. RCPI value is received as 2's complement and 11K spec compliant. Starting with QCA_Networking_2020.SPF.11.2 release, this feature is implemented on IPQ401x, IPQ806x, IPQ807x/IPQ817x/IPQ807xA, and IPQ60xx platforms. Starting with the IPQ5018.ILQ.11.3 release, this functionality is supported on IPQ50xx platforms.

When a locally connected client becomes eligible for steering (that is, RSSI falls below certain threshold as notified by driver), HYD marks the client eligible for steering and request 11k.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

11k response from client contains RCPI and helps HYD find suitable candidate AP for this client. RCPI field is a 8-bit field in 11k response and is used to find potential candidate APs where a client can be steered. RCPI field tells how good client can see these other APs. Candidate AP needs to be better than current AP by certain threshold, then only we steer this client.

RCPI is a 8-bit value in the packet, sent as HEX value. 802.11 standard states to send RCPI between 0 to 255 i.e. positive value, but not many clients follow spec and send RCPI value as 2's compliment which represents negative value. When it's used by client to send positive value (as per spec), then it can be from 0 to 255 value. When it is used by client to send negative dBm value, value can be from -127 to +127.

- When multiple beacons reports are received in a 11k Response from client, RcpiType is set to 1 (i.e. 11k complaint client) only if client sends RCPI as per 11k spec in at least one of these beacon reports.
- This feature is only supported for SON.
- RcpiType persistency will be maintained only when persist feature is enabled.
- RcpiType is updated to all Mesh nodes with this TLV, (IEEE1905_QCA_TYPE_ ASSOCIATED_STATIONS_WITH_CLASS).
- Customer has to apply persistence, lbd config in all nodes in Mesh, if stadb info of all clients need to be persistent, across hyd restarts and reboots.
- Because RcpiType is updated to stadb, it is necessary to enable lbd persistent feature as follows.

```
uci set lbd.Persist.PersistFile='/etc/persist/Persist.dat'
uci set lbd.Persist.PersistPeriod=10
uci commit
lbd
/etc/init.d/hyd restart
```

8.71 Configure advanced CAC implementation for MAP r1

Advanced CAC implementation for MAP r1 is supported, starting with QCA_networking_ 2020.SPF.11.2 release on IPQ401x, IPQ806x, PQ807x/IPQ817x/IPQ807xA, and IPQ60xx/IPQ50xx platforms. The Controller is informed whenever there is a radar detection within the mesh so that it can take appropriate action in selecting another appropriate fronthaul or backhaul channel for the mesh. Wireless backhaul connectivity is maintained during radar detection on DFS channel.

Before the implementation of this feature, if radar is detected on fronthaul of AP, driver will send CSA downstream to other nodes within the mesh and change the channel to a random channel. If radar is detected on backhaul, driver level RCSA is sent for backhaul STA to reverse notify backhaul BSS of channel change.

This feature adds support:

1. To provide config methods to set fallback frequency.
2. To detect radar event in easy mesh and generate 1905 messages and send to controller as per MAP specification.

3. For controller to take necessary steps to ensure all Agents in mesh are on the fallback frequency channel.
4. To work in co-existence with existing CSA and RCSA.

8.71.1 Configuration of fallback frequency

The fallback frequency must be configured separately for the driver and HYD.

The options to configure fallback frequency for driver are:

- cfg80211tool wifix setNxtRadarFreq <freq_value>
- uci set wireless.wifix.setNxtRadarFreq=<freq_value>

The options to configure fallback frequency for Hyd are:

fallback_freq_5GH is the fallback frequency to be set for 5G high radio.

fallback_freq_5GL is the fallback frequency to be set for 5G low radio.

Either, configure the following two commands in conjunction:

- uci set hyd.MultiAP.fallback_freq_5GH=<freq_value> AND
- uci set hyd.MultiAP.fallback_freq_5GL=<freq_value>

Or, the following two commands in conjunction:

- option fallback_freq_5GH 'freq_value' AND
- option fallback_freq_5GL ‘freq_value’

It is important to note that setNxtRadarFreq for driver and fallback_freq in HYD must be the same and configured on both.

8.72 UCI support to configure individual VAPs independently

Starting with QCA_Networking_2020.SPF.11.2 release, this feature is introduced on IPQ40xx, IPQ806x, IPQ807x/IPQ807xA/IPQ817x, IPQ60xx/IPQ50xx, and QCN90xx or QCN602x platforms. Starting with the IPQ5018.ILQ.11.4 release, this feature is extended to QCN61xx chipsets attached to IPQ50xx platforms. The capability to add individual VAP configuration independently is introduced. If a change in configuration applies to only one VAP, only that VAP is impacted.

If a VAP param impacts a radio param (or other VAPs), then that parameter is used only on a specific VAP, called primary VAP, that is, global params of primary VAP alone will be considered. Any one VAP can be designated as a primary VAP.

Existing wifi script functionality is modified such that radio specific and VAP specific configurations can be separated and applied accordingly as per the user requirement.

The UCI commands of WLAN QSDK will apply the configuration to all the VAPs at the same time. Even if there is a change in configuration of only one VAP, all VAPs are destroyed and created for configuration.

Instead of such a functionality, individual VAP configuration is added independently so that if there is a change in configuration on one VAP, only that VAP is affected.

It is assumed that if there is a VAP param, which impacts a radio param (or other VAPs) then that should be used only on a specific VAP, called primary VAP (global params of primary VAP alone will be honored). Any one VAP can be designated as a primary VAP.

This feature is applicable only for cfg80211 configuration. This is not applicable for qcawifi.sh script.

Existing wifi scripts and qcawificfg80211.sh scripts already have support for radio specific configuration

In wifi script, support is available for accepting radio interface name and same is passed to qcawificfg80211.sh script. The wifi script is modified to accept one more argument from CLI and pass it to qcawificfg80211.sh script.

In qcawificfg80211.sh script, currently, it applies configuration by iterating through all VAPs of that radio. To apply configuration to specific VAP, first, segregate radio specific and VAP specific functionality and define different function for VAP.

```
enable_vifs_qcawificfg80211() {
…
}
```

The enable_vifs_qcawificfg80211() is called from enable_qcawificfg80211() conditionally in two places

If user expects configuration to apply for specific VAP, then they need to pass interface name to wifi scripts. In that case it is called immediately after entering enable_qcawificfg80211() and exits so that radio specific configuration can be skipped

If user did not pass VAP interface name, then this function is called from enable_ qcawificfg80211() after applying radio specific configurations.

For disabling case also, if VAP interface name is passed, conditionally apply disable commands for that interface only.

8.73 Configure zero BSS credential MAP agent onboarding support

Starting with QCA_Networking_2020.SPF.11.2 release, zero BSS credential AP Extender onboarding is supported on IPQ807x/IPQ807xA platforms.

The following scenarios and factors are considered while designing this feature.

