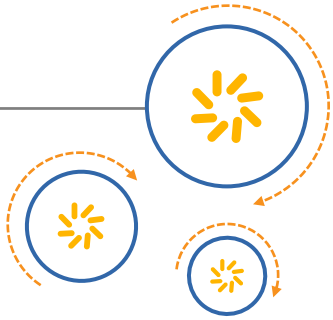
Qualcomm Atheros, Inc.



QCA99xx WLAN TLV Commands

User Guide

80-Y8050-10 Rev. B

October 30, 2015

For additional information or to submit technical questions go to https://createpoint.qti.qualcomm.com/

Qualcomm Atheros, Inc.
1700 Technology Drive
San Jose, CA 95110
U.S.A.

Revision history

Revision	Date	Description
A	October 2014	Initial release
B	October 2015	Updated the document for QCA9984
		Updated Table 3-3, Table 3-5, and Table 3-11

Contents

Figures

Tables

1 Introduction

1.1 Purpose

This document specifies the WLAN TLV (type length value) commands and responses that are transacted between host application (ex. PC) and WLAN firmware. FTM (factory test mode) diagnostic command packets used on WLAN non-signaling testing embed these TLV commands and responses within DIAG (diagnostic task or service). This document covers all WLAN TLV commands that can be performed from a host application.

NOTE: WLAN TLV command and response packets described in this document are applicable for QCA99xx (QCA9980, QCA9982, QCA9984, QCA9990 and QCA9992) chipsets.

1.2 Scope

This document is intended for engineers using WLAN TLV on a host application to communicate with QCA targets in FTM mode. This document assumes that the reader is familiar with WLAN physical layer channels and concepts.

1.3 Conventions

Function declarations, function names, type declarations, and code samples appear in a different font, e.g., `#include`.

If you are viewing this document using a color monitor, or if you print this document to a color printer, red typeface indicates data types, blue typeface indicates attributes, and green typeface indicates system attributes.

2 TLV Stream Format

This section defines the TLV stream format. It can also be used a reference to debug the contents of a TLV stream. The stream is shown in little-endian format.

Please refer to *QCA99XX FTM API - WLAN Non-signaling Commands User Guide* (80-Y8050-11) for details on how DIAG packets are sent from QMSL or QDART. WLAN DIAG packets encapsulate the TLV stream defined on this document.

2.1 TLV Stream

A TLV stream contains a 28-byte stream header and its payload. It represents a command from the host or a response from target.

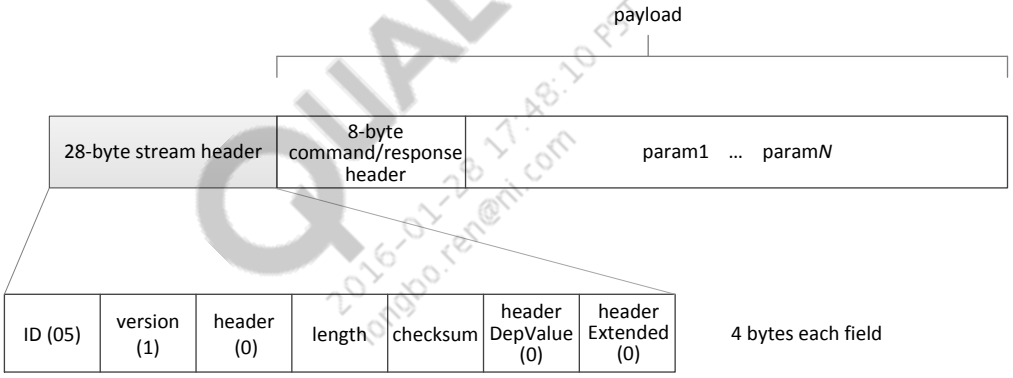


Figure 2-1 TLV steam format

Table 2-1 TLV stream header fields

Header filed	Description
ID	Constant
version	Constant
header	Constant
length	Length of the stream payload including the command header and all the parameters
checksum	16-bit checksum of the whole stream with the checksum field pre-set to 0
headerDepValue	Constant
headerExtended	Constant

2.2 Payload

Typically, the payload contains an 8-byte command or response header and its parameters. Payload size is 2K. The header has a 4-byte command op-code and 4-byte number of parameters:

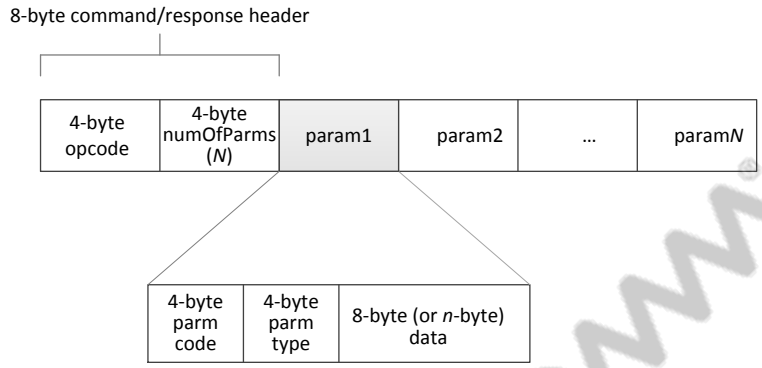


Figure 2-2 Command/response header and its N parameters

Each parameter field is composed of a 4-byte parameter code, 4-byte parameter type, and its data. Based on its type, the data length can be 8 bytes or varied.

Parameters of a command/response can be added to the stream in any order.

3 TLV Commands/Responses

3.1 Opcodes

Table 3-1 Opcodes

Opcode ID	Value	Description
_OP_TX	1	Tx command
_OP_RX	2	Rx command
_OP_PM	5	Sleep command
_OP_GENERIC_RSP	6	Tx/Rx response
_OP_GENERIC_NART_CMD	8	Generic command
_OP_GENERIC_NART_RSP	9	Generic response
_OP_TX_STATUS	10	Tx status response
_OP_RX_STATUS	11	Rx Status response

3.2 Parameters

Each command/response has its own set of parameters. Each parameter is defined with code, type, and size.

Parameter structure

```
typedef struct _parmVal {
    A_UINT32    val16;
    A_UINT32    val32;
} __ATTRIB_PACK _PARM_VAL;


typedef struct _parmOneOf {
    A_UINT32     parmCode;
    A_UINT32     parmType;
    union {
        A_UINT8 addr[8];
        _PARM_VAL value;
    } parmValue;
} __ATTRIB_PACK  _PARM_ONEOF;
```

Parameter type

```
typedef enum {
    _PARM_RESERVED = 0,
    _PARM_U8,           // unsigned char
    _PARM_U16,          // unsigned short
    _PARM_U32,          // unsigned integer
    _PARM_S8,           // signed char
    _PARM_S16,          // signed short
    _PARM_S32,          // signed integer
    _PARM_DATA,         // array of unsigned chars
} _PARM_TYPE;
```

Parameter data can be 8-byte or *n*-byte in length. The parameter types can be defined as unsigned/signed char, unsigned/signed short, or unsigned/signed integer, but the data always occupies 8 bytes in the stream with 0s padding.

NOTE: An array parameter with length less than 7 should be defined as type _PARM_U8, e.g. MAC address.

Table 3-2 shows how data is stored in a TLV stream based on its type.

Table 3-2 Data stored in stream (where xx is data)

Parameter type	# of bytes in stream	Data in stream
_PARM_U8/_PARM_S8	8	…. xx 00 00 00 00 00 00 ….
_PARM_U16/_PARM_S16	8	…. xx xx 00 00 00 00 00 ….
_PARM_U32/_PARM_S32	8	…. 00 00 00 00 xx xx xx xx ….
Array of _PARM_U8/_PARM_S8 (size <= 6)	8	…. xx xx xx xx xx xx 00 00 ….
Array of _PARM_U16/_PARM_S16	Not supported	—
Array of _PARM_U32/_PARM_S32	Not supported	—
_PARM_DATA (size = 0xssss)	n	…. ss ss 00 00 00 00 00 00 xx xx xx xx xx xx xx xx xx xx …. (0xssss bytes)

NOTE: Since the payload size is 2048 bytes, the total length of the parameters added to the stream should not be greater than that.

3.3 Tx Command (_OP_TX = 1)

Table 3-3 Parameters in Tx command

Name	Code	Type	Length (bytes)	Description
channel	0	3	4	Frequency, e.g. 2412
txMode	1	3	4	Transmit mode: ▪ 0 – OFF ▪ 1 – SINE ▪ 2 – FRAME ▪ 3 – TX99 ▪ 4 – TX100 ▪ 7 – CWTONE ▪ 8 – CLPCPKT
antenna	52	3	4	Antenna: 0 or 1
enANI	53	3	4	Enable ANI: 0 or 1
scramblerOff	54	3	4	Scrambler off: 0 or 1
aifsn	55	3	4	Inter-frame spacing
txPattern	57	3	4	Transmit pattern: ▪ 0 – ZEROES_PATTERN ▪ 1 – ONES_PATTERN ▪ 2 – REPEATING_10 ▪ 3 – PN7_PATTERN ▪ 4 – PN9_PATTERN ▪ 5 – PN15_PATTERN ▪ 6 – USER_DEFINED_PATTERN
shortGuard	58	3	4	Enable short guard: 0 or 1
numPackets	59	3	4	Number of transmit packets

Name	Code	Type	Length (bytes)	Description
wlanMode	60	3	4	WLAN mode: ▪ 0 – TCMD_WLAN_MODE_NOHT ▪ 1 – TCMD_WLAN_MODE_HT20 ▪ 2 – TCMD_WLAN_MODE_HT40PLUS ▪ 3 – TCMD_WLAN_MODE_HT40MINUS ▪ 4 – TCMD_WLAN_MODE_CCK ▪ 5 – TCMD_WLAN_MODE_VHT20 ▪ 6 – TCMD_WLAN_MODE_VHT40PLUS ▪ 7 – TCMD_WLAN_MODE_VHT40MINUS ▪ 8 – TCMD_WLAN_MODE_VHT80_0 ▪ 9 – TCMD_WLAN_MODE_VHT80_1 ▪ 10 – TCMD_WLAN_MODE_VHT80_2 ▪ 11 – TCMD_WLAN_MODE_VHT80_3 ▪ 12 – TCMD_WLAN_MODE_VHT80P80 ▪ 13 – TCMD_WLAN_MODE_VHT160 ▪ 21 – TCMD_WLAN_MODE_VHT80P80_0 ▪ 22 – TCMD_WLAN_MODE_VHT80P80_1 ▪ 23 – TCMD_WLAN_MODE_VHT80P80_2 ▪ 24 – TCMD_WLAN_MODE_VHT80P80_3 ▪ 25 – TCMD_WLAN_MODE_VHT80P80_4 ▪ 26 – TCMD_WLAN_MODE_VHT80P80_5 ▪ 27 – TCMD_WLAN_MODE_VHT80P80_6 ▪ 28 – TCMD_WLAN_MODE_VHT80P80_7 ▪ 29 – TCMD_WLAN_MODE_VHT160_0 ▪ 30 – TCMD_WLAN_MODE_VHT160_1 ▪ 31 – TCMD_WLAN_MODE_VHT160_2 ▪ 32 – TCMD_WLAN_MODE_VHT160_3 ▪ 33 – TCMD_WLAN_MODE_VHT160_4 ▪ 34 – TCMD_WLAN_MODE_VHT160_5 ▪ 35 – TCMD_WLAN_MODE_VHT160_6 ▪ 36 – TCMD_WLAN_MODE_VHT160_7
txChain0	61	3	4	Tx chain mask
tpcm	65	3	4	TPC mode: ▪ 0 – TPC_TX_PWR ▪ 1 – TPC_FORCED_GAIN ▪ 2 – TPC_TGT_PWR ▪ 3 – TPC_TX_FORCED_GAIN ▪ 4 – TPC_FORCED_GAINIDX ▪ 5 – TPC_FORCED_TGTPWR
flags	66	3	4	Miscellaneous flags: Bit: 7 6 5 4 3 2 1 0 x: Enable STBC (0 or 1) x : Enable LDPC (0 or 1) x : Enable DPD (0 or 1) x : Enable Tx status per rate (0 or 1) x : Enable process rate order (0 or 1) x : Unused x : Enable MPS x : unused

Name	Code	Type	Length (bytes)	Description
agg	67	3	4	Aggregation (1 to 64)
broadcast	68	3	4	Enable broadcast (0 or 1)
bandwidth	69	3	4	Bandwidth: ▪ 50 – HALF_SPEED_MODE ▪ 51 – QUARTER_SPEED_MODE ▪ Otherwise 0
bssid	70	1	6	BSSID
txStation	71	1	6	Tx station ID
rxStation	72	1	6	Rx station ID
dutyCycle	74	1	1	Duty cycle
nPattern	75	1	1	Size of USER_DEFINED_PATTERN
dataPattern	77	1	40	Data of USER_DEFINED_PATTERN
rateBitIndex0	78	2	2	Rate bit index in rate mask:
channel2	158	3	4	Frequency for secondary channel, e.g. 5500

Name	Code	Type	Length (bytes)	Description
				▪ 1 – RATE_1Mbps ▪ 2 – RATE_2Mbps_L ▪ 3 – RATE_2Mbps_S ▪ 4 – RATE_5_5Mbps_L ▪ 5 – RATE_5_5Mbps_S ▪ 6 – RATE_11Mbps_L ▪ 7 – RATE_11Mbps_S ▪ 8 – RATE_6Mbps ▪ 9 – RATE_9Mbps ▪ 10 – RATE_12Mbps ▪ 11 – RATE_18Mbps ▪ 12 – RATE_24Mbps ▪ 13 – RATE_36Mbps ▪ 14 – RATE_48Mbps ▪ 15 – RATE_54Mbps ▪ 16 – RATE_MCS_0_20 ▪ 17 – RATE_MCS_1_20 ▪ 18 – RATE_MCS_2_20 ▪ 19 – RATE_MCS_3_20 ▪ 20- – RATE_MCS_4_20 ▪ 21 – RATE_MCS_5_20 ▪ 22 – RATE_MCS_6_20 ▪ 23 – RATE_MCS_7_20 ▪ 24 – RATE_MCS_0_40 ▪ 25 – RATE_MCS_1_40 ▪ 26 – RATE_MCS_2_40 ▪ 27 – RATE_MCS_3_40 ▪ 28 – RATE_MCS_4_40 ▪ 29 – RATE_MCS_5_40 ▪ 30 – RATE_MCS_6_40 ▪ 31 – RATE_MCS_7_40 ▪ 32 – RATE_MCS_8_20 ▪ 33 – RATE_MCS_9_20 ▪ 34 – RATE_MCS_10_20 ▪ 35 – RATE_MCS_11_20 ▪ 36 – RATE_MCS_12_20 ▪ 37 – RATE_MCS_13_20 ▪ 38 – RATE_MCS_14_20 ▪ 39 – RATE_MCS_15_20 ▪ 40 – RATE_MCS_8_40 ▪ 41 – RATE_MCS_9_40 ▪ 42 – RATE_MCS_10_40 ▪ 43 – RATE_MCS_11_40 ▪ 44 – RATE_MCS_12_40 ▪ 45 – RATE_MCS_13_40 ▪ 46 – RATE_MCS_14_40 ▪ 47 – RATE_MCS_15_40 ▪ 48 – RATE_MCS_16_20 ▪ 49 – RATE_MCS_17_20 ▪ 50 – RATE_MCS_18_20