- After agent onboarding, bring down all BSS in 2G on agent side via controller bss-policy
- After agent onboarding, take up all BSS in 2G on agent side via controller bss-policy

- After agent onboarding, bring down all BSS in 5G Low on agent side via controller bss-policy
- After agent onboarding, take up all BSS in 5G Low on agent side via controller bss-policy
- After agent onboarding, bring down all BSS in both 2G and 5G Low on agent side via controller bss-policy
- After agent onboarding, take up all BSS in both 2G and 5G Low on agent side via controller bss-policy
- Ensure hyd and wsplcd daemons are running all cases and td output are converged properly

This feature can be enabled or disabled in hyd uci configuration by setting to 1 or 0 respectively.

```
uci set hyd.MultiAP.EnableZeroBss='1'
```

After tear down of all AP VAPs in a radio and when there are no BSSs after tearing down in a radio (say 2G or 5G Low), there is no way for getting ap radio basic capabilities from underlying layer. To handle the situation of when there are no BSSs in a radio, the default values of ap radio basic capabilities TLV for our device can be retrieved from uci configuration if it is already configured for each radio.

In this case, for IPQ807x 3-radio board, the default values of ap radio basic capability for 5G Low and 2G are configured in both wsplcd and hyd uci config files, which are read and used by wsplcd and hyd daemons respectively. The format of AP Radio Basic Capabilities TLV can be found in Multi-AP technical specification document.

For wsplcd component, the basic radio capability is read from wslpcd uci configuration file as shown in following format in hexadecimal values.

```
# Byte 1: Maximum BSS support (0x02)
# Byte 2: Operating class count (0x07)
# Byte 3: Operating class (0x73)
# Byte 4: Maximum transmit power (0xe2)
# Byte 5: Number of non-operable channels (0x00)
# Refer AP Radio Basic Capabilities TLV format
```

For example.

```
option 5GBasic '02 07 73 e2 00 74 e2 00 75 e2 00 76 e2 00 77 e2 00 78 e2
00 80 e2 04 6a 7a 8a 9b'
option 2GBasic '02 03 51 e2 02 0c 0d 53 e2 00 54 e2 02 0c 0d'
```

For hyd component, the same preceding AP radio basic capability TLV values can be used in configuration and can be read from BasicCap5G and BasicCap2G options from /etc/config/hyd file.

```
option BasicCap5G '02 07 73 e2 00 74 e2 00 75 e2 00 76 e2 00 77 e2 00 78
e2 00 80 e2 04 6a 7a 8a 9b'
option BasicCap2G '02 03 51 e2 02 0c 0d 53 e2 00 54 e2 02 0c 0d'
```

The reason for keeping this capability value in UCI is that, there is no way to retrieve this value at runtime from the underlying layer, whenever is needed, as there are no VAPs present in zero BSS case.

The AP radio basic capability is needed by the controller to determine the agent and its details such as the number of radios supported by agent, Maximum transmit power by radio, maximum number

of BSS supported by each radio, operating class of each radio and non-operable channels list, which comes as part of M1 from agent.

On extender node, the hyd and wsplcd configuration files contain ap radio basic capability for the HK3R SBS (RDP385) platforms as default values since this feature has been primarily developed for HK 3R. Also, ensure that the AP radio basic capability has been updated in UCI config files with new values if this feature needs to be enabled in platforms other than HK3R platform.

However, this feature is not supported in other platforms except IPQ807x/IPQ807xA three-radio SBS platforms.

An important point to note is that, the feature is spread in both components hyd and wsplcd and feature can be enabled by setting global UCI option EnableZeroBss in hyd config. Both hyd and wsplcd daemons should be restarted if zero BSS feature is enabled or disabled at runtime dynamically.

```
uci set hyd.MultiAP.EnableZeroBss=1 // 1 - enable,  0 - disable
uci commit hyd
/etc/init.d/hyd restart
/etc/init.d/wsplcd restart
```

8.74 Enhancements to NSS offloading statistics

Starting with QCA_Networking_2020.SPF.11.3 release, this feature is introduced on IPQ40xx, IPQ806x, IPQ807x/IPQ807xA/IPQ817x, IPQ60xx/IPQ50xx, and QCN90xx or QCN602x platforms. NSS Wi-Fi offload supports the following statistics as part of this enhancement. Types of supported statistics are as follows:

- Per client (MAC address)
- Per direction (uplink/downlink)
- Per access category

1. Number of packets (MSDUs) and bytes, successful and with errors – Tx: tx_success, tx_failed and Rx: to_stack, rx_failed
2. Average physical-layer data rate
3. Average latency
4. RSSI and SNR
5. RSSI and chan_nf
6. Number of retries per MSDU – Tx: tx_retry, Rx: rx_retry.
7. Number of bytes, tried, successful, with errors – Tx: ucast, bcast, tx_failed and Rx: unicast, bcast, rx_failed
8. Number of PPDUs, successful, with errors and number of retries
9. Histogram of MCS, spatial streams

Starting with QCA_Networking_2020.SPF.11.3 release, this feature is introduced on IPQ807x/IPQ807xA/IPQ817x, IPQ50xx, IPQ60xx, and QCN90xx/QCN602x platforms. NSS Wi-Fi offload supports the following statistics as part of this enhancement.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Per client per TID latency histogram for the following delays are supported for display in statistical counters:

- Software Enqueue delay
- HW Tx completion delay
- HW reap to stack indication delay

PPDU histogram is not supported because the PPDU indication always received in host even in NSS offload enabled mode as part of monitor status ring. In offload mode, as packets are fast forward, the “HW reap to stack indication delay” is not computed.

The feature is enabled based on the module parameter. The module param values can be changed by the global INI value. The feature is disabled by default with INI value zero. Customer can change the value by 1 to enable the feature. The feature is supported in both offload and non-offload mode. This feature is supported in all memory profiles.

Use the following UCI command to enable or disable the feature.

```
uci set wireless.qcawifi=qcawifi
uci set wireless.qcawifi.peer_ext_stats=1
uci commit
```

After this feature is enabled, the status can be displayed by the command “iwpriv wifi0 fc_peer_ stats <peer_macID>.

8.75 Overriding the default driver behavior for 6 GHz-only AP

In 6 GHz frames RNR advertisement in “6Ghz-only AP” case in bcn/prb/fils frames. This advertisement to be selective advertisement of non Tx VAPs. Only those non Tx APs not in the MBSS IE of that frame to be present in RNR IE.

This default behavior can be overridden by enabling “rnr_6ghz_driver_override=1” in ini/global.ini. Alternatively, use the following in /etc/config/wireless:

```
config qcawifi 'qcawifi’
   option rnr_6ghz_driver_override ‘1’
   wifi unload
   wifi load
```

This config will enable RNR IE advertisement in “Non 6Ghz-only AP” case.

Selective inclusion of Non Tx APs can be enabled/disabled in either of the following ways:

- Changing the ini/global.ini config “rnr_selective_nontx_add=1” using:

  ```
  config qcawifi 'qcawifi’
  option rnr_selective_nontx_add ‘0’
  ```

  in etc/config/wireless followed by

  ```
  wifi unload; wifi load
  ```

- cfg80211tool wifiX selective_rnr_nontx 1/0

If selective inclusion is disabled, RNR IE in this 6 GHz frame to have all Non Tx AP info

8.76 Configure EMA AP using .ini and UCI settings

Users can configure EMA AP using .ini settings and UCI parameters.