Name	Code	Type	Length (bytes)	Description
				▪ 51 – RATE_MCS_19_20 ▪ 52 – RATE_MCS_20_20 ▪ 53 – RATE_MCS_21_20 ▪ 54 – RATE_MCS_22_20 ▪ 55 – RATE_MCS_23_20 ▪ 56 – RATE_MCS_16_40 ▪ 57 – RATE_MCS_17_40 ▪ 58 – RATE_MCS_18_40 ▪ 59 – RATE_MCS_19_40 ▪ 60 – RATE_MCS_20_40 ▪ 61 – RATE_MCS_21_40 ▪ 62 – RATE_MCS_22_40 ▪ 63 – RATE_MCS_23_40 ▪ 64 – RATE_AC_MCS_0_20 ▪ 65 –RATE_AC_MCS_1_20 ▪ 66 – RATE_AC_MCS_2_20 ▪ 67 – RATE_AC_MCS_3_20 ▪ 68 – RATE_AC_MCS_4_20 ▪ 69 – RATE_AC_MCS_5_20 ▪ 70 – RATE_AC_MCS_6_20 ▪ 71 – RATE_AC_MCS_7_20 ▪ 72 – RATE_AC_MCS_8_20 ▪ 73 – RATE_AC_MCS_9_20 ▪ 76 – RATE_AC_MCS_0_40 ▪ 77 – RATE_AC_MCS_1_40 ▪ 78 – RATE_AC_MCS_2_40 ▪ 79 – RATE_AC_MCS_3_40 ▪ 80 – RATE_AC_MCS_4_40 ▪ 81 – RATE_AC_MCS_5_40 ▪ 82 – RATE_AC_MCS_6_40 ▪ 83 – RATE_AC_MCS_7_40 ▪ 84 – RATE_AC_MCS_8_40 ▪ 85 – RATE_AC_MCS_9_40 ▪ 88 – RATE_AC_MCS_0_80 ▪ 89 – RATE_AC_MCS_1_80 ▪ 90 – RATE_AC_MCS_2_80 ▪ 91 – RATE_AC_MCS_3_80 ▪ 92 – RATE_AC_MCS_4_80 ▪ 93 – RATE_AC_MCS_5_80 ▪ 94 – RATE_AC_MCS_6_80 ▪ 95 – RATE_AC_MCS_7_80 ▪ 96 – RATE_AC_MCS_8_80 ▪ 97 – RATE_AC_MCS_9_80 ▪ 100 – RATE_AC_MCS_0_20_2S ▪ 101 – RATE_AC_MCS_1_20_2S ▪ 102 – RATE_AC_MCS_2_20_2S ▪ 103 – RATE_AC_MCS_3_20_2S ▪ 104 – RATE_AC_MCS_4_20_2S ▪ 105 – RATE_AC_MCS_5_20_2S ▪ 106 – RATE_AC_MCS_6_20_2S

Name	Code	Type	Length (bytes)	Description
				▪ 107 – RATE_AC_MCS_7_20_2S ▪ 108 – RATE_AC_MCS_8_20_2S ▪ 109 – RATE_AC_MCS_9_20_2S ▪ 112 – RATE_AC_MCS_0_40_2S ▪ 113 – RATE_AC_MCS_1_40_2S ▪ 114 – RATE_AC_MCS_2_40_2S ▪ 115 – RATE_AC_MCS_3_40_2S ▪ 116 – RATE_AC_MCS_4_40_2S ▪ 117 – RATE_AC_MCS_5_40_2S ▪ 118 – RATE_AC_MCS_6_40_2S ▪ 119 – RATE_AC_MCS_7_40_2S ▪ 120 – RATE_AC_MCS_8_40_2S ▪ 121 – RATE_AC_MCS_9_40_2S ▪ 124 – RATE_AC_MCS_0_80_2S ▪ 125 – RATE_AC_MCS_1_80_2S ▪ 126 – RATE_AC_MCS_2_80_2S ▪ 127 – RATE_AC_MCS_3_80_2S ▪ 128 – RATE_AC_MCS_4_80_2S ▪ 129 – RATE_AC_MCS_5_80_2S ▪ 130 – RATE_AC_MCS_6_80_2S ▪ 131 – RATE_AC_MCS_7_80_2S ▪ 132 – RATE_AC_MCS_8_80_2S ▪ 133 – RATE_AC_MCS_9_80_2S ▪ 136 – RATE_AC_MCS_0_20_3S ▪ 137 – RATE_AC_MCS_1_20_3S ▪ 138 – RATE_AC_MCS_2_20_3S ▪ 139 – RATE_AC_MCS_3_20_3S ▪ 140 – RATE_AC_MCS_4_20_3S ▪ 141 – RATE_AC_MCS_5_20_3S ▪ 142 – RATE_AC_MCS_6_20_3S ▪ 143 – RATE_AC_MCS_7_20_3S ▪ 144 – RATE_AC_MCS_8_20_3S ▪ 145 – RATE_AC_MCS_9_20_3S ▪ 148 – RATE_AC_MCS_0_40_3S ▪ 149 – RATE_AC_MCS_1_40_3S ▪ 150 – RATE_AC_MCS_2_40_3S ▪ 151 – RATE_AC_MCS_3_40_3S ▪ 152 – RATE_AC_MCS_4_40_3S ▪ 153 – RATE_AC_MCS_5_40_3S ▪ 154 – RATE_AC_MCS_6_40_3S ▪ 155 – RATE_AC_MCS_7_40_3S ▪ 156 – RATE_AC_MCS_8_40_3S ▪ 157 – RATE_AC_MCS_9_40_3S ▪ 160 – RATE_AC_MCS_0_80_3S ▪ 161 – RATE_AC_MCS_1_80_3S ▪ 162 – RATE_AC_MCS_2_80_3S ▪ 163 – RATE_AC_MCS_3_80_3S ▪ 164 – RATE_AC_MCS_4_80_3S ▪ 165 – RATE_AC_MCS_5_80_3S

				▪ 166 – RATE_AC_MCS_6_80_3S ▪ 167 – RATE_AC_MCS_7_80_3S ▪ 168 – RATE_AC_MCS_8_80_3S ▪ 169 – RATE_AC_MCS_9_80_3S ▪ 192 – RATE_VHT20_NSS4_MCS0 ▪ 193 – RATE_VHT20_NSS4_MCS1 ▪ 194 – RATE_VHT20_NSS4_MCS2 ▪ 195 – RATE_VHT20_NSS4_MCS3 ▪ 196 – RATE_VHT20_NSS4_MCS4 ▪ 197 – RATE_VHT20_NSS4_MCS5 ▪ 198 – RATE_VHT20_NSS4_MCS6 ▪ 199 – RATE_VHT20_NSS4_MCS7 ▪ 200 – RATE_VHT20_NSS4_MCS8 ▪ 201 – RATE_VHT20_NSS4_MCS9 ▪ 202 – RATE_VHT40_NSS4_MCS0 ▪ 203 – RATE_VHT40_NSS4_MCS1 ▪ 204 – RATE_VHT40_NSS4_MCS2 ▪ 205 – RATE_VHT40_NSS4_MCS3 ▪ 206 – RATE_VHT40_NSS4_MCS4 ▪ 207 – RATE_VHT40_NSS4_MCS5 ▪ 208 – RATE_VHT40_NSS4_MCS6 ▪ 209 – RATE_VHT40_NSS4_MCS7 ▪ 210 – RATE_VHT40_NSS4_MCS8 ▪ 211 – RATE_VHT40_NSS4_MCS9 ▪ 212 – RATE_VHT80_NSS4_MCS0 ▪ 213 – RATE_VHT80_NSS4_MCS1 ▪ 214 – RATE_VHT80_NSS4_MCS2 ▪ 215 – RATE_VHT80_NSS4_MCS3 ▪ 216 – RATE_VHT80_NSS4_MCS4 ▪ 217 – RATE_VHT80_NSS4_MCS5 ▪ 218 – RATE_VHT80_NSS4_MCS6 ▪ 219 – RATE_VHT80_NSS4_MCS7 ▪ 220 – RATE_VHT80_NSS4_MCS8 ▪ 221 – RATE_VHT80_NSS4_MCS9 ▪ 224 - RATE_VHT160_NSS1_MCS0 ▪ 225 - RATE_VHT160_NSS1_MCS1, ▪ 226 - RATE_VHT160_NSS1_MCS2, ▪ 227 - RATE_VHT160_NSS1_MCS3, ▪ 228 - RATE_VHT160_NSS1_MCS4, ▪ 229 - RATE_VHT160_NSS1_MCS5, ▪ 230 - RATE_VHT160_NSS1_MCS6, ▪ 231 - RATE_VHT160_NSS1_MCS7, ▪ 232 - RATE_VHT160_NSS1_MCS8, ▪ 233 - RATE_VHT160_NSS1_MCS9, ▪ 234 - RATE_VHT160_NSS2_MCS0, ▪ 235 - RATE_VHT160_NSS2_MCS1, ▪ 236 - RATE_VHT160_NSS2_MCS2, ▪ 237 - RATE_VHT160_NSS2_MCS3, ▪ 238 - RATE_VHT160_NSS2_MCS4, ▪ 239 - RATE_VHT160_NSS2_MCS5, ▪ 240 - RATE_VHT160_NSS2_MCS6, ▪ 241 - RATE_VHT160_NSS2_MCS7,

Name	Code	Type	Length (bytes)	Description
				▪ 242 - RATE_VHT160_NSS2_MCS8, ▪ 243 - RATE_VHT160_NSS2_MCS9
txPower0	86	2	2	Tx power
pktLen0	94	2	2	Packet length
ir	110	3	4	—
gainIdx	144	2	2	Gain index
dacGain	145	5	2	DAC gain
paConfig	146	2	2	PA setting