INI settings for EMA AP

Table 8-13 INI settings for EMA AP

INI	Description	Example
mbss_ie_enable	A 32 bit bitmask, the lower 16 bit of which specifies the pdev for which the feature needs to be enabled. Bit corresponding to 6Ghz radio is reserved as mbssid is mandatory for 6Ghz. Use the 16 msb of this bitmask as per-pdev ema_ap feature enable/disable bits usage: uci set wireless.qcawifi=qcawifi uci set wireless.qcawifi.mbss_ie_enable=0x00000001 uci commit This INI is only applicable to lower band.	0x00000005 enables EMA mode for pdevs 1 and 3 in all socs 0x00050005 enables mbssid (disables EMA mode) for pdevs 1 and 3 in all socs
ema_ap_vendor_ ie_size_low	A 32-bit configuration param. Each 4-bit value indicates size of vendor_ie in a non-Tx profile such that if the 4-bit value is n (decimal) then the size of vendor_ie per non_tx profile is 2^n, where 0 < n < 8. 4 bits starting at bit position 4* (idx -1) will determine the reserved vendor ie size of VAP with bss-index idx. LSB 4 bits will correspond to VAP with bss-index 1 and MSB 4 bit to that with bss-index 8 usage: uci set wireless.qcawifi=qcawifi uci set wireless.qcawifi. ema_ap_vendor_ie_size_low =0x55556666 uci commit This INI is only applicable to lower and higher bands.	0x55556666 reserves 2^6 = 64 bytes for the non-Tx vaps with bss-idxes 1-4 and 2^5 = 32 bytes for the non-Tx vaps with bss-idxes 5-8
ema_ap_vendor_ ie_size_high	A 32-bit configuration param with same semantics above but correspond to VAPs with bss-indices 8 to 15. MSB 4 bits correspond to Tx-VAP and bears no significance (don't care) usage: uci set wireless.qcawifi=qcawifi uci set wireless.qcawifi. ema_ap_vendor_ie_size_high =0x5555555 uci commit This INI is only applicable to lower and higher bands.	0x5555555 reserves 2^5 = 32 bytes for the non-Tx vaps with bss-idxes 9-15

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 8-13 INI settings for EMA AP (cont.)

INI	Description	Example
beacon_common_part_size_for_ema	Size of common part in beacon usage: uci set wireless.qcawifi=qcawifi uci set wireless.qcawifi. ema_ap_beacon_common_part_size = [200, 600] uci commit This INI is only applicable to lower and higher bands.	Limit is 200-685 bytes. Default value is 600. Note: More useful for customer as common part of the beacon can be less than 600. QCA internal 600 may be required considering 1000 clients and considering possibility of transient IEs being injected dynamically
ema_ap_num_max_vaps	Specifies maximum number of ema vaps supported usage: uci set wireless.qcawifi=qcawifi uci set wireless.qcawifi. ema_ap_num_max_vaps = 16 uci commit This INI is only applicable to lower and higher bands.	

The disable_6ghz_mbssid INI, which provided configurability to change the behavior (enable/disable) of MBSSID (and EMA) being forced in 6 GHz for pdevs only has been removed in QCA_Networking_2021.SPF.11.4 release. This INI was available only in the SPF 11.3 release., where the default value of this INI was 1 keeping the feature disable in 6 GHz.

cfg80211tool parameters

Enter the `cfg80211tool wifiN non_inherit_enable 0/1` command to enable/disable non-inheritance feature for EMA AP.

8.77 6 GHz security compliance with 802.11 specification

Starting with QCA_Networking_2020.SPF.11.3 release, this feature is introduced on QCN90xx/QCN602x platforms.

802.11 specification mandates that all Pre-RSNA security methods are to be deprecated. Open Security is to be replaced with OWE, password-based authentication should use SAE, and the Enterprise Security here is Suite-B/Suite-B-192. PMF is mandatory in 6 GHz. In addition to these requirements, WFA certification requires the following:

- WPA3 Personal Transition modes (SAE+PSK2)
- OWE Transition mode (Open+OWE)
- WPA3-SAE

Currently, WPS support for SAE is enabled because hostapd supports it.

By default, on 6 GHz, only the aforementioned securities are supported. Any legacy configuration causes the VAP to not come up.

If you want to enable legacy securities, you can use following command with the UCI option, en_6g_sec_comp, which is introduced as follows:

```
wireless.@wifi-iface[0].en_6g_sec_comp='0'
```

By default, the en_6g_sec_comp is always 1, which means 6 GHz band is strictly WPA3 only. However, the user can make it '0', which causes all legacy security features to be supported per VAP. An additional UCI option is also available to allow only selected key managements as follows:

```
wireless.@wifi-iface[0].keymgmt_mask_6g='0xC30E00'
```

8.78 Configure WDS extended feature

Starting with QCA_Networking_2020.SPF.11.3 release, this feature is introduced on IPQ40xx, IPQ806x, IPQ807x/IPQ807xA/IPQ817x, IPQ50xx, IPQ60xx, and QCN90xx/QCN602x platforms. Starting with the IPQ5018.ILQ.11.4 release, this feature is extended to QCN61xx chipsets attached to IPQ50xx platforms.

Support is introduced for WDS Extended feature in both NSS offload enabled and disabled mode. The backhaul multicast /broadcast should be a unicast frame so that it can be transmitted with high rate and reliable transmission. Multicast and broadcast domain separation among backhaul and fronthaul.

Enter the following UCI configuration to enable the WDS extended feature.

```
uci set wireless.qcawifi=qcawifi
uci set wireless.qcawifi.wds_ext=1
uci commit wireless
```

This is a radio-level configuration. Feature should be configured explicitly on backhaul AP for the feature to be active.

8.79 Configure Mesh 2.0 architecture

Mesh 2.0 architecture makes use of additional UCI config parameters to execute the intended application in the desired configured mode. Rest of the config parameters are carried forward from the current architecture. There are few additional parameters which will be mandated starting with QCA_Networking_2021.SPF.11.4 to make sure only ezmesh is initialized by default. The following is the list of changes which is common for any of ezmesh operating modes,

1. Replace all '*uci set hyd'* with *'uci set ezmesh'*
2. Replace all '*uci set lbd*' to '*uci set ezlbd'*
3. Repacd has to be enabled with ezmesh mode by default, "*uci set repacd.repacd.Ezmesh='1'* "
4. Ezmesh uses '*emt'* for debug terminal.
5. Tmp file name starts with '*ezmesh*'.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

6. Configurations are part of '*/etc/config/ezmesh'* and '*/etc/config/ezlbd'*

The following is the list of modes supported in ezmesh and each section will have the packaging and config details.

Ezmesh Full Mode

Full mode will be the default mode and it is equivalent to existing MAP mode where it will be using single ALID, and both controller and Agent modules will be part of the same binary. Premium profile by default will have ezmesh package installed and following commands needs to execute to start the DUT in Full mode.

- wifimesh config map # Used to link the wsplcd package with ezmesh
- uci set ezmesh.config.DualAlid='0' # By default this value will be set to 0
- uci set repacd.repacd.Ezmesh='1' # Enabled to start ezmesh, 0 for proprietary SON

Ezmesh Dual Mode

Dual mode will be used to run both Co-located Agent and Co-located Controller on the same DUT. By default, br-lan mac address will be used as ALID for Co-located Agent. By default, Co-located Controller will carry the mac id of '*eth2*' as its ALID and is user configurable. Both cannot make use of the same ALID. In this mode it is mandatory to have the Ethernet port '*eth2*' physically connected to a system or to a switch and it should be in up-state. Premium profile by default will have ezmesh package alone however it is required to install separate '*ezmeshAgent'* only and '*ezmeshController*'only ipk manually. Post that under /usr/sbin there will be '*ezmeshAgent'* and '*ezmeshCtrl'* installed. Use following config to run Dual mode.