3.4 Example of TLV Tx Command

```
05 00 00 00 01 00 00 00 00 00 00 00 F8 00 00 00 62 F4 00 00 00 00 00 00 00
00 00 00 01 00 00 00 0F 00 00 00 00 00 00 00 03 00 00 00 00 00 00 00 71 09
00 00 44 00 00 00 03 00 00 00 00 00 00 00 00 00 00 00 46 00 00 00 01 00 00
00 50 55 55 55 55 05 00 00 47 00 00 00 01 00 00 00 20 22 22 22 22 02 00 00
48 00 00 00 01 00 00 00 FF FF FF FF FF FF 00 00 41 00 00 00 03 00 00 00 00
00 00 00 01 00 00 00 42 00 00 00 03 00 00 00 00 00 00 00 18 00 00 00 3C 00
00 00 03 00 00 00 00 00 00 00 00 00 00 00 01 00 00 00 03 00 00 00 00 00 00
00 03 00 00 00 37 00 00 00 03 00 00 00 00 00 00 00 01 00 00 00 5E 00 00 00
02 00 00 00 DC 05 00 00 00 00 00 00 3D 00 00 00 03 00 00 00 00 00 00 00 01
00 00 00 6E 00 00 00 03 00 00 00 00 00 00 00 01 00 00 00 56 00 00 00 02 00
00 00 9B FF 00 00 00 00 00 00 92 00 00 00 02 00 00 00 00 00 00 00 00 00 00
00
```

The stream header is highlighted in green. The values of its fields are:

- ID = 5
- Version = 1
- Header = 0
- Length = 0xf8 = 248 bytes
- Checksum = 0xF462

The checksum function is:

```
A_UINT16 computeChecksumOnly(A_UINT16 *pStream, A_UINT16 length)
{
    A_UINT16 sum = 0, i;
    for (i = 0; i < length; i++) { sum ^= *pStream ++; }
    sum = 0xffff ^ sum;
    return(sum);
}
```

Where:

- pStream – Indicates the pointer to the stream
- Length – Indicates the length of the stream/2

The command header is highlighted in red. The opcode is _OP_TX and there are 15 parameters.

Table 3-4 lists the parameters in the example (highlighted in orange).

Table 3-4 Parameters in the TLV stream

Parameter field	Name	Code	Type	Length	Value
00 00 00 00 03 00 00 00 00 00 00 00 71 09 00 00	channel	0	3	4	2417
44 00 00 00 03 00 00 00 00 00 00 00 00 00 00 00	broadcast	68	3	4	0
46 00 00 00 01 00 00 00 50 55 55 55 55 05 00 00	bssid	70	1	6	50 55 55 55 55 05
47 00 00 00 01 00 00 00 20 22 22 22 22 02 00 00	txStation	71	1	6	20 22 22 22 22 02
48 00 00 00 01 00 00 00 FF FF FF FF FF FF 00 00	rxStation	72	1	6	FF FF FF FF FF FF
41 00 00 00 03 00 00 00 00 00 00 00 01 00 00 00	tpcm	65	3	4	TPC_FORCED_GAIN
42 00 00 00 03 00 00 00 00 00 00 00 18 00 00 00	flags	66	3	4	Enable Tx status per rate | Enable process rate in order
3C 00 00 00 03 00 00 00 00 00 00 00 00 00 00 00	wlanMode	60	3	4	TCMD_WLAN_MODE_NOHT
01 00 00 00 03 00 00 00 00 00 00 00 03 00 00 00	txMode	1	3	4	TX99
37 00 00 00 03 00 00 00 00 00 00 00 01 00 00 00	aifsn	55	3	4	1
5E 00 00 00 02 00 00 00 DC 05 00 00 00 00 00 00	pktLen0	94	2	2	1500
3D 00 00 00 03 00 00 00 00 00 00 00 01 00 00 00	txChain0	61	3	4	0x00000001
6E 00 00 00 03 00 00 00 00 00 00 00 01 00 00 00	ir	110	3	4	0x00000001
56 00 00 00 02 00 00 00 9B FF 00 00 00 00 00 00	txPower0	86	2	2	0xFF9B (invalid)
92 00 00 00 02 00 00 00 00 00 00 00 00 00 00 00	paConfig	146	2	2	00 00

For the parameters not presented in the stream, the target will use the pre-defined default values for those parameters.

3.5 Rx Command (_OP_RX = 2)

Table 3-5 Parameters in Rx command

Name	Code	Type	Length (bytes)	Description
channel	0	3	4	Frequency, e.g. 2412
rxMode	1	3	4	Receive mode: ▪ 0 – TCMD_CONT_RX_PROMIS ▪ 1 – TCMD_CONT_RX_FILTER ▪ 2 – TCMD_CONT_RX_REPORT ▪ 3 – TCMD_CONT_RX_SETMAC ▪ 4 – TCMD_CONT_RX_SET_ANT_SWITCH_TABLE
enANI	2	3	4	Enable ANI: 0 or 1
antenna	3	3	4	Antenna: 0 or 1

Name	Code	Type	Length (bytes)	Description
wlanMode	4	3	4	WLAN mode: ▪ 0 – TCMD_WLAN_MODE_NOHT ▪ 1 – TCMD_WLAN_MODE_HT20 ▪ 2 – TCMD_WLAN_MODE_HT40PLUS ▪ 3 – TCMD_WLAN_MODE_HT40MINUS ▪ 4 – TCMD_WLAN_MODE_CCK ▪ 5 – TCMD_WLAN_MODE_VHT20 ▪ 6 – TCMD_WLAN_MODE_VHT40PLUS ▪ 7 – TCMD_WLAN_MODE_VHT40MINUS ▪ 8 – TCMD_WLAN_MODE_VHT80_0 ▪ 9 – TCMD_WLAN_MODE_VHT80_1 ▪ 10 – TCMD_WLAN_MODE_VHT80_2 ▪ 11 – TCMD_WLAN_MODE_VHT80_3 ▪ 12 - TCMD_WLAN_MODE_VHT80P80 ▪ 13 - TCMD_WLAN_MODE_VHT160 ▪ 21 - TCMD_WLAN_MODE_VHT80P80_0 ▪ 22 - TCMD_WLAN_MODE_VHT80P80_1 ▪ 23 - TCMD_WLAN_MODE_VHT80P80_2 ▪ 24 - TCMD_WLAN_MODE_VHT80P80_3 ▪ 25 - TCMD_WLAN_MODE_VHT80P80_4 ▪ 26 - TCMD_WLAN_MODE_VHT80P80_5 ▪ 27 - TCMD_WLAN_MODE_VHT80P80_6 ▪ 28 - TCMD_WLAN_MODE_VHT80P80_7 ▪ 29 - TCMD_WLAN_MODE_VHT160_0 ▪ 30 - TCMD_WLAN_MODE_VHT160_1 ▪ 31 - TCMD_WLAN_MODE_VHT160_2 ▪ 32 - TCMD_WLAN_MODE_VHT160_3 ▪ 33 - TCMD_WLAN_MODE_VHT160_4 ▪ 34 - TCMD_WLAN_MODE_VHT160_5 ▪ 35 - TCMD_WLAN_MODE_VHT160_6 ▪ 36 - TCMD_WLAN_MODE_VHT160_7
rxChain	5	3	4	Rx chain mask
expectedPkts	6	3	4	Number of packets expected to receive
bc	8	3	4	Enable broadcast (0 or 1)
bandwidth	9	3	4	Bandwidth: ▪ 50 – HALF_SPEED_MODE ▪ 51 – QUARTER_SPEED_MODE ▪ Otherwise 0
addr	13	1	6	Rx station ID
bssid	14	1	6	BSSID