- wifimesh config map # Used to link the wsplcd package with ezmesh
- uci set ezmesh.config.DualAlid='1' # By default this value will be set to 0.
- uci set repacd.repacd.Ezmesh='1' # Enabled to start ezmesh, 0 for proprietary SON.
- uci set ezmesh.config.DualAlidIntf='eth2'# Co-located Alid for Controller
- uci set ezmesh.config.DualAlidM='Dual'# Run Co-located Controller and Co-located Agent

1. Dual application - one for Controller and another for Agent running in this mode
2. Use 'emt' to enter agent prompt
3. Use 'telnet 127.0.0.1 8888' for Controller prompt
4. Both will show each other as its direct neighbor

Ezmesh Controller only Mode

Controller only mode will be used to run QCA Controller alone in the DUT. Controller will carry the mac id of '*eth2'* by default as its ALID. In this mode it is mandatory to have the Ethernet port '*eth2'* physically connected to a system or to a switch and it should be in up-state. Premium profile by default will have ezmesh package alone however we need to install separate *ezmeshController* only ipk manually. Post that under /usr/sbin there will be *ezmeshCtrl* installed. Use following config to run in Controller only mode.

- wifimesh config map # Used to link the wsplcd package with ezmesh

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

- uci set ezmesh.config.DualAlid='1' # By default this value will be set to 0.
- uci set repacd.repacd.Ezmesh='1' # Enabled to start ezmesh, 0 for proprietary SON.
- uci set ezmesh.config.DualAlidIntf='eth2'# Co-located Alid for Controller
- uci set ezmesh.config.DualAlidM='Controller'# Controller Only mode

1. ezmeshCtrl alone will be running.
2. Use ‘emt’ for Controller prompt.

Ezmesh Agent only Mode

Agent only mode will be used to run QCA Agent alone in the DUT. Premium profile by default will have ezmesh package alone however we need to install separate ezmeshAgent only ipk manually. Post that under /usr/sbin there will be ezmeshAgent installed. Use following config to run in Agent only mode.

- wifimesh config map # Used to link the wsplcd package with ezmesh
- uci set ezmesh.config.DualAlid='1' # By default this value will be set to 0.
- uci set repacd.repacd.Ezmesh='1' # Enabled to start ezmesh, 0 for proprietary SON.
- uci set ezmesh.config.DualAlidIntf='eth2'# Co-located Alid for Controller
- uci set ezmesh.config.DualAlidM=Agent# Agent Only mode

1. ezmeshAgent alone will be running.
2. Use ‘emt’ to enter Agent prompt.

Package install/uninstall

By default, premium profile will have ezmesh , ezmeshAgent , ezmeshCtrrl, and its corresponding libraries will be installed.

If co-located mode is not preinstalled in any low mem platform, following is the procedure to install the same.

1. Uninstall the existing ezmesh package using opkg remove qca-ezmesh
2. Use TFTP to transfer the ezmesh init, Agent, and controller ipk from its relevant NHSS.
3. Install ezmesh-iniit ipk first followed by ezmesh-ctrl , ezmesh-agent.
4. To switch ack to ezmesh mode remove the packge using opkg
   a. remove qca-ezmesh-init
   b. remove qca-ezmesh-ctrl
   c. remove qca-ezmesh-agent
5. Then install qca-ezmesh-init followed qca-ezmesh.

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:14 PDT panyafeng@zte.com.cn

8.80 Implementation of mixed backhaul encryption across radios

CAP's wsplcd will have the capability to send multiple M2 messages with the credential cloning over multiple SSID. With this feature, backhaul VAP credentials are also included in one of the M2 message. To indicate this M2 is for a backhaul VAP, network type and backhaul_ap field are set and send along with deep cloning parameters.

RE on receiving M2 message with backhaul AP VAP credential it will configure one of its dummy AP-VAP with received credentials and same credentials is applied to the backhaul STA-VAP. Since backhaul AP-VAP credentials is used to configure the STA-VAP credentials so, backhaul AP-VAP is must in CAP & RE (dummy VAP in case of RE) to clone STA VAP credentials using wsplcd.

If RE gets onboarded with WPS, then best-uplink will be kicked only after cloning is completed by wsplcd. This is because during wps push button only one band's backhaul STA will be connected to CAP and other STA VAP credentials in other bands will be cloned using wsplcd so all STA-VAP will have valid config and will associate to CAP only after wsplcd cloning. After cloning is performed, wsplcd will intimate "cloning_done" to repacd using pipe and repacd will start the best-uplink selection logic.

When guest config is enabled, typically each radio will have two VAPs (a FH VAP and a BH VAP). Instead of using “wps_enhc” which has the limitations, the management of backaul VAPs is handled by “wsplcd”. Managing backhaul VAPs by “wsplcd” enables supporting different encryption for each radio. Also, as wsplcd manages both fronthaul and backhaul VAPs, then “wsplcd registrar” should have multiple M2 support in auto-configuration 1905 Message which is made available as a part the credential cloning over multiple SSID, which is limited to fronthaul alone and must be extended to backhaul.

To support both FH and BH in multiple M2 message, the BSS type should be mentioned in each M2 message using vendor specific TLV. Based on the BSS type, the network name can be updated as ‘lan’ or ‘backhaul’. The wireless configuration "backhaul_ap" is set to 1 for backhaul VAP alone and identified its BSS Type as Backhaul.

The STA VAP’s credentials will be cloned with same credentials of Backhaul AP VAP with the help of the config:

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

When the WPS is initiated between CAP and RE, only one STA VAP will be connected to establish the connection and other STA VAP cloning will be performed using wsplcd.

The repacd creates the VAP in two-radio devices only. Configuring of STA VAP must be performed manually or via WPS push. wsplcd will configure STA VAPs only after cloning is completed in RE.

To disable “wps_enhc” and instead manage backhaul VAPs by “wsplcd”, the following config is used :

```
uci set wireless.qcawifi=qcawifi
uci set wireless.qcawifi.wps_pbc_extender_enhance=0
uci set wireless.qcawifi.wps_pbc_overwrite_ap_settings_all=0
uci set wireless.qcawifi.wps_pbc_overwrite_sta_settings_all=0
```

To clone the STA VAP's credentials to Backhaul AP VAPs, the following config is used:

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

wsplcd_unmanaged should be set to 0 in both CAP and RE for the backhaul VAP to clone its credentials to RE.

'configSta should be set in RE. ex: uci set wsplcd.config.ConfigSta=1

8.81 Configuration changes for credential cloning over multiple SSIDs

- "wireless.wifi-iface[0].SteeringDisabled" is used to disable steering for specific VAP. If SteeringDisabled is set, then bsteer event will be disabled and station will not be managed by stadb. Default Value: 0.
- "repacd.repacd.AdditionalFHCount" Wi-Fi SON config for repacd to auto create additional VAPs with default ssid_x and key. Default Value: 0. For example:

  ```
  uci set repacd.repacd.AdditionalFHCount=3
  ```

- "repacd.MAPConfig.AdditionalFH" MAP config for repacd to get additional VAPs ssid:key:authType. Default Value: NULL. If key is NULL AP will be configured in open mode. authType 0 = wpa2, 1 = mixed mode. For example:

  ```
  uci set repacd.MAPConfig.AdditionalFH="Adi_fbss1:Adifbsspri:0,Adi_
  fbss2:Adifbsspri2:0,Adi_fbss3:Adifbsspri3:0"
  ```

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:17:44 GMT pan.yafeng@zte.com.cn

8.82 Configure DPP R2 using UCI and ACFG commands

For communicating with the hostapd driver, the hostapd cli commands need to be triggered to pass on the values to enable the DPP features. For WFA plugfest these commands were automated using sigma daemon running on AP. As the DPP R2 feature is commercialiazed, these commands need to be send using UCI commands and ACFG tool based commands.

The following table shows the commands to bring up the AP with DPP mode and have the AP device perform authentication and configuration using DPP. In order to support these hostapd_cli commands, the hostapd scripts are modified to send these cli commands based on the uci set parameters in the wireless config file.

Table 8-14 UCI parameters to configure DPP R2

Functionality	Hostapd_cli command
Generating bootstrapping key	run_hostapd_cli $ifname "/var/run/hostapd-$device" "DPP_BOOTSTRAP_GEN $dpp_ type $dpp_curve $channel mac=$(cat /sys/class/net/$ifname/address | sed 's/://g') $dpp_key"
Initiator initiating authentication	run_hostapd_cli $ifname "/var/run/hostapd-$device" "dpp_auth_init peer=1 role=enrollee"
Responder listening on a particular frequency	run_hostapd_cli $ifname "/var/run/hostapd-$device" "dpp_listen $dpp_freq role=enrollee qr=mutual
Enabling DPP over TCP	run_hostapd_cli $ifname "/var/run/hostapd-$device" "dpp_aut h_init peer=2 own=1 role=enrollee tcp_addr=$dpp_over_ tcp"
Enabling DPP chirp	run_hostapd_cli $ifname "/var/run/hostapd-$device" "dpp_chirp own=1 iter=10 listen=$dpp_freq"

The following table lists the added UCI parameters defined for DPP R2 commercialization:

Table 8-15 UCI parameters to configure DPP R2

UCI param	Value	Usage
dpp_auth_role	<initiator> OR <responder> (By default value is initiator)	option dpp_auth_role ‘initiator’
dpp_over_tcp	0 or 1 or <IP address value>	option dpp_over_tcp ‘1’ (AP configured as responder) Or option dpp_over_tcp ‘192.165.100.149’ (AP configured as initiator)

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 8-15 UCI parameters to configure DPP R2 (cont.)

dpp_controller	‘ipaddr=<value 1> pkhash=<value 2>’	option dpp_controller ‘ ipaddr=192.165.100.169 pkhash=10de7238ebddbb736c1ceaf595fbb0d1d21d5581382756866677999a4f856cb2” (AP configured as a relay)
dpp_chirp	0 or 1	option dpp_chirp ‘1’
dpp_mud_url	<www.tms.wifi.org>	option dpp_mud_url ‘www.tms.wifi.org’
dpp_freq	frequency value in Mhz	option dpp_freq ‘2437’
dpp_pfs	0 or 1	option dpp_pfs ‘1’

For example:

```
config wifi-iface
   option device 'wifi1'
   option network 'lan'
   option mode 'ap'
   option encryption 'dpp'
   option dpp '1'
   option dpp_curve 'BP-256'
   option dpp_type 'qr-code'  (Default case)
   option dpp_over_tcp '192.165.100.117'
```

8.83 Configure service prioritization using UCI parameters

Wi-Fi SON is involved in proprietary signaling for Qualcomm APs to achieve service prioritization (SP) across mesh. This section discusses the setting of the SP rules and sending SP rules using 1905 messaging. This feature prioritizes scheduling latency sensitive TIDs by adding support for latency configuration and latency metric in scheduler.

New UCI configuration parameters are introduced to execute service prioritization in the desired configured mode. There are UCI configs for Hyd and Ezmesh, depending on the module that is used.

Hyd Module:

1. hyd.MAPSPSettings.EnableSP—This config enables SP feature.
2. hyd.MAPSPSettings. EnableEnhancedSP—This config indicates whether the Enhanced SP is enabled on the device. A value of 1 indicates that the device can handle Enhanced SP parameter. Enhanced SP rules are not sent out to devices that do not have this parameter enabled. Default value is 0.
3. hyd.MAPSPSettings.EnableQSP—This config indicates whether the Q SP is enabled on the device. A value of 1 indicates that the device can handle Q SP parameter. QSP rules are not sent out to devices that do not have this parameter enabled. Default value is 0.
4. hyd.MultiAP.EnableQCapability—This config indicates whether the device is a Q capable device. A value of 1 indicates that the device is Q capable and 0 indicates it is not Q capable.

This is responsible for sending and receiving Q capability query and response. Default value is 0.

5. hyd.MAPSPSettings.SPRulesConfPath—This config specifies the path location of the service prioritization rules config file. This config is to be set on the controller alone. Default value is '/etc/hyd/map/sp-rules.conf'.
6. hyd.MAPSPSettings.MaxSPRules=10—This config indicated the maximum number of rules that can be set from sp-rules config file. Default value is 0. For Controller, this config specifies maximum number of rules that can be read and set from sp-rules config file. For Agent, this config specifies the maxSPrules in r2APcapabilities. This value should be the same on Controller and agents.

Ezmesh Module:

1. ezmesh.MAPSPSettings.EnableSP—This config enables SP feature.
2. ezmesh.MAPSPSettings. EnableEnhancedSP—This config indicates whether the Enhanced SP is enabled on the device. A value of 1 indicates that the device can handle Enhanced SP parameter. Enhanced SP rules are not sent out to devices that do not have this parameter enabled. Default value is 0.
3. ezmesh.MAPSPSettings.EnableQSP—This config indicates whether the Q SP is enabled on the device. A value of 1 indicates that the device can handle Q SP parameter. QSP rules are not sent out to devices that do not have this parameter enabled. Default value is 0.
4. ezmesh.MultiAP.EnableQCapability—This config indicates whether the device is a Q capable device. A value of 1 indicates that the device is Q capable and 0 indicates it is not Q capable. This is responsible for sending and receiving Q capability query and response. Default value is 0.
5. ezmesh.MAPSPSettings.SPRulesConfPath—This config specifies the path location of the service prioritization rules config file. This config is to be set on the controller alone. Default value is '/etc/ezmesh/sp-rules.conf'.
6. ezmesh.MAPSPSettings.MaxSPRules=10—This config indicated the maximum number of rules that can be set from sp-rules config file. Default value is 0.