Name	Code	Type	Length (bytes)	Description
flags	20	3	4	Miscellaneous flags: Bit: 7 6 5 4 3 2 1 0 x: Unused x : Unused x : Unused x : Unused x : Enable process rate in order (0 or 1) x : Enable Rx status per rate (0 or 1) x : Unused x : Unused
rateMask0	21	3	4	Rate mask: ▪ 1Mbps – Bit0 ▪ 2Mbps_L – Bit1 ▪ 2Mbps_S – Bit2 ▪ 5_5Mbps_L – Bit3 ▪ 5_5Mbps_S – Bit4 ▪ 11Mbps_L – Bit5 ▪ 11Mbps_S – Bit6 ▪ 6Mbps – Bit8 ▪ 9Mbps – Bit9 ▪ 12Mbps – Bit10 ▪ 18Mbps – Bit11 ▪ 24Mbps – Bit12 ▪ 36Mbps – Bit13 ▪ 48Mbps – Bit14 ▪ 54Mbps – Bit15 ▪ MCS_0_20 – Bit16 ▪ MCS_1_20 – Bit17 ▪ MCS_2_20 – Bit18 ▪ MCS_3_20 – Bit19 ▪ MCS_4_20 – Bit20 ▪ MCS_5_20 – Bit21 ▪ MCS_6_20 – Bit22 ▪ MCS_7_20 – Bit23 ▪ MCS_0_40 – Bit24 ▪ MCS_1_40 – Bit25 ▪ MCS_2_40 – Bit26 ▪ MCS_3_40 – Bit27 ▪ MCS_4_40 – Bit28 ▪ MCS_5_40 – Bit29 ▪ MCS_6_40 – Bit30 ▪ MCS_7_40 – Bit31

Name	Code	Type	Length (bytes)	Description
rateMask1	22	3	4	Rate mask: ▪ MCS_8_20 – Bit0 ▪ MCS_9_20 – Bit1 ▪ MCS_10_20 – Bit2 ▪ MCS_11_20 – Bit3 ▪ MCS_12_20 – Bit4 ▪ MCS_13_20 – Bit5 ▪ MCS_14_20 – Bit6 ▪ MCS_15_20 – Bit7 ▪ MCS_8_40 – Bit8 ▪ MCS_9_40 – Bit9 ▪ MCS_10_40 – Bit10 ▪ MCS_11_40 – Bit11 ▪ MCS_12_40 – Bit12 ▪ MCS_13_40 – Bit13 ▪ MCS_14_40 – Bit14 ▪ MCS_15_40 – Bit15 ▪ MCS_16_20 – Bit16 ▪ MCS_17_20 – Bit17 ▪ MCS_18_20 – Bit18 ▪ MCS_19_20 – Bit19 ▪ MCS_20_20 – Bit20 ▪ MCS_21_20 – Bit21 ▪ MCS_22_20 – Bit22 ▪ MCS_23_20 – Bit23 ▪ MCS_16_40 – Bit24 ▪ MCS_17_40 – Bit25 ▪ MCS_18_40 – Bit26 ▪ MCS_19_40 – Bit27 ▪ MCS_20_40 – Bit28 ▪ MCS_21_40 – Bit29 ▪ MCS_22_40 – Bit30 ▪ MCS_23_40 – Bit31

Name	Code	Type	Length (bytes)	Description
rateMask2	23	3	4	Rate Mask: ▪ AC_MCS_0_20 – Bit0 ▪ AC_MCS_1_20 – Bit1 ▪ AC_MCS_2_20 – Bit2 ▪ AC_MCS_3_20 – Bit3 ▪ AC_MCS_4_20 – Bit4 ▪ AC_MCS_5_20 – Bit5 ▪ AC_MCS_6_20 – Bit6 ▪ AC_MCS_7_20 – Bit7 ▪ AC_MCS_8_20 – Bit8 ▪ AC_MCS_9_20 – Bit9 ▪ AC_MCS_0_40 – Bit12 ▪ AC_MCS_1_40 – Bit13 ▪ AC_MCS_2_40 – Bit14 ▪ AC_MCS_3_40 – Bit15 ▪ AC_MCS_4_40 – Bit16 ▪ AC_MCS_5_40 – Bit17 ▪ AC_MCS_6_40 – Bit18 ▪ AC_MCS_7_40 – Bit19 ▪ AC_MCS_8_40 – Bit20 ▪ AC_MCS_9_40 – Bit21 ▪ AC_MCS_0_80 – Bit24 ▪ AC_MCS_1_80 – Bit25 ▪ AC_MCS_2_80 – Bit26 ▪ AC_MCS_3_80 – Bit27 ▪ AC_MCS_4_80 – Bit28 ▪ AC_MCS_5_80 – Bit29 ▪ AC_MCS_6_80 – Bit30 ▪ AC_MCS_7_80 – Bit31

Name	Code	Type	Length (bytes)	Description
rateMask3	24	3	4	Rate mask: ▪ AC_MCS_8_80 – Bit0 ▪ AC_MCS_9_80 – Bit1 ▪ AC_MCS_0_20_2S – Bit4 ▪ AC_MCS_1_20_2S – Bit5 ▪ AC_MCS_2_20_2S – Bit6 ▪ AC_MCS_3_20_2S – Bit7 ▪ AC_MCS_4_20_2S – Bit8 ▪ AC_MCS_5_20_2S – Bit9 ▪ AC_MCS_6_20_2S – Bit10 ▪ AC_MCS_7_20_2S – Bit11 ▪ AC_MCS_8_20_2S – Bit12 ▪ AC_MCS_9_20_2S – Bit13 ▪ AC_MCS_0_40_2S – Bit16 ▪ AC_MCS_1_40_2S – Bit17 ▪ AC_MCS_2_40_2S – Bit18 ▪ AC_MCS_3_40_2S – Bit19 ▪ AC_MCS_4_40_2S – Bit20 ▪ AC_MCS_5_40_2S – Bit21 ▪ AC_MCS_6_40_2S – Bit22 ▪ AC_MCS_7_40_2S – Bit23 ▪ AC_MCS_8_40_2S – Bit24 ▪ AC_MCS_9_40_2S – Bit25 ▪ AC_MCS_0_80_2S – Bit28 ▪ AC_MCS_1_80_2S – Bit29 ▪ AC_MCS_2_80_2S – Bit30 ▪ AC_MCS_3_80_2S – Bit31

Name	Code	Type	Length (bytes)	Description
rateMask4	25	3	4	Rate mask: ▪ AC_MCS_4_80_2S – Bit0 ▪ AC_MCS_5_80_2S – Bit1 ▪ AC_MCS_6_80_2S – Bit2 ▪ AC_MCS_7_80_2S – Bit3 ▪ AC_MCS_8_80_2S – Bit4 ▪ AC_MCS_9_80_2S – Bit5 ▪ AC_MCS_0_20_3S – Bit8 ▪ AC_MCS_1_20_3S – Bit9 ▪ AC_MCS_2_20_3S – Bit10 ▪ AC_MCS_3_20_3S – Bit11 ▪ AC_MCS_4_20_3S – Bit12 ▪ AC_MCS_5_20_3S – Bit13 ▪ AC_MCS_6_20_3S – Bit14 ▪ AC_MCS_7_20_3S – Bit15 ▪ AC_MCS_8_20_3S – Bit16 ▪ AC_MCS_9_20_3S – Bit17 ▪ AC_MCS_0_40_3S – Bit20 ▪ AC_MCS_1_40_3S – Bit21 ▪ AC_MCS_2_40_3S – Bit22 ▪ AC_MCS_3_40_3S – Bit23 ▪ AC_MCS_4_40_3S – Bit24 ▪ AC_MCS_5_40_3S – Bit25 ▪ AC_MCS_6_40_3S – Bit26 ▪ AC_MCS_7_40_3S – Bit27 ▪ AC_MCS_8_40_3S – Bit28 ▪ AC_MCS_9_40_3S – Bit29
rateMask5	26	3	4	Rate mask: ▪ AC_MCS_0_80_3S – Bit0 ▪ AC_MCS_1_80_3S – Bit1 ▪ AC_MCS_2_80_3S – Bit2 ▪ AC_MCS_3_80_3S – Bit3 ▪ AC_MCS_4_80_3S – Bit4 ▪ AC_MCS_5_80_3S – Bit5 ▪ AC_MCS_6_80_3S – Bit6 ▪ AC_MCS_7_80_3S – Bit7 ▪ AC_MCS_8_80_3S – Bit8 ▪ AC_MCS_9_80_3S – Bit9

Name	Code	Type	Length (bytes)	Description
rateMask6	29	3	4	▪ RATE_VHT20_NSS4_MCS0 – Bit0 ▪ RATE_VHT20_NSS4_MCS1– Bit1 ▪ RATE_VHT20_NSS4_MCS2 – Bit2 ▪ RATE_VHT20_NSS4_MCS3 – Bit3 ▪ RATE_VHT20_NSS4_MCS4 – Bit4 ▪ RATE_VHT20_NSS4_MCS5 – Bit5 ▪ RATE_VHT20_NSS4_MCS6 – Bit6 ▪ RATE_VHT20_NSS4_MCS7 – Bit7 ▪ RATE_VHT20_NSS4_MCS8 – Bit8 ▪ RATE_VHT20_NSS4_MCS9 – Bit9 ▪ RATE_VHT40_NSS4_MCS0 – Bit10 ▪ RATE_VHT40_NSS4_MCS1– Bit11 ▪ RATE_VHT40_NSS4_MCS2 – Bit12 ▪ RATE_VHT40_NSS4_MCS3– Bit13 ▪ RATE_VHT40_NSS4_MCS4 – Bit14 ▪ RATE_VHT40_NSS4_MCS5 – Bit15 ▪ RATE_VHT40_NSS4_MCS6 – Bit16 ▪ RATE_VHT40_NSS4_MCS7 – Bit17 ▪ RATE_VHT40_NSS4_MCS8 – Bit18 ▪ RATE_VHT40_NSS4_MCS9 – Bit19 ▪ RATE_VHT80_NSS4_MCS0 – Bit20 ▪ RATE_VHT80_NSS4_MCS1 – Bit21 ▪ RATE_VHT80_NSS4_MCS2 – Bit22 ▪ RATE_VHT80_NSS4_MCS3 – Bit23 ▪ RATE_VHT80_NSS4_MCS4 – Bit24 ▪ RATE_VHT80_NSS4_MCS5 – Bit25 ▪ RATE_VHT80_NSS4_MCS6 – Bit26 ▪ RATE_VHT80_NSS4_MCS7 – Bit27 ▪ RATE_VHT80_NSS4_MCS8 – Bit28 ▪ RATE_VHT80_NSS4_MCS9 – Bit29

Name	Code	Type	Length (bytes)	Description
rateMask7	30	3	4	▪ RATE_VHT160_NSS1_MCS0 – Bit0 ▪ RATE_VHT160_NSS1_MCS1 – Bit1 ▪ RATE_VHT160_NSS1_MCS2 – Bit2 ▪ RATE_VHT160_NSS1_MCS3 – Bit3 ▪ RATE_VHT160_NSS1_MCS4 – Bit4 ▪ RATE_VHT160_NSS1_MCS5– Bit5 ▪ RATE_VHT160_NSS1_MCS6– Bit6 ▪ RATE_VHT160_NSS1_MCS7– Bit7 ▪ RATE_VHT160_NSS1_MCS8– Bit8 ▪ RATE_VHT160_NSS1_MCS9 --Bit9 ▪ RATE_VHT160_NSS2_MCS0 – Bit10 ▪ RATE_VHT160_NSS2_MCS1 – Bit11 ▪ RATE_VHT160_NSS2_MCS2 -- Bit12 ▪ RATE_VHT160_NSS2_MCS3 – Bit13 ▪ RATE_VHT160_NSS2_MCS4– Bit14 ▪ RATE_VHT160_NSS2_MCS5– Bit15 ▪ RATE_VHT160_NSS2_MCS6– Bit16 ▪ RATE_VHT160_NSS2_MCS7– Bit17 ▪ RATE_VHT160_NSS2_MCS8– Bit18 ▪ RATE_VHT160_NSS2_MCS9 –Bit19
channel2	35	3	4	▪ Frequency for secondary channel, e.g. 5500

3.6 Sleep Command (_OP_PM = 5)

Table 3-6 Parameters in sleep command

Name	Code	Type	Length (in bytes)	Description
mode	0	3	4	Sleep mode: ▪ 1 – TCMD_PM_WAKEUP ▪ 2 – TCMD_PM_SLEEP ▪ 3 – TCMD_PM_DEEPSLEEP

3.7 Tx/Rx Response (_OP_GENERIC_RSP = 6)

This response is returned from target when a Tx or Rx command has been received.

Table 3-7 Parameters in Tx/Rx response

Name	Code	Type	Length (in bytes)	Description
status	0	3	4	Status – 0 if OK; error otherwise

3.8 Generic Command (_OP_GENERIC_NART_CMD = 8)

Generic command is used to carry other commands besides Tx, Rx, and Sleep commands.

Table 3-8 Parameters in generic command

Name	Code	Type	Length (in bytes)	Description
commandId	0	3	4	Command ID: ▪ 0 – INIT_F2_CMD_ID ▪ 2 – MEM_WRITE_CMD_ID ▪ 3 – MEM_READ_CMD_ID ▪ 4 – REG_READ_CMD_ID ▪ 5 – REG_WRITE_CMD_ID ▪ 6 – CFG_READ_CMD_ID ▪ 10 – MEM_WRITE_BLOCK_CMD_ID ▪ 11 – MEM_READ_BLOCK_CMD_ID ▪ 30 – M_RESET_DEVICE_CMD_ID ▪ 174 – OTP_WRITE_CMD_ID ▪ 175 – OTP_READ_CMD_ID ▪ 176 – OTP_RESET_CMD_ID ▪ 177 – EFUSE_READ_CMD_ID ▪ 183 – M_RX_DATA_STOP_CMD_ID ▪ 184 – M_RX_DATA_STATUS_CMD_ID ▪ 186 – M_TX_DATA_STOP_CMD_ID ▪ 187 – M_TX_DATA_STATUS_CMD_ID ▪ 188 – EFUSE_WRITE_CMD_ID ▪ 189 – M_EEPROM_WRITE_ITEMS_CMD_ID ▪ 190 – M_STICKY_WRITE_CMD_ID ▪ 191 – M_STICKY_CLEAR_CMD_ID ▪ 192 – OTP_LOAD_CMD_ID
param1	1	3	4	Parameter 1
param2	2	3	4	Parameter 2
param3	3	3	4	Parameter 3
data	4	1	1024	Data if any

The param1, param2, param3 and data fields of each subcommand will be described in next chapter.

3.9 Generic Response (_OP_GENERIC_NART_RSP = 9)

Table 3-9 Parameters in generic response

Name	Code	Type	Length (in bytes)	Description
commandId	0	3	4	See Table 3-8
status	0	3	4	Status – 0 if OK; error otherwise
length	0	3	4	Length of data
data	4	1	1024	Data if any

The length and data fields of the responses will be described in next chapter.

3.10 Tx Status Response (_OP_TX_STATUS = 10)

This response is returned from the target when a generic command OP_GENERIC_NART_CMD with command ID M_TX_DATA_STOP_CMD_ID or M_TX_DATA_STATUS_CMD_ID has been received.

The parameter numOfReports indicates the number of status reports returned from target. Each report states the status of the transmission rate specified in rateBitIndex0 in the Tx command. The report contains the parameters from totalPackets to dacGain in Table 3-10.

Table 3-10 Parameters in Tx status response

Name	Code	Type	Length (in bytes)	Description
numOfReports	0	3	4	Number of reports. Each report contains all the parameters below.
totalPackets	1	3	4	Total packets received
goodPackets	2	3	4	Number of good packets
underruns	3	3	4	Number of under-run packets
otherError	4	3	4	Number of other error packets
excessRetries	5	3	4	Number of excess retries
rateBit	6	3	4	Rate bit (see rate bit index in rate mask in Table 3-3)
shortRetry	7	6	4	Short retry
longRetry	8	6	4	Long retry
startTime	9	3	4	Time to start transmitting
endTime	10	3	4	Time to end transmitting
byteCount	11	3	4	Number of bytes counted
dontCount	12	3	4	Number of bytes not counted
rssi	13	6	4	RSSI histogram for good packets
rssic0	14	6	4	Chain0 RSSIC histogram for good packets
rssic1	15	6	4	Chain1 RSSIC histogram for good packets
rssic2	16	6	4	Chain2 RSSIC histogram for good packets
rssie0	17	6	4	Chain0 RSSIE histogram for good packets
rssie1	18	6	4	Chain1 RSSIE histogram for good packets
rssie2	19	6	4	Chain2 RSSIE histogram for good packets
thermCal	20	3	4	Thermal value for calibration
pdadc	21	3	4	Power detector ADC
paCfg	22	3	4	PA configuration
gainIdx	23	6	4	Gain index
dacGain	24	6	4	DAC gain

3.11 Rx Status Response (_OP_RX_STATUS = 11)

This response is returned from the target when a generic command OP_GENERIC_NART_CMD with command ID M_RX_DATA_STOP_CMD_ID or M_RX_DATA_STATUS_CMD_ID has been received.

The parameter numOfReports indicates the number of status reports returned from the target. Each report states the status of each receiving rate specified in rateMask0 to rateMask5 in the Rx command. The report contains the parameters from totalPackets to badevm2 in Table 3-11.