Enable ECM EasyMesh classifier

```
echo 1 > /sys/kernel/debug/ecm/ecm_classifier_emesh/latency_config_
enabled
insmod qca-nss-mscs
```

8.84 Configure dynamic role selection of CAP/RE (Controller/Agent) using UCI parameters

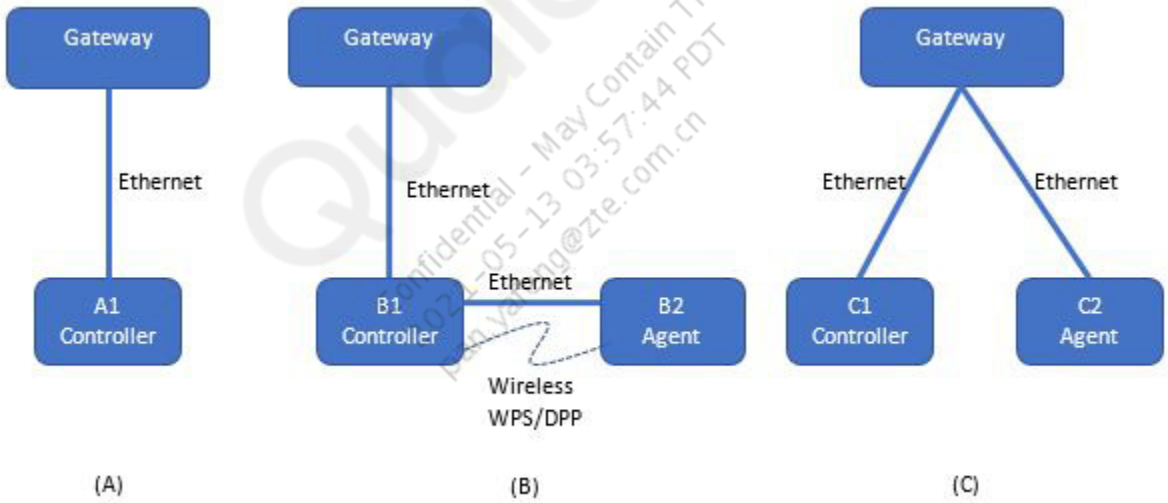
Starting with QCA_Networking_2021.SPF.11.4 release, this feature is implemented on IPQ401x, IPQ806x, IPQ807x/IPQ817x/IPQ807xA, IPQ50xx, IPQ60xx, and QCN90xx/QCN60xx/QCN61xx platforms.

The capability to dynamically select a Controller in EasyMesh network, based on certain criteria (for example, lowest MAC address), is implemented. User space application will be responsible for monitoring and making this role selection decision. User space application will use certain set

of APIs/commands to fetch information from EasyMesh module WSPLCD using UCI calls and shared file. The *Mesh 2.0 API Reference* (80-YA984-9) and *Mesh 2.0 Layer Definitions API Reference* (80-YA984-10) describe the API functionalities. This is a Qualcomm-to-Qualcomm feature that supports EasyMesh r1.

The following are certain use case scenarios in which this dynamic role selection of CP/RE is required:

1. Case 1: If the Gateway does not support mesh, then nodes, supporting both SON and Easymesh, shall configure itself as Controller or Agent based on the topology and algorithm. If the Controller device is removed, all agents shall reboot and re-perform the role selection again.
   a. The only node is connected to Gateway through Ethernet; it is the controller.
   b. Only one node is connected to Gateway through Ethernet; it is the controller. Other nodes are connected to the node as agent.
   c. Two or more nodes are connected to Gateway through Ethernet; one of the nodes should be selected as controller based on a self-defined algorithm, other nodes are agent.



2. Case 2: If the GW support mesh (EZM controller), upon receiving the message from GW, then nodes are configured as agent.
3. Case 3: If no GW, or connecting to SFU (bridge), then CAP/controller works in router mode, role definition as Case 1.

The topology is nodes are running Easymesh in bridge mode. From the requirements, focus is on the scenario 3 of Case 1. Other scenarios and cases can be a special form or be considered incorporated in the scenario.

The application will be started after system bootup and check the network status at the beginning before setting or making decision to set to Controller/Agent. If the node has no IP address or the gateway cannot be reached, the node will not start the process.

A node will announce its capabilities in the 1905 AP Configuration Search message and collect the capabilities from the neighbor nodes to make decision.

The following are the UCI parameters added to the wsplcd application:

- wsplcd.config.EnableNBTLV—Enable the TLV support
- wsplcd.config.NBTLVbuff <Vendor TLV Buffer> —TLV parameters to be sent as vendor tlv in search msg
- wsplcd.config.ResponseRcvd —Check if response for search msg received from controller

The autorole daemon is implemented to enable and configure the dynamic role selection of CAP/RE (Controller/Agent) functionality. The following are the UCI parameters that can be configured for this capability:

- uci set autorole.config.Enable: Enable the dynamic role selection of CAP/RE (Controller/Agent) feature.

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

- uci set autorole.config.RunMode: “Auto” or empty: dynamically select the role based on algorithm. “Controller” or “Agent”: Set to the given role.
- uci set autorole.config.Network: The network to check the gateway. The default is ‘lan’.
- uci set autorole.config.WaitTime: Time value to check the neighbor nodes.
- uci set autorole.config.ControllerCheck: Flag to enable the check of availability of controller for Agent.
- uci set autorole.wifi.FronthaulSSID/ uci set autorole.wifi.FronthaulKey / uci set autorole.wifi.BackhaulSSID / uci st autorole.wifi.BackhaulKey: Wi-Fi configuration for Controller role to be applied to repacd and wireless settings.

8.85 Configure optional IEEE 802.11 IE support for non-Tx VAP in MBSSID/EMA using .ini and UCI parameters

Starting with QCA_Networking_2021.SPF.11.4 release, this feature is implemented on QCN90xx/QCN602x attached to IPQ807xA, IPQ60xx platforms. Support is implemented for 11k, 11r, 11v, 11u (Hotspot), and other optional features for a combination of Tx and non-Tx VAPs. Until QCA_Networking_2021.SPF.11.3 release, optional feature support in a non-Tx VAP was possible through inheritance (must be enabled in Tx-vap) only. This feature is now extended to all types of VAPs, regardless of whether they are Tx or non- Tx VAPs.

This feature extends the MBSSID/EMA framework to support any VAP-level feature generically. This framework supports any IEEE80211 IE addition per-VAP, regardless of it being Tx or non-Tx.

Tx	non-Tx VAP1	non-Tx VAP2
✓	✓	✓
✓	✓	×
✓	×	✓
✓	×	×
×	✓	✓
×	✓	×
×	×	✓
×	×	×

Figure 8-7 Combination of feature enable/disable across one Tx and two non-Tx VAPs

Prior to this enhancement, non-Tx VAP used to support an optional feature such as 11k/r/u only through inheritance when the feature is enabled in Tx VAP. This feature retains that mode using a knob.

Additionally, this feature brings the flexibility of updating the resource configuration for optional and vendor IE sections dynamically by bringing down the Tx-vap (so, consequently all other non-Tx VAPs).

If certain IE needs to be dropped due to size limitations, then the following conditions apply:

- If the event has occurred during NL80211_START_AP then fail the VDEV up event.
- If the event occurs dynamically by wlanconfig/hostapd_cli commands then fail the requested command whenever the IE cannot be accommodated in the VAP's available space
- If the event occurs due to dynamic enablement of a feature then the IE is dropped and beaconing continues at its current operating state.