Table 3-11 Parameters in Rx status response

Name	Code	Type	Length (in bytes)	Description
numOfReports	0	3	4	Number of reports. Each report contains all the parameters below.
totalPackets	1	3	4	Total packets received
goodPackets	2	3	4	Number of good packets
otherError	3	3	4	Number of other error packets
crcPackets	4	3	4	Number of CRC error packets
decrypErrors	5	3	4	Number of decrypted packets
rateBit	6	3	4	Rate bit (see rate bit index in rate mask in Table 3-10)
startTime	7	3	4	Time to start receiving
endTime	8	3	4	Time to end receiving
byteCount	9	3	4	Number of bytes counted
dontCount	10	3	4	Number of bytes not counted
rssi	11	6	4	RSSI histogram for good packets
rssic0	12	6	4	Chain0 RSSIC histogram for good packets
rssic1	13	6	4	Chain1 RSSIC histogram for good packets
rssic2	14	6	4	Chain2 RSSIC histogram for good packets
rssie0	15	6	4	Chain0 RSSIE histogram for good packets
rssie1	16	6	4	Chain1 RSSIE histogram for good packets
rssie2	17	6	4	Chain2 RSSIE histogram for good packets
evm0	18	6	4	Chain0 EVM histogram for good packets
evm1	19	6	4	Chain1 EVM histogram for good packets
evm2	20	6	4	Chain2 EVM histogram for good packets
badrssi	21	6	4	RSSI histogram for bad packets
badrssic0	22	6	4	Chain0 RSSIC histogram for bad packets
badrssic1	23	6	4	Chain1 RSSIC histogram for bad packets
badrssic2	24	6	4	Chain2 RSSIC histogram for bad packets
badrssie0	25	6	4	Chain0 RSSIE histogram for bad packets
badrssie1	26	6	4	Chain1 RSSIE histogram for bad packets
badrssie2	27	6	4	Chain2 RSSIE histogram for bad packets
badevm0	28	6	4	Chain0 EVM histogram for bad packets
badevm1	29	6	4	Chain1 EVM histogram for bad packets
badevm2	30	6	4	Chain2 EVM histogram for bad packets
noisefloor0	37	6	4	Chain0 noisefloor calibration value
noisefloor1	38	6	4	Chain1 noisefloor calibration value
noisefloor2	39	6	4	Chain2 noisefloor calibration value
noisefloor3	40	6	4	Chain3 noisefloor calibration value

4 Generic NART Sub-Commands/Responses

4.1 INIT_F2_CMD_ID (0)

This command gets the device information.

Table 4-1 INIT_F2_CMD_ID command request

Field	Size (bytes)	Description
param1	4	Device ID

Table 4-2 INIT_F2_CMD_ID response

Field	Size (bytes)	Description
length	4	Size of the returned data = 20 bytes
data	20	data[3:0] – Device number data[7:4] – Memory physical address data[11:8] – Memory size data[15:12] – Target ID data[19:16] – Board data address

4.2 MEM_WRITE_CMD_ID (2)

Despite the name, this command only writes 4 bytes of memory.

Table 4-3 MEM_WRITE_CMD_ID command request

Field	Size (bytes)	Description
param1	4	Address
param2	4	Value
param3	4	Size = 32 (in bits which is 4 bytes)

Table 4-4 MEM_WRITE_CMD_ID response

Field	Size (bytes)	Description
length	4	Size of the returned data = 0 bytes

4.3 MEM_READ_CMD_ID (3)

Despite the name, this command only reads 4 bytes of memory.

Table 4-5 MEM_READ_CMD_ID command request

Field	Size (bytes)	Description
param1	4	Address
param2	4	Size = 32 (in bits which is 4 bytes)

Table 4-6 MEM_READ_CMD_ID response

Field	Size (bytes)	Description
length	4	Size of the returned data = 4 bytes
data	4	data[3:0] = value

4.4 REG_READ_CMD_ID (4)

This command reads a register.

Table 4-7 REG_READ_CMD_ID command request

Field	Size (bytes)	Description
param1	4	Address

Table 4-8 REG_READ_CMD_ID response

Field	Size (bytes)	Description
length	4	Size of the returned data = 4 bytes
data	4	data[3:0] = value

4.5 REG_WRITE_CMD_ID (5)

This command writes to a register.

Table 4-9 REG_WRITE_CMD_ID command request

Field	Size (bytes)	Description
param1	4	Address
param2	4	Value

Table 4-10 REG_WRITE_CMD_ID response

Field	Size (bytes)	Description
length	4	Size of the returned data = 0 bytes

4.6 CFG_READ_CMD_ID (6)

This command reads 4 bytes of configuration area.

Table 4-11 CFG_READ_CMD_ID command request

Field	Size (bytes)	Description
param1	4	Offset
param2	4	Size = 32 (in bits which is 4 bytes)

Table 4-12 CFG_READ_CMD_ID response

Field	Size (bytes)	Description
length	4	Size of the returned data = 4 bytes
data	4	data[3:0] = value

4.7 MEM_WRITE_BLOCK_CMD_ID (10)

This command writes to a block of memory.

Table 4-13 MEM_WRITE_BLOCK_CMD_ID command request

Field	Size (bytes)	Description
param1	4	Size of data
param2	4	Address
data	4	Values

Table 4-14 MEM_WRITE_BLOCK_CMD_ID response

Field	Size (bytes)	Description
length	4	Size of the returned data = 4 bytes
data	4	data[3:0] = address

4.8 MEM_READ_BLOCK_CMD_ID (11)

This command reads a block of memory.

Table 4-15 MEM_READ_BLOCK_CMD_ID command request

Field	Size (bytes)	Description
param1	4	Address
param2	4	Size to read

Table 4-16 MEM_READ_BLOCK_CMD_ID response

Field	Size (bytes)	Description
length	4	Size of the returned data = n bytes
data	n	data[*n*-1:0] = data

4.9 M_RESET_DEVICE_CMD_ID (30)

This command resets the device.

Table 4-17 M_RESET_DEVICE_CMD_ID command request

Field	Size (bytes)	Description
param1	4	Frequency, e.g. 2412
param2	4	WLAN mode: ▪ 0 – TCMD_WLAN_MODE_NOHT ▪ 1 – TCMD_WLAN_MODE_HT20 ▪ 2 – TCMD_WLAN_MODE_HT40PLUS ▪ 3 – TCMD_WLAN_MODE_HT40MINUS ▪ 4 – TCMD_WLAN_MODE_CCK ▪ 5 – TCMD_WLAN_MODE_VHT20 ▪ 6 – TCMD_WLAN_MODE_VHT40PLUS ▪ 7 – TCMD_WLAN_MODE_VHT40MINUS ▪ 8 – TCMD_WLAN_MODE_VHT80_0 ▪ 9 – TCMD_WLAN_MODE_VHT80_1 ▪ 10 – TCMD_WLAN_MODE_VHT80_2 ▪ 11 – TCMD_WLAN_MODE_VHT80_3

Table 4-18 M_RESET_DEVICE_CMD_ID response

Field	Size (bytes)	Description
length	4	Size of the returned data = 0 bytes

4.10 OTP_WRITE_CMD_ID (174)

This command writes an OTP stream to OTP.

Table 4-19 OTP_WRITE_CMD_ID command request

Field	Size (bytes)	Description
param1	4	Size of OTP stream = n bytes
data	n	data[*n*-1:0] = data

Table 4-20 OTP_WRITE_CMD_ID response

Field	Size (bytes)	Description
length	4	Size of the returned data = 0 bytes

4.11 OTP_READ_CMD_ID (175)

This command reads an OTP stream from OTP.

Table 4-21 OTP_READ_CMD_ID command

Field	Size (bytes)	Description
param1	4	Max size to read

Table 4-22 OTP_READ_CMD_ID response

Field	Size (bytes)	Description
length	4	Size of the returned data = n bytes
data	n	data[3:0] = size of OTP stream = s bytes data[s+3:4] = OTP stream data[*n*-1:s+4] = padding if any

4.12 OTP_RESET_CMD_ID (176)

This command resets the OTP before reading or writing OTP stream.

Table 4-23 OTP_RESET_CMD_ID command

Field	Size (bytes)	Description
param1	4	Reset ID: ▪ 1 – OTPSTREAM_READ_APP ▪ 2 – OTPSTREAM_WRITE_APP

Table 4-24 OTP_RESET_CMD_ID response

Field	Size (bytes)	Description
length	4	Size of the returned data = 0 bytes

4.13 EFUSE_READ_CMD_ID (177)

This command reads OTP.