This feature requires an additional input parameter ema_ap_optional_ie_size in the preceding function.

ema_ap_optional_ie_size defines the optional IE section to be used uniformly across all VAPs.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Config type	Config ID	Description
INI	ema_ap_optional_ie_size	Size in bytes of Optional IE section of non-Tx profiles. Note that this common size will be applicable to all the non-Tx VAPs at init. The default value is 100. This value is required to be >=30 in current release. Otherwise, VAP bringup might fail. Usage: uci set wireless.qcawifi=qcawifi uci set wireless.qcawifi. ema_ap_optional_ie_size=100 uci commit

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

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_AFGHANISTAN	AF	4
CTRY_ALBANIA	AL	8
CTRY_ALGERIA	DZ	12
CTRY_AMERICAN_SAMOA	AS	16
CTRY_ARGENTINA	AR	32
CTRY_ARUBA	AW	533
CTRY_ARMENIA	AM	51
CTRY_AUSTRALIA	AU	36
CTRY_AUSTRIA	AT	40
CTRY_AZERBAIJAN	AZ	31
CTRY_BAHAMAS	BS	44
CTRY_BAHRAIN	BH	48
CTRY_BANGLADESH	BD	50
CTRY_BARBADOS	BB	52
CTRY_BERMUDA	BM	60
CTRY_BELARUS	BY	112
CTRY_BELGIUM	BE	56
CTRY_BELIZE	BZ	84
CTRY_BHUTAN	BT	64
CTRY_BOLIVIA	BO	68
CTRY_BOSNIA_HERZEGOWINA	BA	70
CTRY_BRAZIL	BR	76
CTRY_BRUNEI_DARUSSALAM	BN	96
CTRY_BULGARIA	BG	100
CTRY_BURKINA_FASO	BF	854
CTRY_CANADA	CA	124
CTRY_CAYAMAN_ISLANDS	KY	136

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

Table A-1 Country code definitions (cont.)

Country definition	Country string	Country ID
CTRY_CENTRAL AFRICA REPUBLIC	CF	140
CTRY_CHAD	TD	148
CTRY_COTE D'IVOIRE	CI	384
CTRY_CHILE	CL	152
CTRY_CHINA	CN	156
CTRY_CHRISTMAS_ISLAND	CX	162
CTRY_COLOMBIA	CO	170
CTRY_COSTA_RICA	CR	188
CTRY_CROATIA	HR	191
CTRY_CURACAO	CW	531
CTRY_CYPRUS	CY	196
CTRY_CZECH_REPUBLIC	CZ	203
CTRY_DENMARK	DK	208
CTRY_DOMINICA	DM	212
CTRY_DOMINICAN_REPUBLIC	DO	214
CTRY_ECUADOR	EC	218
CTRY_EGYPT	EG	818
CTRY_EL_SALVADOR	SV	222
CTRY_ESTONIA	EE	233
CTRY_ETHIOPIA	ET	231
CTRY_FAROE_ISLANDS	FO	234
CTRY_FINLAND	FI	246
CTRY_FRANCE	FR	250
CTRY_FRENCH_GUIANA	GF	254
CTRY_GEORGIA	GE	268
CTRY_GERMANY	DE	276
CTRY_GHANA	GH	288
CTRY_GIBRALTER	GI	292
CTRY_GREECE	GR	300
CTRY_GREENLAND	GL	308
CTRY_GRENADA	GD	312
CTRY_GUADELOUPE	GP	316
CTRY_GUAM	GU	304
CTRY_GUATEMALA	GT	320
CTRY_HONDURAS	HN	340
CTRY_HONG_KONG	HK	344

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

Table A-1 Country code definitions (cont.)

Country definition	Country string	Country ID
CTRY_HUNGARY	HU	348
CTRY_ICELAND	IS	352
CTRY_INDIA	IN	356
CTRY_INDONESIA	ID	360
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_JAPAN15	JP	4015
CTRY_JORDAN	JO	400
CTRY_KAZAKHSTAN	KZ	398
CTRY_KENYA	KE	404
CTRY_KOREA_ROC	KR	410
CTRY_KUWAIT	KW	414
CTRY_LATVIA	LV	428
CTRY_LEBANON	LB	422
CTRY_LESOTHO	LS	426
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_MALAWI	MW	454
CTRY_MALAYSIA	MY	458
CTRY_MALDIVES	MV	462
CTRY_MALTA	MT	470
CTRY_MARSHALL_ISLANDS	MH	584
CTRY_MARTINIQUE	MQ	474
CTRY_MAURITANIA	MR	478
CTRY_MAURITIUS	MU	480
CTRY_MAYOTTE	YT	175
CTRY_MEXICO	MX	484
CTRY_MICRONESIA	FM	583

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

Table A-1 Country code definitions (cont.)

Country definition	Country string	Country ID
CTRY_MOLDOVA	MD	498
CTRY_MONACO	MC	492
CTRY_MOROCCO	MA	504
CTRY_MONGOLIA	MN	496
CTRY_MONTENEGRO	ME	499
CTRY_NETHERLANDS	NL	528
CTRY_NETHERLANDS_ANTILLES	AN	530
CTRY_NEW_ZEALAND	NZ	554
CTRY_NICARAGUA	NI	558
CTRY_NIGERIA	NG	566
CTRY_NORTHERN MARIANA ISLANDS	MP	580
CTRY_NORWAY	NO	578
CTRY_OMAN	OM	512
CTRY_PAKISTAN	PK	586
CTRY_PANAMA	PA	591
CTRY_PALAU	PW	585
CTRY_PAPUA NEW GUINEA	PG	598
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_REUNION	RE	638
CTRY_ROMANIA	RO	642
CTRY_RUSSIA	RU	643
CTRY_RWANDA	RW	646
CTRY_SAUDI_ARABIA	SA	682
CTRY_ST_BARTHELEMY	BL	652
CTRY_ST_KITTS_NEVIS	KN	659
CTRY_ST_MARTIN	MF	663
CTRY_ST_PIERRE_MIQUELON	PM	666
CTRY_ST_VINCENT_GRENADIENS	VC	670
CTRY_SAMOA	WS	882
CTRY_SERBIA	RS	688

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

Table A-1 Country code definitions (cont.)

Country definition	Country string	Country ID
CTRY_SENEGAL	SN	686
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_ST_LUCIA	LC	662
CTRY_SURINAME	SR	740
CTRY_SWEDEN	SE	752
CTRY_SWITZERLAND	CH	756
CTRY_TAIWAN	TW	158
CTRY_TANZANIA	TZ	834
CTRY_THAILAND	TH	764
CTRY_TOGO	TG	768
CTRY_TRINIDAD_Y_TOBAGO	TT	780
CTRY_TUNISIA	TN	788
CTRY_TURKEY	TR	792
CTRY_TURKS_CAICOS	TC	796
CTRY_UAE	AE	784
CTRY_UGANDA	UG	800
CTRY_UKRAINE	UA	804
CTRY_UNITED_KINGDOM	GB	826
CTRY_UNITED_STATES	US	840
CTRY_URUGUAY	UY	858
CTRY_UZBEKISTAN	UZ	860
CTRY_VANUATU	VU	548
CTRY_VENEZUELA	VE	862
CTRY_VIET_NAM	VN	704
CTRY_VIRGIN_ISLANDS	VI	850
CTRY_YEMEN	YE	887
CTRY_WALLIS_FUTUNA	WF	876
CTRY_ZIMBABWE	ZW	716
CTRY_XA	XA	4100
CTRY_MYANMAR	MM	104
CTRY_CAMBODIA	KH	116

Qualcomm Confidential – May Contain Trade Secrets 2021-05-13 03:57:44 PDT pan.yafeng@zte.com.cn

Table A-1 Country code definitions (cont.)