Table 4-25 EFUSE_READ_CMD_ID command

Field	Size (bytes)	Description
param1	4	Expected read size, max = 255 (0xFF) bytes
param2	4	Start offset

Table 4-26 EFUSE_READ_CMD_ID response

Field	Size (bytes)	Description
length	4	Size of the returned data = n bytes
data	20	data[0] = (*n*-1) data[*n*-1:1] = OTP data

4.14 M_RX_DATA_STOP_CMD_ID (183)

This command stops the receiving.

The M_RX_DATA_STOP_CMD_ID command does not take any argument.

See 3.11 for the response.

4.15 M_RX_DATA_STATUS_CMD_ID (184)

This command queries the receiving status.

The M_RX_DATA_STATUS_CMD_ID command does not take any argument.

See 3.11 for the response.

4.16 M_TX_DATA_STOP_CMD_ID (186)

This command stops the transmission.

Table 4-27 M_TX_DATA_STOP_CMD_ID command request

Field	Size (bytes)	Description
param1	4	Report needed indicator (0: not needed, 1: needed)

See 3.10 for the response.

4.17 M_TX_DATA_STATUS_CMD_ID (187)

This command queries the transmission status.

Table 4-28 M_TX_DATA_STATUS_CMD_ID command request

Field	Size (bytes)	Description
param1	4	Stop transmitting indicator (0: continue, 1: stop)

See 3.10 for the response.

4.18 EFUSE_WRITE_CMD_ID (188)

This command writes raw data to OTP.

Table 4-29 EFUSE_WRITE_CMD_ID command

Field	Size (bytes)	Description
param1	4	Size of data to write = *n* bytes
param2	4	Start OTP offset
data	4	data [*n*-1:0] = data

Table 4-30 EFUSE_WRITE_CMD_ID response

Field	Size (bytes)	Description
length	4	Size of the returned data = 0 bytes

4.19 M_EEPROM_WRITE_ITEMS_CMD_ID (189)

This command writes raw data to board data in memory.

Table 4-31 M_EEPROM_WRITE_ITEMS_CMD_ID command

Field	Size (bytes)	Description
param1	4	Number of items
data	4	Each item contains: ▪ 2-byte offset ▪ 1-byte size ▪ Size bytes of item data

Table 4-32 M_EEPROM_WRITE_ITEMS_CMD_ID response

Field	Size (bytes)	Description
length	4	Size of the returned data = 0 bytes

4.20 M_STICKY_WRITE_CMD_ID (190)

This command sends sticky writes to target.

Table 4-33 M_STICKY_WRITE_CMD_ID command

Field	Size (bytes)	Description
param1	4	Size of data field = n bytes
param2	4	Number of sticky writes
data	4	data[3:0] = number of sticky writes data[*n*-1:4] = sticky write items Each sticky write item defined as: ▪ 1st 4 bytes = mode/address: ▫ Bit 0:19 – register address ▫ Bit 20:23 – mode: – OTP_CONFIG_MODE_COMMON = 1 – OTP_CONFIG_MODE_2MODAL = 2 – OTP_CONFIG_MODE_5MODAL = 3 ▫ Bit 24 – applied pre/post of HW calibration: 0 = pre, 1 = post ▪ 2nd 4 bytes = register mask ▫ If the mode is 1, the next 4-byte would be the value. ▫ If the mode is 2, the next 4-byte would be the value for 2G, then the one after the next would be the value for 5G. ▫ If the mode is 3, the next five 4-byte values would be the values for 2G_VHT20, 2G_VHT40, 5G_VHT20, 5G_VHT40 and 5G_VHT80 respectively. Then, next sticky write item

Table 4-34 M_STICKY_WRITE_CMD_ID response

Field	Size (bytes)	Description
length	4	Size of the returned data = 0 bytes

4.21 M_STICKY_CLEAR_CMD_ID (191)

This command sends sticky clear request to target.

Table 4-35 M_STICKY_CLEAR_CMD_ID command

Field	Size (bytes)	Description
param1	4	Size of data field = n bytes
param2	4	Number of sticky clears
data	4	Data[3:0] = number of sticky clears Data[*n*-1:4] = sticky clear items Each sticky clear item defined as: ▪ 1st 4 bytes = mode/address: ▫ Bit 0:19 – register address ▫ Bit 20:23 – mode: – OTP_CONFIG_MODE_COMMON = 1 – OTP_CONFIG_MODE_2MODAL = 2 – OTP_CONFIG_MODE_5MODAL = 3 ▫ Bit 24 – applied pre/post of hardware calibration: 0 = pre, 1 = post ▪ 2nd 4 bytes = register mask Then, next sticky clear item

Table 4-36 M_STICKY_CLEAR_CMD_ID response

Field	Size (bytes)	Description
length	4	Size of the returned data = 20 bytes

4.22 NV_SET_MAC_ADDR_CMD_ID (198)

This command sends the request to target to set MAC address.

Table 4-37 NV_SET_MAC_ADDR_CMD_ID command

Field	Size (bytes)	Description
param1	4	Size of data field = n bytes(5 + (6*nMac));
data	(5 + (6*nMac))	data [0] = 0x01; data [1] = 0x08; data [2] = 0x04; data [3] = 0x06; data [4] = nMac; MAC0(6bytes) + MAC1(6bytes) +….+ MACn-1(6bytes)

Table 4-38 NV_SET_MAC_ADDR_CMD_ID response

Field	Size (bytes)	Description
status	4	0: successfully; others: failure
length	4	Size of the returned data = 0 bytes

4.23 NV_GET_MAC_ADDR_CMD_ID (235)

Table 4-39 NV_ GET_MAC_ADDR_CMD_ID command

Field	Size (bytes)	Description
param1	4	Ignore

Table 4-40 NV_GET_MAC_ADDR_CMD_ID response

Field	Size (bytes)	Description
status	4	0: successfully; others: failure
length	4	Size of the returned MAC address
data	6	6 bytes of MAC address

4.24 M_EEEPROM_BLOCK_READ_ID (200)

Table 4-41 M_EEEPROM_BLOCK_READ_ID command

Field	Size (bytes)	Description
param1	4	Total size of EEPROM
param2	4	Offset address of EEPROM to read from
param3	4	Block size (less than 1000 bytes one time)

Table 4-42 M_EEEPROM_BLOCK_READ_ID response

Field	Size (bytes)	Description
Status	4	0: successfully; others: failure
length	4	offset + block size + block data length
data	Block size + 4	block size (2bytes) + offset (2bytes) + block data

4.25 M_EEEPROM_BLOCK_WRITE_ID (201)

Table 4-43 M_EEEPROM_BLOCK_WRITE_ID command

Field	Size (bytes)	Description
param1	4	Total size of EEPROM
param2	4	Offset address of EEPROM to read from
param3	4	Block size (less than 1000 bytes one time)

Field	Size (bytes)	Description
data	Block size	Block data

Table 4-44 M_EEEPROM_BLOCK_WRITE_ID response

Field	Size (bytes)	Description
status	4	0: successfully; others: failure
length	4	offset + block size
data	4	block size (2bytes) + offset (2bytes)

4.26 M_WRITE_FW_BD_ID (232)

Table 4-45 M_WRITE_FW_BD_ID command

Field	Size (bytes)	Description
param1	4	Ignore
param2	4	Ignore
param3	4	Ignore

Table 4-46 M_WRITE_FW_BD_ID response

Field	Size (bytes)	Description
status	4	0: successfully; others: failure
length	4	The length of firmware board data (0 bytes)

4.27 M_READ_FW_BD_ID (233)

Table 4-47 M_READ_FW_BD_ID command

Field	Size (bytes)	Description
param1	4	Total size of board data in firmware
param2	4	Offset address of board data in firmware
param3	4	Block size (less than 1000 bytes one time)

Table 4-48 M_READ_FW_BD_ID response

Field	Size (bytes)	Description
status	4	0: successfully; others: failure
length	4	offset + block size + block data length
data	Block size + 4	block size (2bytes) + offset (2bytes) + block data

4.28 M_READ_FW_BD_SIZE_ID (234)

Table 4-49 M_READ_FW_BD_ID command

Field	Size (bytes)	Description
param1	4	Ignore
param2	4	Ignore
param3	4	Ignore

Table 4-50 M_READ_FW_BD_ID response

Field	Size (bytes)	Description
status	4	0: successfully; others: failure
length	4	The length of firmware board data
data	2	Board data size