Country definition	Country string	Country ID
CTRY_HAITI	HT	332
CTRY_NAMIBIA	NA	516
CTRY_NEPAL	NP	524
CTRY_ALAND_ISLANDS	AX	248
CTRY_ANDORRA	AD	20
CTRY_ANTIGUA_AND_BARBUDA	AG	28
CTRY_CAMEROON	CM	120
CTRY_COOK_ISLANDS	CK	184
CTRY_FALKLAND ISLANDS	FK	238
CTRY_FRENCH SOUTHERN TERRITORIES	TF	260
CTRY_GUERNSEY	GG	831
CTRY_HEARD_ISLAND_AND_MCDONALD_ ISLANDS	HM	334
CTRY_ISLE_OF_MAN	IM	833
CTRY_JERSEY	JE	832
CTRY_MONTSERRAT	MS	500
CTRY_NEW CALCEDONIA	NC	540
CTRY_NIUE	NU	570
CTRY_NORFOLK ISLAND	NF	574
CTRY_SAINT_HELENA_ASCENSION_AND_ TRISTAN_DA_CUNHA	SH	654
CTRY_SAN MARINO	SM	674
CTRY_SINT MAARTEN (DUTCH PART)	SX	534
CTRY_SAO TOME AND PRINCIPE	ST	678
CTRY_SVALBARD AND JAN MAYEN	SJ	744
CTRY_UNITED_STATES_MINOR_OUTLYING_ ISLANDS	UM	581
CTRY_VIRGIN_ISLANDS_BRITISH	VG	92
CTRY_ANGUILLA	AI	660
CTRY_FRENCH_PLOYNESIA	PF	258
CTRY_ISLE_OF_MAN	IM	833
CTRY_GUYANA	GY	328
CTRY_HOLY_SEE	VA	336

NOTE Starting with QCA_Networking_2017.SPF.5.0.3, the following country codes are deprecated and are returned as invalid if entered using the iwpriv wifi0 setCountryID command:

- CTRY_JAPAN through CTRY_JAPAN5 with corresponding country codes of 392 through 397
- CTRY_JAPAN6 through CTRY_JAPAN8 with corresponding country codes of 4006 through 4008
- CTRY_JAPAN10 through CTRY_JAPAN14 with corresponding country codes of 4010 through 4014
- CTRY_JAPAN16 through CTRY_JAPAN47 with corresponding country codes of 4016 through 4047
- CTRY_JAPAN49 through CTRY_JAPAN54 with corresponding country codes of 4049 through 4054
- CTRY_JAPAN49 through CTRY_JAPAN54 with corresponding country codes of 4049 through 4054
- CTRY_JAPAN56 through CTRY_JAPAN59 with corresponding country codes of 4056 through 4059
- CTRY_IRAN (IR) with the country ID of 364
- CTRY_KOREA_NORTH (KP) with the country ID of 408
- CTRY_SYRIA (SY) with the country ID of 760

NOTE Starting with QCA_Networking_2017.SPF.6.0, The following country codes are deprecated and are returned as invalid if entered using the `iwpriv wifi0 setCountryID` command:

- CTRY_KOREA_ROC3 with country code of 412
- CTRY_AUSTRALIA2 with country code of 5000
- CTRY_ARGENTINA2 with country code of 5003
- CTRY_CANADA2 with country code of 5001
- CTRY_UNITED_STATES2 through CTRY_UNITED_STATES3 with corresponding country codes of 841 and 843
- CTRY_UNITED_STATES_FCC49 with country code of 842

NOTE Starting with QCA_Networking_2017.SPF.6.0, enhancements are implemented to certain country code definitions to support regulatory domains. The following country codes are newly introduced:

- CTRY_NAMIBIA with the country string of NA and country ID of 516
- CTRY_NEPAL with the country code of NE and country ID of 524
- CTRY_DOMINICA with the country code of DM and country ID of 212
- CTRY_IRAQ with the country code of IQ and country ID of 368
- CTRY_SENEGAL with the country code of SN and country ID of 686
- CTRY_RWANDA with the country code of RW and country ID of 646

- CTRY_PALAU with the country code of PW and country ID of 585
- CTRY_PAPUA NEW GUINEA with the country code of PG and country ID of 598
- CTRY_SERBIA with the country code of RS and country ID of 688
- CTRY_UGANDA with the country code of UG and country ID of 800
- CTRY_BANGLADESH with the country code of BD and country ID of 50
- CTRY_BARBADOS with the country code of BB and country ID of 52
- CTRY_BERMUDA with the country code of BM and country ID of 60
- CTRY_CENTRAL AFRICA REPUBLIC with the country code of CF and country ID of 140
- CTRY_CHAD with the country code of TD and country ID of 148
- CTRY_COTE D'IVOIRE with the country code of CI and country ID of 384
- CTRY_ETHIOPIA with the country code of ET and country ID of 231
- CTRY_MALTA with the country code of MT and country ID of 470
- CTRY_MONGOLIA with the country code of MN and country ID of 496
- CTRY_MONTENEGRO with the country code of ME and country ID of 499
- CTRY_NIGERIA with the country code of NG and country ID of 566
- CTRY_NORTHERN MARIANA ISLANDS with the country code of MP and country ID of 580
- CTRY_JAPAN with the country code of JP and country ID of 4015

NOTE Starting with the QCA_Networking_2019.SPF.10.0,release, CTRY_MYANMAR with the country code of MM and country ID of 104 for Myanmar is introduced.

NOTE Starting with QCA_Networking_2019.SPF.11.0 release, the enhancements are implemented to certain country code definitions to support regulatory domains. The following country codes are newly introduced:

- CTRY_ALAND_ISLANDS with country string of AX and country code of 248
- CTRY_ANDORRA with country string of AD and country code of 20
- CTRY_ANTIGUA_AND_BARBUDA with country string of AG and country code of 28
- CTRY_CAMERON with country string of CM and country code of 120
- CTRY_COOK_ISLANDS with country string of CK and country code of 184
- CTRY_FALKLAND ISLANDS with country string of FK and country code of 238
- CTRY_FRENCH SOUTHERN TERRITORIES with country string of TF and country code of 260
- CTRY_GUERNSEY with country string of GG and country code of 831

- CTRY_HEARD_ISLAND_AND_MCDONALD_ ISLANDS with country string of HM and country code of 334
- CTRY_ISLE_OF_MAN with country string of IM and country code of 833
- CTRY_JERSEY with country string of JE and country code of 832
- CTRY_MONTSERRAT with country string of MS and country code of 500
- CTRY_NEW CALCEDONIA with country string of NC and country code of 540
- CTRY_NIUE with country string of NU and country code of 570
- CTRY_NORFOLK ISLAND with country string of NF and country code of 574
- CTRY_SAINT_HELENA_ASCENSION_AND_TRISTAN_DA_CUNHA with country string of SH and country code of 654
- CTRY_SAN MARINO with country string of SM and country code of 674
- CTRY_SINT MAARTEN (DUTCH PART) with country string of SX and country code of 534
- CTRY_SAO TOME AND PRINCIPE with country string of ST and country code of 678
- CTRY_SVALBARD AND JAN MAYEN with country string of SJ and country code of 744
- CTRY_UNITED_STATES_MINOR OUTLYING ISLANDS with country string of UM and country code of 581
- CTRY_VIRGIN_ISLANDS_BRITISH with country string of VG and country code of 92

NOTE Starting with QCA_Networking_2021.SPF.11.4 release, the enhancements are implemented to certain country code definitions to support regulatory domains. The following country codes are newly introduced:

- CTRY_ANGUILLA with the country code of AI and country ID of 660
- CTRY_FRENCH_POLYNESIA with country string of PF and country code of 258
- CTRY_GUYANA with country string of GY and country code of 328
- CTRY_HOLY_SEE with country string of VA and country code of 336
- CTRY_ISLE_OF_MAN with the country code of IM and country ID of 833